Vertex Standard

ユーザーズマニュアル

LYrmDLL （WinSock32版）

株式会社バーテックス　スタンダード　システム機器事業部

Version 2.0
2000/11/01
（Ver2.032）

目次

１．はじめに

ＬＹｒｍＤｌｌはＬＡＮアダプタを使用し、ネットーワーク環境でウィンドウズアプリケーションから八重洲無線株式会社製の無線モデム（YRM-311, YRM-211, YSM-2400DN）を制御するためのソフトウェアです。

ネットワークプロトコルはＴＣＰ／ＩＰに対応しており、ソケットインターフェースを使用して通信制御を行います。
その為に、実行前にオペレーティングシステム上に TCP/IP のプロトコルがインストールされている事を確認してください。

本ＤＬＬは以下のＯＳにて確認済みです。

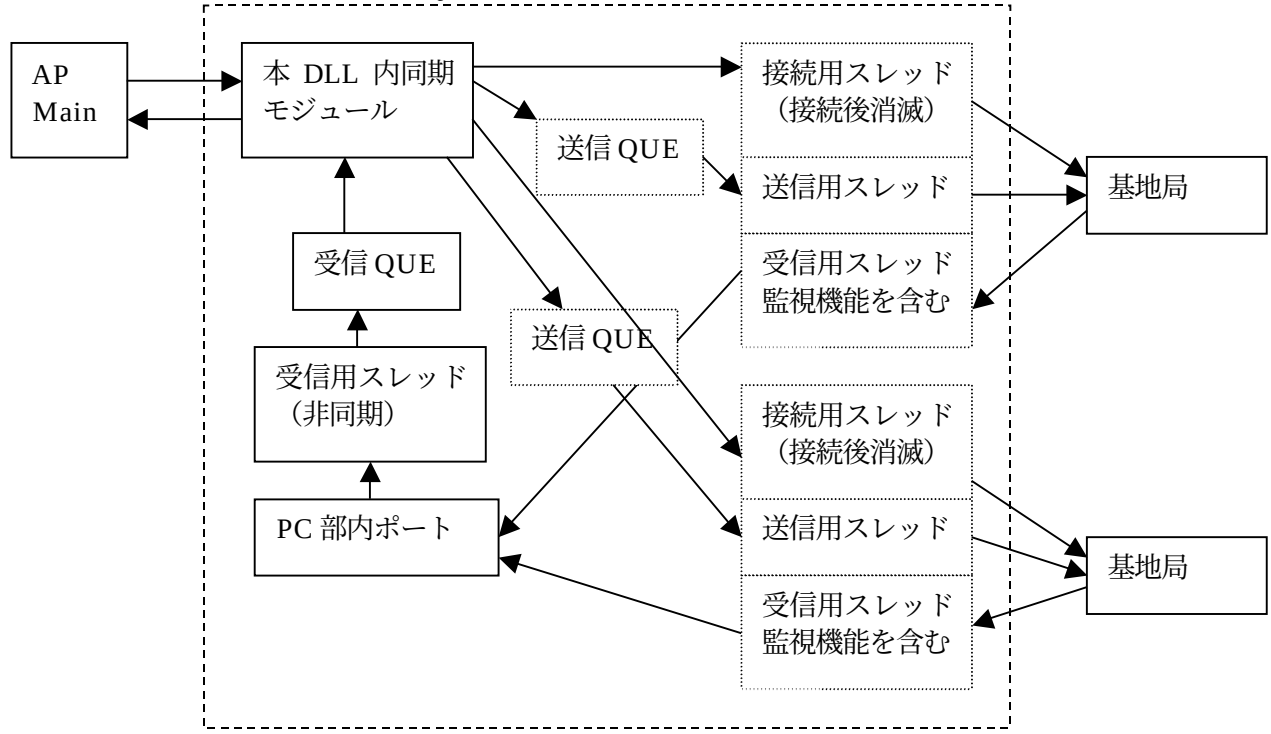
Microsoft(R) Windows 95
Microsoft(R) Windows 98
Microsoft(R) Windows NT 4.0

注：基地局とはＬＡＮアダプタと無線モデムの組み合わせを表します。

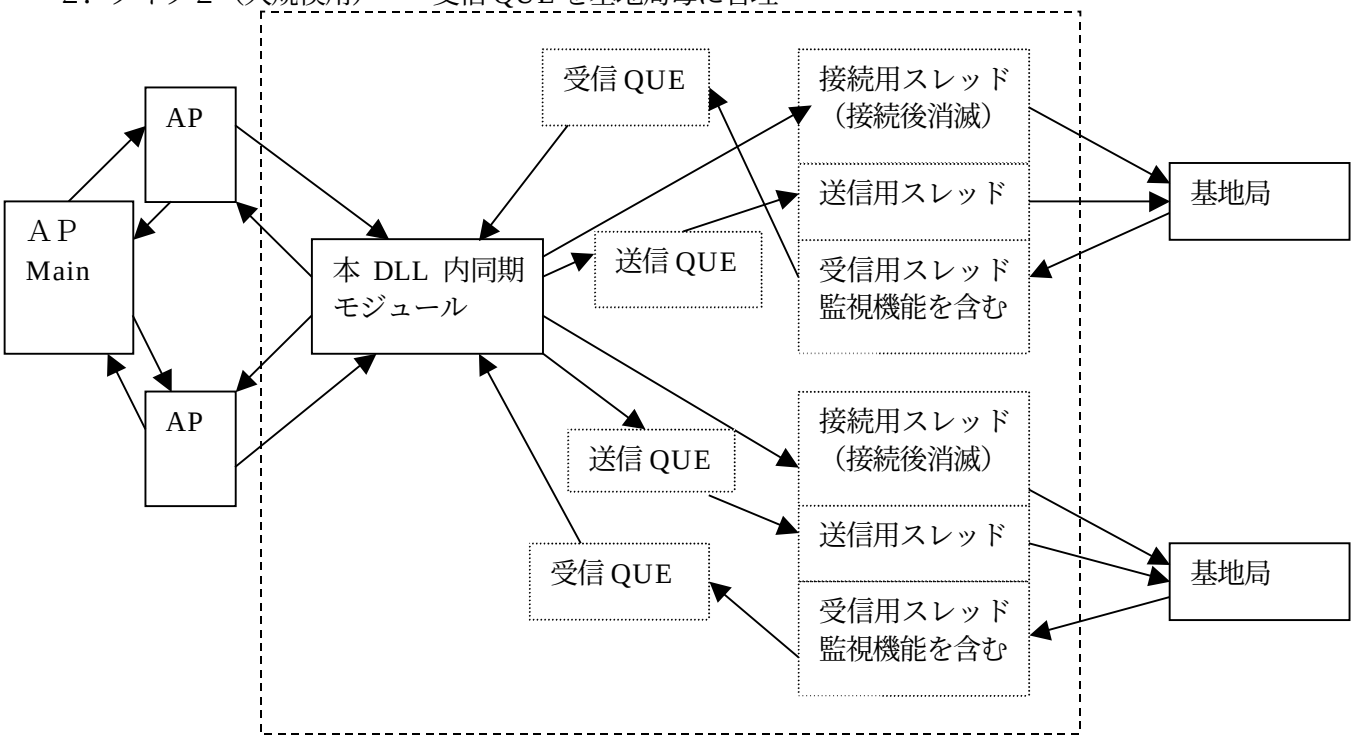
２．動作概要

本 DLL のモジュールは、AP と同期を取り動作するもの（モード設定）と
接続用スレッド（接続後消滅）と受信用スレッド、送信用スレッドに分かれます。
本 DLL の使用方法は以下の２タイプに分かれます。

１．タイプ１（小規模用）　　受信 QUE を１個にて管理



２．タイプ２（大規模用）　　受信 QUE を基地局毎に管理



アプリケーションの作成について

本ＤＬＬファンクションは、接続、送受信共に非同期で行われます。
その為、接続確立と送信処理はファンクション呼出し後、必ず成否の判定フクションの呼出しを行ってから次のファンクション呼出しを実行してください。
（応答待ちの間に、ＡＰ側の内部作業を行う様にコーデイングすると効率が良くなります。）

① LYrmInit を呼出して本ＤＬＬの初期設定を行う。

② LYrmOpenThread(LYrmAllOpenThread) にて基地局との接続命令を発行する。

③ ＡＰ側の初期化処理等

④ LYrmStatusCheck を呼出し基地局との接続状態を確認する。

⑤ LYrmRecvData を呼出し受信データを取得する。
QueMode = 0(１個のキューで管理)の場合はハンドル番号=0 にて呼出しを行う
QueMode = 1(基地局毎にキューを確保)の場合は②で取得したハンドル番号にて呼出しを行う

⑥ レスポンスデータの編集等を行う。

⑦ LYrmSendData(LYrmSendDataB) にて送信側にレスポンスを通知する

⑧ ＡＰ側の内部作業等の実行。

⑨ LYrmCmdCheck を呼出し送信が正常に行われた事を確認。

④～⑨を繰り返す

⑩ LYrmTerm を呼出して処理終了。

３．ＤＬＬファンクション

ＬＹｒｍＤｌｌはアプリケーションが利用できる、以下のファンクションを提供します。

LYrmInit	・・・	共通情報の設定を行う。 必ず、他のファンクションにさきがけて呼出してください。
LYrmTerm	・・・	終了処理。 必ず、終了時に呼出してください。
LYrmOpenAllThread	・・・	複数の基地局をＯＰＥＮする。 １回の送受信毎に OPEN, CLOSE を行う必要はありません。
LYrmOpenThread	・・・	指定基地局をＯＰＥＮする。
LYrmAllClose	・・・	複数の基地局をＣＬＯＳＥする。
LYrmClose	・・・	指定基地局をＣＬＯＳＥする。
LYrmStatusCheck	・・・	基地局の状態を取得する。
LYrmCmdCheck	・・・	送信コマンドに対する返信結果を取得する。
LYrmRecvData	・・・	データの受信を行う。
LYrmRecvDataLen	・・・	データの受信を行う（リターン値に受信データ BYTE 数をセット）
LYrmGetRID	・・・	基地局の Gr. ID. ＮＯを取得する。
LYrmSendData	・・・	テキストデータの送信を行う。
LYrmSendDataNoLnk	・・・	テキストデータの送信を行う。[LNK]レスポンス無し
LYrmSendDataB	・・・	バイナリーデータの送信を行う。
LYrmSendDataBNoLnk	・・・	バイナリーデータの送信を行う。[LNK]レスポンス無し
LYrmSendSTO	・・・	「ＳＴＯ」コマンドの送信を行う。
LYrmSendCON	・・・	「ＣＯＮ」コマンドの送信を行う。
LYrmSendDCN	・・・	「ＤＣＮ」コマンドの送信を行う。
LYrmSendFree	・・・	任意の(YSM-2400, YRM-311, YRM-211) コマンドの送信
LYrmFileTrans	・・・	ファイルの送受信を行う。
LYrmFileTrans_Stop	・・・	ファイル送受信の強制停止を行う。
LYrmFileTrans_Status	・・・	ファイル送受信状態のチェックを行う。

ＬＹｒｍＩｎｉｔ

宣言形式

BOOL LYrmInit(LYrmCommonInfo *Inf, DWORD My_Port, DWORD ResTime)

```
Declare Function LYrmInit Lib "Lyrmdll.dll" (ByRef CommonInfo As info,
                    ByVal My_Port As Long, ByVal ResTime As Long) As Long
```

（info）はユーザー定義型

```
Type info       ' ユーザー定義型を作成します。
    DoubleDelete As Long
    QueMode As Long
    MaxQueCnt As Long
    MaxSndQueCnt As Long
    CheckTime As Long
    LogFilename As String * 256
    LogMode As Long
    AutoConnect As Long
End Type
```

目的

共通情報の設定を行う
ＤＬＬ起動時１回設定を行い、２回目以降の設定は無効となる

引数

LYrmCommonInfo	*Inf	共通情報
DWORD	My_Port	QueMode=0 の場合の自 PC 内部ポート番号を指定 My_Port=0 にすると自動的に「４４７８０」を使用する。
DWORD	ResTime	返信レスポンス待機時間（単位＝秒）

LyrmCommonInfo の説明

```
typedef struct _LYrmCommonInfo {            // 受信スレッド確立情報
        DWORD       DoubleDelete;   // 0=同一 HT よりの同一受信データを生かす
                                    // 1=同一 HT よりの同一受信データを削除
        DWORD       QueMode;        // 0=受信キューを１個にて管理
                                    // 1=受信キューを基地局毎に確保
        DWORD       MaxQueCnt;      // 受信キューの確保数
        DWORD       MaxSndQueCnt;   // 送信キューの確保数
        DWORD       CheckTime;      // 基地局の監視タイマー間隔（秒）
        char        *LogFilename;   // ログファイル名称（フルパスで表記）
                                    // 文字列の終端は null(0)をｾｯﾄしてください
        DWORD       LogMode;        // 0=ログファイル出力無し
                                    // 1=全ての基地局の情報を１ファイルへ出力
                                    // 2=LogFilename+IP ｱﾄﾞﾚｽにてファイル出力
        DWORD       AutoConnect;    // 0=エラー切断時の自動再接続機能無し
                                    // 1=      〃              有り
} LYrmCommonInfo;
```

注記

必ず他のファンクションにさきがけて呼出してください。
MaxQueCnt は設定値が５以下の場合は、本 DLL が受信キューの数を５とセットします。
MaxS ndQueCnt は設定値が３以下の場合は、本 DLL が送信キューの数を３とセットします。

使用例

```
LYrmCommonInfo          ComI;
BOOL                    Ret;

ComI.DoubldeDelete      = 1;      // 同一 HTID のﾃﾞｰﾀを削除する
ComI.QueMode            = 0;      // 受信ｷｭｰを１個で管理する
ComI.MaxQueCnt          = 100;    // 受信ｷｭｰの確保数を 100
ComI.MaxSndQueCnt       = 50;     // 送信ｷｭｰの確保数を 50
ComI.CheckTime          = 10;     // 監視間隔を 10 秒
// ﾛｸﾞﾌｧｲﾙ名称 = “C:¥Temp¥LYrmDll.Log”
strcpy(ComI.LogFilename, “C:¥¥TEMP¥¥LYrmDll.Log”);
ComI.LogMode            = 1;      // 1 個のﾛｸﾞﾌｧｲﾙへ出力
ComI.AutoConnect        = 1;      // 自動再接続機能

// 内部ﾎﾟｰﾄ番号=44780, 返信レスポンス時間=30 秒にて呼出し
Ret = LYrmInit(&ComI, 0, 30);
```

ＬＹｒｍＴｅｒｍ

宣言形式

BOOL　LYrmTerm( )

Declare Function LYrmTerm Lib "Lyrmdll.dll" ( ) As Long

目的

アプリケーションの終了時に必ずコールしてください。
現在、接続中の基地局を全て切断し内部メモリーを解放します。

引数

無し

注記

必ずアプリケーション終了時に呼出してください。

使用例

```
BOOL            Ret;

Ret = LYrmTerm();
```

ＬＹｒｍｏｐｅｎＡｌｌＴｈｒｅａｄ

宣言形式

BOOL　LYrmOpenAllThread(DWORD　n, char *IPADRS,
DWORD *PortNo, DWORD *handle)

Declare Function LYrmOpenAllThread Lib "LYrmDll.dll" (ByVal n As Long,
ByVal ipad As String, ByRef PortNo As Any, ByRef handle As Any) As Long

目的

受信、送信スレッドの起動（複数基地局との接続）
接続用のスレッド（接続完了後自動消滅）を起動しますので本関数は待機状態には
なりません。
接続完了の確認は、**LyrmStatusCheck** 関数を使用してください。

引数

DWORD	n	接続を行う基地局の数
char	*IPADRS	基地局のＩＰアドレス NULL (0) で区切って複数設定を行う。
DWORD	*PortNo	基地局にポート番号
DWORD	*handle	基地局を特定するハンドル番号を返す 以降の関数は全てこのハンドル番号を使用する。
BOOL	関数値	TRUE=エラー無し、FALSE=エラー有り FALSE が返された場合は(**LyrmStatusCheck**)にて確認

注記

本ファンクション及び **LYrmOpenThread** が正常に終了すれば、
無線モデム（基地局）へ「**STO, RID**」のコマンドを送信します。

ＳＯＴコマンドでは、下記の３項目の判定を行います。内容が違う場合に設定を行ないます。
１）データ受信時のヘッダの有無　　・・・有り
２）データ受信時の終端コードに有無　　・・・終端コード有り
３）終端コードの指定　　・・・ＣＲとする

ＲＩＤコマンドでは、返信データより無線モデムの種類(YRM-311, YRM-211)を判定します。

使用例

```
BOOL        Ret;
Char        IPadrs[40];
Int         Portno[2];
Int         handle[2];

Strcpy(IPadrs, ”192.168.0.1”);      strcpy(IPadrs[strlen(Ipadrs)+1], ”192.168.0.2”);
Portno[0] = 1111;                   Portno[1] = 1112;
// IP ｱﾄﾞﾚｽ[192.168.0.1]ﾎﾟｰﾄ[1111] と IP ｱﾄﾞﾚｽ[192.168.0.2]ﾎﾟｰﾄ[1112]の２局を OPEN
Ret = LYrmAllOpenThread(2, IPadrs, Portno, handle);]
```

ＬＹｒｍＯｐｅｎＴｈｒｅａｄ

宣言形式

BOOL　LYrmOpenThread(char IPADRS, DWORD PortNo, DWORD *handle)

Declare Function LYrmOpenThread Lib "LYrmDll.dll" (ByVal ipadrs As String,
ByVal PortNo As Long, ByRef handle As Long) As Long

目的

受信用スレッドの起動（指定基地局接続）
接続用のスレッド（接続完了後自動消滅）を起動しますので本関数は待機状態には
なりません。
接続完了の確認は、**LyrmStatusCheck** 関数を使用してください。

引数

char	* IPADRS	基地局のＩＰアドレス 文字列の終端は null(0)をｾｯﾄしてください。
DWORD	PortNo	基地局にポート番号
DWORD	*handle	基地局を特定するハンドル番号を返す 以降の関数は全てこのハンドル番号を使用する。
BOOL	関数値	TRUE=エラー無し、FALSE=エラー有り FALSE が返された場合は(**LyrmStatusCheck**)にて確認

使用例

```
BOOL        Ret;
Char        IPadrs[20];
Int         Portno;
Int         handle;

Strcpy(IPadrs, ”192.168.0.1”);        Portno = 1111;
// IP ｱﾄﾞﾚｽ[192.168.0.1]ﾎﾟｰﾄ[1111]の局を OPEN
Ret = LYrmOpenThread( IPadrs, Portno, &handle);]
```

ＬＹｒｍＡｌｌＣｌｏｓｅ

宣言形式

BOOL　LYrmAllClose( )

Declare Function LYrmAllClose Lib "LYrmDll.dll" ( ) As Long

目的

基地局との接続を切断して受信、送信スレッドを終了する。
アプリケーション終了時は、**LyrmTerm** をコールすれば本ファンクションの
機能も含んでいますのでコールする必要はありません。

引数

BOOL	関数値	TRUE=エラー無し、FALSE=エラー有り FALSE が返された場合は(**LyrmStatusCheck**)にて確認

使用例

```
BOOL        Ret;

Ret = LYrmAllClose();
```

ＬＹｒｍＣｌｏｓｅ

宣言形式

BOOL　LYrmClose(DWORD n, DWORD *handle)

```
Declare Function LYrmClose Lib "LYrmDll.dll" (ByVal n As Long,
                                              ByRef handle As Any) As Long
```

目的

IP アドレスにて指定された基地局との接続を切断して受信スレッドを
終了する

引数

DWORD	n	接続を切断する基地局数
DWORD	*handle	接続を切断するする基地局のハンドル番号
BOOL	関数値	TRUE=エラー無し、FALSE=エラー有り FALSE が返された場合は(**LyrmStatusCheck**)にて確認

使用例

```
BOOL            Ret;
Int             handle[2];

Handle[0] = 3;              handle[1] = 5;
// ﾊﾝﾄﾞﾙ番号[ 3, 5]  の２局を CLOSE
Ret = LYrmClose(2, handle);
```

ＬＹｒｍＳｔａｔｕｓＣｈｅｃｋ

宣言形式

BOOL　LYrmStatusCheck(DWORD n, DWORD *handle,　DWORD *status)

Declare Function LYrmStatusCheck Lib "LYrmDll.dll" (ByVal n As Long,
ByRef　handle As Any, ByRef Status As Any) As Long

目的

基地局との通信状況（ステータス）を取得する。

引数

DWORD	n	ステータスを取得する基地局数 -1 をセットすると全ての基地局をチェックする。
DWORD	*handle	ステータスを取得する基地局のハンドル番号
DWORD	*status	0=エラー無し, 9=該当するハンドル番号無し 2=PC 内部ポート接続中 1=基地局接続中、 -1=基地局との接続が切れている可能性あり、 -2= PC 内部ポートとの接続が切れている可能性あり、 -3=無線モデムとの接続をリトライ回数分試みたが、 接続出来なかった、 -4=メモリーのアロック又はフリーエラー
BOOL	関数値	TRUE=エラー無し FALSE=指定ポートの内１ヶ所以上でエラー発生

注記

PC 内部ポートの情報（２、－２）は、指定基地局が全てエラー無しの場合に
Status[0]に値をセットされる。
メモリーエラーは、基地局と内部ポートの情報に優先して status[0]へセットされる。

使用例

```
BOOL        Ret;
Int         handle[2];
Int         status[2];

handle[0] = 3;      handle[1] = 5;
// ﾊﾝﾄﾞﾙ番号[ 3, 5]  の２局のｽﾃｰﾀｽを取得
Ret = LYrmStatusCheck(2, handle, status);
```

ＬＹｒｍＣｍｄＣｈｅｃｋ

宣言形式

BOOL　LYrmCmdCheck(DWORD mode, DWORD　handle,　DWORD n,
DWORD *htid , DWORD *status)

Declare Function LYrmCmdCheck Lib "LYrmDll.dll" (ByVal mode As Long,
ByVal handle As Any,ByVal n As Long, ByRef htid As Long, ByRef Status As Any) As Long

目的

基地局との通信状況（ステータス）を取得する。

引数

DWORD	mode	0=通常送信,　1=ローミング送信
DWORD	handle	ステータスを取得する基地局のハンドル番号
DWORD	n	返信ステータスを取得する送信相手先 HTID 数
DWORD	*htid	返信ステータスを取得する送信相手先 HTID [STO, OBL ..]等の自局送信の場合は htid=0 をセット すれば取得できます。
DWORD	*status	0=エラー無し, -1=返信データ未受信 1 = ”NID”,　2 = “BSY”,　3 = “LNG”,　4 = “ENG” 5 = “CER”,　6=”MER”,　7=”LBT”,　8=”MNG” 9=ﾊﾝﾄﾞﾙ番号指定ｴﾗｰ 10=送信ｷｭｰが一杯になっている 21=ﾛｰﾐﾝｸﾞ時の基地局検索ｴﾗｰ
BOOL	関数値	TRUE=エラー無し FALSE=指定ポートの内１ヶ所以上でエラー発生

注記

無線モデムが YSM-2400 でない場合はﾛｰﾐﾝｸﾞ指示は無効となります。

使用例

```
BOOL            Ret;
Int             handle;
Int             mode;
Int             htid[2]
Int             status[2];

Mode = 0;       // 通常時のﾁｪｯｸ
handle = 3;
htid[0] = 0;    htid[1] = 0201
// ﾊﾝﾄﾞﾙ番号=3 の基地局の自局と 0201 への送信ｺﾏﾝﾄﾞｽﾃｰﾀｽを取得
Ret = LYrmCmdCheck(mode, handle, 2, htid, status);
```

ＬＹｒｍＲｅｃｖＤａｔａ

宣言形式

BOOL　LYrmRecvData(DWORD *handle, DWORD *Nbyte, char *data, DWORD *htid)

Declare Function LYrmRecvData Lib "LYrmDll.dll" (ByRef handle As Long,
ByRef Nbyte As Long, Byval Data As String, ByRef htid As Long) As Long

目的

受信キュー（指定基地局）にデータが存在すればデータを AP 側へ渡します。
データが存在しない場合も、制御を戻しますからＡＰ側が待ち状態にはなりません。

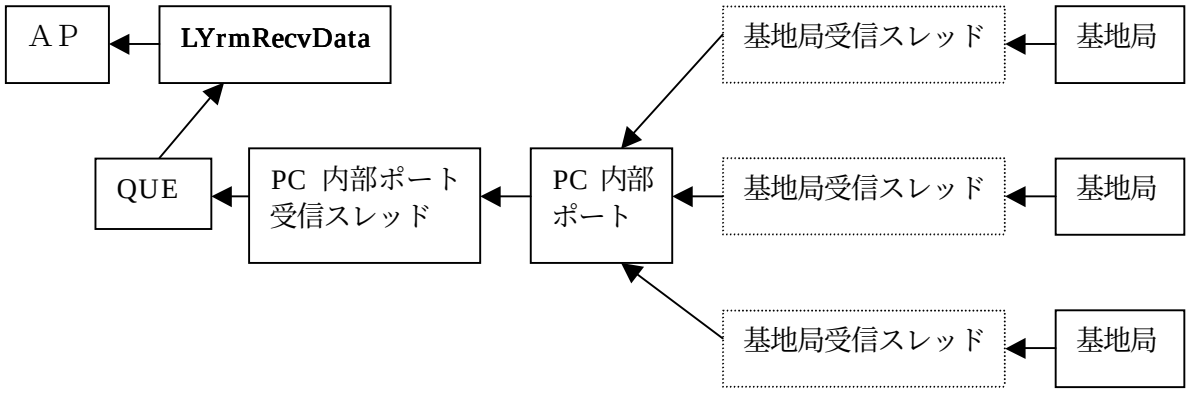
引数

DWORD	*handle	受信した基地局のハンドル番号を返す。 QueMode=1 の場合は、受信基地局の特定の為指定を行う。
DWORD	*Nbyte	受信データ BYTE 数 Nbyte=0 の場合は、受信データが無しです。
char	*data	受信データ（AP 側にて 8192 の領域をあらかじめ確保する）
DWORD	*htid	受信元 HTID 番号 千と百の位がｸﾞﾙｰﾌﾟ NO、十と一の位が IDNO を表す
BOOL	関数値	TRUE=エラー無し、FALSE=エラー有り FALSE の場合は **LyrmStatusCheck, LyrmCmdCheck** にて エラー原因を調査してください。

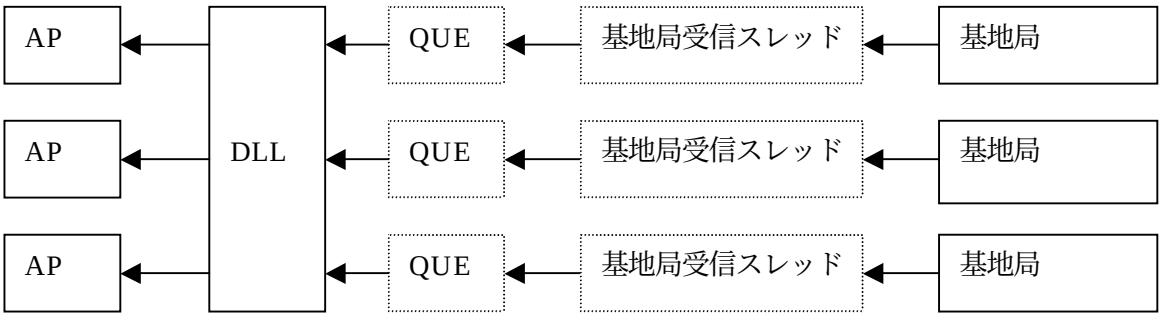
注記

● 受信キューの確保方法による動作の違いについて

LyrmCommonInfo.QueMode = 0 の場合
受信スレッド間の割込みによるデータの破壊と、ＦＩＦＯの確定の為に下図の様な処理を行う。

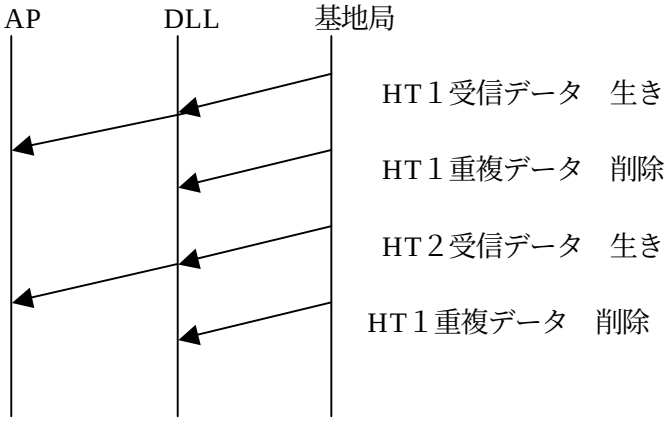


LyrmCommonInfo.QueMode = 1 の場合



● データの重複判定について

CommonInfo のメンバーDoubleDelete =1 の場合は下図の様な処理になる。



DoubleDelete に１をセットすると同一子機からの同一データを複数受信した場合は、先に到着したデータを生かし、後から受信したデータを削除する。

注：重複データチェックのバッファは受信先 ID 番号（１～９９）のみでの識別をおこなっており別グループでも ID 番号が同じなら同一バッファへ格納されます。

使用例

```
BOOL        Ret;
char        Rbuf[8192];
int         handle, Nbyte, htid;

// QueMode = 1 の場合は受信先基地局のﾊﾝﾄﾞﾙ番号を指定する
// QueMode = 0 の場合は指定は行いません。

Ret = LYrmRecvData(&handl, &Nbyte, Rbuf, &htid);
```

ＬＹｒｍＲｅｃｖＤａｔａＬｅｎ

宣言形式

int　LYrmRecvDataLen(DWORD *handle, DWORD *Nbyte, char *data, DWORD *htid)

Declare Function LYrmRecvDataLen Lib "LYrmDll.dll" (ByRef handle As Long,
ByRef Nbyte As Long, Byval Data As String, ByRef htid As Long) As Long

目的

受信キュー（指定基地局）にデータが存在すればデータを AP 側へ渡します。
データが存在しない場合も、制御を戻しますからＡＰ側が待ち状態にはなりません。

引数

DWORD	*handle	受信した基地局のハンドル番号を返す。 QueMode=1 の場合は、受信基地局の特定の為指定を行う。
DWORD	*Nbyte	受信データ BYTE 数 Nbyte=0 の場合は、受信データが無しです。
char	*data	受信データ（AP 側にて 8192 の領域をあらかじめ確保する）
DWORD	*htid	受信元 HTID 番号 千と百の位がｸﾞﾙｰﾌﾟ NO、十と一の位が IDNO を表す
int	関数値	1 以上なら受信データ有り（受信ﾃﾞｰﾀ BYTEs 数） 0 の場合は受信データ無し, -1 の場合は受信エラーなので **LyrmStatusCheck, LyrmCmdCheck** にてエラー原因を 調査してください。

注記

本関数の使用方法は LyrmRecvData とほぼ同じですが関数のリターン値が受信データがあればその BYTE 数となります。

使用例

```
int             nByte;
char            Rbuf[8192];
int             handle, Nbyte, htid;

// QueMode = 1  の場合は受信先基地局のﾊﾝﾄﾞﾙ番号を指定する
// QueMode = 0  の場合は指定は行いません。

nByte = LYrmRecvDataLen(&handl, &Nbyte, Rbuf, &htid);
```

ＬＹｒｍＧｅｔＲＩＤ

宣言形式

int　LYrmGetRID (DWORD　handle)

Declare Function LYrmGetRID Lib "LYrmDll.dll" (ByVal handle As Long) As Long

目的

「RID」コマンドを本ＤＬＬが、基地局監視の為に使用しておりＡＰ側からの
発行が出来ない為に、基地局の ｸﾞﾙｰﾌﾟ NO と ID.NO を返す関数を用意します。

引数

DWORD	handle	ｸﾞﾙｰﾌﾟ NO と ID.NO を取得したい基地局のハンドル番号。
DWORD	関数値	ｸﾞﾙｰﾌﾟ NO と ID.NO（－１は取得不可） 千と百の位がｸﾞﾙｰﾌﾟ NO、十と一の位が IDNO を表す。 -1 の場合は **LyrmStatusCheck, LyrmCmdCheck** にて エラー原因を調査してください。

使用例

```
int             htid;
int             hndle;

hndle = 1;
// ﾊﾝﾄﾞﾙ番号 1 の基地局の ID 情報取得
htid = LYrmGetRID(hndle);
```

ＬＹｒｍＳｅｎｄＤａｔａ

宣言形式

BOOL　LYrmSendData(DWORD mode, DWORD　handle, char *data, DWORD　htid)

Declare Function LYrmSendData Lib "LYrmDll.dll" (ByVal mode As Long,
ByVal handle As Long,ByVal Data As String, ByVal htid As Long) As Long

目的

テキストデータを指定の HT、又は基地局へ送信する。
送信は、非同期で行われるのでＡＰ側が待機状態にはなりません。

引数

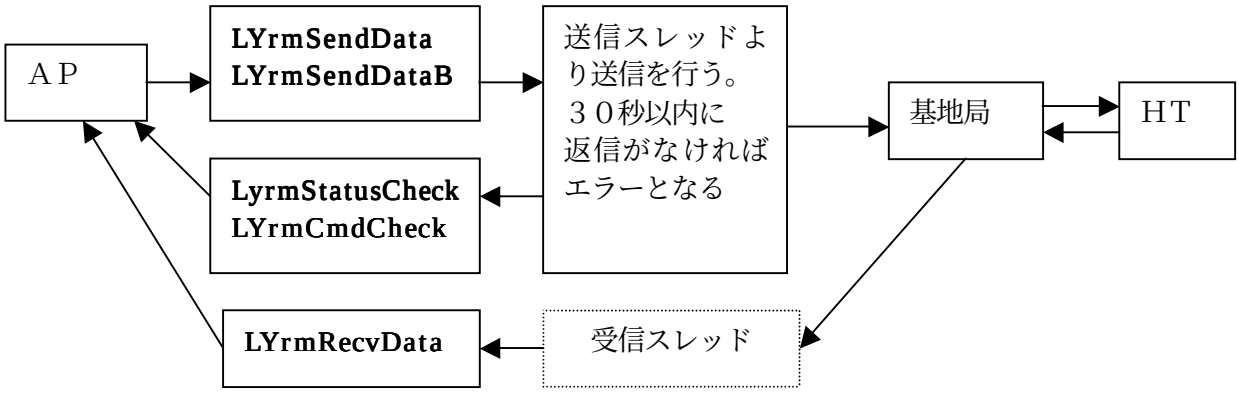
DWORD	mode	0=通常送信,　1=ローミング送信（注１）
DWORD	handle	送信先ハンドル番号を指定する
char	*data	送信データ（文字列の終端は null(0)をｾｯﾄしてください） 終端コード（CR）は本 DLL 内部にて付加するので AP 側では付加しない。
DWORD	htid	送信先 HTID 番号 送信先 HTID が０の場合は指定基地局内の全ての HT へ 送信を行う。
BOOL	関数値	TRUE=成功、FALSE=失敗, FALSE の場合は **LyrmStatusCheck, LyrmCmdCheck** にて エラー原因を調査してください。

注記

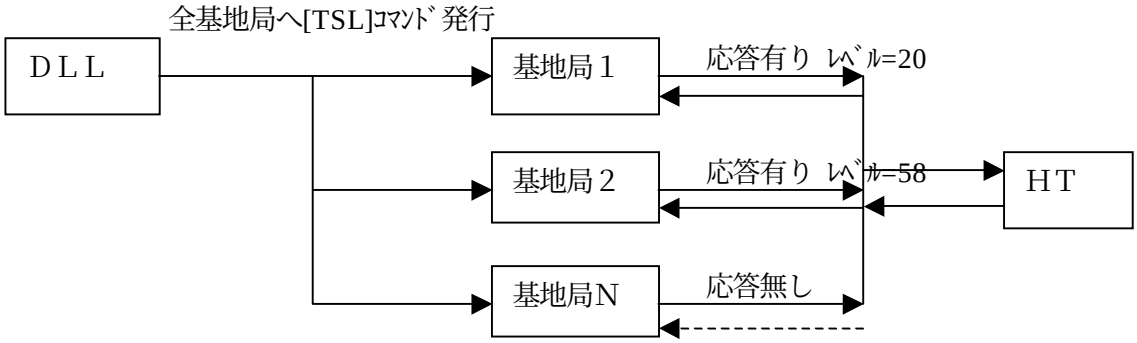
YRM-311 の場合は通常送信と、回線保持モード送信の判定は本 DLL 内部にて行いますから
ＡＰ側は留意する必要はありません。

無線モデムが YSM-2400 でない場合はﾛｰﾐﾝｸﾞ指示は無効となります。

● 送信処理の概要



●ローミング送信処理の概要



全基地局へ [TSL] ｺﾏﾝﾄﾞを発行して、応答のあった基地局で受信レベルの高い基地局を送信相手先とする。
上記の例だと基地局２が送信相手先となる。

使用例

```
BOOL        Ret;
int         handl;, htid、mode;
char        Sbuf[256];

mode = 0;       // ﾛｰﾐﾝｸﾞ指示無し
handl = 1;
htid = 102;
strcpy(Sbuf, “SEND DATA 0123456789ABCDEFG”);
// ﾊﾝﾄﾞﾙ番号 1 の基地局から HTID=0102 へﾃｷｽﾄﾃﾞｰﾀを送信
Ret = LYrmSendData(mode, handl, Sbuf, htid);
```

注１：mode=1(ﾛｰﾐﾝｸﾞ送信)時, handle(送信基地局ハンドル番号)が設定されていれば、
handle 指定の基地局で通常送信を行ってＮＧとなった場合にローミング送信を行う。
handle 番号が無効(=0)の場合は最初からローミング送信を行なう。

ＬＹｒｍＳｅｎｄＤａｔａＮｏＬｎｋ

宣言形式

BOOL　LYrmSendDataNolnk(DWORD mode, DWORD　handle, char *data, DWORD htid)

```
Declare Function LYrmSendDataNoLnk Lib "LYrmDll.dll" (ByVal mode As Long,
        ByVal handle As Long, ByVal Data As String, ByVal htid As Long) As Long
```

目的

テキストデータを指定の HT、又は基地局へ「ＬＮＫ」レスポンス無しで送信する。
送信は、非同期で行われるのでＡＰ側が待機状態にはなりません。

引数

DWORD	mode	0=通常送信,　1=ローミング送信（注１）
DWORD	handle	送信先ハンドル番号を指定する
char	*data	送信データ（文字列の終端は null(0)をｾｯﾄしてください） 終端コード（CR）は本 DLL 内部にて付加するので AP 側では付加しない。
DWORD	htid	送信先 HTID 番号 送信先 HTID が０の場合は指定基地局内の全ての HT へ 送信を行う。
BOOL	関数値	TRUE=成功、FALSE=失敗, FALSE の場合は **LyrmStatusCheck, LyrmCmdCheck** にて エラー原因を調査してください。

注記

YRM-211 の場合は通常送信として動作します。
無線モデムが YSM-2400 でない場合はﾛｰﾐﾝｸﾞ指示は無効となります。

使用例

```
BOOL            Ret;
int             handl;, htid, mode;
char            Sbuf[256];

mode = 1;       // ﾛｰﾐﾝｸﾞ指示有り
handl = 1;
htid = 102;
strcpy(Sbuf, “SEND DATA 0123456789ABCDEFG”);
// ﾊﾝﾄﾞﾙ番号 1 の基地局から HTID=0102 へﾃｷｽﾄﾃﾞｰﾀを送信
Ret = LYrmSendDataNoLnk(mode, handl, Sbuf, htid);
```

注１：mode=1(ﾛｰﾐﾝｸﾞ送信)時, handle(送信基地局ハンドル番号)が設定されていれば、
handle 指定の基地局で通常送信を行ってＮＧとなった場合にローミング送信を行う。
handle 番号が無効(=0)の場合は最初からローミング送信を行なう。

ＬＹｒｍＳｅｎｄＤａｔａＢ

宣言形式

BOOL　LYrmSendDataB(DWORD mode, DWORD　handle, DWORD Nbyte, char *data, DWORD htid)

Declare Function LYrmSendDataB Lib "LYrmDll.dll" (ByVal mode As Long, ByVal handle As Long,ByVal Nbyte As Long, ByRef Data As Byte, ByVal htid As Long) As Long

目的

バイナリーデータを指定の HT、又は基地局へ送信する。
送信は、非同期で行われるのでＡＰ側が待機状態にはなりません。

引数

DWORD	mode	0=通常送信,　1=ローミング送信（注１）
DWORD	handle	送信先ハンドル番号を指定する
DWORD	Nbyte	送信データ BYTE 数
char	*data	送信データ
DWORD	htid	送信先 HTID 番号 送信先 HTID が０の場合は指定基地局内の全ての HT へ送信を行う。
BOOL	関数値	TRUE=成功、FALSE=失敗, FALSE の場合は **LyrmStatusCheck, LyrmCmdCheck** にてエラー原因を調査してください。

注記

YRM-311 の場合は通常送信と、回線保持モード送信の判定は本 DLL 内部にて行いますからＡＰ側は留意する必要はありません。
無線モデムが YSM-2400 でない場合はﾛｰﾐﾝｸﾞ指示は無効となります。

使用例

```
BOOL        Ret;
int         handl;, htid, Nbyte, mode;
char        Sbuf[256];

mode = 0;       // ﾛｰﾐﾝｸﾞ指示無し
handl = 1;
htid = 102;
Nbyte = 128;
memset(Sbuf, 0x0f, Nbyte);
// ﾊﾝﾄﾞﾙ番号 1 の基地局から HTID=0102 へﾊﾞｲﾅﾘｰﾃﾞｰﾀを送信
Ret = LYrmSendDataB(mode, handl, Nbyte, Sbuf, htid);
```

注１：mode=1(ﾛｰﾐﾝｸﾞ送信)時, handle(送信基地局ハンドル番号)が設定されていれば、
handle 指定の基地局で通常送信を行ってＮＧとなった場合にローミング送信を行う。
handle 番号が無効(=0)の場合は最初からローミング送信を行なう。

ＬＹｒｍＳｅｎｄＤａｔａＢＮｏＬｎｋ

宣言形式

BOOL　LYrmSendDataBNoLnk(DWORD mode, DWORD　handle, DWORD Nbyte, char *data, DWORD htid)

Declare Function LYrmSendDataBNoLnk Lib "LYrmDll.dll" (ByVal mode As Long, ByVal handle As Long,ByVal Nbyte As Long, ByRef Data As Byte, ByVal htid As Long) As Long

目的

バイナリーデータを指定の HT、又は基地局へ「ＬＮＫ」レスポンス無しで送信する。
送信は、非同期で行われるのでＡＰ側が待機状態にはなりません。

引数

DWORD	mode	0=通常送信,　1=ローミング送信（注１）
DWORD	handle	送信先ハンドル番号を指定する
DWORD	Nbyte	送信データ BYTE 数
char	*data	送信データ
DWORD	htid	送信先 HTID 番号 送信先 HTID が０の場合は指定基地局内の全ての HT へ送信を行う。
BOOL	関数値	TRUE=成功、FALSE=失敗, FALSE の場合は **LyrmStatusCheck, LyrmCmdCheck** にてエラー原因を調査してください。

注記

YRM-211 の場合は通常送信として動作します。
無線モデムが YSM-2400 でない場合はﾛｰﾐﾝｸﾞ指示は無効となります。

使用例

```
BOOL        Ret;
int         handl;, htid, Nbyte、mode;
char        Sbuf[256];

mode = 1;       // ﾛｰﾐﾝｸﾞ指示有り
handl = 1;
htid = 102;
Nbyte = 128;
memset(Sbuf, 0x0f, Nbyte);
// ﾊﾝﾄﾞﾙ番号 1 の基地局から HTID=0102 へﾊﾞｲﾅﾘｰﾃﾞｰﾀを送信
Ret = LYrmSendDataBNoLnk(mode, handl, Nbyte, Sbuf, htid);
```

注１：mode=1(ﾛｰﾐﾝｸﾞ送信)時, handle(送信基地局ハンドル番号)が設定されていれば、
handle 指定の基地局で通常送信を行ってＮＧとなった場合にローミング送信を行う。
handle 番号が無効(=0)の場合は最初からローミング送信を行なう。

ＬＹｒｍＳｅｎｄＳＴＯ

宣言形式

BOOL LYrmSendSTO(DWORD　handle, DWORD mode, DWORD *status)

Declare Function LYrmSendSTO Lib "LYrmDll.dll" (ByVal handle As Long,
ByVal mode As Long, ByRef Status As Any) As Long

目的

「STO」コマンドを指定の基地局へ送信する。

引数

DWORD	handle	送信先ハンドル（基地局）を指定する
DWORD	mode	0：現在の設定内容の取得, 1：status の内容をセットする
DWORD	*status	mode=1 の場合の設定内容 Status[8] の 8 個の要素を持っています。
BOOL	関数値	TRUE=成功、FALSE=失敗 FALSE の場合は **LyrmStatusCheck, LyrmCmdCheck** にて エラー原因を調査してください。

注記

mode=0(現状の取得)の場合は、**LyrmRecvData**　にて内容を取得します。
なおデータ受信時のヘッダ＝有、データ受信時の改行コード＝有、改行コード指定＝ＣＲのみ
上記３種の設定は変更不可となります。

使用例

```
BOOL        Ret;
int         handl;, mode;
int         sto[8];

handl = 1;
// mode = 1 にて送信する場合は、設定値を  sto[0] ～ sto[7]へ設定
// mode = 0  現在値を取得する
mode = 0
// ﾊﾝﾄﾞﾙ番号１の基地局のﾊﾟﾗﾒｰﾀを取得する
Ret = LYrmSendSTO(handl, mode, sto);
```

ＬＹｒｍＳｅｎｄＣｏｎ

宣言形式

BOOL LYrmSendCON(DWORD handel , DWORD　htid)

Declare Function LYrmSendCON Lib "LYrmDll.dll" (ByVal handle As Long,
ByVal htid As Long) As Long

目的

回線保持「CON」コマンドを指定の基地局へ送信する。（YRM-311 のみ使用可能）

引数

DWORD	handle	送信先ハンドル（基地局）を指定する
DWORD	htid	回線保持要求を発行する相手先のｸﾞﾙｰﾌﾟ NO と IDNO 千と百の位がｸﾞﾙｰﾌﾟ NO、十と一の位が IDNO を表す
BOOL	関数値	TRUE=成功、FALSE=失敗

使用例

```
BOOL        Ret;
int         hndl, htid;

hndl = 1;       htid = 102;
// ﾊﾝﾄﾞﾙ番号１の基地局と HTID=0102 のﾊﾝﾃﾞｨｰﾀｰﾐﾅﾙ(若しくは基地局)を回線保持状態とする
Ret = LYrmSendCON(hndl, htid);
```

ＬＹｒｍＳｅｎｄＤＣＮ

宣言形式

BOOL LYrmSendDCN(DWORD　handel)

Declare Function LYrmSendDCN Lib "LYrmDll.dll" (ByVal handle As Long) As Long

目的

回線保持解除「DCN」コマンドを指定の基地局へ送信する。（YRM-311 のみ使用可能）

引数

DWORD	handle	送信先ハンドル（基地局）を指定する
BOOL	関数値	TRUE=成功、FALSE=失敗

使用例

```
BOOL            Ret;
int             hndl;

hndl = 1;
// ﾊﾝﾄﾞﾙ番号１の基地局とﾊﾝﾃﾞｨｰﾀｰﾐﾅﾙ(若しくは基地局)の回線保持状態を解除とする
Ret = LYrmSendDCN(hndl);
```

ＬＹｒｍＳｅｎｄＦｒｅｅ

宣言形式

BOOL LYrmSendFREE(DWORD t handel, DOWRD Nbyte, char　send_data)

Declare Function LYrmSendFREE Lib "LYrmDll.dll" (ByVal handle As Long,
ByVal Nbyte As Long, Byval Data As String) As Long

目的

send_data にて指定した内容を指定の基地局へ送信する。

引数

DWORD	handle	送信先ハンドル（基地局）を指定する
DWORD	Nbyte	送信データバイト数
char	*send_data	送信データ（コマンド）を全てセットする。 「CR、LF」は不要
BOOL	関数値	TRUE=成功、FALSE=失敗

注記

基地局からの返信データは、**LyrmRecvData**　にて内容を取得する。

- 以下のコマンドは送信不可。
  「RID」、「STO」、「POL」、「PCL」、「TSL」、「TST」、「TSC」の７種

使用例

```
BOOL        Ret;
int         hndl, Nbyte;
char        Sbuf[16];

hndl = 1;
strcpy(Sbuf, “OBL”);
Nbyte = strlen(Sbuf);
// ﾊﾝﾄﾞﾙ番号１の基地局へOBL（ﾊﾞｯﾃﾘﾚﾍﾞﾙ出力要求）ｺﾏﾝﾄﾞを送信
Ret = LYrmSendFree(hndl, Nbyte, Sbuf);
```

ＬＹｒｍＦｉｌｅＴｒａｎｓ

宣言形式

BOOL LYrmFileTrans (DWORD mode, DOWRD N, DWORD *htid, DWORD *handle, char *f_name)

Declare Function LYrmFileTrans Lib "LYrmDll.dll" (ByVal mode As Long, ByVal N As Long, ByRef htid as Long, ByRef handle As Long, Byval f_name As String) As Long

目的

指定ファイルの転送（ＵＰ，ＤＯＷＮ　ＬＯＡＤ）を行う。

引数

DWORD	mode	0=DOWN LOAD(ﾓﾃﾞﾑ -> HT), 1=UP LOAD(HT -> ﾓﾃﾞﾑ)
DWORD	N	送受信ﾃﾞｰﾀ数
DWORD	*htid	送受信相手先 HTID 番号 (N 個有り)
DWORD	*handle	送受信対象ハンドル（基地局）を指定する(N 個有り) Handle[n]=0 の場合はローミング指示となります
char	*f_name	DOWN　LOAD 時は送信ﾌｧｲﾙ名（ﾌｫﾙﾀﾞｰまで指定する）、 ＵＰ　ＬＯＡＤ時は受信ﾌｫﾙﾀﾞｰ名 NULL (0) で区切って複数設定を行う。
BOOL	関数値	TRUE=成功、FALSE=失敗 複数ＨＴを対象とした場合は１ＨＴでもエラーがあると FALSE となります。 各ＨＴ毎の状態は「LYrmFileTrans_Stop」を呼出て 確認してください。

注記

・無線モデムが YSM-2400 でない場合は本関数呼出は無効となります。
・一度に実行できる転送件数は５０件です。
・DOWN LOAD 時のファイル名の拡張子はプログラムファイル=".MOT"、
データファイル=".DAT" / ".MST"以外の指定は無効となります。

ファイル転送時の注意事項

・ファイル転送開始前にＨＴの
「SYSTEM MODE MENU　－＞　SET －＞ Loader －＞ SET Loader Port」で
2：Rf　Port を選択してください。
・DOWN　LOAD 時はＨＴの「SYSTEM MODE MENU －＞　DOWN LOAD」待機
状態にＨＴをセットしてから、本ファンクションを呼出してください。
・UP　LOAD 時はＨＴの
「SYSTEM MODE MENU　－＞　SET －＞ Loader －＞ Set Rf Loader」で
Rt　UpLoad　Start の設定を 2：Private に設定してください。
実際のファイル転送時は、本ファクションを呼出し基地局を待機状態にしてから
「SYSTEM MODE MENU －＞　UP LOAD」を指示してください。
・ユーザーアプリケーションでは、rsloader および progloader 関数を使用します。

転送ファイルの構成

・プログラムファイルは、Hitachi Embedded Workshop で作成された “MOT”ファイルを使用すればそのまま転送が行なえます。

・転送ファイルの構成は 「Vertex Standard」リファレンスマニュアルの６章（ファイル転送仕様）を参照してください。
なお、本ＤＬＬにて伝送制御キャラクターの付加は行います。

使用例

```
BOOL        Ret;
Int         l,mode, cnt;
int         hndl[2], htid[2];;
char        f_buf[128];

mode = 0;       // DOWN LOAD(ﾓﾃﾞﾑ -> HT),
cnt = 2;        // ファイル転送数２件
hndl[0] = 1;    // ﾊﾝﾄﾞﾙ番号１の基地局よりファイル送信
htid[0] = 501;  // HTID = 501
hndl[1] = 0;    // 基地局指定無し（ローミング転送）
htid[1] = 610;  // HTID = 610
// DOWN LOAD ファイル名のセット
strcpy(f_buf, “C:¥¥TEMP¥¥YSB_2400.MOT”);
l = strlen(f_buf);
strcpy(&f_buf[l+1], “C:¥¥TEMP¥¥TEST.DAT”);
// ファイル転送
Ret = LYrmFileTrans (mode, cnt, htid, hndl, f_buf):
```

ＬＹｒｍＦｉｌｅＴｒａｎｓ＿Ｓｔｏｐ

宣言形式

BOOL LYrmFileTrans_Stop (DOWRD N, DWORD *htid)

Declare Function LYrmFileTrans_Stop Lib "LYrmDll.dll" (ByVal N As Long, ByRef htid as Long) As Long

目的

実行中のファイル転送処理を強制停止させます。

引数

DWORD	N	強制停止対象ＨＴ数 “-1”を指定した場合は現在処理中の全ファイル転送を停止します。
DWORD	*htid	強制停止対象 HTID 番号 (N 個有り)
BOOL	関数値	TRUE=成功、FALSE=失敗

注記

本ファンクション呼出しを行なった直後に、ファイル転送処理が即時終了はしないので LYrmFileTrans_Status を呼出して状態を確認して次の処理を行なってください。

使用例

```
BOOL            Ret;
Int             htid[2];

Htid[0] = 1101;     htid[1] = 1201;
// HTID=1101 と HTID=1201 のファイル転送強制停止
Ret = LYrmFileTrans_Stop( 2, htid):
// 全ファイル転送処理を停止する。
Ret = LYrmFileTrans_Stop(-1, htid);
```

ＬＹｒｍＦｉｌｅＴｒａｎｓ＿Ｓｔａｔｕｓ

宣言形式

BOOL LYrmFileTrans_Status (DOWRD N, DWORD htid, DWORD status)

Declare Function LYrmFileTrans_StatusLib "LYrmDll.dll" (ByVal N As Long,
ByRef htid as Long, ByRef status As Long) As Long

目的

指定ファイルの転送（ＵＰ，ＤＯＷＮ　ＬＯＡＤ）を行う。

引数

DWORD	N	ステータス取得数
DWORD	*htid	ステータス取得相手先 HTID 番号 (N 個有り)
DWORD	*status	取得したステータス値(N 個有り) 1=処理待ち, 2=処理中, 3=処理正常終了, -1=ﾕｰｻﾞｰ強制停止 21=ﾛｰﾐﾝｸﾞ検出不可(ﾛｰﾐﾝｸﾞ指定時のみ), 31=ﾌｧｲﾙ転送処理ｴﾗｰ
BOOL	関数値	TRUE=成功、FALSE=失敗 (status=21, 31 の場合)

使用例

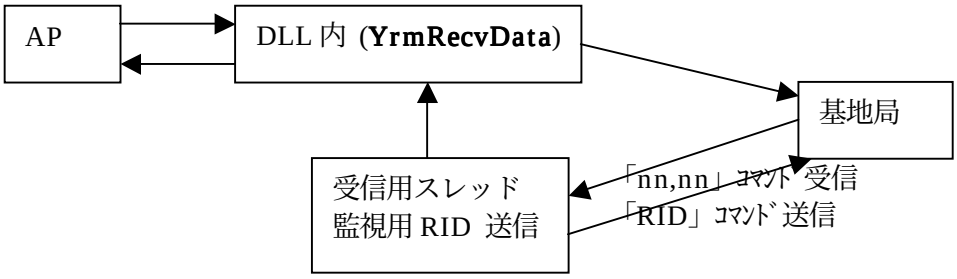
```
BOOL            Ret;

// ファイル転送強制停止
Ret = LYrmFileTrans_Stop( ):
```

４．基地局監視機能

受信スレッド内監視を行う為にＡＰが待機状態になる事はない。

監視方法は、受信スレッド内にて指定時間(CommonInfo. CheckTime) 受信データが無い場合は「RID」コマンドを送信して待機する。
受信無しの状態で、３０秒間経過した場合は、エラーとする。



ステータス取得ファンクション(**LYrmStatusCheck**)を呼び出してエラー状況を確認する。

- LAN アダプタのエラー時は、自動再接続フラグ(CommonInfo. AutoConnect) がセットされていれば受信スレッドを終了して基地局再接続を行う。
  自動再接続時は、何らかのデータを受信すると[STO, RID]コマンドの発行を中止して接続完了と判定します。

  注意：データ受信と監視機能はスレッドにて動作しているために、
  ステータス取得ファンクション(**LYrmStatusCheck**)の発行時期によって実際の状況と異なる場合がある。

５．通信状況とエラー表示（ログ出力）

通信状況とエラー情報をコンソール（ＤＬＬ内部にて自動作成）とエラーログファイルへ出力する。

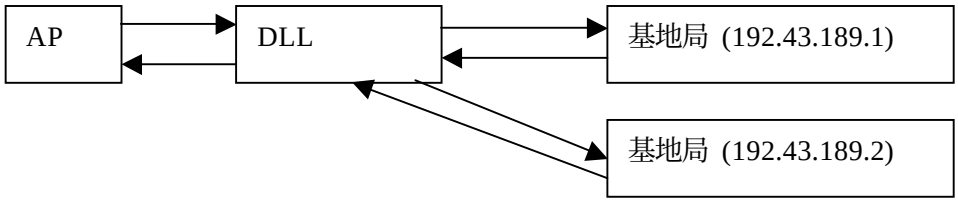
報告内容

１：受信データ
２：受信コマンド、返信コマンド
３：送信データ
４：監視スレッドよりの報告情報
５：基地局接続回数
６：エラー情報

● 出力情報には、各々IP アドレスと PORTNO. を付加する。

ログファイルへ出力方法

CommonInfo.LogFilename =”C:¥TMP¥INFFOLOG.LOG”
CommonInfo.LogMode = 1 の場合 は全てのメッセージを”C:¥TMP¥INFFOLOG.LOG”へ出力



CommonInfo.LogMode = 2 の場合 はメッセージを”C:¥TMP¥INFFOLOG192_43_189_1.LOG”と
”C:¥TMP¥INFFOLOG192_43_189_2.LOG”へ出力

出力項目の制御について

ＤＬＬの格納フォルダーの有る（存在しなければ自動作成）「LYrmDll.ini」ファイルの DEBUGLEVEL 項目の数値によって出力内容を制御できます。
DEBUGLEVEL=0　の場合は、デバッグ情報は出力しない。
DEBUGLEVEL=1　の場合は、エラー情報のみ出力します。
DEBUGLEVEL=2　の場合は、エラー情報と基地局の OPEN, CLOSE に関する情報を出力します。
DEBUGLEVEL=3　の場合は、上記情報に加えて送受信データも出力します。

注意：自動作成時の初期値は２となっています。

ログファイル出力例

● 項目　：　内部ポート、ＤＬＬ共通の情報
■ 項目　：　基地局（受信関連）の情報
▲ 項目　：　基地局（送信関連）の情報
★ 項目　：　エラー情報

********** LOG FILE OPEN [E:¥TMP¥YRMLOG.LOG] **********
12:30:29.120 ● デバッグコンソール作成完了
12:30:29.320 ● LEADNT40SRVBACK 内部ポート オープン完了
12:30:39.515 ■ 128.1.29.220 基地局オープン開始
12:30:39.525 ■ 128.1.29.220 への connect が可能かどうか select() で調査中
12:30:40.546 ■ 128.1.29.220 に connect() 中(1)
12:30:40.556 ■ 128.1.29.220 に connect() 成功
12:30:40.566 ▲ 128.1.29.220 へ 15BYTE(CR 含)送信[STO ,,0,1,0,,,0dH]
12:30:40.696 ■ 128.1.29.220 からデータを 4BYTE 受信[END0dH]
12:30:40.706 ■ 128.1.29.220 から 3BYTE 受信キューへセット[END]
12:30:40.837 ▲ 128.1.29.220 へ 4BYTE(CR 含)送信[RID0dH]
12:30:40.887 ■ 128.1.29.220 からデータを 6BYTE 受信[01,000dH]
12:30:40.897 ■ 128.1.29.220 から 5BYTE 受信キューへセット[01,00]
12:30:40.987 ■ 128.1.29.220 基地局オープン完了
12:30:46.154 ■ 128.1.29.230 基地局オープン開始
14:51:45.298 ★ エラー詳細番号 (errdetail=87)
14:51:45.368 ■ 128.1.29.220 shutdown() および closesocket() 中 at SockShutdown()
14:51:45.528 ■ 128.1.29.220 shutdown() および closesocket() 完了
14:51:45.709 ★ エラー詳細番号 (errdetail=87)
14:51:45.779 ■ 128.1.29.230 shutdown() および closesocket() 中 at SockShutdown()
14:51:45.929 ■ 128.1.29.230 shutdown() および closesocket() 完了
14:51:46.119 ● LEADNT40SRVBACK 内部ポート shutdown() および closesocket() 中 at SockShutdown()
14:51:46.259 ● LEADNT40SRVBACK 内部ポート shutdown() および closesocket() 完了
********** LOG FILE CLOSE [E:¥TMP¥YRMLOG.LOG] **********

注意：送信と受信のデータの最大ダンプ数は多量のデータをコンソールとログファイルへ出力すると非同期にて動作しているスレッド処理が遅延するために６４ＢＹＴＥに制限しています。

６．注意事項

３２ｂｉｔのみの対応

本ＤＬＬは３２ｂｉｔアプリケーションのみの対応となっており１６ｂｉｔアプリケーションでは使用できません。

非同期送信に関する注意

同一 HT（無線モデム）へ連続して送信する場合は、LyrmCmdCheck にて前回の送信ステータスが”-1”でなくなった事を確認してデータ送信関数を呼出してください。
送信相手先以外の HT（無線モデム）のステータスは確認する必要はありません。

HT（ハンディーターミナル）との接続に関す注意

ＨＴを、同一基地局に複数台接続する場合に、
通信処理がぶつかった時に一方のＨＴの処理が遅延させられますので、以下の注意点に留意してください。

① ＨＴ側の再発呼回数と再送回数を共に８以上に設定。

② 固定チャンネル方式とする。（チャネル指定のある機種のみ)
ＭＣＡ方式より約１５０ｍｓｅｃ前後通信処理速度が速くなります。
ただし、同一エリアに複数の基地局を設置してHTがどの基地局とも通信できるようにシステムを構成する場合には、ＭＣＡ方式にすれば空いている基地局探しが容易かつスピーディです。

③ ＯＳはＷｉｎｄｏｗｓＮＴを使用。
マルチスレッド処理実行に際して動作が Windows95(98) より安定する様です。

以上の注意点は同一基地局に複数台のＨＴを接続する場合のみで、ＨＴが１台のみなら考慮する必要はありません。

７．サンプルプログラム

DISK2 ディレクトリィを参照してください。