特定小電力ワイヤレスモデム

ＲＳ－２３２Ｃ準拠　429MHz 帯

M1

セットアッププログラム
ユーザーズマニュアル

&

エアーモニタＤＬＬの使い方

CIRCUIT DESIGN, INC.

目 次

使用上の注意

◆ このセットアッププログラムはM1ワイヤレスモデム専用のプログラムです。他の機器に対しては使用する事はできません。M1本体のユーザーズマニュアルを十分理解した上でご使用下さい。
弊社では、誤った使い方によるいかなる損害に対しも一切の責任を負いません。

◆ 本プログラムならびに本書の内容のコピー、転載は無断で行わないで下さい。
著作権法により禁止されています。

技術的なお問い合わせ

弊社営業部、技術部では製品に関連する技術的なお問い合わせを随時受け付けております。
開発環境、状況、問題となっている事柄などを具体的にとりまとめ、先ず営業部までご連絡下さい。

ご連絡、問い合わせ先

各種問い合わせは、弊社営業部まで下記のいずれかの方法でご連絡下さい。
また、弊社webには技術情報ならびに新しい情報、Q&Aなどが掲載されていますのでご覧下さい。

ポイント:Eメールによるお問い合わせが、簡潔で間違いが無く、内容が伝えやすいのでとても便利です。

■ インターネットメール
Eメールアドレス: nbd@circuitdesign.jp
宛先: 営業部

■ webアドレス
web URL: http://www.circuitdesign.jp/

■ 電話
電話番号: 0263-82-1024
担当部署: 営業部
受け付け時間: 9:00 ～ 17:30 (平日)

■ FAX.
FAX番号: 0263-82-1016
宛先: ㈱サーキットデザイン 営業部

■ 郵便
郵便番号: 399-8303
住所: 長野県南安曇郡穂高町穂高 7557-1
宛名: ㈱サーキットデザイン 営業部

第1章 M1 セットアッププログラムについて

M1セットアッププログラムはM1の各種パラメータを設定するツールです。
また、各種コマンドを使ったデモンストレーションとなっており、簡単な通信テストができるのでアプリケーションプログラマーの参考になります。
また、エアーモニタフォームはDLLとしてありますので、ユーザプログラムから利用できます。

☞注意:このプログラムはM1専用プログラムです。他の機器に対しては使用することができません。

1．1 動作環境

- CPU： Intel Pentium 166MHz 以上(Intel PentiumII 330MHz 以上推奨) ※1
- メモリ： 48MB 以上(96MB 以上推奨)
- HDD： 1MB 以上(設定を保存する場合は、別途1KB 必要)
- OS： MicroSoft 日本語版 Windows95/98/2000/XP ※2

※1 インテル製CPU以外での動作確認は行っておりません。
※2 WindowsNT・Me での動作確認は行っておりません
日本語版以外のOSでは動作しないことがあります

第2章 インストール方法

M1 のセットアップ（各種動作パラメータの設定）は「M1 セットアッププログラム」（M1SET.EXE）を用いて行う事ができます。

ヒント:設定したパラメータはM1内部の不揮発性メモリに保存されますので、電源を切っても失われることはありません。

2．1 インストール

◆ M1セットアッププログラムのインストール

1、セットアッププログラムをインストールする前に、コンピュータ上で起動している全てのアプリケーションを閉じて下さい。
2、CDをCDトレイに入れるとセットアッププログラムの自動インストール画面が立ち上がります。
3、画面の指示に従ってインストールを進めて下さい。
4、M1専用フォルダが「Program Files」フォルダに作制され、スタートアップメニューの「プログラムメニュー」に「M1 WirelessModem」として登録されます。

◆ インストーラが自動的に立ち上がらない場合

次の手順でインストールして下さい。

1、スタートメニューの「ファイル名を指定して実行(R)」をクリックして下さい。
2、ウィンドウ画面の「参照(B)」をクリックして下さい。ファイルウィンドウが開くので、マイコンピュータのCD－ROMデバイスを選択してインストールディスク上の「SETUP.EXE」を実行するとインストールが始まります。
3、画面の指示に従ってインストールを進めて下さい。
4、M1専用フォルダが「Program Files」フォルダに作制され、スタートアップメニューの「プログラムメニュー」に「M1 WirelessModem」として登録されます。

2．2 アンインストール

◆ M1セットアッププログラムのアンインストール

コントロールパネルの「アプリケーションの追加と削除」のリストの中にある本プログラムを指定し、指示に従い削除して下さい。
なお、エアーモニタ機能で保存したファイルエクステンションが‘. ADT’のファイルは削除されませんので、手動で削除して下さい。

第3章 使い方

◆ M1セットアッププログラムの実行

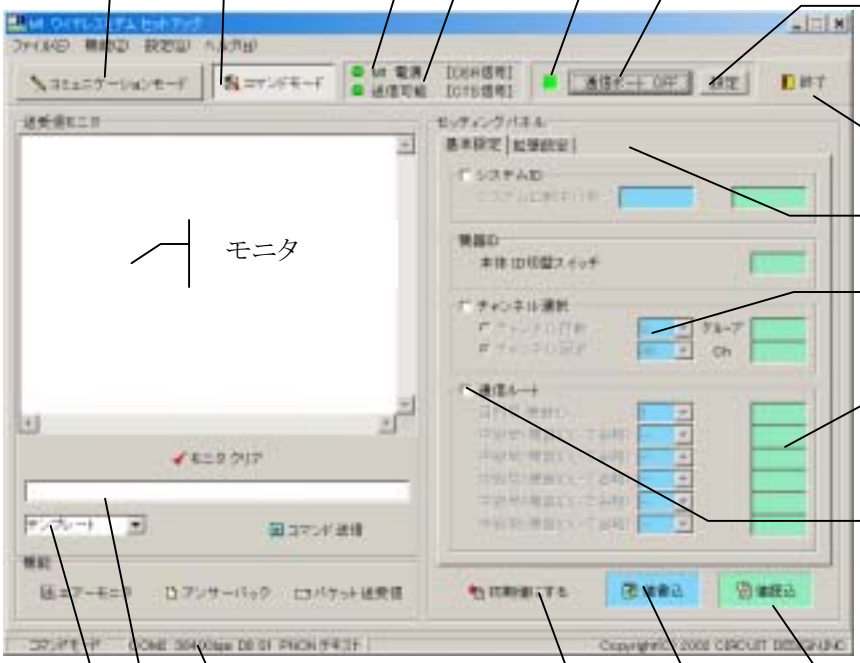
☞注意:実行する前にM1本体と付属RS232Cケーブルで接続し、電源を入れて下さい。

スタートアップメニューの「プログラムメニュー」の「M1 WirelessModem」内にある
「M1 セットアップ プログラム」を実行して下さい。

プログラムが起動したら、次の手順でセットアップを行います。通信先の機器も設定します。

1、通信設定ボタンで通信パラメータの設定をします。
2、通信ポートボタンでポートに接続して下さい。正常に接続されれば、ボタン左の緑インジケータが点灯します。
3、コマンドモードボタンでコマンドモードにします。
4、セッティングパネルで各種パラメータの設定を行って下さい。

◆ メイン画面（基本設定）



コミュニケーションモード

コマンドモード

M1電源
M1が接続されていると緑表示です

CTS信号の状態を表します

ポートインジケータ
ポートが接続されると緑表示です

通信ポートボタン
通信ポートの接続、切断をします

通信設定ボタン
通信パラメータを設定します

プログラム終了

セッティングパネル

モニタ

青：書込み値表示

緑：読込み値表示

チェックがある項目だけ書込みできます

値読込みボタン
設定値の読込みをします

ステータスバー

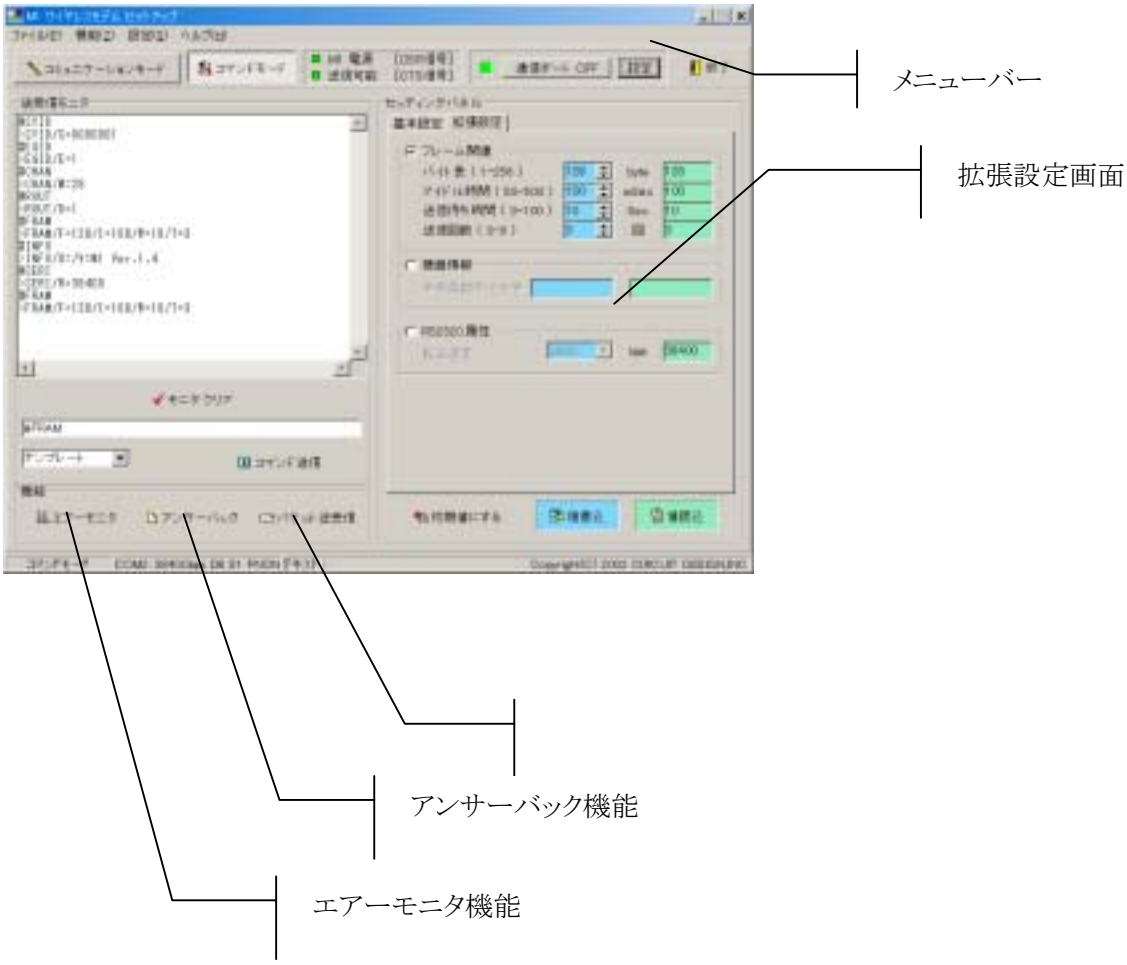
コマンドを書込みます

値書込みボタン
チェックがある項目の書込みをします

コマンドのテンプレートです

書込み欄を初期化します

◆ メイン画面（拡張設定）



◆ 通信するには

次のように設定されていないと通信できません。

❑1と2は通信元と通信先で同じに設定します。

1、システムID

2、チャンネル選択方式で自動の場合のグループ番号または、固定の場合のチャンネル番号

❑通信ルートの‘目的局’は双方の機器で、通信先の機器IDを指定します。

◆ 通信ポート関連ボタン、表示

1、通信設定ボタンを押すと、RS232Cポートの接続ポートと通信速度を設定することができます。

2、通信ポートボタンがOFFの時ボタンを押すと、COMポートに接続することができます。正常に接続できた時はポートインジケータが緑になります。通信ポートボタンがONの時ボタンを押すと、COMポートから切断されます。

‘M1電源’インジケータはM1が接続されていて電源が入っていれば緑表示になります。電源が入っていない場合は赤表示です。この処理はDSR信号ラインを見ています。

‘送信可能’インジケータは、DTE上にあるこのセットアッププログラムに対して、DCEであるM1が受信可能かどうかを知らせる信号です。M1が受信可能な場合は緑表示になります。この処理はCTS信号ラインを見ています。RTS信号ラインと共に使用し、ハードウェアフロー制御を行います。

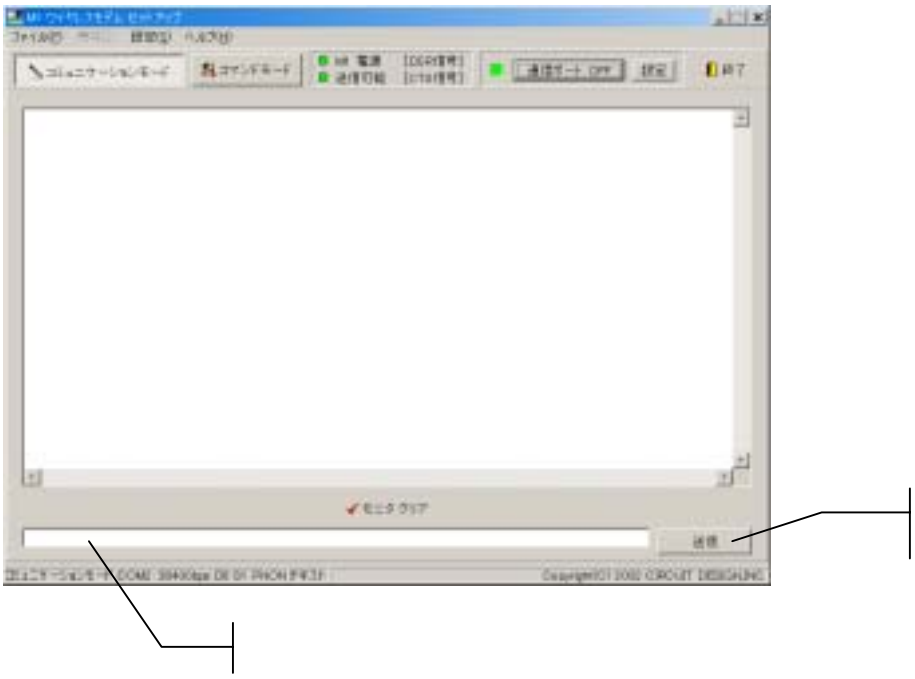
◆ セッティングパネル概要

セッティングパネルは基本設定ページと拡張設定ページに分かれています。
既に設定されている値の確認は‘値読込み’ボタンを押すことで、右の緑エリアに表示されます。
新たに設定するには各設定項目のチェックをし、左の青エリアに設定値を書込み‘値書込み’ボタンを押します。
‘初期値にする‘ボタンは、各項目の設定値書込み青エリアの値を初期値に戻します。書込みボタンを押さない限り本体には書き込みされません。

拡張設定でRS232属性の転送速度の変更を行った場合は、本体の電源を再投入した時点で変更されます。
‘送受信モニタ’には本プログラムとM1との間で交わされているコマンドやレスポンスが表示されます。

☞**重要**:セッティングパネル上の各設定項目は、項目パネルのチェックが入っているものだけ書き込むことができます。

◆ コミュニケーションモード画面



コミュニケーションモードで通信テストを行うことができます。このテストを行うには通信先のセットアッププログラムもコミュニケーションモードにして下さい。
送信テキストラインに電文を書込み送信ボタンを押すと送信されます。

◆ エアーモニタ機能

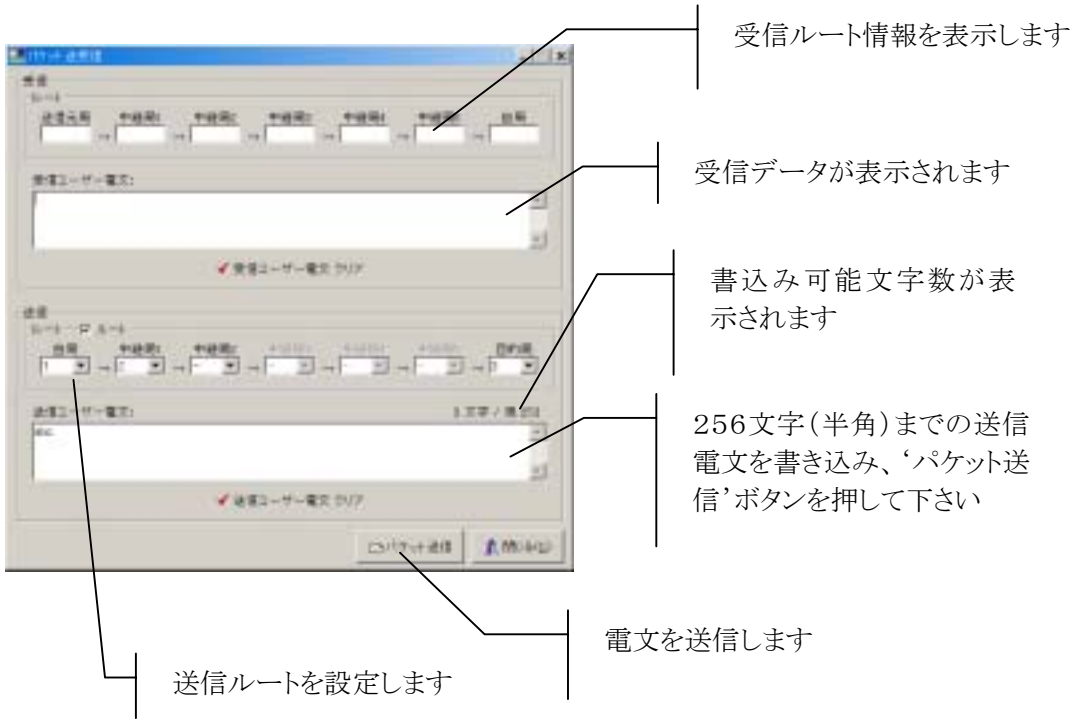
エアーモニタは429MHz帯フィールドの7チャンネルから46チャンネルまでの電界強度を測定し表示します。
通信時点におけるフィールドのノイズや混信特性の確認に利用できます。
詳細はDLLの呼び出し方と一緒に、第4章で説明します。

◆ アンサーバック機能

機器 ID を指定してリーフノード(子機)の受信電波強度を測定・表示します。
電波強度レベルは5段階(0、1、2、3、4)で表示(電波強度レベルLEDに対応)します。

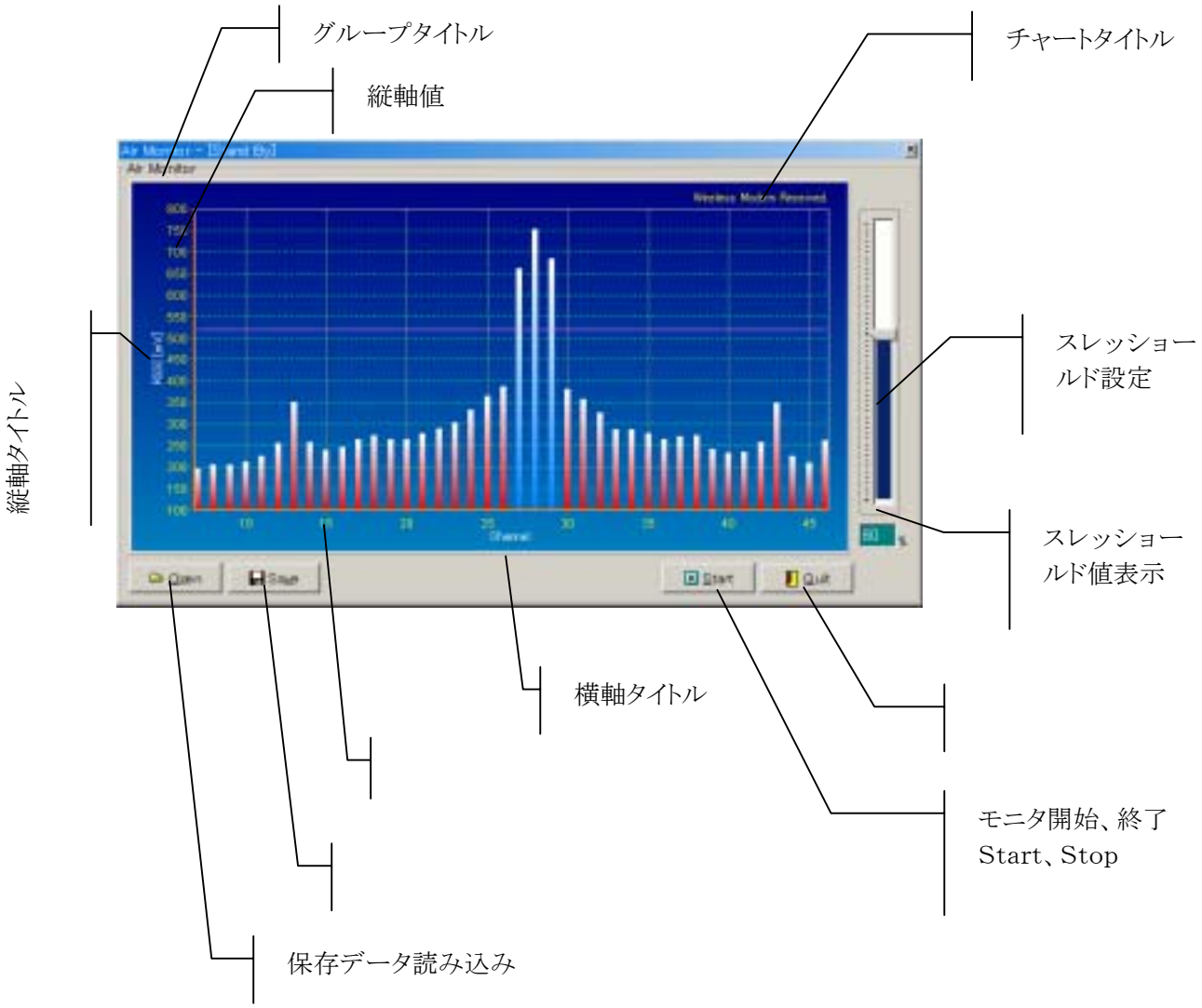
◆ パケット送受信機能

この機能は $ SENDコマンドのデモです。



第4章　エアーモニタとDLLの使い方

エアーモニタは429MHz帯フィールドの7チャンネルから46チャンネルまでの電界強度を測定し表示します。
通信時点におけるフィールドのノイズや混信特性の確認に利用できます。
スレッショールドレベルがマウス操作で簡単に設定することができ、スレッショールドのを超えたレベルはグラフが色分けされ視認性に優れています。
このエアーモニタはDLLの形でも供給されるので、モニタプログラムを作制しなくてもユーザーアプリケーションに組み込んで使用することができます。VC＋＋、C＋＋Builder、Delphi、VBなどの開発言語から簡単にアクセスできます。
下図は28チャンネルに電波が出ている時の様子です。



4．1 エアーモニタ機能

セットアッププログラムのエアーモニタボタンを押すと、上のようなフォームが表示されます。

❑モニタのスタート、ストップ

‘Start’ボタンを押ししばらくすると現在のフィールドの電界強度が表示されます。。
モニタを中止する場合は、‘Stop’ボタンを押して下さい。

❑スレッショールドレベルの設定

電界強度の表示はスレッショールドレベルの設定によって色分けされます。レベルはマウスで簡単にスライド変化させることができます。レベルは%表示でスラードバーの下に表示されます。

❑モニタデータの保存と読込み

‘Save’ボタンを押すとモニタ画面情報をファイルに保存することができます。保存したデータはいつでも読み込むことができます。ファイルのエクステンションは‘．ADT’です。

4．2 エアーモニタDLLの使い方

ユーザプログラムに、エアーモニタDLLを使ってエアーモニタ機能を持たせることができます。
このDLLは Windows の開発言語VC＋＋、C＋＋Builder、Delphi、VBなどから呼び出すことができます。

DLLファイル名： Analyze32.dll

☞注意:DLLはアプリケーションプログラムと同じフォルダに置いて下さい。

◆ DLLファイルをロードするには

DLLファイル（ファイル名：Analyze32.dll）をロードするには次のようにします。

(1) DLLハンドル変数の宣言

グローバル変数で下記のようにDLLハンドル変数の宣言を行います。

```
static HINSTANCE HInstanceDLL = 0;
```

グローバル変数とするのはDLLアンロード時にこの変数が必要になるからです。

(2) DLLのロード

LoadLibrary() 関数（WindowsAPI）を使います。

```
HInstanceDLL = LoadLibrary("ANALYZE32.DLL");
if (!HInstanceDLL) {
    MessageBox(HWindow, "DLL がロードできません。", "AnalyzeTest", MB_OK);
    PostQuitMessage(0);                    // 終了
    return;
}
```

DLLのロードはユーザーアプリケーションの初期化部分（あるいはエアーモニタ表示が必要となる前処理において）で行います。

LoadLibrary() 関数の戻り値が NULL の場合，DLLのロードに失敗しましたから，適切なエラー処理を行って下さい。（上記の例ではアプリケーションを終了しています。）

◆ DLLファイルのアンロード

エアーモニタ表示DLLが不要になったら(ユーザーアプリケーション終了時など)DLLをアンロードして下さい。
アンロードには FreeLibrary() 関数(WindowsAPI)を使います。

・例　　:　　if (HInstanceDLL)
　　FreeLibrary(HInstanceDLL);　　// Analyze32.dll をアンロードする

◆ DLL各関数の使い方

DLL内関数のアドレスを求めます。(DLLをロードしただけではDLL内関数の呼び出しができません。)

(1) 関数へのポインタを宣言

DLL内の各関数は下記のように宣言されています。

```
extern "C" {
    BOOL _declspec(dllexport) _stdcall DLLShow(void);
    BOOL _declspec(dllexport) _stdcall DLLHide(void);
    BOOL _declspec(dllexport) _stdcall DLLTitle(const ANALYZE_TITLE *);
    BOOL _declspec(dllexport) _stdcall DLLStatus(ANALYZE_STATUS * const);
    BOOL _declspec(dllexport) _stdcall DLLUpdate(const ANALYZE_UPDATE *);
    BOOL _declspec(dllexport) _stdcall DLLControl(const ANALYZE_CONTROL *);

  int WINAPI DllEntryPoint(HINSTANCE, DWORD, void *);
}
```

これに対応する変数(関数へのポインタ)をユーザーアプリケーションで下記のように宣言します。

```
#include "Analyze32.h"                    //※1
static BOOL _stdcall (FAR *LpfnDLLShow)(void);
static BOOL _stdcall (FAR *LpfnDLLHide)(void);
static BOOL _stdcall (FAR *LpfnDLLTitle)(const ANALYZE_TITLE *);
static BOOL _stdcall (FAR *LpfnDLLStatus)(ANALYZE_STATUS * const);
static BOOL _stdcall (FAR *LpfnDLLUpdate)(const ANALYZE_UPDATE *);
static BOOL _stdcall (FAR *LpfnDLLControl)(const ANALYZE_CONTROL *);
```

※1　関数とのパラメータの受け渡しは構造体へのポインタを用いますので, "Analyze32.h" をユーザーアプリケーションで #include して下さい。

(2) 関数アドレスの取得

関数アドレスの取得には GetProcAdress() 関数(WindowsAPI)を使います。

```
LpfnDLLShow = (BOOL (_stdcall *)(void))GetProcAddress(HInstanceDLL, "DLLShow");
LpfnDLLHide = (BOOL (_stdcall *)(void))GetProcAddress(HInstanceDLL, "DLLHide");
LpfnDLLTitle = (BOOL (_stdcall *)(const ANALYZE_TITLE *))GetProcAddress(HInstanceDLL,"DLLTitle");
LpfnDLLStatus = (BOOL (_stdcall *)(ANALYZE_STATUS * const))GetProcAddress(HInstanceDLL, "DLLStatus");
LpfnDLLUpdate= (BOOL (_stdcall *)(const ANALYZE_UPDATE*))GetProcAddress(HInstanceDLL, "DLLUpdate");
LpfnDLLControl=(BOOL(_stdcall*)(constANALYZE_CONTROL ))GetProcAddress(HInstanceDLL, "DLLControl");
```

上記の例ではチェックしていませんが, GetProcAddress() 関数が NULL を返す場合, 関数アドレスの解決に失敗したことになります。(DLL内関数の呼び出しは出来ません。)関数へのポインタ変数の宣言が正しいか調べて下さい。

(3) DLL内関数の呼び出し

DLL関連の初期化(DLLファイルのロード, 関数アドレスの取得など)が正常に処理されていれば, DLL内関数の呼び出しは一般の関数の呼び出しと同様に行えます。

・例　　:　　LpfnDLLShow();　　// エアーモニタ画面を表示する。

◆ DLL内関数リファレンス

DLL内関数の説明です。DLL内関数の戻り値はすべて BOOL 型となります。

(1) DLLShow() 関数 －エアーモニタ画面の表示

・宣言　　:　　BOOL _declspec(dllexport) _stdcall DLLShow(void);
・戻り値　　:　　成功:TRUE(1) / 失敗: FALSE(0)
・引数　　:　　なし(void)

・例　　:　　LpfnDLLShow();　　// エアーモニタ画面を表示する。

(2) DLLHide() 関数 －エアーモニタ画面の非表示

・宣言　　:　　BOOL _declspec(dllexport) _stdcall DLLHide(void);
・戻値　　:　　成功:TRUE(1) / 失敗: FALSE(0)
・引数　　:　　なし(void)

・例　　:　　LpfnDLLShow();　　// エアーモニタ画面を非表示にする。

(3) DLLTitle() 関数 －画面タイトルの設定

・宣言　　:　　BOOL _declspec(dllexport) _stdcall DLLTitle(const ANALYZE_TITLE *);
・戻値　　:　　成功:TRUE(1) / 失敗: FALSE(0)
・引数　　:　　設定用 ANALYZE_TITLE 構造体へのポインタ

・ANALYZE_TITLE 構造体

```
typedef struct _ANALYZE_TITLE {
    char *lpszTitleAxisLeft;        // 左目盛りタイトル
    char *lpszTitleAxisBottom;      // 下目盛りタイトル
    char *lpszTitleChart;           // 右上(チャート)タイトル
    char *lpszTitleGroup;           // 左上(全体)タイトル
} ANALYZE_TITLE;
```

・機能　　:　　エアーモニタ画面4カ所のタイトル設定を行います。設定するタイトル(文字列へのポインタ)を ANALYZE_TITLE 構造体の各メンバに代入して下さい。タイトルを変更しない場合は必ず (char *)NULL とします。

・例　　:　　左目盛りタイトルを "Level" → "RSSI[dBuv]" に変更

```
static char buf[] = "RSSI[dBuV]";
static ANALYZE_TITLE atTitle;

atTitle.lpszTitleAxisLeft = (char *)buf;        // "RSSI[dBuV]" に変更
atTitle.lpszTitleAxisBottom = (char *)NULL;     // 変更無し
atTitle.lpszTitleChart = (char *)NULL;          // 変更無し
atTitle.lpszTitleGroup = (char *)NULL;          // 変更無し

LpfnDLLTitle((const ANALYZE_TITLE *)&atTitle);
```

(4) DLLStatus() 関数 －ステータス取得

・宣言　　:　BOOL _declspec(dllexport) _stdcall DLLStatus(ANALYZE_STATUS * const);
・戻値　　:　成功:TRUE(1) / 失敗: FALSE(0)
・引数　　:　格納用 ANALYZE_STATUS 構造体へのポインタ

・ANALYZE_STATUS 構造体

```
typedef struct _ANALYZE_STATUS {
    int Start;                          // 測定フラグ(0:停止／1:測定中)
    int Threshold;                      // 閾値(%)
} ANALYZE_STATUS;
```

・機能　　:　エアーモニタ表示のステータス(測定フラグ, 閾値)を取得します。

・例　　:　エアーモニタ画面のステータス(測定中／待機中)を取得して表示する。

```
static ANALYZE_STATUS asStatus;

LpfnDLLStatus((ANALYZE_STATUS * const)&asStatus);
MessageBox(HWindow, asStatus.Start ? "測定中" : "待機中", "AnalyzeTest", MB_OK);
```

(5) DLLUpdate() 関数 －エアーモニタデータ(レベル)の更新

・宣言　　:　BOOL _declspec(dllexport) _stdcall DLLUpdate(const ANALYZE_UPDATE *);
・戻値　　:　成功:TRUE(1) / 失敗: FALSE(0)
・引数　　:　設定用 ANALYZE_UPDATE 構造体へのポインタ

・ANALYZE_UPDATE 構造体

```
typedef struct _ANALYZE_UPDATE {
    int *Level;                         // レベルデータブロックへのポインタ
    int Length;                         // レベルデータ長(40 固定)
} ANALYZE_UPDATE;
```

・機能　　:エアーモニタデータを更新します。ステータスが”測定中”の場合は表示に反映されます。

・例　　:　ランダムなレベルを設定

```
static ANALYZE_UPDATE auUpdate;
static int Level[40];

for (int i = 0; i < 40; i++) {
    Level[i] = random(100);                 // 適当な値(Level)を作成
}

auUpdate.Level = Level;                     // 先頭アドレスをセット
auUpdate.Length = 40;                       // 全40ch
LpfnDLLUpdate((const ANALYZE_UPDATE *)&auUpdate);
```

(6) DLLControl() 関数 －エアーモニタ画面・パラメータ設定

・宣言　　:　　BOOL_declspec(dllexport)_stdcallDLLControl(const NALYZE_CONTROL *);
・戻値　　:　　成功:TRUE(1) / 失敗: FALSE(0)
・引数　　:　　設定用 ANALYZE_CONTROL 構造体へのポインタ

・ANALYZE_UPDATE 構造体

```
typedef struct _ANALYZE_CONROL {
    int AxisRightMaximum;                    // 左目盛り最大値(0～1000)
    int AxisRightMinimum;                    // 左目盛り最小値(0～1000)
    int Threshold;                           // 閾値(0～100)
} ANALYZE_CONTROL;
```

・機能　　:　　左目盛りの最大値・最小値, 閾値を設定します。
AxisRightMaximum が AxisRightMinimum より小さい場合エラー(FALSE を返す)となります。各メンバは上記(コメント部)範囲内にして下さい。

・例　　:　　最大値[300], 最小値[20], 閾値[30%]に設定する。

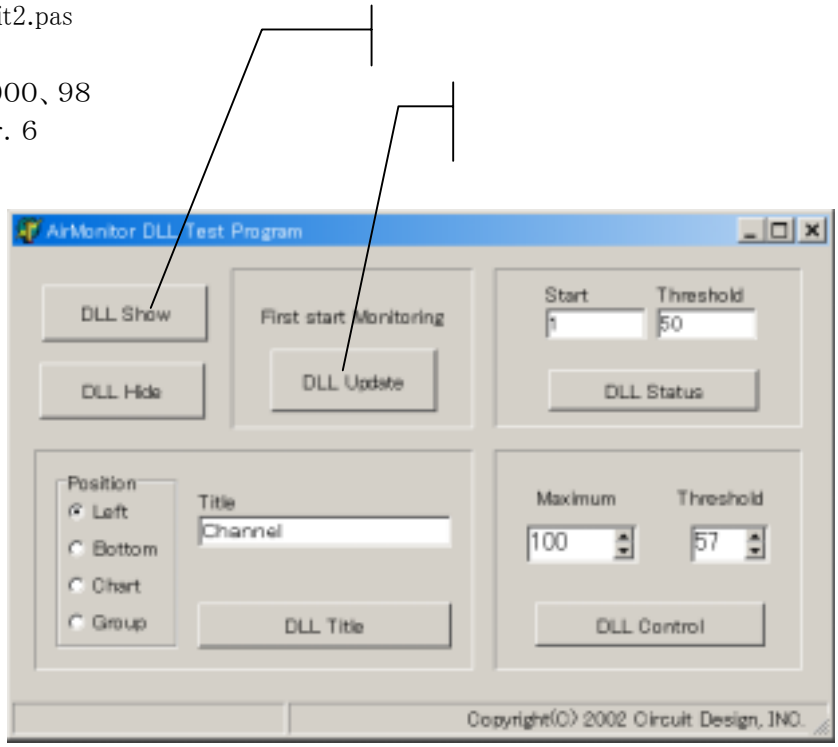
```
static ANALYZE_CONTROL acControl;

acControl.AxisRightMaximum = 300;            // 最大値:300
acControl.AxisRightMinimum = 20;             // 最小値:20
acControl.Threshold = 30;                    // 閾値:30%

LpfnDLLControl((const ANALYZE_CONTROL *)&acControl);
```

4．3　参考：エアーモニタDLLテストプログラム

　このプログラムはDelphiユーザのためのエアーモニタDLLテスト用参考プログラムです。
CDには次のファイルが入っています。実行ファイルでテストしながらプログラムソースファイルをご覧下さい。
DLLファイルは実行ファイルと同じフォルダに置いて下さい。

DLLファイル名：Analyze32.dll
実行ファイル名：AirMonitorDLL.exe
ソースファイル名：SAUnit2.pas

動作環境：　Windows2000、98
作制言語：Delphi　Ver. 6



◆　使い方

　各ボタンのキャプションはDLL内部関数に対応しています。
まず、‘DLL　Show’ボタンでDLLを表示して下さい。

❑DLLShow関数、DLLHide関数のテスト
　　‘DLL　Show’ボタンで表示、‘DLL　Hide’ボタンで表示が消えます。
❑DLLTitle関数のテスト
　　タイトルの変更はTitle欄に文字を入れ、変更位置をPositionで選び、DLL　Titleボタンを押して下さ　　い。
❑DLLUpdate関数のテスト
　　まず、DLL画面で‘Start’ボタンを押してモニタをスタートさせて下さい。
　　‘DLL　Update’ボタンを押す度にランダムデータを作り出し表示します。
❑DLLStatus関数のテスト
　　‘DLL　Status’ボタンを押すとDLLからモニタの開始、停止状態とスレッショルド値を読み込みます。
❑DLLControl関数のテスト
　　縦軸の目盛りとスレッショルド値を設定します。スピンボタンで設定してから‘DLL　Control’ボタンを押して下さい。

このユーザーズマニュアルの記載内容については万全を期しておりますが、
万一不明な点、不備な点などがありましたら、弊社窓口にご連絡下さい。

このマニュアルの内容は、予告無く変更することがあります。ご了承下さい。

M1 セットアッププログラム
ユーザーズマニュアル
Jul. 31 2002
発行: 株式会社サーキットデザイン

〒399-8303 長野県南安曇郡穂高町穂高 7557-1
株式会社サーキットデザイン
TEL:(0263)82-1024 FAX:(0263)82-1016
e-mail: nbd@circuitdesign.jp
web: http://www.circuitdesign.jp/