ＡＬＧＯ省配線システム

リファレンスマニュアル

INtime
ComHsUart.RSL

目　次

第１章 関数一覧

第２章 関数仕様

第１章　関数一覧

１）シリアルポートアクセス関数

関　数	機　　能
OpenComm( )	通信デバイスハンドルをオープンします。
CloseComm( )	通信デバイスのハンドルをクローズします。
WriteComm( )	通信デバイスにデータを送信します。
ReadComm( )	通信デバイスからデータの受信を行います。
PurgeComm( )	通信デバイスの出力バッファまたは入力バッファにあるすべての文字を破棄します。
ClearCommError( )	通信エラーの情報を取得して、通信デバイスの現在の状態を通知します。取得後、通信エラーをクリアします。
SetCommState( )	デバイスコントロールブロックの仕様にしたがい通信デバイスの構成をします。
GetCommState( )	通信デバイスの現在の制御設定を取得します。
SetCommTimeouts( )	通信デバイスの送受信時のタイムアウトパラメータの設定をします。
GetCommTimeouts( )	通信デバイスで実行されるすべての読み書き操作のタイムアウトパラメータを取得します。
SetCommMask( )	通信デバイスで監視する一連のイベントを指定します。
GetCommMask( )	通信デバイスのイベントマスクの値を取得します。
WaitCommEvent( )	通信デバイスでイベントが発生するのを待機します。
ResetCommEvent( )	指定された通信イベントオブジェクトを非シグナル状態に設定します。
SetCommConfig( )	通信デバイスの現在の構成を設定します。
GetCommConfig( )	通信デバイスの現在の構成を取得します。
GetCommProperties( )	通信デバイスの通信プロパティの情報を取得します。
SetupComm( )	通信デバイスの通信パラメータを初期化します。
EscapeCommFunction( )	通信デバイスに拡張機能を実行するよう指示します。
GetCommModemStatus( )	モデムの制御レジスタ値を取得します。

第２章　関数仕様

２－１　ライブラリ使用方法

ライブラリを使用したアプリケーション開始のフローチャートを以下に示します。

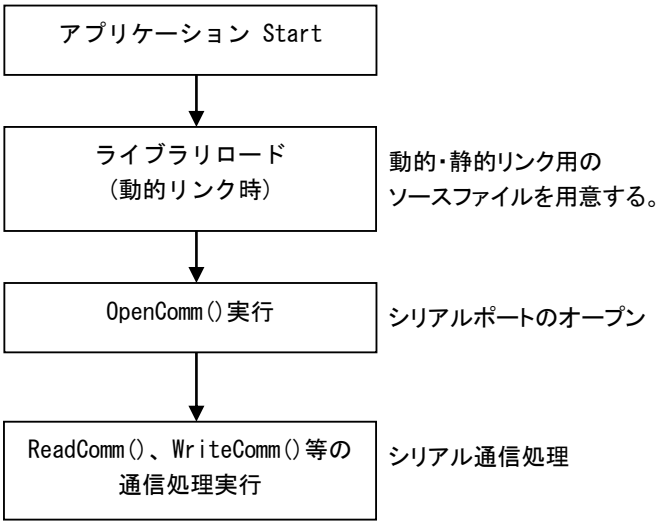

図 2-1-1. フローチャート

ライブラリロード後、シリアルポートのオープンを行うことでシリアルポートへアクセス可能となります。

２－２　シリアルポートアクセス関数

OpenComm 関数

機能　通信デバイスハンドルをオープンします。

書式
```
COMMHANDLE OpenComm (
    LPCSTR lpCommName,
    DWORD dwFlags
);
```

引数　lpCommName：
　NULL で終わるデバイス名称文字列へのポインタを指定します。

デバイス	lpCommName
SIO1	“COM1”
SIO2	“COM2”

dwFlags：
　定義しません。0 を指定します。

戻り値　デバイスハンドル：
　成功。
INVALID_COMM_HANDLE：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー　E_EXIST：
　COMM チャンネルは閉じられているか、既に他のプロセスによって使用中です。
E_PARAM：
　無効なパラメータです。

説明　通信デバイスハンドルをオープンします。
本関数をコール後、戻り値で返されるデバイスハンドルを使用することでシリアルポートにアクセス可能となります。

CloseComm 関数

機能　通信デバイスのハンドルをクローズします。

書式
```
BOOL CloseComm (
    COMMHANDLE hComm
);
```

引数　hComm：
オープンした通信デバイスのハンドル。

戻り値　値 0 以外：
成功。
値 0：
失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー　E_DISCONNECTED：
COMM チャンネルは閉じられています。

説明　通信デバイスハンドルをクローズします。
本関数をコール後、シリアルポートへのアクセスは不可となります。

WriteComm 関数

機能　通信デバイスにデータを送信します。

書式

```
BOOL WriteComm (
    COMMHANDLE hComm,
    LPVOID lpBuffer,
    DWORD nNumberOfBytesToWrite,
    LPDWORD lpNumberOfBytesWritten,
    LPVOID lpOverlapped
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。
lpBuffer：
　通信デバイスに送信するデータを保持しているバッファへのポインタを指定します。
nNumberOfBytesToWrite：
　送信対象のバイト数を指定します。
lpNumberOfBytesWritten：
　実際に書き込まれたバイト数を格納する領域へのポインタを指定します。関数から制御が返ると、この変数に、実際に書き込まれたバイト数が格納されます。何らかの作業やエラーのチェックを行う前に、WriteComm はこの値を 0 に設定します。
lpOverlapped：
　サポートしていません。NULL を指定します。

戻り値

値 0 以外：
　成功。
値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー

E_BUSY：
　送信操作は、既に実行中です。
E_DISCONNECTED：
　COMM チャンネルは閉じられています。
E_IO：
　ハードウェアエラーが発生しました。
E_PARAM：
　無効なパラメータです。

説明　通信デバイスに指定したデータを送信します。
この関数は、エラー発生、DCB 構造体の fAbortOnError が TRUE の場合処理を終了します。

ReadComm 関数

機能　通信デバイスからデータを読み込みます。

書式
```
BOOL ReadComm (
    COMMHANDLE hComm,
    LPVOID lpBuffer,
    DWORD nNumberOfBytesToRead,
    LPDWORD lpNumberOfBytesRead,
    LPVOID lpOverlapped
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。

lpBuffer：
　読み込みデータを格納する領域のポインタを指定します。

nNumberOfBytesToWrite：
　読み取り対象のバイト数を指定します。

lpNumberOfBytesWritten：
　実際に読み取ったバイト数が格納される領域のポインタを指定します。何らかの作業やエラーのチェックを行う前に、ReadComm はこの値を 0 に設定します。

lpOverlapped：
　サポートしていません。NULL を指定します。

戻り値

値 0 以外：
　成功。

値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー

E_BUSY：
　送信操作は、既に実行中です。

E_DISCONNECTED：
　COMM チャンネルは閉じられています。

E_IO：
　ハードウェアエラーが発生しました。

E_PARAM：
　無効なパラメータです。

説明

通信デバイスからデータを読み込みます。
指定されたバイト数の読み取りが終わるか、エラーが発生した場合にこの関数は戻ります。SetCommTimeouts 、GetCommTimeouts の各関数は、この通信タイムアウト値の設定と取得を行います。タイムアウト値の設定を怠ると、予測できない結果が生じることがあります。通信タイムアウトの詳細については、COMMTIMEOUTS 構造体を参照してください。

PurgeComm 関数

機能　通信デバイスの出力バッファまたは入力バッファにあるすべての文字を破棄します。

書式
```
BOOL PurgeComm (
    COMMHANDLE hComm,
    DWORD dwFlags
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。

dwFlags：
　実行する操作を指定します。次の定数を組み合わせて渡すことができます。

dwFlags	内容
PURGE_TXABORT	操作が完了していないものも含め、未処理の書き込みのすべてを中止し、ただちに制御を返します。
PURGE_RXABORT	操作が完了していないものも含め、未処理の読み取りのすべてを中止し、ただちに制御を返します。
PURGE_TXCLEAR	出力バッファの内容を消去します(デバイスドライバの出力バッファが存在する場合)。
PURGE_RXCLEAR	入力バッファの内容を消去します(デバイスドライバの入力バッファが存在する場合)。

戻り値

値 0 以外：
　成功。

値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー

E_DISCONNECTED：
　COMM チャンネルは閉じられています。

E_PARAM：
　無効なパラメータです。

説明

通信デバイスの出力バッファ、入力バッファにあるすべての文字を破棄します。
未処理の読み取り操作または書き込み操作を中止することもできます。
PurgeComm 関数を使って出力バッファをフラッシュ(内容を消去)すると、削除された文字は送信されません。

ClearCommError 関数

機能　通信エラーの情報を取得して、通信デバイスの現在の状態を通知します。取得後、エラーをクリアします。

書式

```
BOOL ClearCommError (
    COMMHANDLE hComm,
    LPDWORD lpErrors,
    LPCOMSTAT lpStat
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。

lpErrors：
　エラーの種類を示すマスクを受け取る 変数へのポインタを指定します。次の１つ以上のエラーコードが格納されます。

lpErrors	内容
CE_BREAK	ハードウェアがブレーク条件を検出しました。
CE_FRAME	ハードウェアがフレーミングエラーを検出しました。
CE_IOE	サポートしていません。
CE_MODE	サポートしていません。
CE_OVERRUN	文字バッファがいっぱいになりました。次の文字は失われます。
CE_RXOVER	入力バッファのオーバーフローが発生しました。
CE_RXPARITY	ハードウェアがパリティエラーを検出しました。
CE_TXFULL	アプリケーションが文字を送信しようとしましたが、出力バッファがいっぱいでした。

lpStat：
　COMSTAT 構造体へのポインタを指定します。この構造体を使って、デバイスの状態情報を受け取ります。NULL を指定した場合、状態情報を受け取りません。

COMSTAT 構造体

```
typedef struct {
    DWORD   fCtsHold: 1;
    DWORD   fDsrHold: 1;
    DWORD   fRlsdHold: 1;
    DWORD   fXoffHold: 1;
    DWORD   fXoffSent: 1;
    DWORD   fEof: 1;
    DWORD   fTxim: 1;
    DWORD   fReserved: 25;
    DWORD   cbInQue;
    DWORD   cbOutQue;
} COMSTAT;
```

fCtsHold：
　CTS 信号検出待機状態が設定されます。このフィールドが TRUE の場合、送信側は CTS 信号が送信されるのを待っています。
fDsrHold：
　サポートしていません。
fRlsdHold：
　サポートしていません。
fXoffHold：
　XOFF 文字受信後、送信の待機状態が設定されます。このフィールドが TRUE の場合、送信側は XOFF 文字受信したため、送信を待機しています。
fXoffSent：
　XOFF 文字送信後、送信の待機状態が設定されます。このフィールドが TRUE の場合、送信側は送信を待機しています。 XOFF 文字が実際の文字にかかわらず、XON として次のキャラクターをとるシステムに送信される場合、送信を待機しています。
fEof：
　サポートしていません。
fTxim：
　サポートしていません。
fReserved：
　予約されています。使わないで下さい。
cbInQue：
　受信バッファ内にある、読まれていないデータ数(バイト)。
cbOutQue：
　送信バッファ内にある、送信されていないデータ数(バイト)。

戻り値

値 0 以外：
　成功。
値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー

E_DISCONNECTED：
　COMM チャンネルは閉じられています。
E_PARAM：
　無効なパラメータです。

説明

通信エラーの情報を取得して、通信デバイスの現在の状態を通知します。
通信エラーが発生した場合に呼び出し、デバイスのエラーフラグをクリアして次の入出力(I/O)操作を可能にします。
fAbortOnError が TRUE の場合、通信エラーが発生すると通信ソフトウェアがすべての読み書き動作を中止します。ClearCommError 関数を呼び出して通信エラーを確認するまで、新たな読み取り、書き込みは実行されません。

SetCommState 関数

機能　デバイス制御ブロック(DCB 構造体)の指定に従って通信デバイスを構成します。

書式

```
BOOL SetCommState (
    COMMHANDLE hComm,
    LPDCB lpDCB
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。
lpDCB：
　通信デバイスの構成情報が入った DCB 構造体へのポインタを指定します。

DCB 構造体

```
typedef struct {
    DWORD   DCBlength;
    DWORD   BaudRate;
    DWORD   fBinary: 1;
    DWORD   fParity: 1;
    DWORD   fOutxCtsFlow: 1;
    DWORD   fOutxDsrFlow: 1;
    DWORD   fDtrControl: 2;
    DWORD   fDsrSensitivity: 1;
    DWORD   fTXContinueOnXoff: 1;
    DWORD   fOutX: 1;
    DWORD   fInX: 1;
    DWORD   fErrorChar: 1;
    DWORD   fNull: 1;
    DWORD   fRtsControl: 2;
    DWORD   fAbortOnError: 1;
    DWORD   fDummy2: 17;
    WORD    wReserved;
    WORD    XonLim;
    WORD    XoffLim;
    BYTE    ByteSize;
    BYTE    Parity;
    BYTE    StopBits;
    char    XonChar;
    char    XoffChar;
    char    ErrorChar;
    char    EofChar;
    char    EvtChar;
    WORD    wReserved1;
} DCB;
```

DCBlength：
　DBC 構造体の大きさ(バイト数)。
BaudRate：
　通信デバイスのボーレート。 このフィールドは、実際のボーレート値または、下記の定数になります。

BaudRate	内容
CBR_1200	1200
CBR_2400	2400
CBR_4800	4800
CBR_9600	9600
CBR_19200	19200
CBR_38400	38400
CBR_57600	57600
CBR_115200	115200

fBinary：
　バイナリモードが可否。 INtime は非バイナリモードをサポートしないのでこのフィールドは TRUE です。
fParity：
　パリティチェックの可否。このフィールドが TRUE の場合、パリティチェックを実行し、エラーを返します。
fOutxCtsFlow：
　出力フローコントロールとして、CTS(送信可)信号をモニタするかどうかの可否。このフィールドが TRUE の場合、CTS(送信可)信号が再び ON になるまでデータ送信を行いません。
fOutxDsrFlow：
　サポートしていません。
fDtrControl：
　サポートしていません。
fDsrSensitivity：
　サポートしていません。
fTXContinueOnXoff：
　入力バッファがいっぱい且つ、ドライバが XoffChar 文字を送信したときに送信を停止するかどうかを指定します。
　このフィールドが TRUE の場合、XoffChar が送信されたときにもデータの送信は続けられます。
fOutX：
　送信中に XON/XOFF フローコントロールを使用するかを指定します。このフィールドが TRUE の場合、XoffChar を受信したときにデータの送信が中断され、XonChar を受信したときにデータの送信を再開します。
fInX：
　受信中に XON/XOFF フローコントロールを使用するかを指定します。このフィールドが TRUE の場合、受信バッファの空き容量が XoffLim 以下になったときに XoffChar を送信し、受信バッファのデータが XonLim 以下になったときに XonChar を送信します。
fErrorChar：
　パリティエラー時に ErrorChar フィールドに設定された文字に置き換えるかどうかを指定します。このフィールドと fParity フィールドが TRUE の場合に置き換えられます。

fNull：
　NULL バイトを破棄するかどうかを指定します。このフィールドが TRUE の場合、受信時に NULL バイトは破棄されます。
fRtsControl：
　RTS フロー制御。このフィールドは下記の定数の中の 1 つです。

fRtsControl	内容
RTS_CONTROL_DISABLE	デバイスのオープン時に RTS ラインを無効にし、そのまま無効状態を保ちます。
RTS_CONTROL_ENABLE	デバイスのオープン時に RTS ラインを有効にし、そのまま有効状態を保ちます。
RTS_CONTROL_HANDSHAKE	RTS のハンドシェイクを有効にします。先行入力バッファ内のデータが半分以下になると、ドライバは RTS ラインを H レベルにします。また、バッファが 4 分の 3 以上まで埋まると、RTS ラインを L レベルにします。ハンドシェイクが有効になると、アプリケーションが EscapeCommFunction 関数を使用して回線を調整した場合エラーが発生します。
RTS_CONTROL_TOGGLE	サポートしていません。

fAbortOnError：
　エラー発生時に、送受信処理を停止するかどうかを指定します。このフィールドが TRUE の場合、エラーが発生すると送受信処理を停止します。ドライバは、アプリケーションが ClearCommError 関数を呼び出してエラーを認証するまで、これ以降の通信操作を受け付けません。
fDummy2：
　予約されています。使わないで下さい。
wReserved：
　予約されています。使わないで下さい。
XonLim：
　XonChar を送信するまでの受信バッファのデータ容量を指定します。受信バッファの容量が XonLim 以下になれば、XonChar を送信します。
XoffLim：
　XoffChar を送信するまでの受信バッファの空き容量を指定します。受信バッファの空き容量が XoffLim 以下になれば、XoffChar を送信します。
ByteSize：
　送受信するバイトデータのビット数。5～8 ビットの間で指定できます。
Parity：
　パリティ方式を指定します。以下の定数より 1 つ指定します。

Parity	内容
NOPARITY	パリティなし
EVENPARITY	偶数パリティ
ODDPARITY	奇数パリティ
MARKPARITY	マークパリティ（サポートしていません）
SPACEPARITY	空白パリティ（サポートしていません）

StopBits：
　使用するストップビットの数を指定します。以下の定数より１つ指定します。

StopBits	内容
ONESTOPBIT	１ストップビット
ONE5STOPBITS	1.5 ストップビット
TWOSTOPBITS	２ストップビット

XonChar：
　送受信の XON 文字を指定します。
XoffChar：
　送受信の XOFF 文字を指定します。
ErrorChar：
　パリティエラー時にエラー文字と置き換える文字を指定します。
EofChar：
　サポートしていません。
EvtChar：
　イベント通知に使用する文字を指定します。
wReserved1：
　予約されています。使わないで下さい。

戻り値　値 0 以外：
　成功。
値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー　E_DISCONNECTED：
　COMM チャンネルは閉じられています。
E_PARAM：
　無効なパラメータです。

説明　デバイス制御ブロック(DCB 構造体)の指定に従って通信デバイスを構成します。
DCB 構造体のメンバーの一部だけを設定する場合でも、他のメンバーも適切な値にするため、GetCommState 関数を使って DCB 構造体を設定してから、該当するメンバーの値を修正するようにします。
SetCommState 関数は、DCB 構造体の XoffChar メンバーが XoffChar と等しいと失敗します。

GetCommState 関数

機能　通信デバイスの現在の制御設定を取得します。

書式
```
BOOL GetCommState (
    COMMHANDLE hComm,
    LPDCB lpDCB
);
```

引数
hComm：
　通信デバイスのハンドルを指定します。
lpDCB：
　通信デバイスの構成情報が入った DCB 構造体へのポインタを指定します。
　DCB 構造体については、SetCommState 関数を参照してください。

戻り値
値 0 以外：
　成功。
値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー
E_DISCONNECTED：
　COMM チャンネルは閉じられています。
E_PARAM：
　無効なパラメータです。

説明　通信デバイスの現在の制御設定を取得します。

SetCommTimeouts 関数

機能　通信デバイスで実行されるすべての読み書き操作のタイムアウトパラメータを設定します。

書式

```
BOOL SetCommTimeouts (
    COMMHANDLE hComm,
    LPCOMMTIMEOUTS lpCommTimeouts
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。
lpCommTimeouts：
　新しいタイムアウト値が入った COMMTIMEOUTS 構造体へのポインタを指定します。

COMMTIMEOUTS 構造体

```
typedef struct {
    DWORD   ReadIntervalTimeout;
    DWORD   ReadTotalTimeoutMultiplier;
    DWORD   ReadTotalTimeoutConstant;
    DWORD   WriteTotalTimeoutMultiplier;
    DWORD   WriteTotalTimeoutConstant;
} COMMTIMEOUTS;
```

ReadIntervalTimeout：
　2 文字を受信するときに、はじめの文字を受信してから、次の文字を受信するまでの最大の許容時間をミリ秒で指定します。はじめの文字を受信してから次の文字を受信するまでに ReadIntervalTimeout 時間経過すると ReadComm 処理を終了します。タイムアウトを行わない場合は 0 を指定します。
ReadTotalTimeoutMultiplier：
　受信処理でトータルタイムアウトの計算に使用する乗数を指定します(1 文字あたりの受信時間)。受信処理ごとにこの値を受信バイト数と掛けます。
ReadTotalTimeoutConstant：
　受信処理でトータルタイムアウトの計算に使用する定数を指定します(受信最小待ち時間)。受信処理ごとにこの値を受信時間(ReadTotalTimeoutMultiplier との積)と足します。ReadTotalTimeoutMultiplier と ReadTotalTimeoutConstant から求めた値が 0 ならば受信処理でトータルタイムアウト時間を使用しません。
WriteTotalTimeoutMultiplier：
　送信処理でトータルタイムアウトの計算に使用する乗数を指定します(1 文字あたりの送信時間)。送信処理ごとにこの値を受信バイト数と掛けます。
WriteTotalTimeoutConstant：
　送信処理でトータルタイムアウトの計算に使用する定数を指定します(送信最小待ち時間)。送信処理ごとにこの値を送信時間(ReadTotalTimeoutMultiplier との積)と足します。WriteTotalTimeoutMultiplier と WriteTotalTimeoutConstant から求めた値が 0 ならば送信処理でトータルタイムアウト時間を使用しません。

戻り値　値 0 以外：
成功。
値 0：
失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー　E_DISCONNECTED：
COMM チャンネルは閉じられています。
E_PARAM：
無効なパラメータです。

説明　通信デバイスで実行されるすべての読み書き操作のタイムアウトパラメータを設定します。

送受信タイムアウトの値は以下の式で計算されますので、これに従って処理に必要なタイムアウト時間を決定してください。
（受信タイムアウト）＝
ReadTotalTimeoutMultiplier ＊（受信バイト数）＋ ReadTotalTimeoutConstant
（送信タイムアウト）＝
WriteTotalTimeoutMultiplier ＊（送信バイト数）＋ WriteTotalTimeoutConstant

GetCommTimeouts 関数

機能　通信デバイスで実行されるすべての読み書き操作のタイムアウトパラメータを取得します。

書式
```
BOOL GetCommTimeouts (
    COMMHANDLE hComm,
    LPCOMMTIMEOUTS lpCommTimeouts
);
```

引数
hComm：
　通信デバイスのハンドルを指定します。
lpCommTimeouts：
　新しいタイムアウト値が入った COMMTIMEOUTS 構造体へのポインタを指定します。
　COMMTIMEOUTS 構造体については、SetCommTimeouts 関数を参照してください。

戻り値
値 0 以外：
　成功。
値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー
E_DISCONNECTED：
　COMM チャンネルは閉じられています。
E_PARAM：
　無効なパラメータです。

説明　通信デバイスで実行されるすべての読み書き操作のタイムアウトパラメータを取得します。
通信デバイスのタイムアウト値については、SetCommTimeouts 関数を参照してください。

SetCommMask 関数

機能　通信デバイスで監視する一連のイベントを指定します。

書式

```
BOOL SetCommMask (
    COMMHANDLE hComm,
    DWORD dwEvtMask
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。

dwEvtMask：
　監視するイベントを指定します。0 を渡すと、どのイベントも監視しません。次の１つ以上の定数を組み合わせて渡すことができます。

dwEvtMask	内容
EV_BREAK	サポートしていません。
EV_CTS	CTS(送信可)信号の状態が変わったとき。
EV_DSR	サポートしていません。
EV_ERR	回線状態エラーが発生したとき。回線状態エラーには、CE_FRAME、CE_OVERRUN、CE_RXPARITY があります。
EV_RING	サポートしていません。
EV_RLSD	サポートしていません。
EV_RXCHAR	1 文字受信し、入力バッファに入れたとき。
EV_RXFLAG	イベント文字を受信し、入力バッファに入れたとき。イベント文字はデバイスの DCB 構造体で指定します。
EV_TXEMPTY	出力バッファの最後の文字を送信したとき。

戻り値

値 0 以外：
　成功。

値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー

E_DISCONNECTED：
　COMM チャンネルは閉じられています。

E_PARAM：
　無効なパラメータです。

説明

通信デバイスで監視する一連のイベントを指定します。
この関数は、特定の通信資源で監視できる一連のイベントを設定します。 WaitCommEvent 関数に通信資源のハンドルを渡して、イベントが発生するのを待機します。通信資源の現在のイベントマスクを取得するには、GetCommMask 関数を使います。

GetCommMask 関数

機能　通信デバイスのイベントマスクの値を取得します。

書式
```
BOOL GetCommMask (
    COMMHANDLE hComm,
    LPDWORD lpEvtMask
);
```

引数
hComm：
　通信デバイスのハンドルを指定します。
lpEvtMask：
　変数へのポインタを指定します。変数には、現在監視対象になっているイベントを示すイベントマスクが格納されます。
　イベントマスクの詳細は、SetCommMask 関数を参照してください。

戻り値
値 0 以外：
　成功。
値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー
E_DISCONNECTED：
　COMM チャンネルは閉じられています。
E_PARAM：
　無効なパラメータです。

説明
通信デバイスのイベントマスクの値を取得します。
GetCommMask 関数はマスク変数を使って特定の通信資源で監視できるイベントを取得します。
WaitCommEvent 関数に通信資源のハンドルを渡して、それらのイベントの発生を待機します。
通信資源のイベントマスクを変更するには、SetCommMask 関数を使います。

WaitCommEvent 関数

機能　通信デバイスでイベントが発生するのを待機します。

書式
```
BOOL WaitCommEvent (
    COMMHANDLE hComm,
    LPDWORD lpEvtMask,
    DWORD dwTimeoutMs
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。

lpEvtMask：
　発生したイベントの種類を示すマスクが格納領域のポインタを設定します。エラーが発生すると、0 が格納されます。
　イベントマスクの詳細は、SetCommMask 関数を参照してください。

dwTimeoutMs：
　[入力]コールスレッドの待機するミリ秒時間。

dwTimeoutMs	内容
NO_WAIT	スレッドは待機しません。
WAIT_FOREVER	スレッドは要求が満了するまで待機します。
1-655349	コールスレッドは指定したミリ秒時間スリープします。

戻り値

値 0 以外：
　成功。

値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー

E_DISCONNECTED：
　COMM チャンネルは閉じられています。

E_PARAM：
　無効なパラメータです。

説明

通信デバイスでイベントが発生するのを待機します。
この関数で監視するイベントは、デバイスのハンドルに関連付けられているイベントマスクによって示されます。
この関数は、指定された通信資源で一連のイベントを監視します。通信資源の現在のイベントマスクを設定、取得する場合は、それぞれ SetCommMask 関数と GetCommMask 関数を使います。

ResetCommEvent 関数

機能　指定された通信イベントオブジェクトを非シグナル状態に設定します。

書式
```
BOOL ResetCommEvent (
    COMMHANDLE hComm,
    DWORD dwEvents
);
```

引数
hComm：
　通信デバイスのハンドルを指定します。
dwEvents：
　WaitCommEvent で指定したイベントフラグを指定します。
　イベントマスクの詳細は、SetCommMask 関数を参照してください。

戻り値
値 0 以外：
　成功。
値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー
E_DISCONNECTED：
　COMM チャンネルは閉じられています。
E_PARAM：
　無効なパラメータです。

説明　指定された通信イベントオブジェクトを非シグナル状態に設定します。

SetCommConfig 関数

機能　通信デバイスの現在の構成を設定します。

書式

```
BOOL SetCommConfig (
    COMMHANDLE hComm,
    LPCOMMCONFIG lpCC,
    DWORD dwSize
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。
lpCC：
　COMMCONFIG 構造体へのポインタを指定します。

COMMCONFIG 構造体

```
typedef struct {
    DWORD   dwSize;
    WORD    wVersion;
    WORD    wReserved;
    DCB     dcb;
    DWORD   dwProviderSubType;
    DWORD   dwProviderOffset;
    DWORD   dwProviderSize;
    WCHAR   wcProviderData[1];
} COMMCONFIG;
```

dwSize：
　COMMCONFIG 構造体のサイズ(バイト数)。
wVersion：
　COMMCONFIG 構造体のバージョン番号。
wReserved：
　予約されています。使用しないで下さい。
dcb：
　シリアル通信 RS-232 用の DCB 構造体。
dwProviderSubType：
　通信デバイスの種類、データ書式を確認します。このメンバーの値は 1(RS-232) にします。
dwProviderOffset：
　このメンバーの値は 0 にします。
dwProviderSize：
　このメンバーの値は 0 にします。
wcProviderData：
　このメンバーの値は 0 にします。

dwSize：
　lpCC パラメータが示す構造体のサイズ(バイト数)を指定します。

戻り値　値 0 以外：
　成功。
値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー　E_DISCONNECTED：
　COMM チャンネルは閉じられています。
E_PARAM：
　無効なパラメータです。

説明　通信デバイスの現在の構成を設定します。

GetCommConfig 関数

機能　通信デバイスの現在の構成を取得します。

書式
```
BOOL GetCommConfig (
    COMMHANDLE hComm,
    LPCOMMCONFIG lpCC,
    LPDWORD lpdwSize
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。

lpCC：
　COMMCONFIG 構造体へのポインタを指定します。
　COMMCONFIG 構造体については、SetCommConfig 関数を参照してください。

lpdwSize：
　lpCC パラメータが示すバッファのサイズ(バイト数)を指定する変数へのポインタを渡します。関数が成功すると、コピーされたバイト数が変数に格納されます。

戻り値

値 0 以外：
　成功。

値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー

E_DISCONNECTED：
　COMM チャンネルは閉じられています。

E_PARAM：
　無効なパラメータです。

説明　通信デバイスの現在の構成を取得します。

GetCommProperties 関数

機能　通信デバイスの通信プロパティの情報を取得します。

書式

```
BOOL GetCommProperties (
    COMMHANDLE hComm,
    LPCOMMPROP lpCommProp
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。

lpCommProp：
　COMMPROP 構造体へのポインタを指定します。構造体には通信プロパティの情報が格納されます。取得した情報は、SetCommState 関数、SetCommTimeouts 関数、SetupComm 関数等で通信デバイスの設定をするときに使えます。

COMMPROP 構造体

```
typedef struct {
    WORD    wPacketLength;
    WORD    wPacketVersion;
    DWORD   dwServiceMask;
    DWORD   dwReserved1;
    DWORD   dwMaxTxQueue;
    DWORD   dwMaxRxQueue;
    DWORD   dwMaxBaud;
    DWORD   dwProvSubType;
    DWORD   dwProvCapabilities;
    DWORD   dwSettableParams;
    DWORD   dwSettableBaud;
    WORD    wSettableData;
    WORD    wSettableStopParity;
    DWORD   dwCurrentTxQueue;
    DWORD   dwCurrentRxQueue;
    DWORD   dwProvSpec1;
    DWORD   dwProvSpec2;
    WCHAR   wcProvChar[1];
} COMMPROP;
```

wPacketLength：
　COMMPROP 構造体のサイズ(バイト数)。

wVersion：
　構造体のバージョン番号。このメンバーの値は 2 を返します。

dwServiceMask：
　この通信ドライバが実行するサービスのビットマスク値。このメンバーの値は SP_SERIALCOMM になります。

dwReserved1：
　予約されています。使わないで下さい。
dwMaxTxQueue：
　通信デバイスの出力バッファ最大サイズ(バイト数)。
dwMaxRxQueue：
　通信デバイスの入力バッファ最大サイズ(バイト数)。
dwMaxBaud：
　最大許容ボーレート。このメンバーの値は BAUD_USER になります。
dwProvSubType：
　通信デバイスの種類。このメンバーの値は PST_RS232 になります。
dwProvCapabilities：
　通信デバイスが提供している機能のビットマスク値。
dwSettableParams：
　変更可能な通信パラメータを示すビットマスク値。
dwSettableBaud：
　使用できるボーレートを示すビットマスク値。
wSettableData：
　設定できるデータビット数を示すビットマスク値。
wSettableStopParity：
　設定できるストップビットとパリティを示すビットマスク値。
dwCurrentTxQueue：
　ドライバの内部出力バッファサイズ(バイト数)。
dwCurrentRxQueue：
　ドライバの内部入力バッファサイズ(バイト数)。
dwProvSpec1：
　通信デバイス特有データ。このメンバーの値は 0 を返します。
dwProvSpec2：
　通信デバイス特有データ。このメンバーの値は 0 を返します。
wcProvChar：
　通信デバイス特有データ。このメンバーの値は 0 を返します。

戻り値

値 0 以外：
　成功。
値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー

E_DISCONNECTED：
　COMM チャンネルは閉じられています。
E_PARAM：
　無効なパラメータです。

説明

通信デバイスの通信プロパティの情報を取得します。

SetupComm 関数

機能　通信デバイスの通信パラメータを初期化します。

書式
```
BOOL SetupComm (
    COMMHANDLE hComm,
    DWORD dwInQueue,
    DWORD dwOutQueue
);
```

引数
hComm：
　通信デバイスのハンドルを指定します。
dwInQueue：
　デバイスの内部入力バッファの推奨サイズ(バイト数)を指定します。
dwOutQueue：
　デバイスの内部出力バッファの推奨サイズ(バイト数)を指定します。

戻り値
値 0 以外：
　成功。
値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー
E_DISCONNECTED：
　COMM チャンネルは閉じられています。
E_PARAM：
　無効なパラメータです。

説明　通信デバイスの通信パラメータを初期化します。
本関数はサポートしていません。

EscapeCommFunction 関数

機能　通信デバイスに拡張機能を実行するよう指示します。

書式
```
BOOL EscapeCommFunction (
    COMMHANDLE hComm,
    DWORD dwFunc
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。

dwFunc：
　実行する拡張機能のコードを指定します。

dwFunc	内容
CLRDTR	サポートしていません。
CLRRTS	RTS(送信要求)信号を消去します。
SETDTR	サポートしていません。
SETRTS	RTS(送信要求)信号を送信します。
SETXOFF	XOFF 文字を受信したときのように送信を行います。
SETXON	XON 文字を受信したときのように送信を行います。
SETBREAK	サポートしていません。
CLRBREAK	サポートしていません。

戻り値

値 0 以外：
　成功。

値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー

E_DISCONNECTED：
　COMM チャンネルは閉じられています。

E_PARAM：
　無効なパラメータです。

E_CONTEXT：
　フローコントロールの有効時に、フローコントロール回線の変更を試みました。

説明　通信デバイスに拡張機能を実行するよう指示します。

GetCommModemStatus 関数

機能　モデムの制御レジスタ値を取得します。

書式
```
BOOL GetCommModemStatus (
    COMMHANDLE hComm,
    LPDWORD lpModemStat
);
```

引数

hComm：
　通信デバイスのハンドルを指定します。

lpModemStat：
　モデム制御レジスタの現在の状態を示す値が格納される変数へのポインタを指定します。この値は、次の定数の組み合わせになります。

lpModemStat	内容
MS_CTS_ON	CTS(送信可)信号がオンです。
MS_DSR_ON	サポートしていません。
MS_RING_ON	サポートしていません。
MS_RLSD_ON	サポートしていません。

戻り値

値 0 以外：
　成功。

値 0：
　失敗。拡張エラー情報を取得する場合、GetLastRtError 関数を使います。

拡張エラー

E_DISCONNECTED：
　COMM チャンネルは閉じられています。

E_PARAM：
　無効なパラメータです。

E_CONTEXT：
　フローコントロールの有効時に、フローコントロール回線の変更を試みました。

説明　モデムの制御レジスタ値を取得します。
GetCommModemStatus 関数は WaitCommEvent 関数を使って CTS 信号を監視する場合に有効です。信号の状態変化を検出するには、WaitCommEvent 関数を使って待機し、次に GetCommModemStatus 関数を使って変化後の状態を調べます。

このリファレンスマニュアルについて

（１）本書の内容の一部または全部を当社からの事前の承諾を得ることなく、無断で複写、複製、掲載することは固くお断りします。
（２）本書の内容に関しては、製品改良のためお断りなく、仕様などを変更することがありますのでご了承下さい。
（３）本書の内容に関しては万全を期しておりますが、万一ご不審な点や誤りなどお気付きのことがございましたらお手数ですが巻末記載の弊社もしくは、営業所までご連絡下さい。その際、巻末記載の書籍番号も併せてお知らせ下さい。

７６ＩＴ１０００１Ａ　２０１６年　１１月　初版
７６ＩＴ１０００１Ｂ　２０１７年　１月　第２版

株式会社アルゴシステム

本社
〒５８７－００２１　大阪府堺市美原区小平尾６５６番地
ＴＥＬ（０７２）３６２－５０６７
ＦＡＸ（０７２）３６２－４８５６

ホームページ　http://www.algosystem.co.jp/