Panasonic

通信用OCXコントロール

Control CommX

HELPファイル内容

ご使用になる前に

■ 使用環境について

Microsoft Visual Basic Ver.6.0 SP3以上が正常に動作する環境でお使いください。
もしくは、Microsoft Visual Basic .NET2002, C# .NET2002以上が正常に動作する環境でお使いください。

OS	:	Windows95 OSR2(Ver.4.00.950B)以上/ 98/ Me WindowsNT(Ver.4.0以上)/2000/XP
必要ハードディスク容量	:	20MB以上
CPU	:	Pentium 300MHz以上
搭載メモリ	:	128MB以上
画面解像度	:	1024×768 以上
表示色	:	High Color(16ビット)以上

■ キーユニットに関して

本ソフトウェアに同梱されています。
キーユニットを装着しなければ、通信を実行することはできません。
(本ソフトウェアを利用してVisual Basicでプログラミングすることはできますが、通信を実行することはできません。)
通信を実行するためには、本ユニットをプリンタポートまたはUSBポートに装着する必要があります。
キーユニットの装着に関しては、“キーユニットの装着について”をご参照ください。

■ 対応PLC機種

松下電工製プログラマブルコントローラFPシリーズ全機種に対応。

■ 対応ネットワーク

・RS232C(C-NET)接続
・Ethernet接続
・モデム接続

キーユニットの装着について

キーユニットのタイプにより、接続形態が変わります。

■ IBM PC/AT互換機専用
プリンターポート直結型

接続方法：[コンピュータのプリンタポート] - [キーユニット]
キーユニットの先にプリンタケーブルを接続することは可能です。

■ IBM PC/AT互換機・NEC 9821シリーズ共用
USB(Universal Serial Bus) ポート直結型

接続方法：[コンピュータのUSBポート] - [キーユニット]
USBポートを1つ占有します。
キーユニットの先にUSBケーブルを接続することはできません。

注意　コンピュータ側で、USBデバイスが使用できる環境になっていなければご使用になることは出来ません。
詳しくは、各コンピュータのマニュアルをご参照下さい。

なお、キーユニットの形状等につきましては、予告なしに変更することがあります。

使用条件について

本ソフトウェアの使用の条件について、とくにご注意いただく項目について記述します。

■ 対象ユーザー
本ソフトウェアをご利用頂くに際しては、Microsoft Visual Basicに関する充分な知識が必要です。本ソフトウェアは、Microsoft Visual Basicに関する充分な知識があるお客様を対象に開発されております。Microsoft Visual Basicのご利用方法、及び本製品の通信に無関係なMicrosoft Visual Basicのプログラミングに関するご質問はお受けできません。あらかじめご了承ください。

■ 再配布の禁止
本商品に含まれるソフトウェアは、すべて再配布が禁止されています。お客様が本ソフトウェアを使ったアプリケーションを開発した場合で、かつ、そのアプリケーションとともに使用する限りであっても、本ソフトウェアの実行部であるActiveXコントロールを再配布することはできません。

■ 保証の範囲
本ソフトウェアの商品の保証および動作確認の範囲を示します。

本ソフトウェアは、Microsoft Visual Basic 6.0 SP3以上で動作することを保証します。
もしくは、Microsoft Visual Basic .NET2002, C# .NET2002以上で動作することを保証します。(但し、一部使用できない機能があります。下記のMicrosoft Visual Basic .NETまたはC#の使用上の注意を確認してください。)
しかし、これは本ソフトウェアを利用してお客様が開発されたアプリケーションの正常な動作や24時間連続稼動を保証するものではありません。

本ソフトウェアは、Microsoft ExcelのVBA(マクロ)の環境で動作することを確認していますが、動作を保証するものではありません。
しかし、動作確認・保証の範囲外の利用環境における本ソフトウェアの使用を禁止するものではありません。

また、本ソフトウェアに添付されているサンプルコードは、あくまでもお客様の責任範囲でお使い頂きますようお願い致します。サンプルコードに起因する問題によりお客様で損害が発生しても、一切の保証をしないものとします。

その他、本書に記載されていない内容に関しては、弊社のソフトウェア・ライセンス条件に従い解釈されるものとします。

■バージョン互換について

本ソフトウェアは、バージョン間の互換性はありません。
下位バージョンで作成された実行モジュールは再作成する必要があります。

再作成を行わず実行すると、下記のメッセージが表示され実行出来ません。
「コントロール'VB.UserControl'をアクティブ化できませんでした。
このコントロールにこのアプリケーションとの互換性がない可能性があります。
アプリケーションで提供されたバージョンのコントロールを使用しているかどうか確認してください。」

■Microsoft Visual Basic .NETまたはC# 使用上の注意

WP10(弊社PHSデータ通信ユニット)用のActiveXに関しては、Microsoft Visual Basic .NET または C# には対応していません。
また、遠隔地にあるモデムからの送信に関して、コンピュータ側で受信できません。使用できないメソッド、イベントは下記の通りです。

メソッド	ReceivePortOpen
	ReceivePortClose
イベント	OnReceive

お客様が作成されたアプリケーションに関してPortOpenメソッドを実行した後、アプリケーションを終了する前にPortOpenを実行したポートNo.に対して必ずPortCloseメソッドを実行してください。実行しなかった場合、アプリケーションエラーが発生します。

Windowsアプリケーション以外のWebアプリケーションやWebサービス等のアプリケーションは作成できません。

機能概要

■ 本ソフトウェアについて
本ソフトウェアは、弊社FPシリーズPLCの内部データを、コンピュータ上で表示・操作するアプリケーションを容易に構築できるようにするソフトウェア部品です。アプリケーションを作成するにあたり、当社プロトコル(MEWTOCOL)を意識することなくPLCへのアクセスが実現できます。

[特長]

■ コントロールをフォームに貼りつけるだけで、簡単に通信プログラムが作成できます。
当社のPLC通信用プロトコル(MEWTOCOL)に対する知識は、一切不要です。

■ 対応するネットワークを意識する必要もありません。
対応するネットワークの種類によって、基本的には通信コマンドを変更する必要はありません。
使用するネットワーク用の設定画面も、コマンドを1行記述するだけで、起動することができます。

■ 作成したアプリケーションは、弊社製ソフトウェアと同時に通信することが出来ます。
たとえRS232CのCOMポート1が、弊社ツールソフト(*注)に使用されていたとしても、本製品を利用して作成頂いたアプリケーションは、コンピュータの同じCOMポート1を使用して同時に通信することができます。当社ツールソフトの通信を停止させる必要はありません。
これにより、お客様で作成されたアプリケーションのデバッグ効率が飛躍的に向上します。

(*注) 上記に対応している弊社ツールソフトは、現状以下のとおりです。
・PLC用プログラミングツールソフト Control FPWIN GR Ver.1.1以上
・PLC用プログラミングツールソフト Control FPWIN Pro Ver.4.0以上
・表示器用画面作成ツールソフト Terminal GTWIN Ver.1.0以上
・稼働データ収集ソフトPCWAY Ver.2.1以上
・OPCサーバソフト MEWTOCOL OPC Server Ver1.0以上

上記以外のソフトウェア、及び他社のソフトウェアが通信の資源を使用している場合、同時通信はできません。ご注意ください。

[主な通信機能]
本ソフトウェアを使用して実現できるおもな通信機能を簡単に説明します。
詳細は、ヘルプを参照してください。

■連続リード／ライト	:	PLC内の連続した接点やレジスタ、及びPLCに装着されたICカードの情報を読み書きできます。
■ランダムリード	:	種類の異なるデバイスや連続していない接点やレジスタ情報を読み込むことができます。
■PLCステータスリード	:	PLCの状態(RUN/RPOG等)を表示することができます。
■PLC RUN/PROG切替	:	PLCの状態(RUN/RPOG)を変更することができます。
■通信設定表示	:	通信条件の設定内容を表示・変更できます。
■通信条件検索機能	:	RS232C接続時に合致する通信条件を自動的に検索することができます。
■モデム受信接続	:	モデム接続時には、PLCからの受信によってイベントを発生させることが可能です。 Microsoft .NET(VB/C#)に関しては使用できません。
■変換関数	:	2進数 <-> 10進数 <-> 8進数 <-> 16進数 の各々の変換関数も搭載されています。

インストール

■ インストールの手順
インストールは以下の手順で行ってください。
Windows NT/2000/XPでは、"Administrator"権限のあるユーザーでログオンしてください。
1.CD-ROMドライブにCD-ROMをセットします。
2.CD-ROM内の setup.exeを起動します。
3.以降は、画面に表示されるセットアッププログラムの指示に従ってください。

■ インストールされるプログラムフォルダ
インストールされるグループ名は、[¥NAiS Control¥CommX]になります。

■ インストールされるソフトウェア
インストールされると以下のソフトウェアがインストールされます。

・通信用ActiveX	:	Microsoft Visual Basicで使用する通信用部品です。 上記のプログラムフォルダには表示されません。
・オンラインヘルプ	:	本ソフトウェアの使用方法・詳細内容が記載されています。 上記のプログラムフォルダに表示されます。
・サンプルプログラム	:	本ソフトウェアを利用したサンプルプログラムです。 このサンプルプログラムを利用して動作確認を行うことができます。 上記のプログラムフォルダに表示されます。

■ アンインストールの手順
アンインストールする場合は、コントロールパネルの[アプリケーションの追加と削除]を起動し、[CommX]を選択してください。

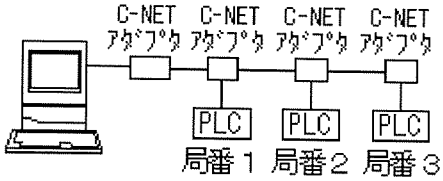
RS232C(C－NET)接続

[1]　コンピュータとPLCを直接接続する場合

■ 自局(局番0)としてアクセスしてください。

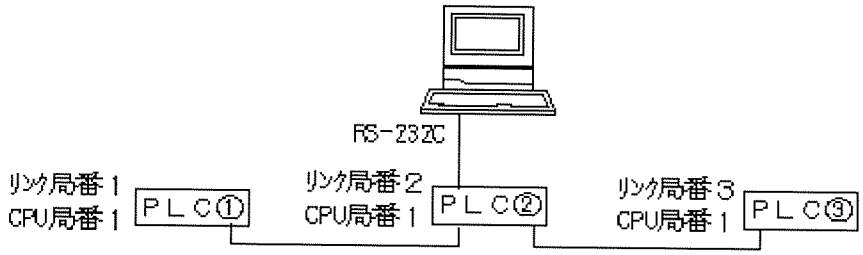
[2]　C-NETアダプタを利用する場合

■ C-NET接続でコンピュータに接続できるPLC局数は、最大32局です。



[3]　MEWNET-H/Pリンクユニットを利用する場合

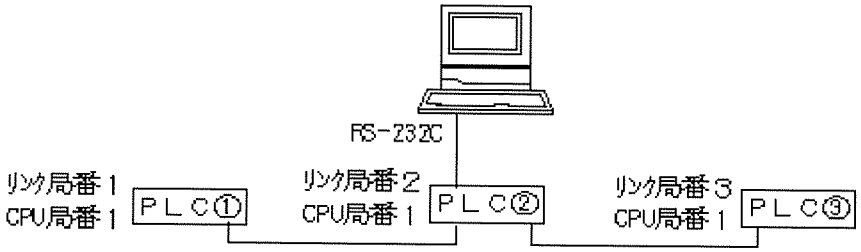
■ MEWNET-H/Pリンクユニットを介して接続できるPLC局数は、最大64局です。



リンク局番とは、リンクユニットに設定されている局番を示します。
CPU局番とは、CPUユニットに設定されている局番を示します。

[4]　MEWNET-Wリンクユニットを利用する場合

■ MEWNET-Wリンクユニットを介して接続できるPLC局数は、最大32局です。



リンク局番とは、リンクユニットに設定されている局番を示します。
CPU局番とは、CPUユニットに設定されている局番を示します。

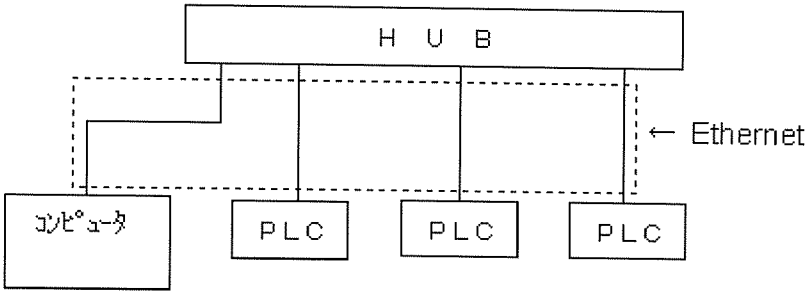
Ethernet接続

Ethernet通信をお使いの場合は、コンピュータ側のIPアドレスの設定や、ET-LANユニット、及びEthernetに関して充分にご理解いただいた上でお使いください。
特に、ET-LANユニットをお使いになられる場合には、「ET-LANユニット導入マニュアル」を充分にご理解いただいた上で、コンピュータとPLCの設定を一致させてお使いください。

Ethernetを使用したネットワークと接続する形態には下記の2種類があります。

[1] MEWNETの各リンク経路を利用しない場合
(Ethernetのみで接続する場合)

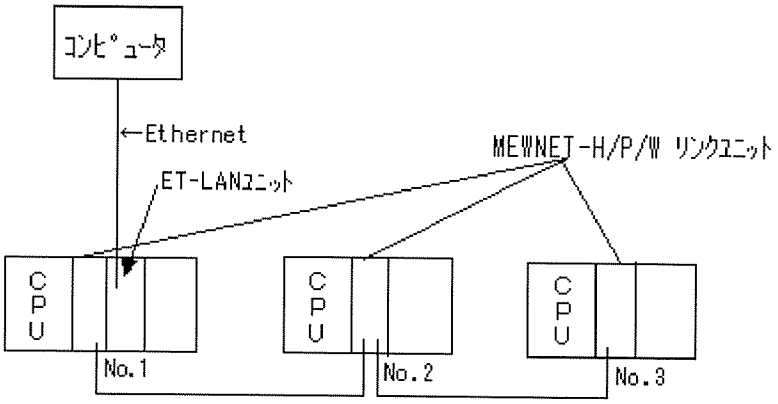
この場合はET-LANユニットが使用できます。
IPアドレスを持つ複数台の機器(PLC等)とEthernetでの接続が可能です。



コンピュータ(又はHUB)とPLCはET-LANユニット、または市販の[Ethernet/RS232C変換ユニット]で接続してください。

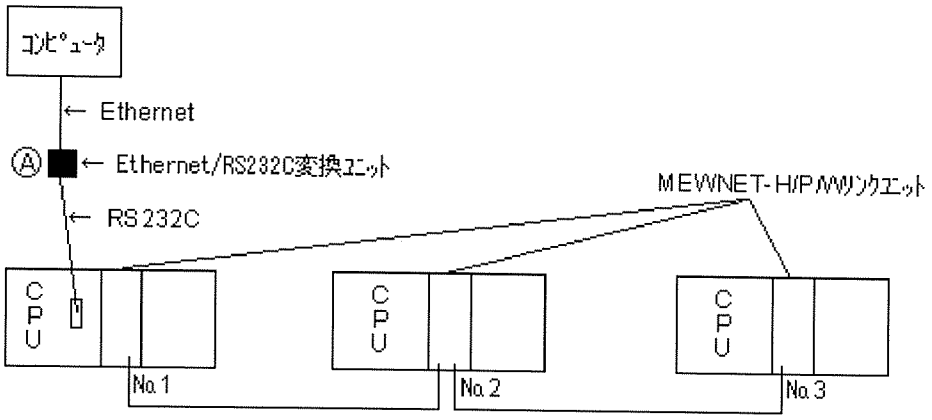
[2] MEWNETの各リンク経路を利用する場合

1)ET-LANユニットを利用する場合



※ No.1のリンクユニットは必ずCPUの横にセットしてください。
※ コンピュータとET-LANユニットを直接接続する時のEthernetケーブルはクロスケーブルになります。

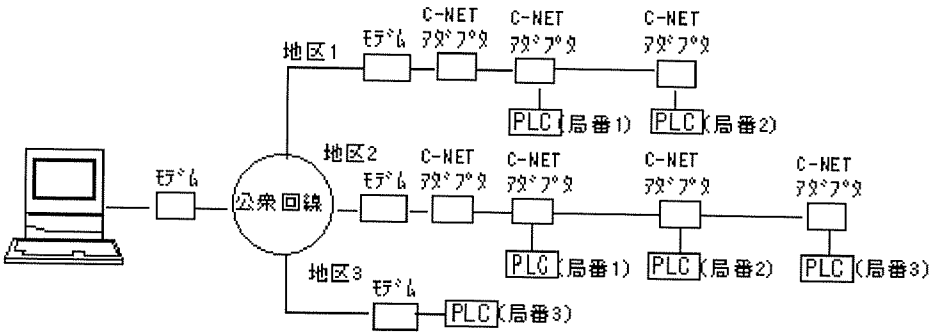
2)CPUのTOOLポートを利用する場合
この場合はET-LANユニットが使用できません。
Ethernetで接続するのは、IPアドレスをもつ機器(Ethernet/RS232C変換ユニット)1台だけになります。
それ以外は、MEWNETの経路を利用して通信します。



※ コンピュータとPLCをET-LANユニットで接続されますと正常に動作できません。
必ず、上図のように市販の[Ethernet/RS232C変換ユニット]を使用してコンピュータとPLCのCPUを接続してください。

モデム接続

公衆回線先の接続地区数は無制限ですが、1つの地区に接続できるPLC局数は、RS232C(C-NET)接続の接続タイプに依存します。



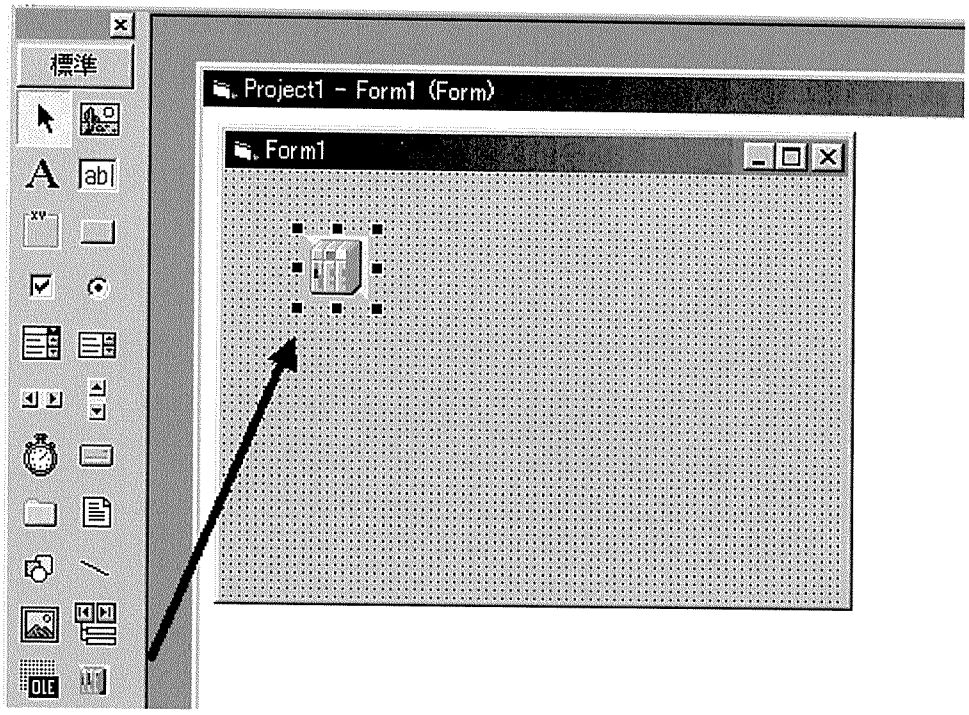
コンポーネントの選択

1. Visual Basicを起動します。
2. Visual Basicの[ファイル]メニューから[新しいプロジェクト]を選択し、<標準EXE>を選択します。
3. [プロジェクト]メニューから[コンポーネント]を選択し、<コンポーネント>ダイアログ画面を表示します。
4. [コントロール]タブを選択し、リストの中から[CommX]をチェックして、[OK]ボタンを押下します。

この時点で、ツールボックスウィンドウにCommXのアイコン  が表示されます。

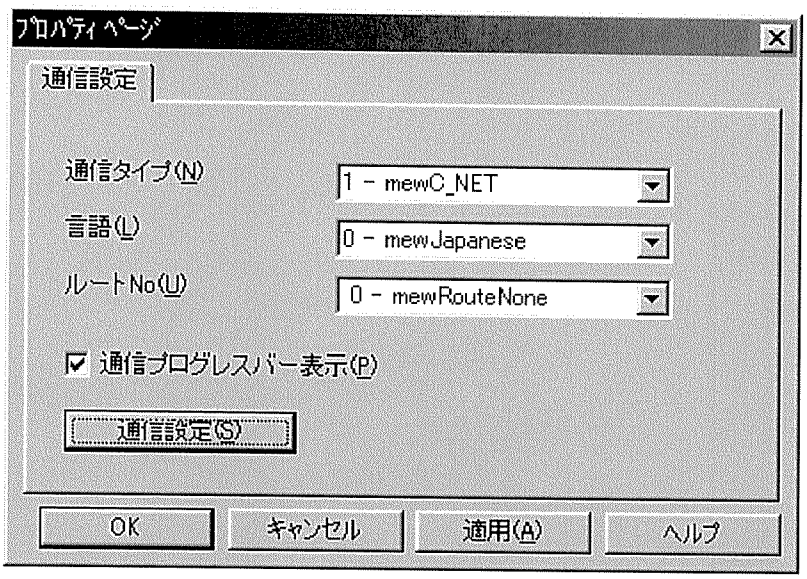
オブジェクトの貼り付け

ツールボックスウィンドウ上のCommXのアイコン  をドラッグし、フォームの上にドロップします。
フォーム上にCommXのアイコンが表示されます。これがCommXのオブジェクトです。



プロパティページの設定

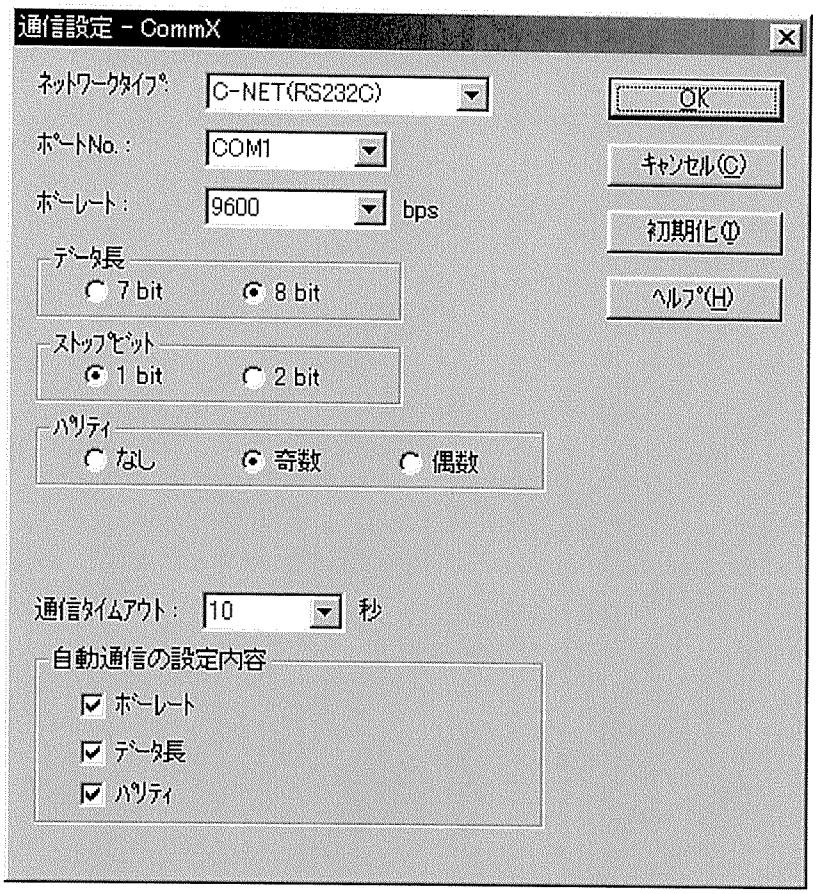
CommXのオブジェクトを選択した状態で、[表示]メニューから[プロパティページ]を選択し設定ダイアログを表示します。
(CommXのオブジェクトを右クリックした後、[プロパティ]を選択しても表示できます)



通信タイプ	:	ネットワークタイプを下記から選択します。 1:mewC_NET　RS232C(C-NET)の場合 5:mewEthernetLocal　Ethernet 通信の場合 6:mewMODEM　モデム通信の場合
言語	:	表示されるエラーメッセージの言語です。 現バージョンでは、下記から選択可能です。 0:mewJapanese(日本語) 1:mewEnglish(英語) 2:mewChineseSimplified(中国語) 5: mewSpanish(スペイン語) 6: mewItalian(イタリア語) 7: mewGermany(ドイツ語) 8: mewFrench(フランス語)
リンクユニット局番	:	通信タイプで選択されたネットワークを使用して、リンクユニットを介して通信をする場合は、下記から選択してください。 0:mewLinkUnitNone(リンクユニットを使用しない) 1:mewRoot1(ルート1を使用する) 2:mewRoot2(ルート2を使用する) 3:mewRoot3(ルート3を使用する) (プログラムからの変更も可能です)
通信プログレスバー表示	:	1パケットで納まらない通信を行う場合に通信の進行状況ダイアログを表示するかどうかを設定します。
通信設定	:	通信条件の詳細を設定します。"通信設定の起動"を参照してください。

通信設定の起動

前述の[プロパティページ]の[通信設定]ボタンを押下し、通信設定ダイアログを表示します。

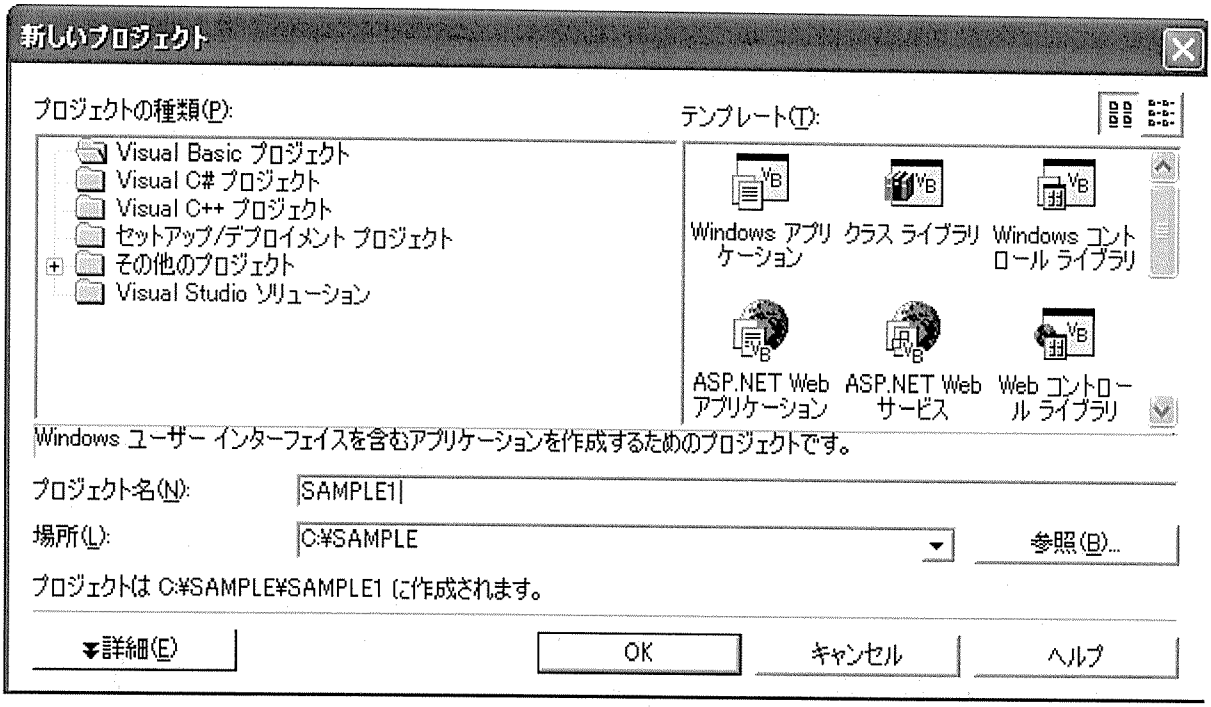


以降の設定は、ネットワークタイプによって異なります。
“各ネットワークでの設定”をご参照ください。

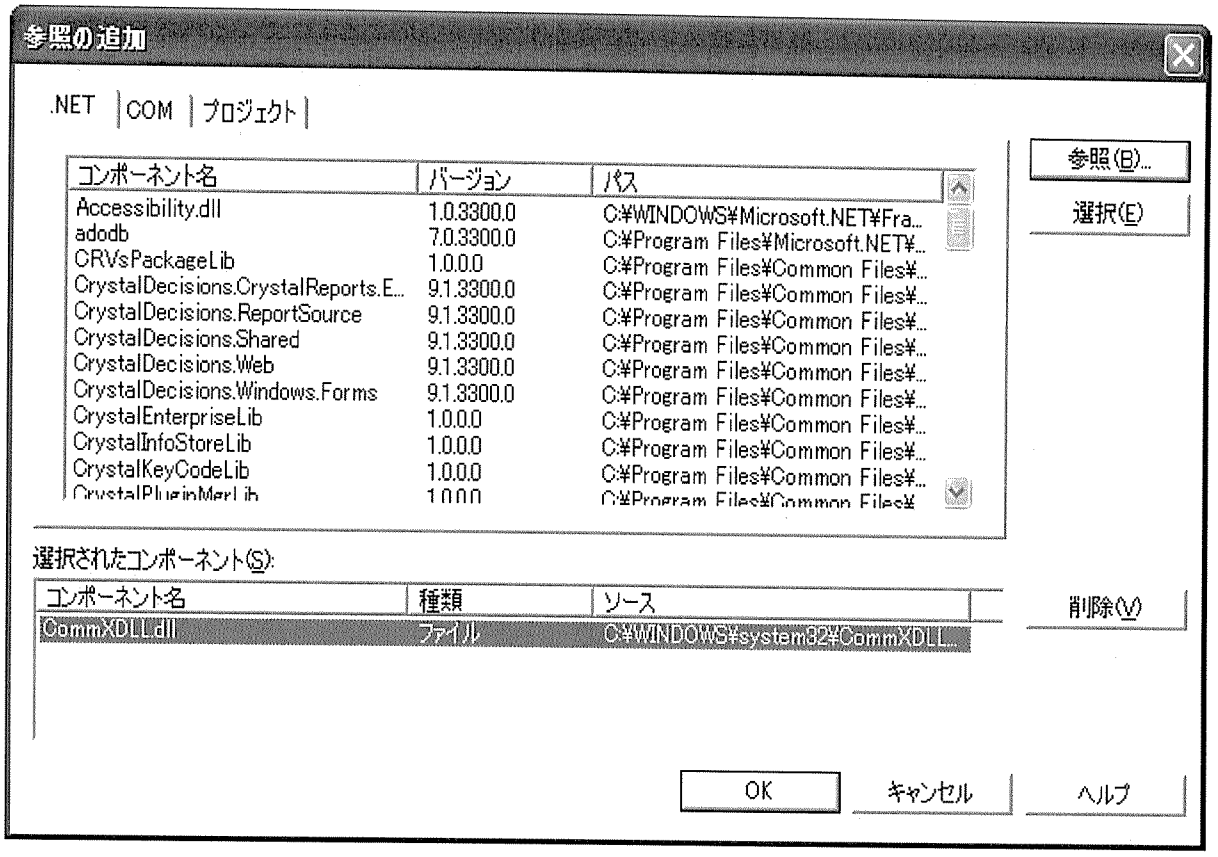
Visual Basic .NETでの初期設定(参照の追加)

新しいプロジェクトを作成される場合はVisual Basic .NETを起動し最初に以下の作業を行ってください。

1. Microsoft Visual Studio .NETを起動します。
2. 新しいVisual Basicプロジェクトを作成します。



3. 「プロジェクト」メニューから「参照の追加」を選択し、<参照の追加>ダイアログ画面を表示します。
4. 「参照」ボタンをクリックし、C:¥Windows¥system32フォルダ内のCommXDLL.DLLを選択し、[OK]ボタンを押下します。



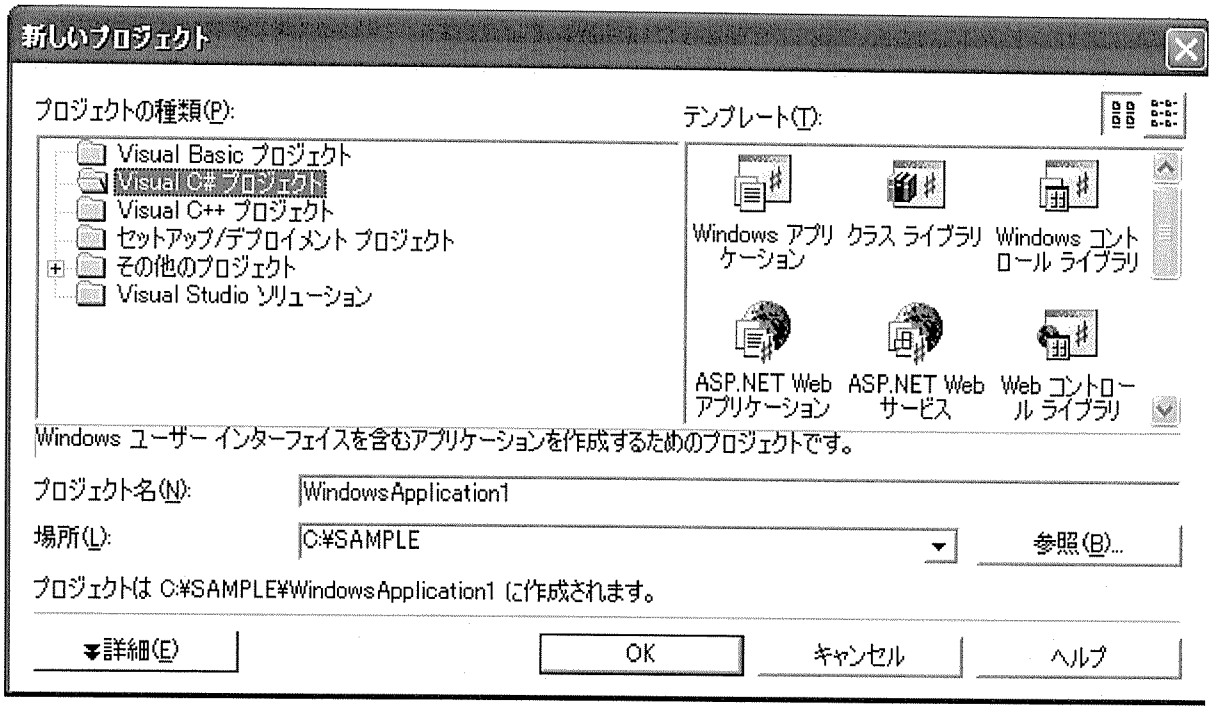
この時点でソリューションエクスプローラの参照設定フォルダ内にCommX_DLLが表示されます。

C# .NETでの初期設定(参照の追加)

新しいプロジェクトを作成される場合は、C# .NETを起動し、最初に以下の作業を行ってください。

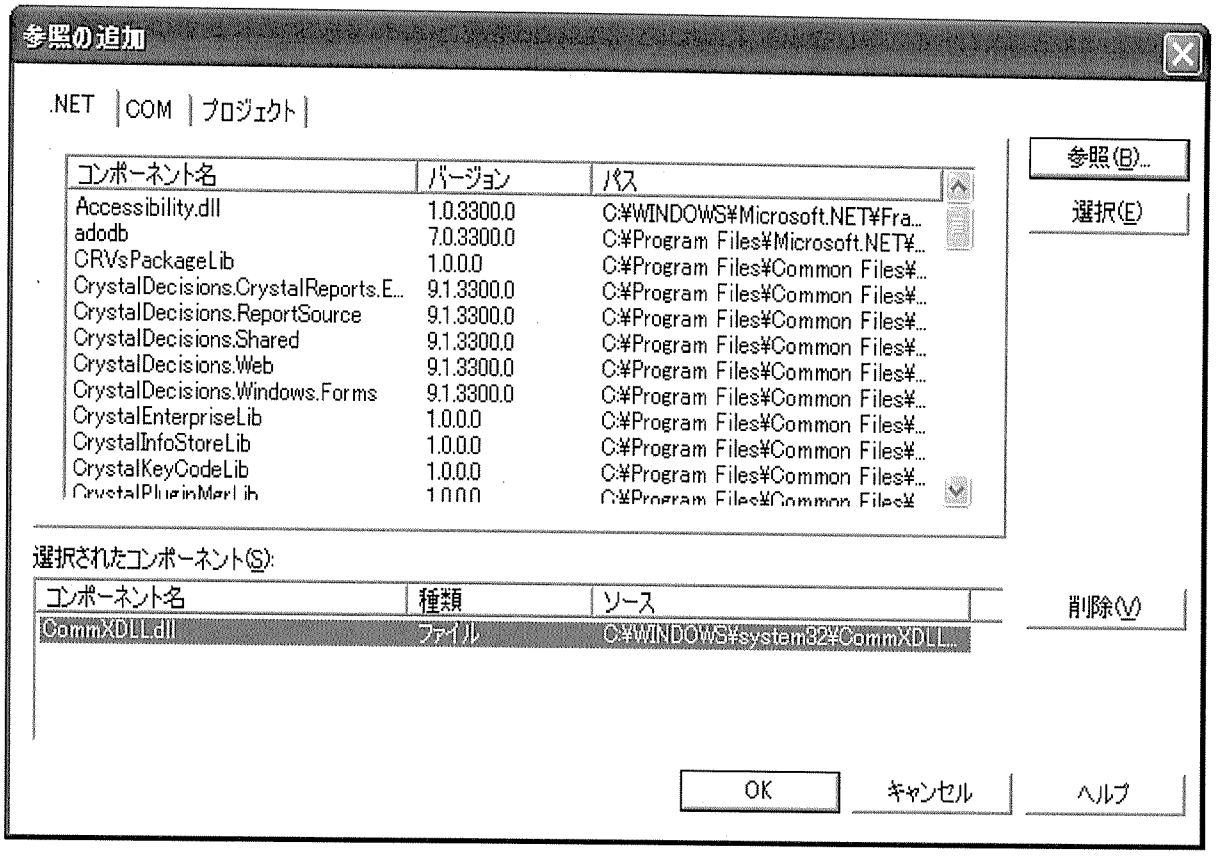
1. Microsoft Visual Studio .NETを起動します。

2. 新しいVisual Basicプロジェクトを作成します。



3. 「プロジェクト」メニューから「参照の追加」を選択し、<参照の追加>ダイアログ画面を表示します。

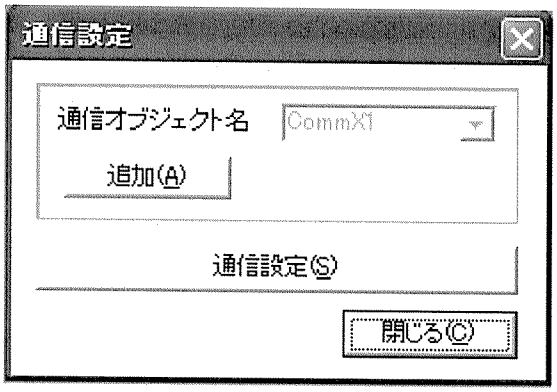
4. 「参照」ボタンをクリックし、C:¥Windows¥system32フォルダ内のCommXDLL.DLLを選択し、[OK]ボタンを押下します。



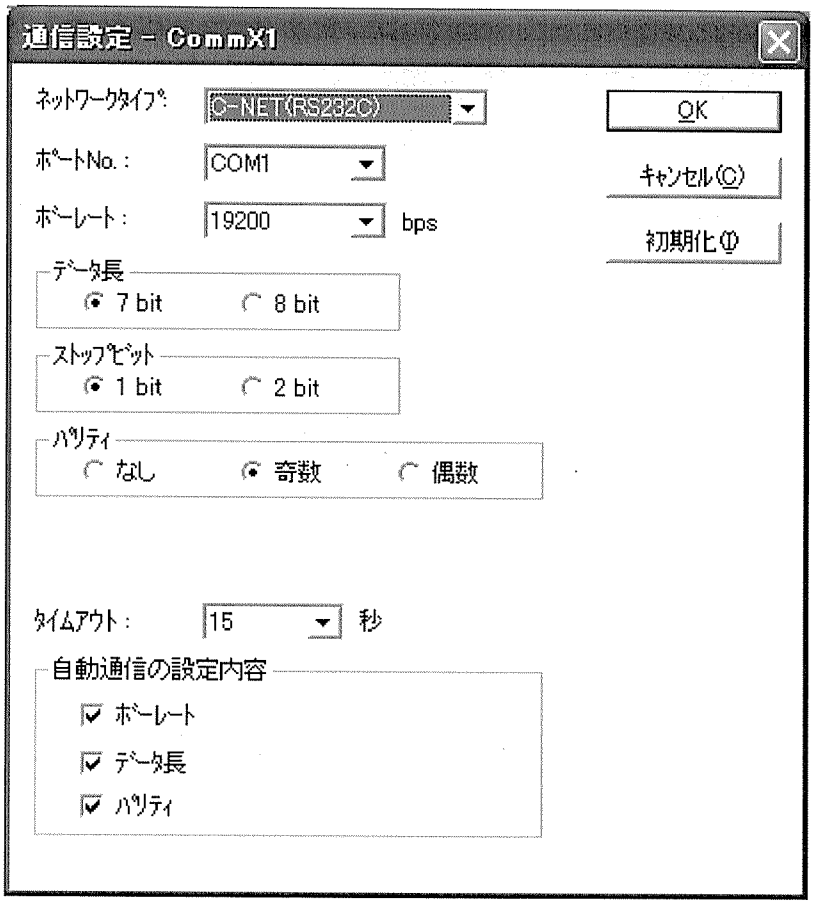
この時点でソリューションエクスプローラの参照設定フォルダ内にCommX_DLLが表示されます。

Visual Basic またはC# .NETでの通信条件の設定

“Visual Basic .NETでの初期設定(参照の追加)”または“C# .NETでの初期設定(参照の追加)”での初期設定が完了したら、通信条件の設定にうつります。
Windows「スタート」メニューの「NAIS Control」の「CommX」→「通信設定(.NET用)」をクリックします。
下記の通信設定ダイアログを表示します。



[通信設定]ボタンを押下し、下記の通信設定ダイアログを表示します。
[追加]ボタンについては、“多系列ネットワーク対応”を参照してください。
表示されるネットワークタイプについては、前回表示時、[OK]ボタンにより保存終了したネットワークタイプが表示します。



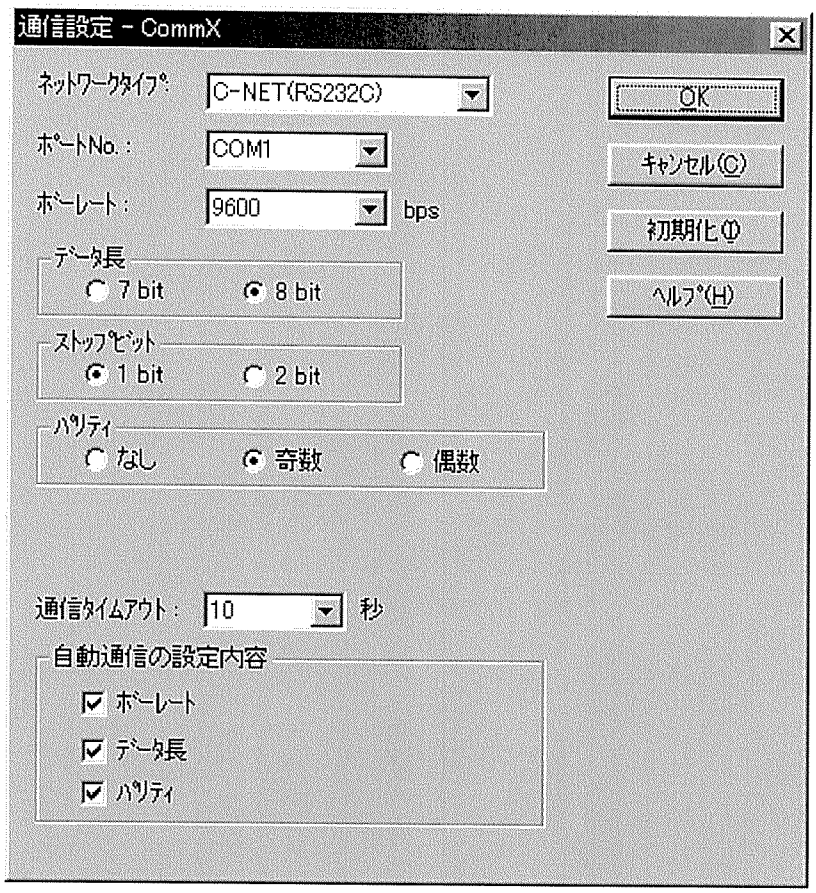
Visual Basic 6.0とは異なり、プロパティページが存在しないため、プログラムコード内で下記プロパティに値を代入し、実行環境を設定します。

通信タイプ : ネットワークタイプを下記から選択します。
NetWorkTypeプロパティに下記の値を代入します。
1:RS232C(C-NET)の場合
5:Ethernet通信の場合
6:モデム通信の場合

言語	:	表示されるエラーメッセージの言語を設定します。 Languageプロパティに下記の値を代入します。 0:日本語 1:英語 2:中国語 4:韓国語 5:スペイン語 6:イタリア語 7:ドイツ語 8:フランス語
リンクユニット局番	:	通信タイプで選択されたネットワークを使用して、リンクユニットを介して通信をする場合は、Routeプロパティに下記の値を代入します。 0:ルート指定なし 1:ルートNo.1 2:ルートNo.2 3:ルートNo.3 10:ルート指定クリア
通信プログレスバー表示	:	1パケットで納まらない通信を行う場合に通信の進行状況ダイアログを表示するかどうかを設定します。ProgressBarプロパティに下記の値を代入します。 True:表示する False:表示しない

RS232C(C－NET)通信

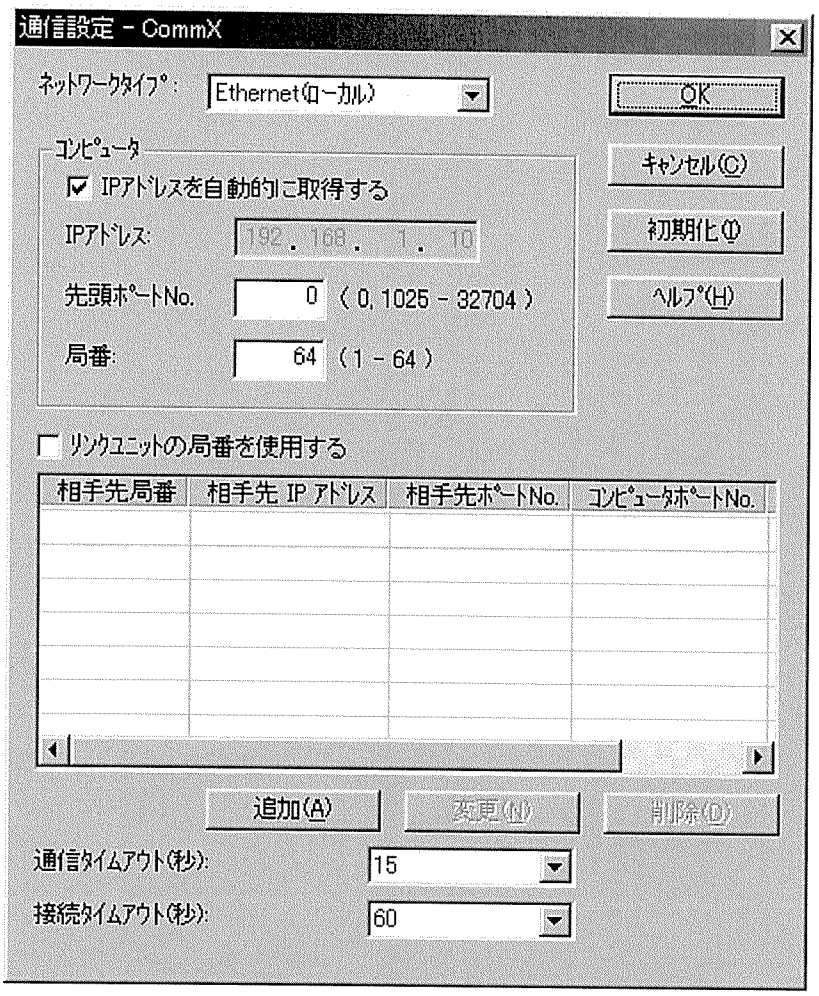
RS232C(C-NET)通信時の登録を以下に説明いたします。



ポートNo.	:	設定するポートNo.をCOM1 ～ COM5から選択してください。 ここに表示されているポートNo.は、あくまでも現状設定しているポートNo.であり、今から使用しようするポートNo.ではありません。 (COM1 ～ COM5の各ポートを使用するときの条件を設定するだけです。) 通信で使用するポートNo.はプログラム内で設定します。
ボーレート	:	1200 ～ 115200 bpsより選択してください。(初期値:9600)
データ長	:	1バイトを何ビットで転送するかを、7bit, 8bitより選択してください。 (初期値:8bit)
ストップビット	:	1bit, 2bitより選択してください。(初期値:1bit)
パリティ	:	なし, 奇数, 偶数より選択してください。(初期値:奇数)
通信タイムアウト	:	PLCとの通信タイムアウト時間(0～60秒)を設定してください。 (初期値:5秒)
自動通信の設定内容	:	PLCと通信条件が異なるときに、合致する条件を検索したい項目にチェックをつけてください。(初期値:全てにチェック) [項目全てにチェックがついていない場合は、自動的に通信条件を検索しません。]

Ethernet通信

Ethernet通信時の登録を以下に説明いたします。



まずは、システムの構成から選定します。

[1]システム構成の選定

まずは、画面中段の [リンクユニットの局番を使用する] の項目に、チェックをつけるかどうか決定してください。

■ MEWNETの各リンク経路を利用しない場合(Ethernetのみで接続する場合)

この場合はET-LANユニットが使用できます。
IPアドレスを持つ複数台の機器(PLC等)とEthernetでの接続が可能です。
上の画面中段の[リンクユニットの局番を使用する] の項目に、チェックをつけないでください。

■ MEWNETの各リンク経路を利用する場合

この場合はET-LANユニットが使用できません。
Ethernetで接続するのは、IPアドレスをもつ機器(Ethernet/RS232C変換ユニット)1台だけになります。
それ以外は、MEWNETの経路を利用して通信します。
上の画面中段の[リンクユニットの局番を使用する] の項目に、チェックをつけてください。

(詳しくは、Ethernet接続を参照してください)

[2]各項目を登録する

各項目の設定方法を以下に記します。

(1)コンピュータ側の登録

IPアドレスを自動的に取得する:
この機能はWindowsのネットワーク設定のIPアドレスの設定と同じです。
Windowsで有効なIPアドレスを取得し、そのIPアドレスを使用して各処理を実行します。

IPアドレス　：コンピュータの設定を自動的に取得し、表示します。
表示されない場合は、コントロールパネルのネットワーク設定等で、TCP/IPのプロパティを設定してください。自分で設定することも可能です。
(OSによって設定方法が異なります。詳しくは各OSマニュアル・ヘルプを参照してください。)

先頭ポートNo.　：0,1025 - 32767の範囲で設定してください。(初期値:0)
0を設定された時は、PLCのET-LANユニット接続オプションのオープン方式のFullpassiveは使用できません。
下記の欄で表示されるコンピュータNo.の先頭ポートNo.を設定します。
他のプログラムが動作する場合は、重複しないよう設定してください。

コンピュータポートNo.の考え方 (MEWNETのリンク経路を使用しない場合のみ)

■ MEWNETのリンク経路を使用しない場合

上記で設定した コンピュータ欄の先頭ポートNo.は、相手先PLC局番1と接続したときに使用されるコンピュータポートNo.となります。
相手先PLC局番が1以外の場合のコンピュータポートNo.の計算方法は、下記のようになります。
[コンピュータポートNo.= コンピュータ欄の先頭ポートNo. + 相手先PLC局番 -1]

例) コンピュータ欄の先頭ポートNo.に1025を設定した時、
・相手先PLC局番が 1の時は、使用するコンピュータポートNo.は1025
計算方法:1025 + 1 - 1

・相手先PLC局番が10の時は、使用するコンピュータポートNo.は1034
計算方法:1025 + 10 - 1

■ MEWNETのリンク経路を使用する場合

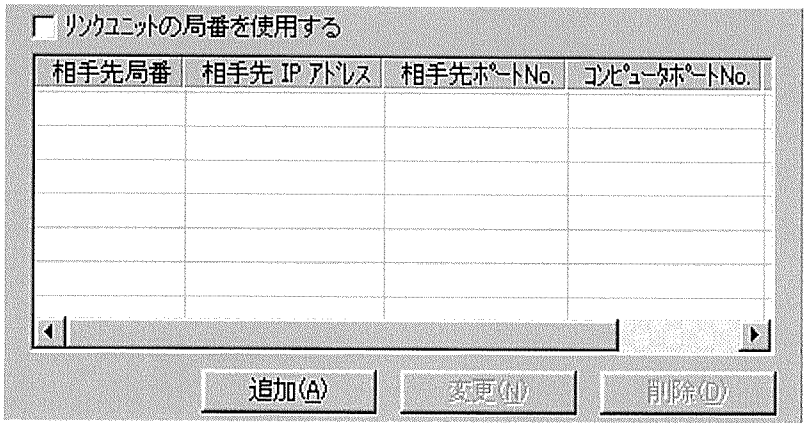
Ethernetで通信する相手先は1台のみです。
上記の考え方とは無関係です。

ET-LANユニットのオープン方式をFull passiveに設定された時は、上記のコンピュータポートNo.をラダー上で記述する必要があります。
[ET-LANユニットのオープン方式に関しては、"ET-LANユニット導入マニュアル"を参照してください。]

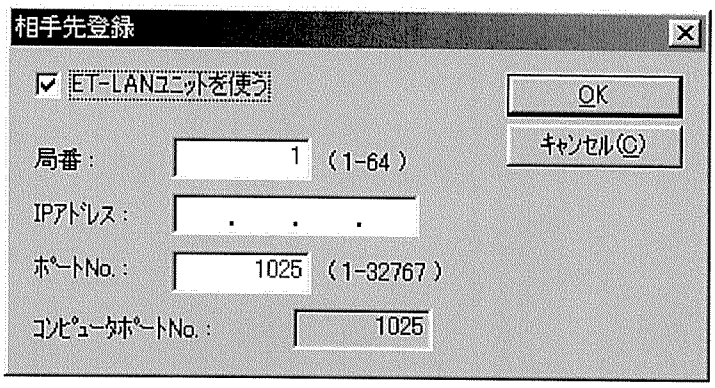
局番　：1 - 64の範囲で設定してください。(初期値:64)
但し、相手先の局番と同じにならないように設定してください。
[ET-ALNユニットを使用しない場合は、局番は関係ありません。]

リンクユニットの局番を使用する:
前述しましたので省略します。

(2)相手先PLCの登録



新規に項目を入力する場合は 追加(A) ボタンをクリックします。
既に入力済みの内容を修正する場合は 変更(N) ボタンをクリックします。
以下の画面が表示されるので必要な各項目の内容を入力してください。



追加(A) ボタンで表示した場合、局番は、自動的に使用していない最小の局番が表示されます。
登録内容は、局番によって昇順で並べ換えられます。

ET-LANユニットを使う：弊社ET-LANユニットを通して、コンピュータ（またはHUB)と接続される時はチェックをつけます。

局番　：1 - 64の範囲で設定してください。
但し、コンピュータの局番と重複しないように設定してください。
通信で使用する相手先局番は、ポートNo.としてプログラム内で設定します。

IPアドレス　：アクセス相手先のIPアドレスを設定してください。

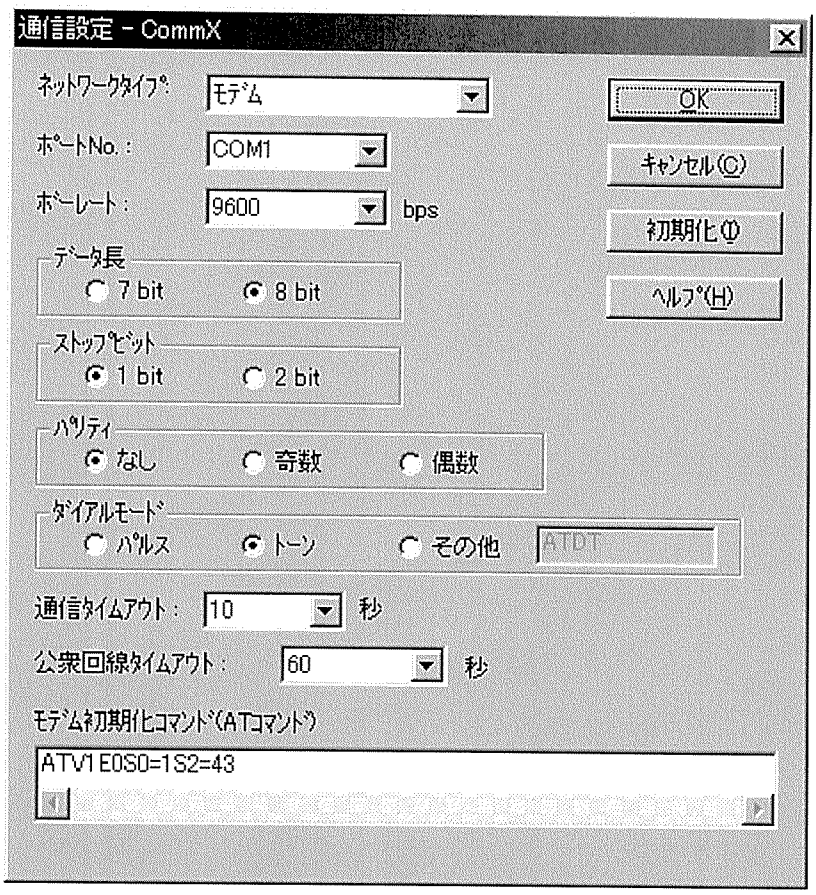
ポートNo.　：1 - 32767の範囲で設定してください。(初期値:1025)

通信タイムアウト　：コネクションが確立された後での、毎回の通信におけるタイムアウト時間を1 - 950秒の範囲で設定して下さい。(初期値:15)
[コネクションが確立されるまでの設定は次項目です。]

接続タイムアウト　：コネクションが確立されるまでのタイムアウト時間を1 - 180秒の範囲で設定してください。(初期値:60)

モデム通信

モデム通信時の登録を以下に説明いたします。



ポートNo. : 設定するポートNo.をCOM1 ～ COM5から選択してください。
ここに表示されているポートNo.は、あくまでも現状設定しているポートNo.であり、今から使用しようするポートNo.ではありません。
(COM1 ～ COM5の各ポートを使用するときの条件を設定するだけです。)
通信で使用するポートNo.はプログラム内で設定します。

ボーレート : 1200 ～ 115200 bpsより選択してください。(初期値:9600)

データ長 : 1バイトを何ビットで転送するかを、7bit, 8bitより選択してください。(初期値:8bit)

ストップビット : 1bit, 2bitより選択してください。(初期値:1bit)

パリティ : なし, 奇数, 偶数より選択してください。(初期値:奇数)

ダイアルモード : 接続する回線の種類を指定してください。(初期値:パルス)
[その他]を選択した場合は、使用するモデムのダイヤルモードを入力してください。

通信タイムアウト : PLCとの通信タイムアウト時間(0～60秒)を設定してください。
(初期値:5秒)

公衆回線タイムアウト :
回線接続時の待ち時間(1～180秒)を設定してください。(初期値:60秒)

モデム初期化コマンド(ATコマンド)
: モデムの初期化コマンド(最大入力文字数 半角80文字) を設定してください。
(初期値:ATV1E0S0=1S2=43)

参考 ATコマンドに対する解説

ATV1E0S0=1S2=43

V1 : リザルトコードを英単語で表示する。(変更可)
E0 : キャラクタエコーしない。(変更不可)
S0=1 : 呼び出し信号回数1回。(変更不可)
S2=43 : エスケープコードに"+"を使う。(変更可)

上記のATコマンドは標準的なモデムを対象にしていますが、使用するモデムとあわない場合

がありますので、使用するモデムのマニュアルを参照して上記のATコマンドの内容が同じかどうか確認してください。

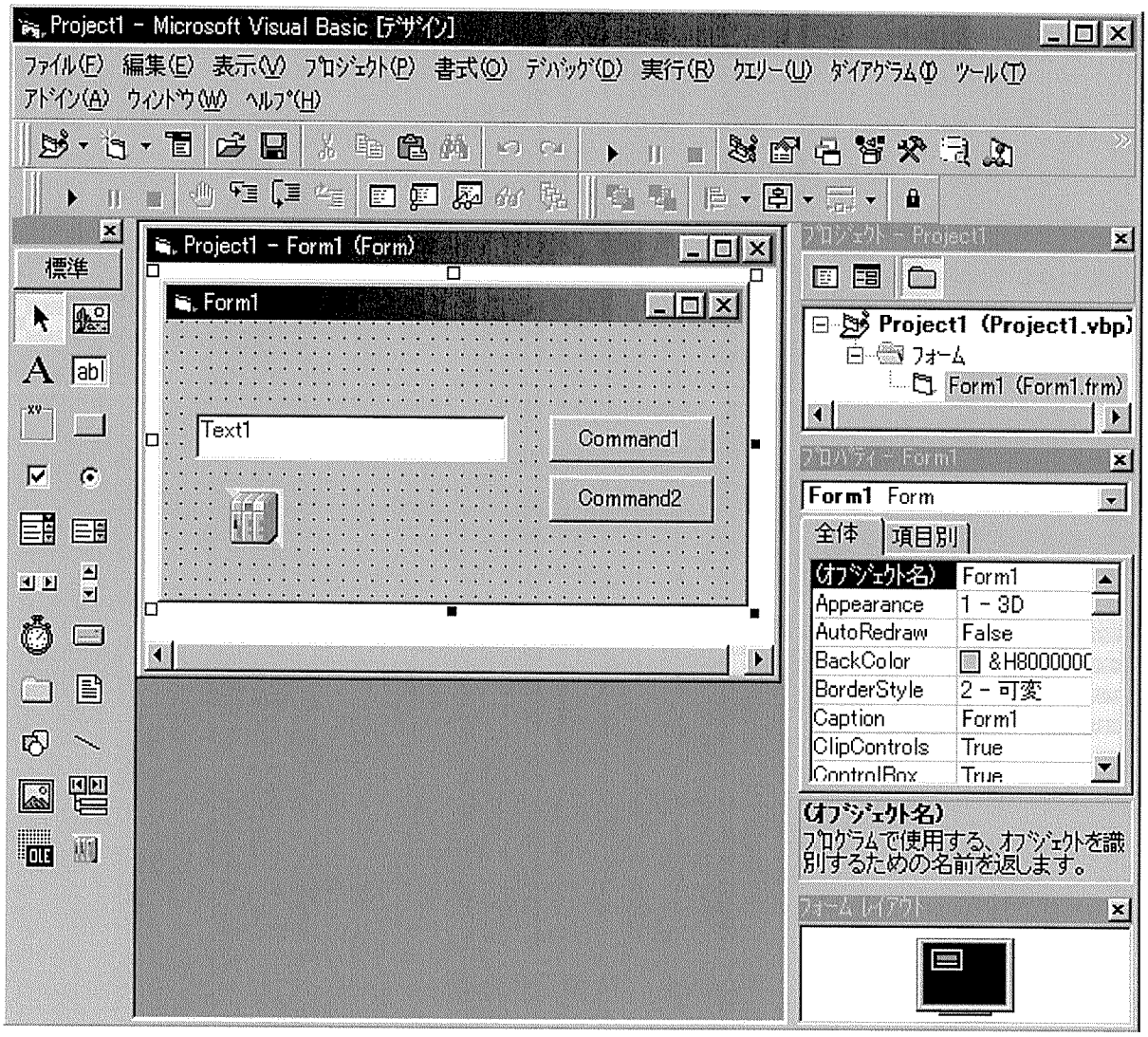
プログラムの主な流れ

この章では、いよいよプログラミングに入っていきます。
Visual Basicの基本的な使用方法に関しては、充分ご理解された上で、プログラミングしてください。すでにネットワークタイプの設定、及び通信条件の設定／登録はすんでいるものとします。まだ設定されていない方は、“プログラム作成準備”を参照して、設定してください。

ここでは、以下のような動作をするサンプルを作成します。
[1] データ読出し
[Command1]ボタンが押されたらPLCの自局(局番0)のデータレジスタ100(DT100)を読み出して、テキストボックスに表示する。
[2] データ書込み
[Command2]ボタンが押されたらテキストボックスに入力された値を、PLCの自局(局番0)のデータレジスタ100(DT100)に書き込む。

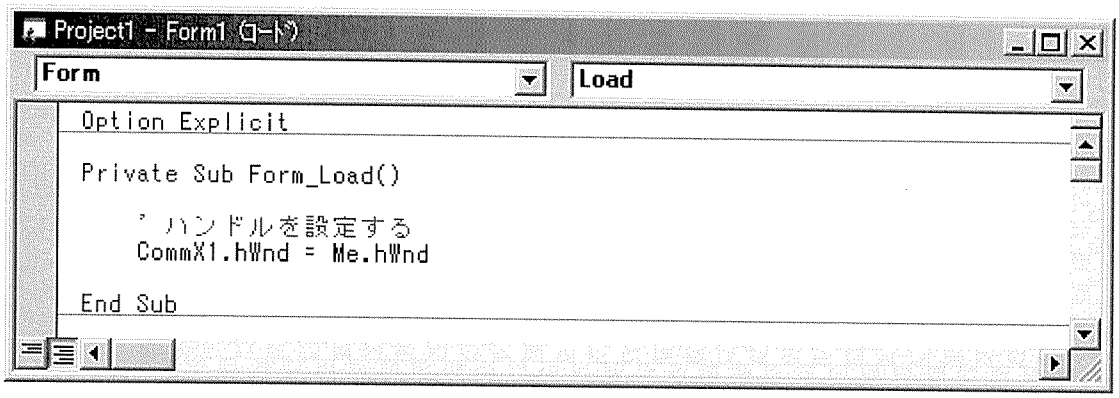
プログラムの主な流れは以下のような手順になります。

■ フォームロード時にハンドルを設定
■ [Command1]押下時に、データ読込み処理
■ [Command2]押下時に、データ書込み処理



ハンドルの設定

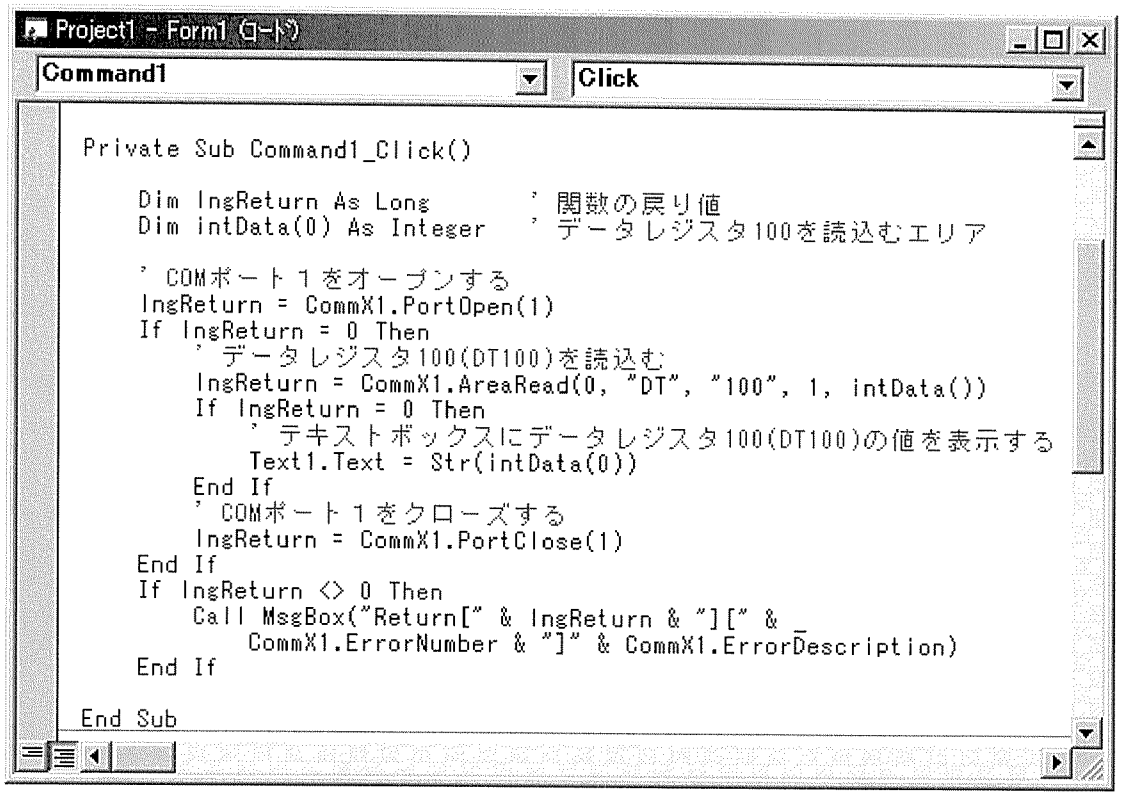
フォームロード時に通信オブジェクト(CommX1)にハンドルを渡します。



Visual Basicの場合は、上記の例のように Me.hWndを渡してください。
Visual Basic Application(Excelのマクロ等)の場合は、ハンドルは0を渡してください。
(CommX1.hWnd = 0)

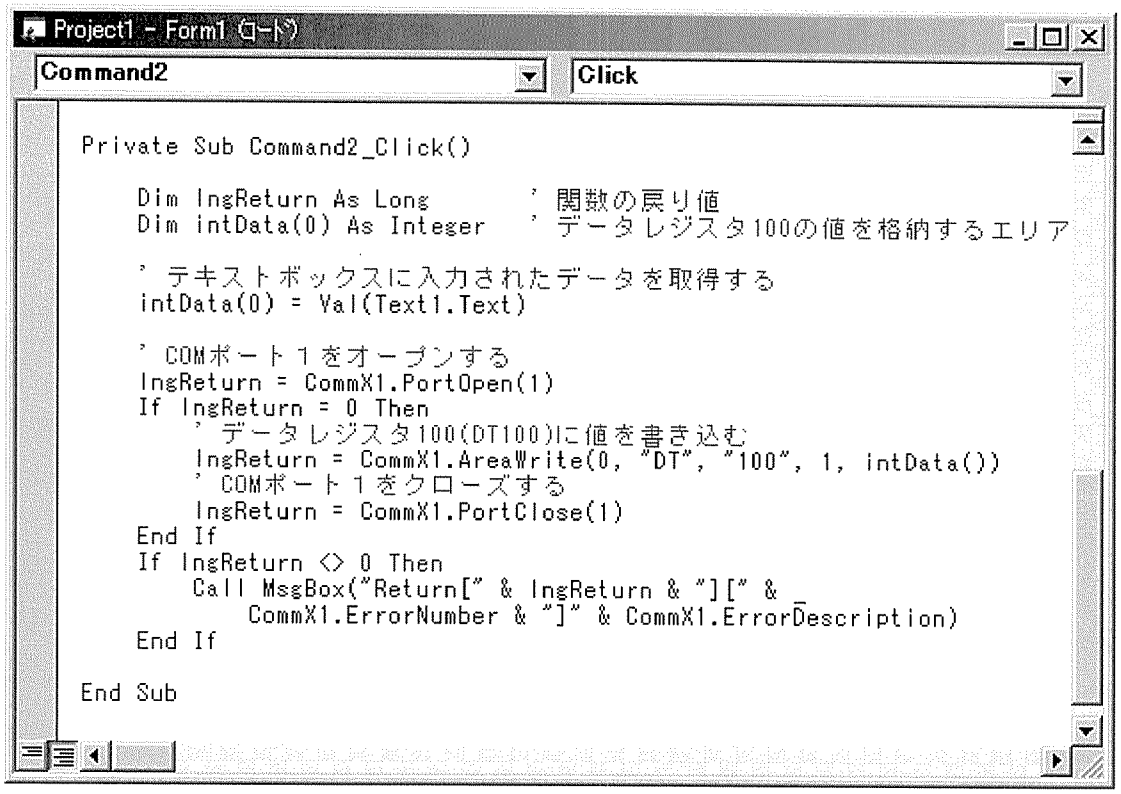
データを読み出す

コマンドボタン押下時に、通信を行います。
通信手順は、以下のような手順で行います。
1. 接続を開始(COMポート1のオープン)
2. 局番0, データレジスタ100(DT100)の値を読み出します。
3. Text1にデータレジスタ100の値を表示します。
4. 接続を終了(COMポート1のクローズ)



データを書き込む

コマンドボタン押下時に、通信を行います。
通信手順は、以下のような手順で行います。
1. 接続を開始(COMポート1のオープン)
2. テキストボックス(Text1)に入力されたデータを取得する。
3. 局番0, データレジスタ100(DT100)に値を書き込む。
4. 接続を終了(COMポート1のクローズ)



参考
本サンプルを作成したプロジェクトは、標準インストールでは以下のフォルダに格納されています。
¥Program Files¥NAiS Control¥CommX¥Sample¥Sample1
各々のサンプルコードを参考にしてください。

Visual Basic .NET

Windowsアプリケーション以外のWebアプリケーションやWebサービス等のアプリケーションは作成できません。
Visual Basic .NETでのプログラミング方法を説明します。

プログラムの主な流れ

Visual Basicの基本的な使用方法に関しては、充分ご理解された上で、プログラミングしてください。すでにネットワークタイプの設定、及び通信条件の設定／登録はすんでいるものとします。まだ設定されていない方は、“プログラム作成準備”を参照して設定してください。

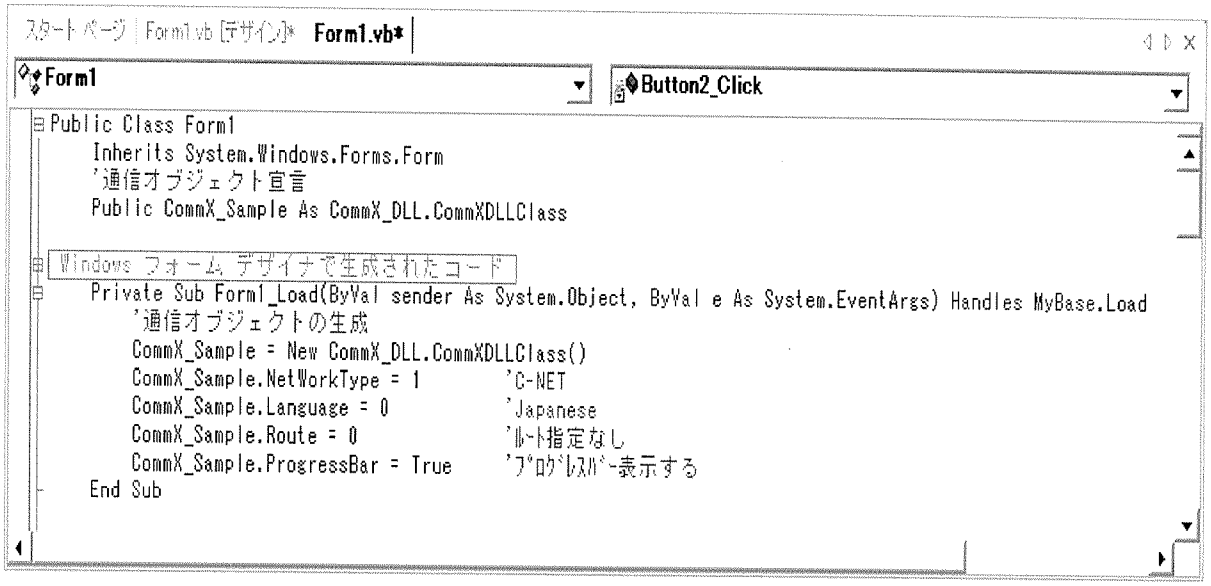
[1]データ読出し
　[DT100 Read]ボタンが押されたらPLCの自局(局番0)のデータレジスタ100(DT100)を読み出して、左横のテキストボックスに表示する。

[2]データ書込み
　[DT100 Write]ボタンが押されたら、左横のテキストボックスに入力された値をPLCの自局(局番0)のデータレジスタ100(DT100)に書き込む。
　COMポートの設定は、[COMポート]右横のテキストボックスに入力する。

プログラムの主な流れは以下のような手順になります。
■フォームロード時に初期処理
■[DT100 Read]押下時に、データ読込み処理
■[DT100 Write]押下時に、データ書込み処理

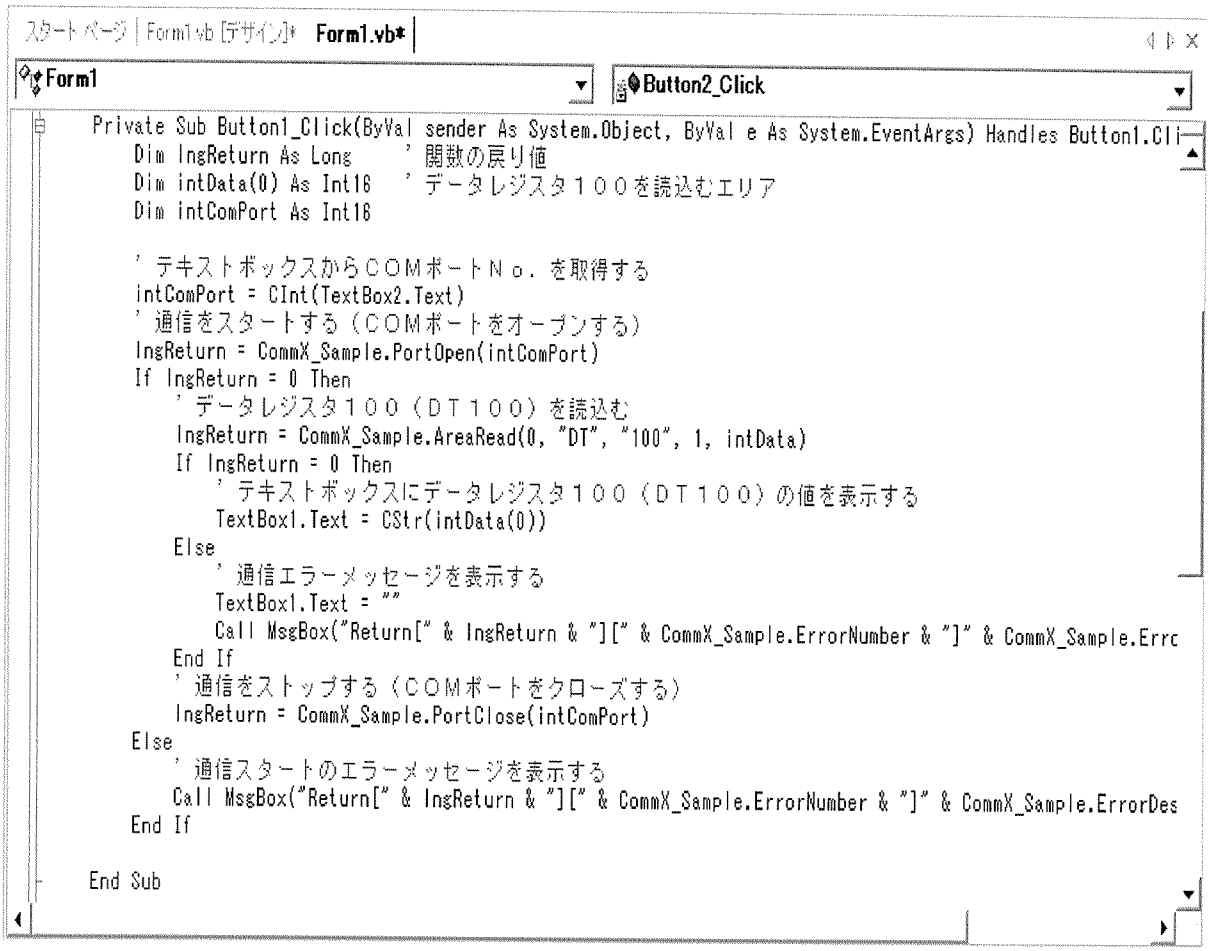
初期処理の設定

通信オブジェクト宣言やフォームロード時に通信オブジェクトの生成や各プロパティを初期設定します。



データを読み出す

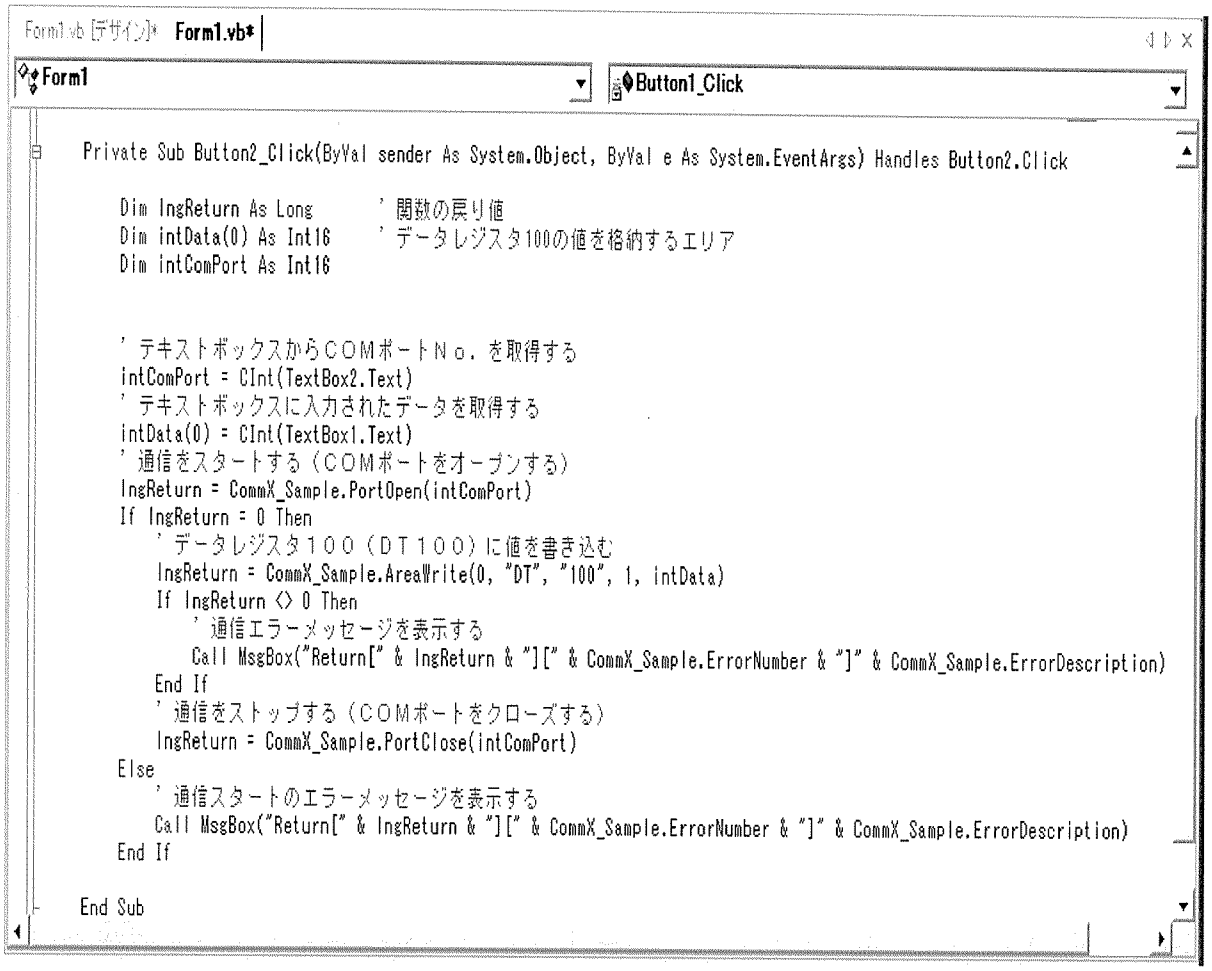
コマンドボタン押下時に通信を行います。
通信手順は以下のような手順で行います。
1. 接続を開始(指定されたCOMポートでオープン)
2. 局番0、データレジスタ100(DT100)の値を読み出します。
3. テキストボックス(TextBox1)にデータレジスタ100の値を表示します。
4. 接続を終了(COMポートのクローズ)




```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Cli
        Dim lngReturn As Long     ' 関数の戻り値
        Dim intData(0) As Int16   ' データレジスタ１００を読込むエリア
        Dim intComPort As Int16

        ' テキストボックスからＣＯＭポートＮｏ．を取得する
        intComPort = CInt(TextBox2.Text)
        ' 通信をスタートする（ＣＯＭポートをオープンする）
        lngReturn = CommX_Sample.PortOpen(intComPort)
        If lngReturn = 0 Then
            ' データレジスタ１００（ＤＴ１００）を読込む
            lngReturn = CommX_Sample.AreaRead(0, "DT", "100", 1, intData)
            If lngReturn = 0 Then
                ' テキストボックスにデータレジスタ１００（ＤＴ１００）の値を表示する
                TextBox1.Text = CStr(intData(0))
            Else
                ' 通信エラーメッセージを表示する
                TextBox1.Text = ""
                Call MsgBox("Return[" & lngReturn & "][" & CommX_Sample.ErrorNumber & "]" & CommX_Sample.Errc
            End If
            ' 通信をストップする（ＣＯＭポートをクローズする）
            lngReturn = CommX_Sample.PortClose(intComPort)
        Else
            ' 通信スタートのエラーメッセージを表示する
            Call MsgBox("Return[" & lngReturn & "][" & CommX_Sample.ErrorNumber & "]" & CommX_Sample.ErrorDes
        End If

    End Sub
```

データを書き込む

コマンドボタン押下時に通信を行います。
通信手順は以下のような手順で行います。
1. 接続を開始(指定されたCOMポートでオープン)
2. テキストボックス(TextBox1)に入力されたデータを取得します。
3. 局番0、データレジスタ100(DT100)に値を書き込みます。
4. 接続を終了(COMポートのクローズ)



参考

本サンプルを作成したプロジェクトは、標準インストールでは以下のフォルダに格納されています。
¥Program Files¥NAiS Control¥CommX¥Samples Vb .NET¥Sample1

注意

PortOpenメソッドを実行した後は、アプリケーションを終了する前にPortOpenを実行したポートNo.に対して、必ず、PortCloseメソッドを実行してください。実行しなかった場合、アプリケーションエラーが発生します。

C# .NET

Windowsアプリケーション以外のWebアプリケーションやWebサービス等のアプリケーションは作成できません。
C# .NETでのプログラミング方法を説明します。

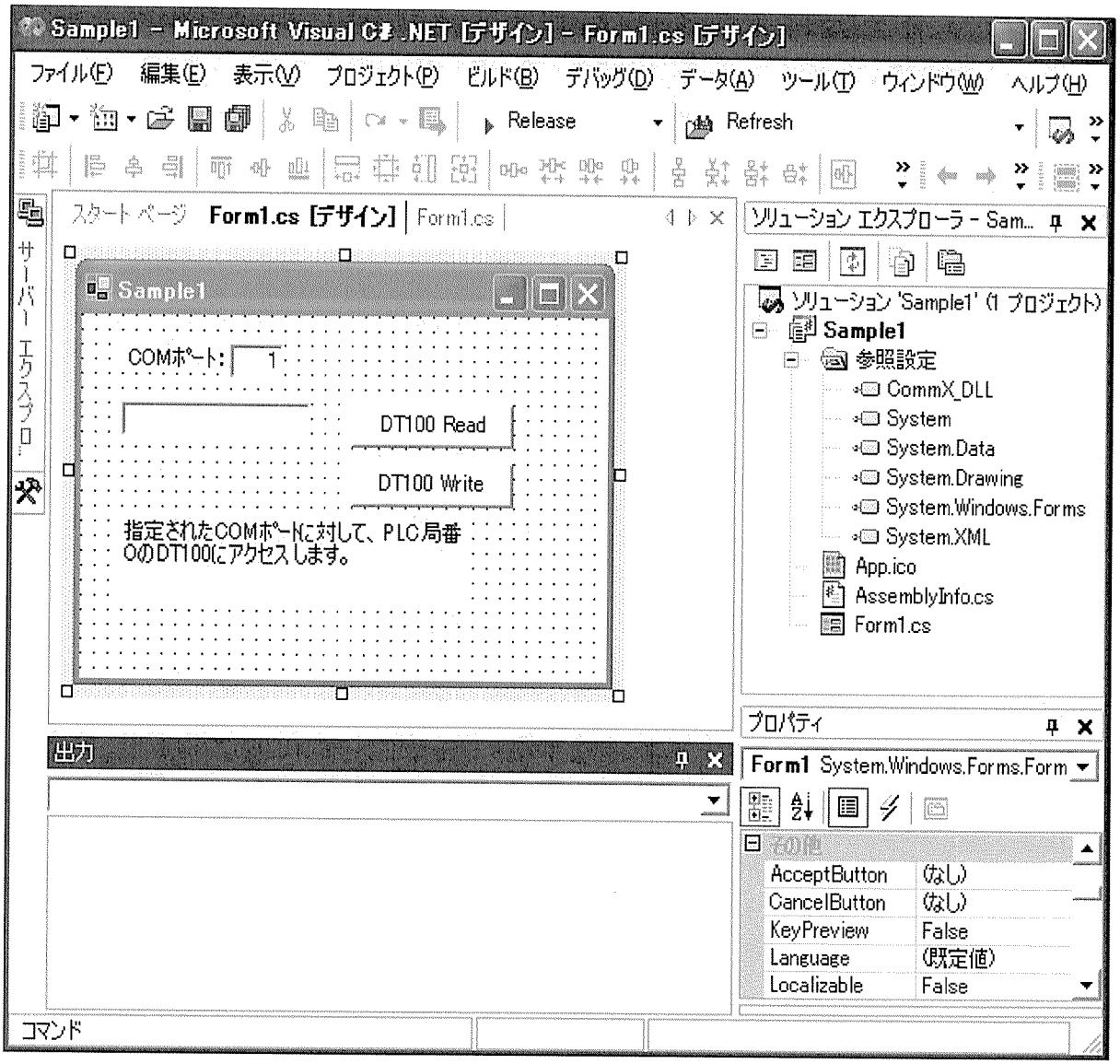
プログラムの主な流れ

C#の基本的な使用方法に関しては、充分ご理解された上で、プログラミングしてください。すでにネットワークタイプの設定、及び通信条件の設定／登録はすんでいるものとします。まだ設定されていない方は、“プログラム作成準備”を参照して設定してください。

[1]データ読出し
　[DT100 Read]ボタンが押されたらPLCの自局(局番0)のデータレジスタ100(DT100)を読み出して、左横のテキストボックスに表示する。

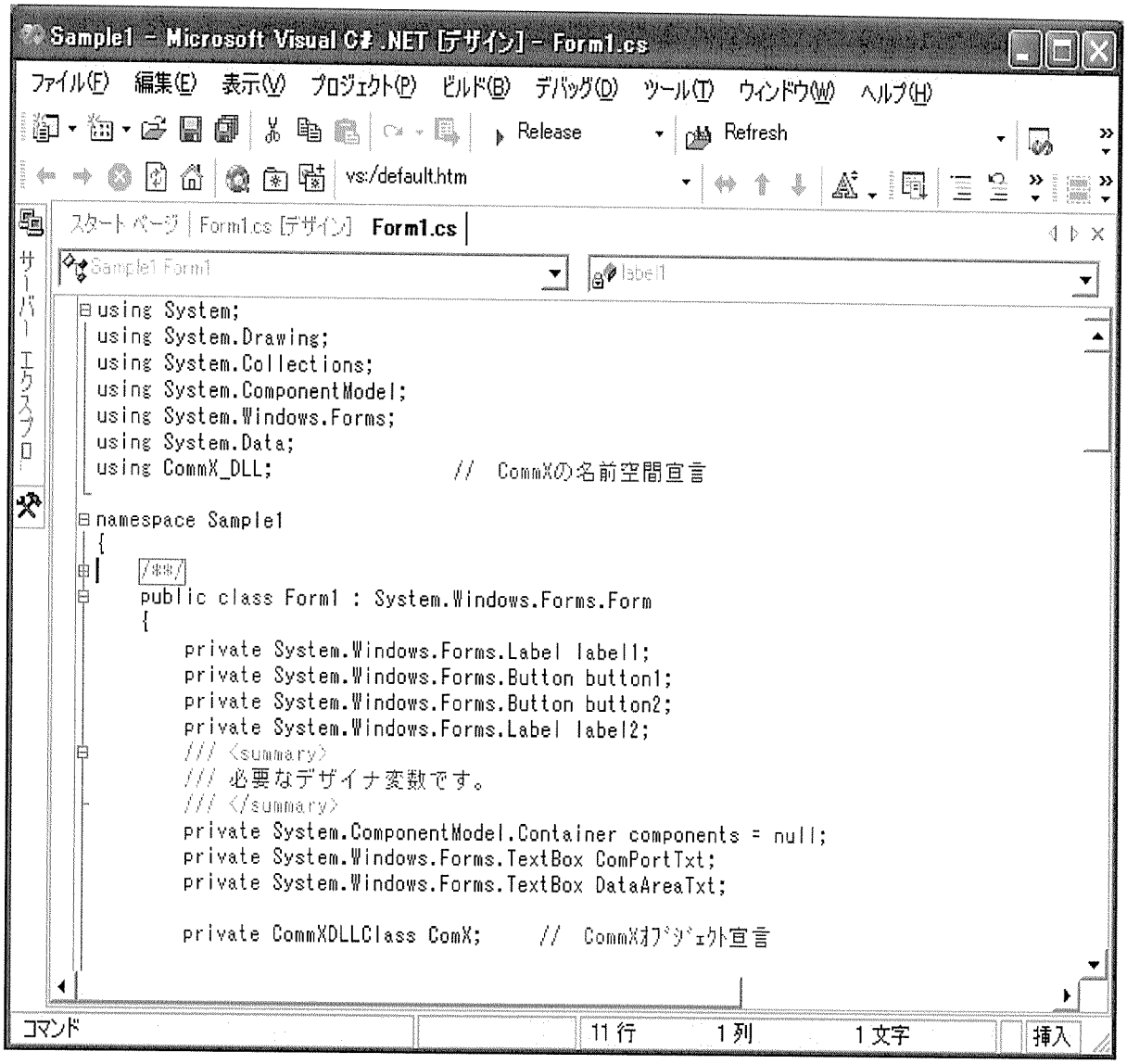
[2]データ書込み
　[DT100 Write]ボタンが押されたら、左横のテキストボックスに入力された値をPLCの自局(局番0)のデータレジスタ100(DT100)に書き込む。
　COMポートの設定は、[COMポート]右横のテキストボックスに入力する。

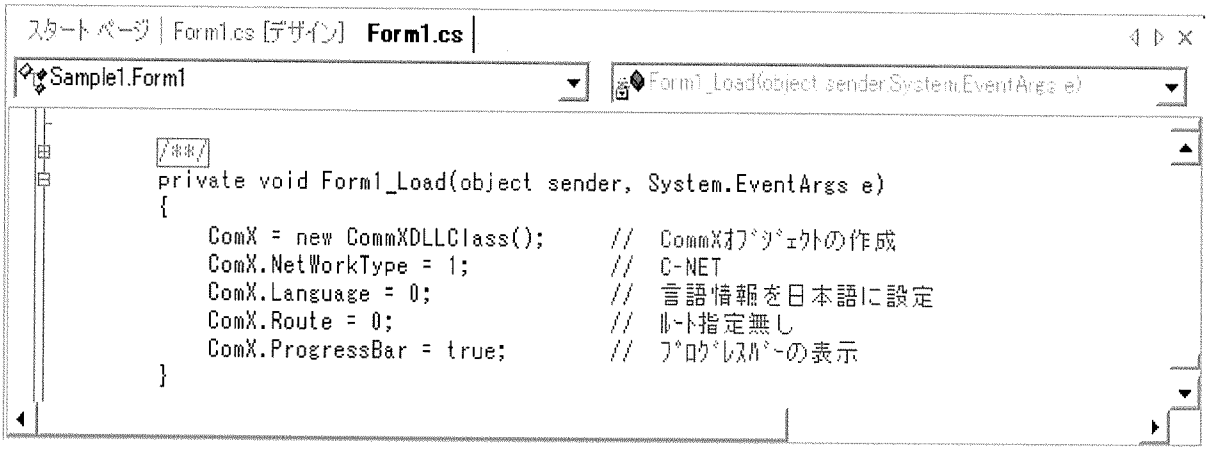
プログラムの主な流れは以下のような手順になります。
■フォームロード時に初期処理
■[DT100 Read]押下時に、データ読込み処理
■[DT100 Write]押下時に、データ書込み処理



初期処理の設定

通信オブジェクト宣言やフォームロード時に通信オブジェクトの生成や各プロパティを初期設定します。





データを読み出す

コマンドボタン押下時に通信を行います。
通信手順は以下のような手順で行います。
1. 接続を開始(指定されたCOMポートでオープン)
2. 局番0、データレジスタ100(DT100)の値を読み出します。
3. DataAreaTxtにデータレジスタ100の値を表示します。
4. 接続を終了(COMポートのクローズ)

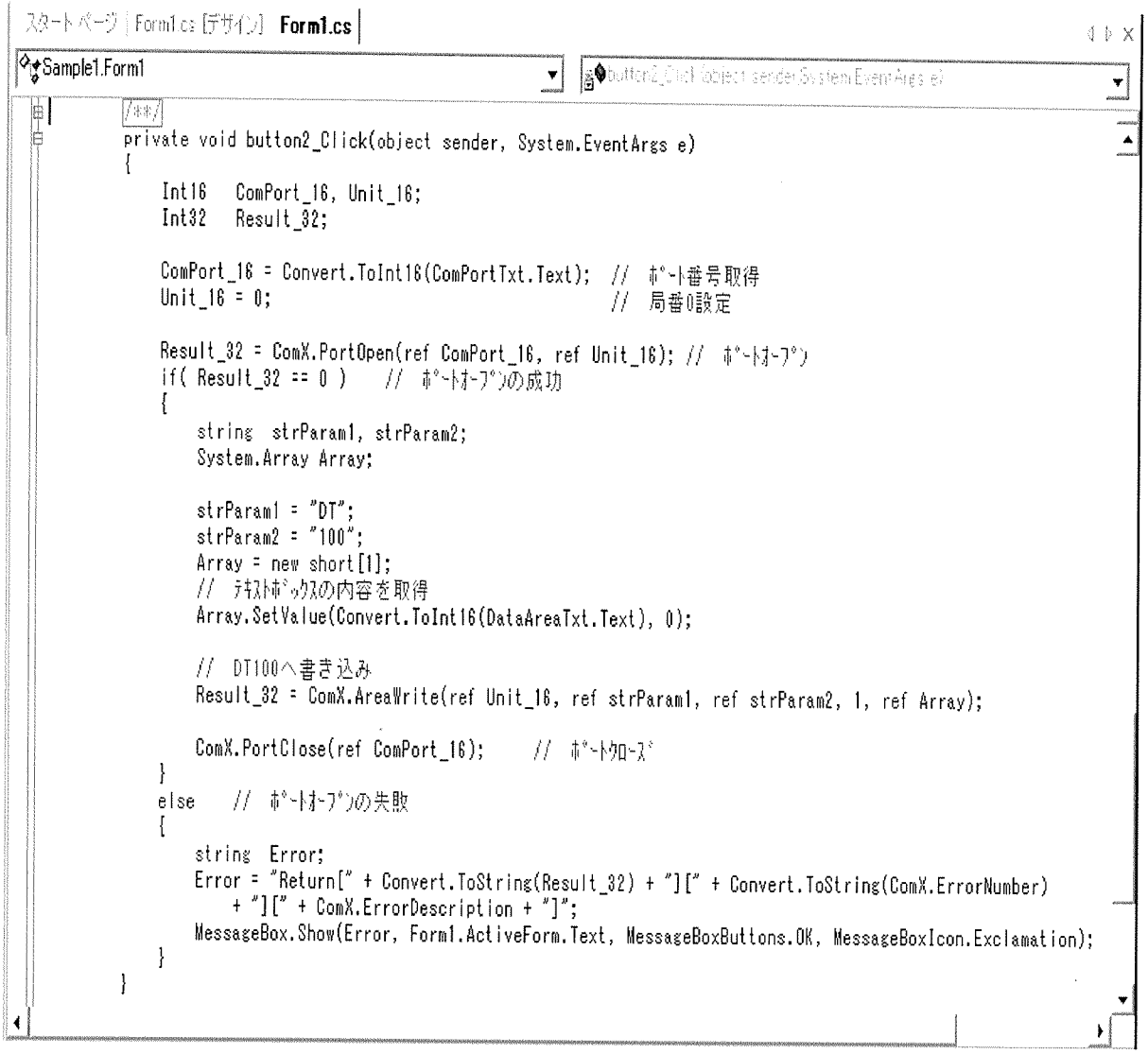
スタート ページ | Form1.cs [デザイン] **Form1.cs**

Sample1.Form1 ▼ | button1_Click(object sender, System.EventArgs e) ▼

```
        /**/
        private void button1_Click(object sender, System.EventArgs e)
        {
            Int16   ComPort_16, Unit_16;
            Int32   Result_32;

            ComPort_16 = Convert.ToInt16(ComPortTxt.Text);  //  ﾎﾟｰﾄ番号取得
            Unit_16 = 0;                                    //  局番0設定

            Result_32 = ComX.PortOpen(ref ComPort_16, ref Unit_16); //  ﾎﾟｰﾄｵｰﾌﾟﾝ
            if( Result_32 == 0 )
            {
                string  strParam1, strParam2;
                System.Array Array;

                strParam1 = "DT";
                strParam2 = "100";
                Array = new short[1];
                //  DT100読込み
                Result_32 = ComX.AreaRead(ref Unit_16, ref strParam1, ref strParam2, 1, ref Array);

                //  読込み値をﾃｷｽﾄに設定
                DataAreaTxt.Text = Convert.ToString(Array.GetValue(0));

                ComX.PortClose(ref ComPort_16);     //  ﾎﾟｰﾄｸﾛｰｽﾞ
            }
            else
            {
                string  Error;
                Error = "Return[" + Convert.ToString(Result_32) + "][" + Convert.ToString(ComX.ErrorNumber)
                    + "][" + ComX.ErrorDescription + "]";
                MessageBox.Show(Error, Form1.ActiveForm.Text, MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
        }
```

データを書き込む

コマンドボタン押下時に通信を行います。
通信手順は以下のような手順で行います。
1. 接続を開始(指定されたCOMポートでオープン)
2. テキストボックス(DataAreaTxt)に入力されたデータを取得します。
3. 局番0、データレジスタ100(DT100)に値を書き込みます。
4. 接続を終了(COMポートのクローズ)




```
        /**/
        private void button2_Click(object sender, System.EventArgs e)
        {
            Int16   ComPort_16, Unit_16;
            Int32   Result_32;

            ComPort_16 = Convert.ToInt16(ComPortTxt.Text);  //  ﾎﾟｰﾄ番号取得
            Unit_16 = 0;                                    //  局番0設定

            Result_32 = ComX.PortOpen(ref ComPort_16, ref Unit_16); //  ﾎﾟｰﾄｵｰﾌﾟﾝ
            if( Result_32 == 0 )    //  ﾎﾟｰﾄｵｰﾌﾟﾝの成功
            {
                string  strParam1, strParam2;
                System.Array Array;

                strParam1 = "DT";
                strParam2 = "100";
                Array = new short[1];
                //  ﾃｷｽﾄﾎﾞｯｸｽの内容を取得
                Array.SetValue(Convert.ToInt16(DataAreaTxt.Text), 0);

                //  DT100へ書き込み
                Result_32 = ComX.AreaWrite(ref Unit_16, ref strParam1, ref strParam2, 1, ref Array);

                ComX.PortClose(ref ComPort_16);     //  ﾎﾟｰﾄｸﾛｰｽﾞ
            }
            else    //  ﾎﾟｰﾄｵｰﾌﾟﾝの失敗
            {
                string  Error;
                Error = "Return[" + Convert.ToString(Result_32) + "][" + Convert.ToString(ComX.ErrorNumber)
                    + "][" + ComX.ErrorDescription + "]";
                MessageBox.Show(Error, Form1.ActiveForm.Text, MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
        }
```

参考

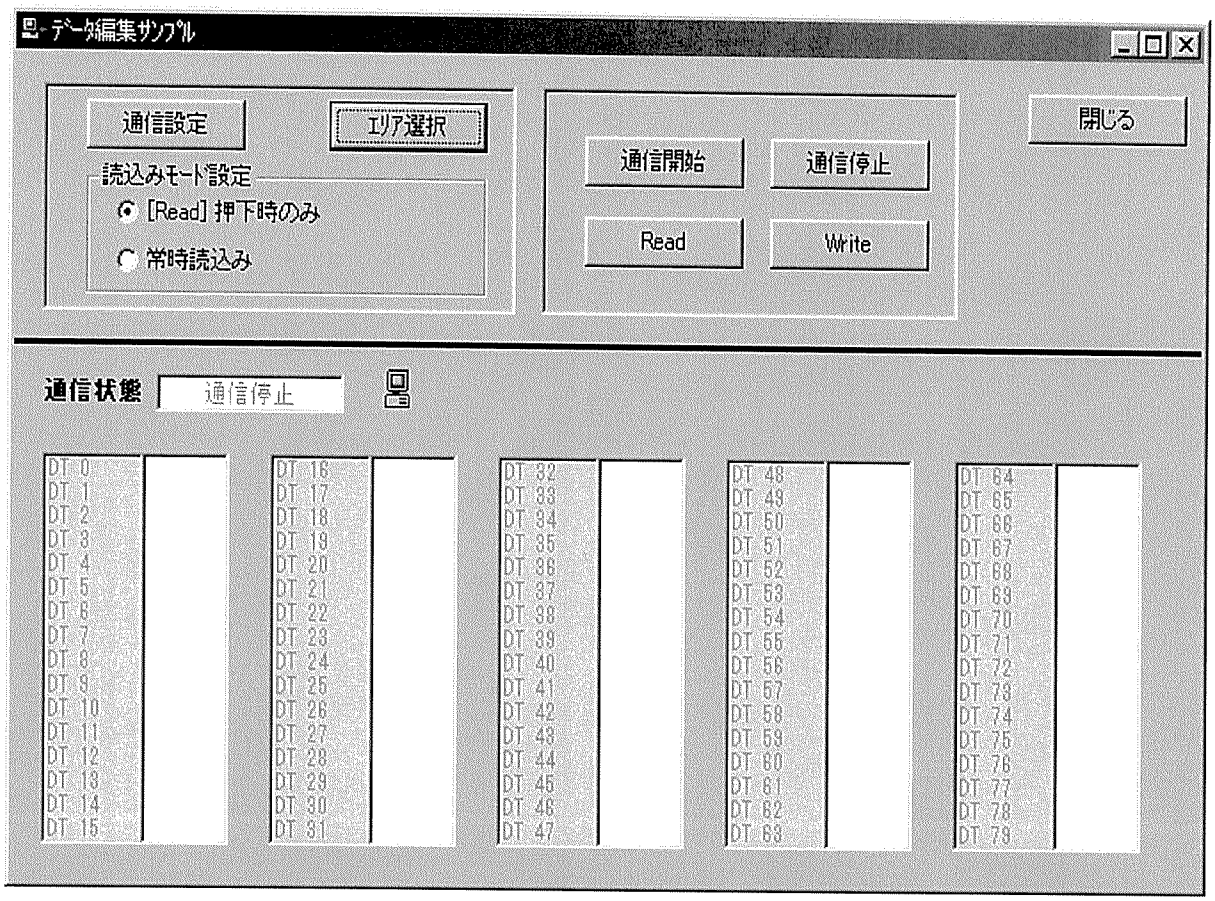
本サンプルを作成したプロジェクトは、標準インストールでは以下のフォルダに格納されています。
¥Program Files¥NAiS Control¥CommX¥Samples C# .NET¥Sample1

注意

PortOpenメソッドを実行した後は、アプリケーションを終了する前にPortOpenを実行したポートNo.に対して、必ず、PortCloseメソッドを実行してください。実行しなかった場合、アプリケーションエラーが発生します。

サンプルプログラムを利用する

本ソフトウェアには、サンプルプログラムを添付しております。
標準インストール時には、[スタート]メニューをクリックして、[プログラム] => [NAiS Control] => [CommX] をポイントし、[Monitorサンプル]を起動してください。
以下の画面が表示されます。

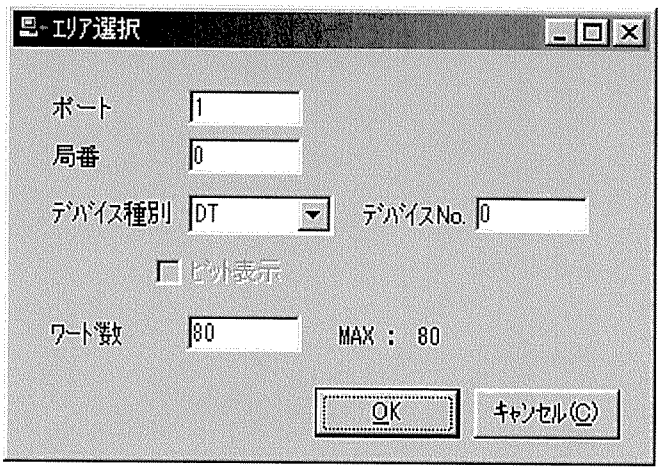


この[Monitorサンプル]は、本ソフトウェアを利用してVisual Basicで作成されております。

利用手順を簡単に説明します。

1. [通信設定]ボタンを押下し、ご利用の通信環境を設定してください。
   詳細は、各種通信設定の登録を参照してください。
   （但し、このサンプルはモデム通信には対応しておりません。）

2. [エリア選択] ボタンを押下し、読込みたいデバイスやNo.を選択します。



・ ワードなら80ワード、ビット表示するなら80点までが連続で読込めます。

3. 読込みモードを設定してください。
   ・[Read]押下時のみ　：　画面上の[Read]ボタンを押下したときだけ読み込みを行い、表示を更新します。
   ・常時読込み　：　100ms毎に読み込みを行い、表示を更新します。

4. [通信開始]ボタンを押しますと、接続を開始します。

5. データの更新は、上記3.の読込みモードに従います。

6. 表示データの値を書き込みたいときには、変更したい値が表示されている欄をクリックして[Write]ボタンを押すか、変更したい値が表示されている欄をダブルクリックしてください。

7. 通信を終了する場合は、[通信停止]ボタンを押してください。

参考
本サンプルを作成したプロジェクトは、標準インストールでは以下のフォルダに格納されています。
¥Program Files¥NAiS Control¥CommX¥Sample¥Monitor
各々のサンプルコードを参考にしてください。

当社の他のツールソフトを利用する

お客様で作成されたアプリケーションが正しく動作するかどうかは、当社の以下のツールを利用して確認することも可能です。

前述したように、本ソフトウェアを利用して作成したアプリケーションは、以下のソフトウェアと同時通信が可能となっております。

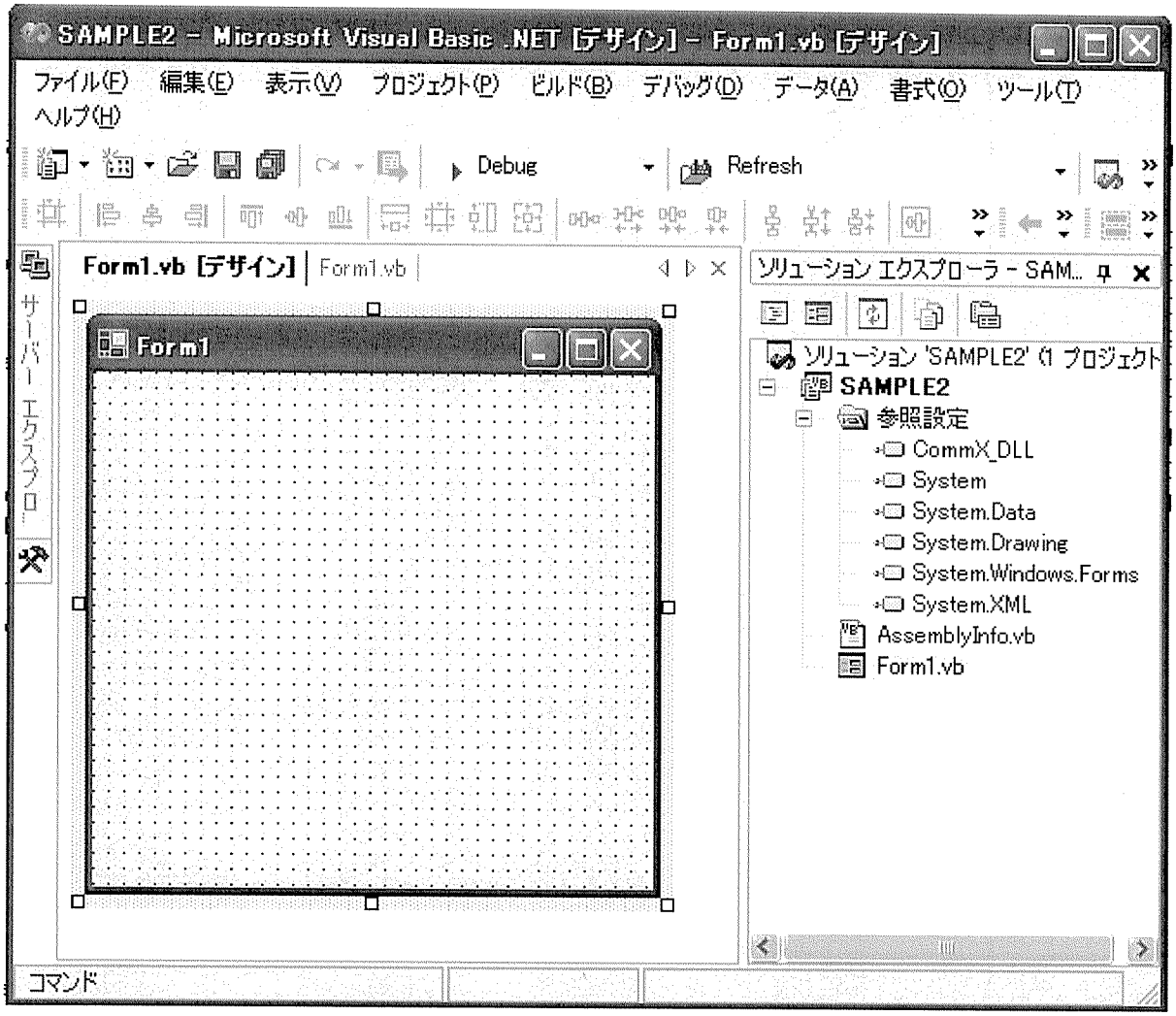
- PLC用プログラミングツールソフト Control FPWIN GR Ver.1.1以上
- PLC用プログラミングツールソフト Control FPWIN Pro Ver.4.0以上
- 表示器用画面作成ツールソフト Terminal GTWIN Ver.1.0以上
- 稼働管理ソフトPCWAY Ver.2.1以上

データを書き込むようなアプリケーションを作成された場合は、上記の各ツールのデータモニタの機能を使用して確認してください。
また、データを読込むようなアプリケーションを作成された場合は、上記の各ツールのデータ書込みの機能を使用して確認してください。

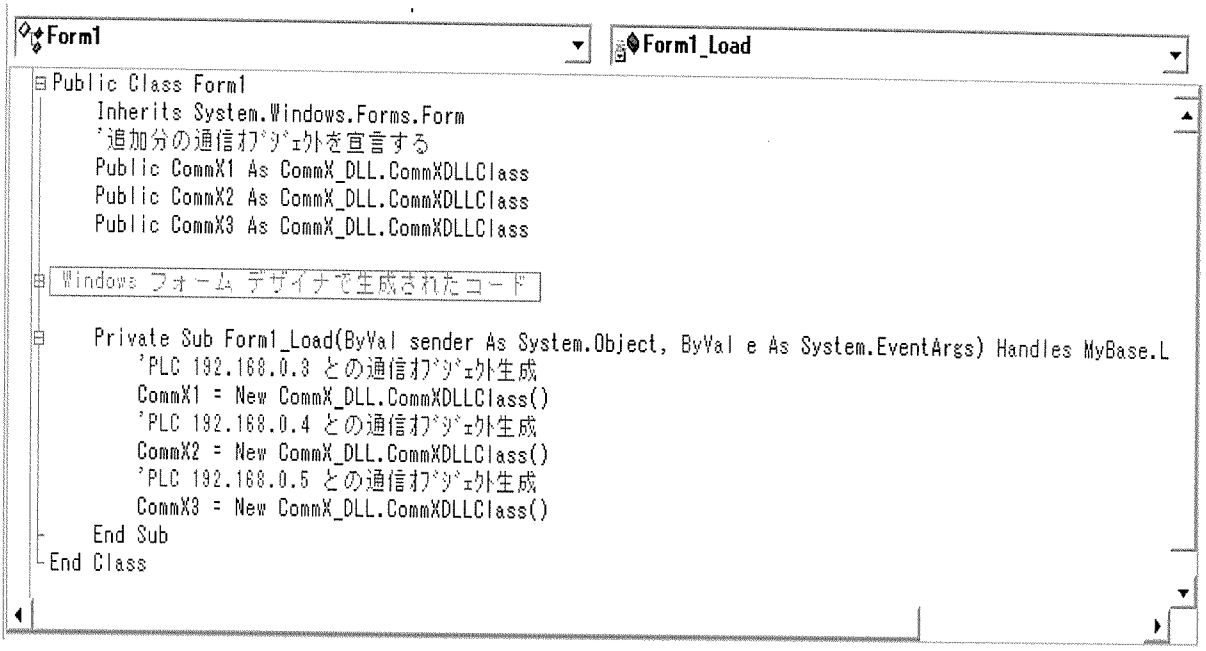
各ツールの使用方法に関しては、各ツールのヘルプをご参照ください。

プログラム作成事例

“設定方法”にて、設定した内容にそってプログラムを作成します。
“Visual Basic .NETでの初期設定(参照の追加)”を参考にして、下記のようにVB .NETの新しいプロジェクトを作成します。



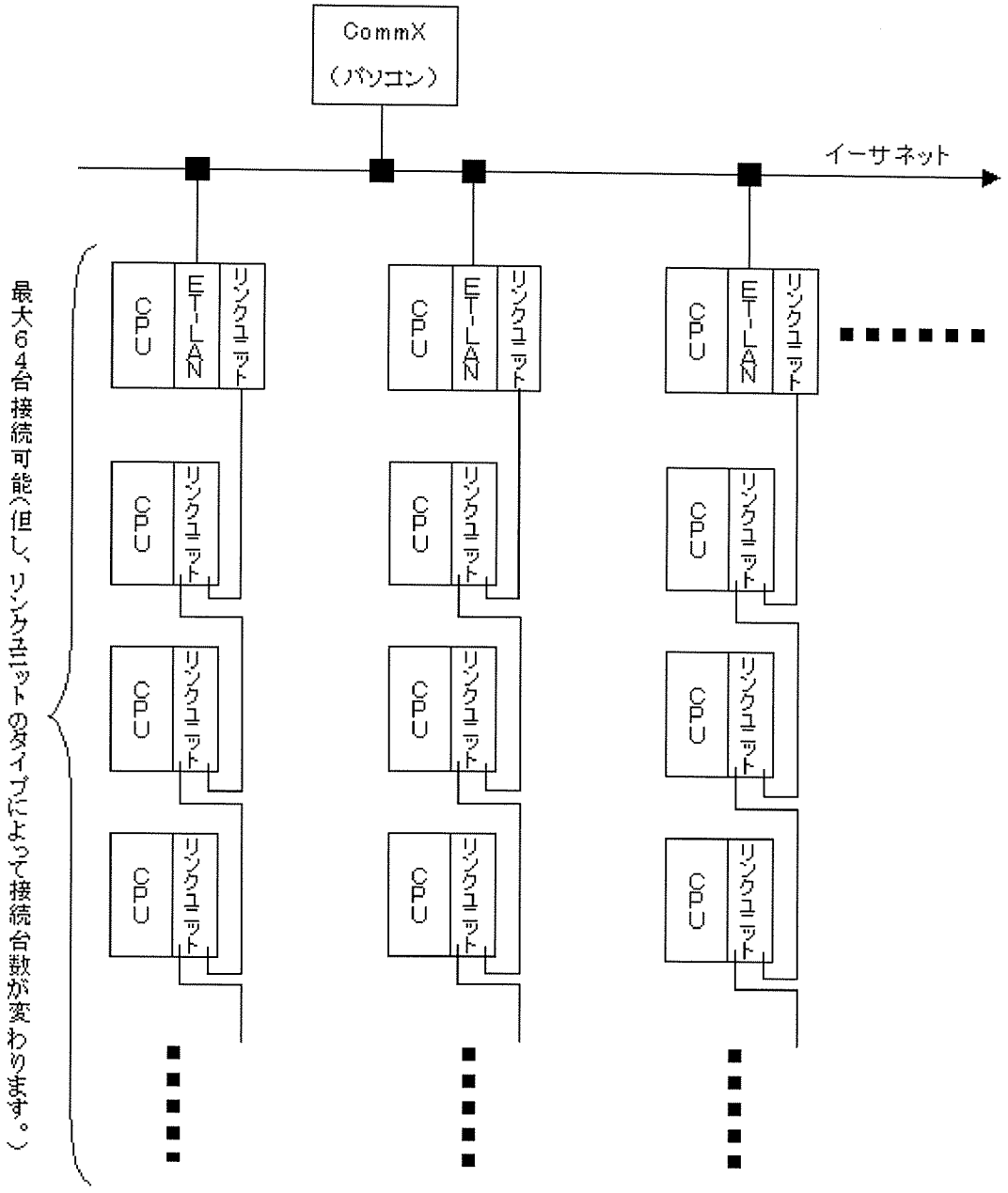
■フォームロード時の初期プログラム



CommX1、CommX2、CommX3の通信オブジェクトを利用して、それぞれのネットワーク系列へアクセスします。
実際のアクセス方法については、“Visual Basic .NET”を参照してください。

対応ネットワーク構成

下記のようなネットワーク構成時に対応します。
下記のすべてのPLCのデバイスへ読書きすることが可能です。

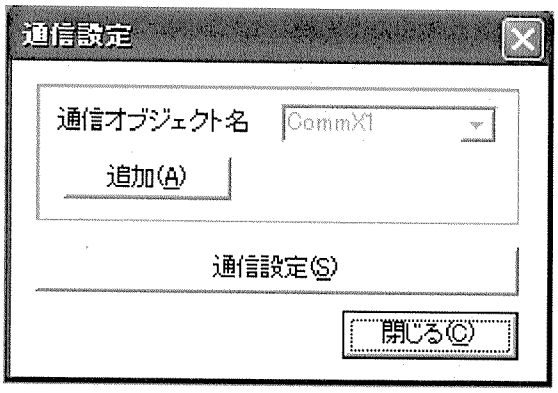


※ET-LANユニットからリンクユニットを経由していないとき、または、そのようなPLCのネットワークが1つのときは、CommXの標準設定でプログラム作成する事ができます。

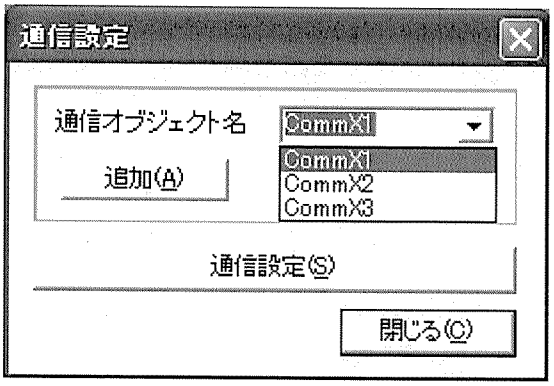
注意
リンクユニットは、MEWNET-W/P/Hの3種類のリンクユニットが使用できます。

設定方法

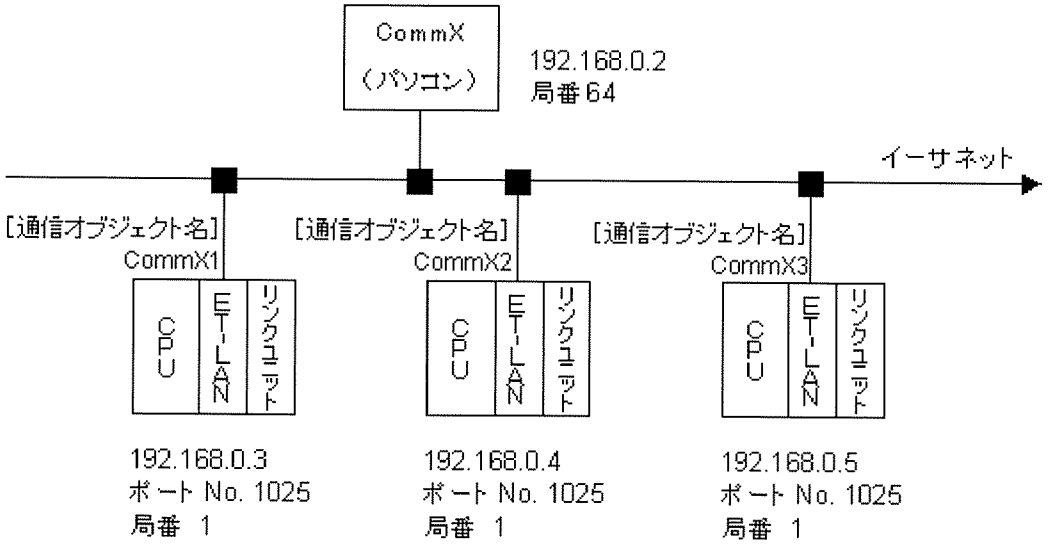
“対応ネットワーク構成”の図のように、ET-LANユニットを3台使用する場合を例にあげて説明します。
まず、通信条件を設定します。
Windows「スタート」メニューの「NAIS Control」の「CommX」-「通信設定(.NET用)」をクリックします。下記の通信設定ダイアログを表示します。



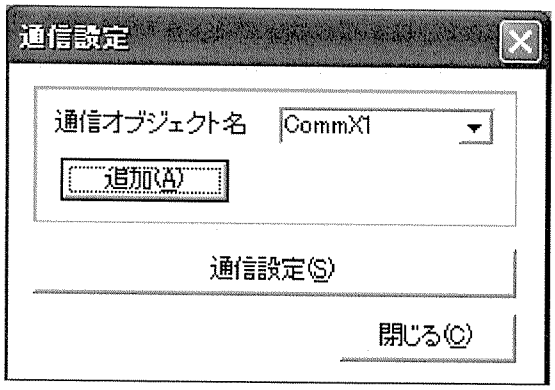
[追加]ボタンを2回クリックします。
下記のように、通信オブジェクト名のコンボボックスが使用可能となって、CommX2とCommX3が追加されます。



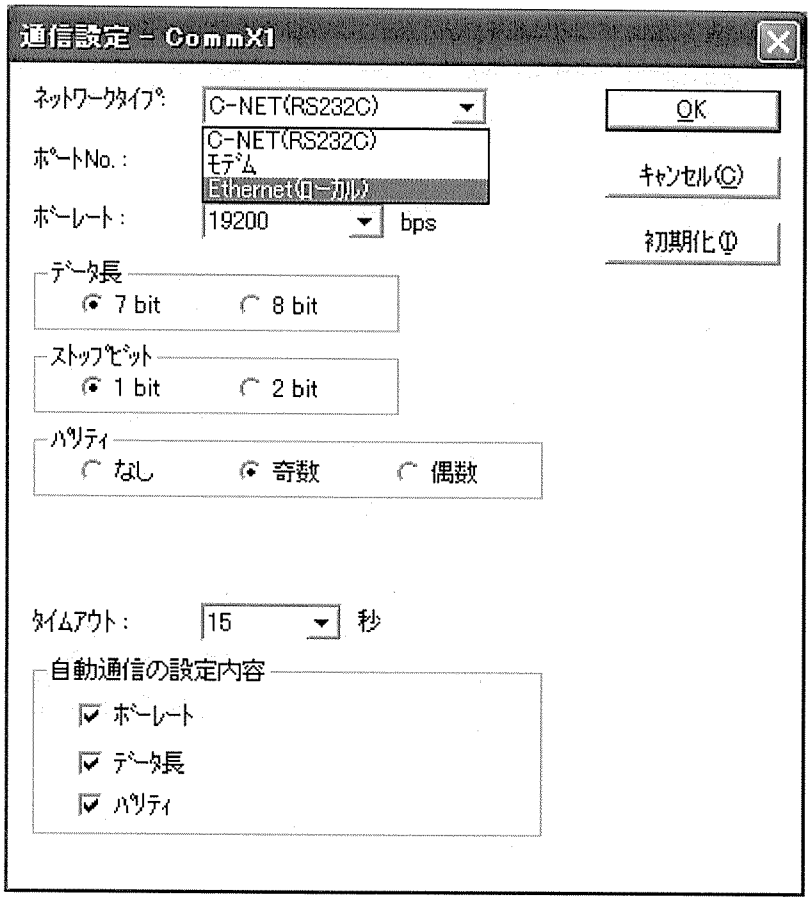
例. 各イーサネットの設定が以下であった時



(1)IPアドレス192.168.0.3のPLC設定
「通信設定(.NET用)」を開いて、下記のように[通信オブジェクト名]コンボボックスを[CommX1]に選択します。



[通信設定]ボタンをクリックして、下記のように[ネットワークタイプ]コンボボックスを[Ethernet(ローカル)]を選択します。



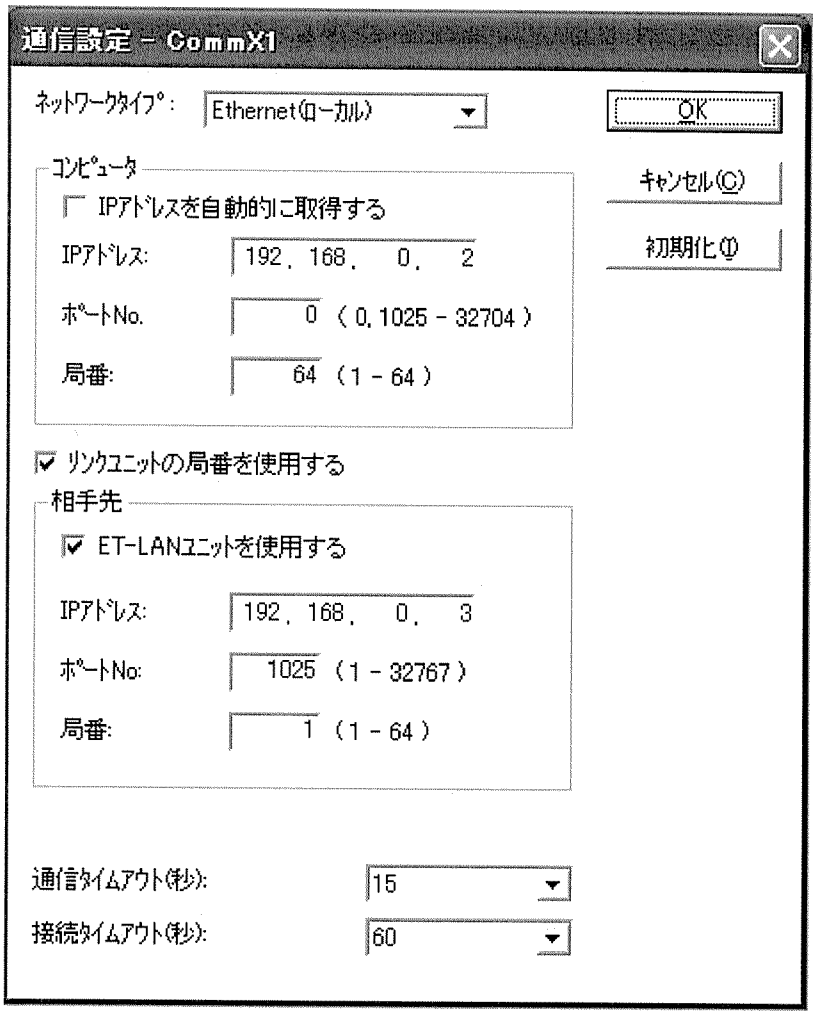
通信設定ダイアログを下記のように設定します。

●パソコン(CommX)側

IPアドレス	:	192.168.0.2
ポートNo.	:	0 (Windowsが未使用ポートNo.を自動設定)
局番	:	64

●PLC側

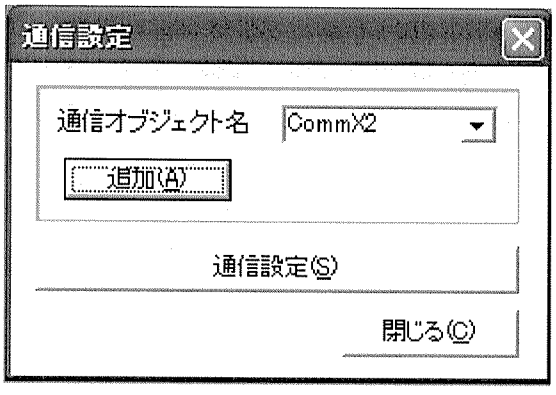
IPアドレス	:	192.168.0.3
ポートNo.	:	1025
局番	:	1



データを入力後、[OK]ボタンを押下して入力データを登録します。

(2)IPアドレス192.168.0.4のPLCの設定

「通信設定(.NET用)」を開いて、下記のように[通信オブジェクト名]コンボボックスを[CommX2]に選択します。



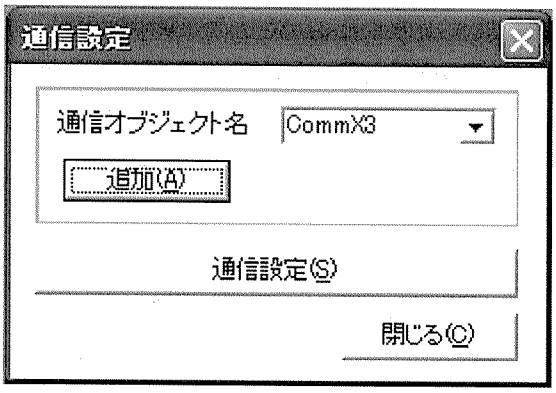
[通信設定]ボタンをクリックして、[ネットワークタイプ]コンボボックスを[Ethernet(ローカル)]を選択します。
通信設定ダイアログ内に下記データを(1)と同じように設定します。

●パソコン(CommX)側
- IPアドレス : 192.168.0.2
- ポートNo. : 0 (Windowsが未使用ポートNo.を自動設定)
- 局番 : 64

●PLC側
- IPアドレス : 192.168.0.4
- ポートNo. : 1025

局番　　：1
データを入力後、[OK]ボタンを押下して入力データを登録します。

(3)IPアドレス192.168.0.5のPLCの設定
「通信設定(.NET用)」を開いて、下記のように[通信オブジェクト名]コンボボックスを[CommX3]に選択します。



[通信設定]ボタンをクリックして、[ネットワークタイプ]コンボボックスを[Ethernet(ローカル)]を選択します。
通信設定ダイアログ内に下記データを(1)と同じように設定します。
●パソコン(CommX)側
　IPアドレス　　：192.168.0.2
　ポートNo.　　：0 (Windowsが未使用ポートNo.を自動設定)
　局番　　：64
●PLC側
　IPアドレス　　：192.168.0.5
　ポートNo.　　：1025
　局番　　：2
データを入力後、[OK]ボタンを押下して入力データを登録します。

プロパティー覧

■一般仕様

hWind

Port

Connection

Route

NetworkType

Language

ProgressBar

Bank

TaskName

LCCommand

ErrorDescription

ErrorNumber

ThreadNumber

■モデム仕様

TelephoneNumber

ReceiveInterval

hWndプロパティ

ウィンドウを識別するハンドルを設定します。値の取得も可能です。
デザイン時には指定できません。

構文
object.hWnd[= *hWnd*]

hWnd プロパティの構文は、次の指定項目から構成されます。

指定項目内容

object	オブジェクトの名前を指定します。必ず指定します。
hWnd	アクティブになっているウィンドウハンドルを指定します。

解説
プログレスバー表示や通信関連のメッセージ表示を行うときに使用します。アクティブになっているウィンドウハンドルを設定してください。(既定値 0)
必ず最初に必要です。

使用例

```
'ハンドルを指定する
object.hWnd = Me.hWnd
'/特に問題ない限り、このサンプルのままでOKです/
```

Portプロパティ

複数のポートがオープンされている場合、今から通信するポートNo.を指定します。値の取得も可能です。
デザイン時には使用できません。

構文
object.Port[= *Integer*]

Port プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。
Integer	使用するポートNo.を指定します。省略可能です。

設定値
引数 Integer の設定値は次のとおりです。

ネットワーク	値の範囲	内容
RS232C(C-NET)	1～5	通信するCOMポートを指定します。 複数のCOMポートをオープンしている場合は、通信するCOMポートを変更する度に、この指定が必要です。(既定値 1)
Ethernet	1～64	■ET-LANユニットを使用する場合 通信する相手先局番をポートNo.として指定します。 複数台のET-LANユニットと通信する場合などは、通信する相手先局を変更する度に、この指定が必要です。(既定値 1) ■ET-LANユニットを使用せず(RS232C <-> Ethernet変換ユニットを使用して)、ルート指定でリンクユニットの局番を使用する場合 ポートNo.として必ず1を指定します。(既定値 1)
モデム	1～5	通信するCOMポートを指定します。 複数のCOMポートを使用してモデムと接続する場合は、通信するCOMポートを変更する度に、この指定が必要です。(既定値 1)

解説
複数のポートをオープンしているときに、これからアクセスするポートNo.を指定する場合に使用します。
先にPortOpenメソッドが発行されている必要があります。

使用例
```
'ポート1を設定する
object.Port = 1
```

Connectionプロパティ

各ポートの通信状態(オープン状態)を取得します。
デザイン時には使用できません。実行時には参照のみ可能です。

構文
object.Connection

Connection プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。

値	内容
0	未接続
1	接続中
2	モデム受信待ち中

解説
指定したポートがオープンしているかどうかを取得します。

使用例

```
'ポート1が接続中であるか確認する
Dim intReturn As Integer
object.Port = 1
intReturn = object.Connection
```

Routeプロパティ

2階層目のネットワークに対して、ルートを指定してアクセスする場合に設定します。
値の取得も可能です。
デザイン時にも使用できます。

構文
object.Route[= *Integer*]

Route プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。
Integer	使用するルートNo.を指定します。省略可能です。

設定値
引数*Integer* の設定値は次のとおりです。

定数	値	内容
mewRouteNone	0	ルート指定なし(既定値)
mewRoute1	1	ルートNo.1
mewRoute2	2	ルートNo.2
mewRoute3	3	ルートNo.3
mewRouteClear	10	ルート指定をクリアする

解説
コンピュータと接続されているPLCがMEWNET-W/-P/-Hのリンクユニットを装着していて、そのリンクユニットに接続されているネットワークを介して他のPLCにアクセスする場合に使用します。
CPUに近い方から下記のユニットを1から数えてください。
・MEWNET-Wリンクユニット
・MEWNET-Pリンクユニット
・MEWNET-Hリンクユニット
・ET-LANユニット
・コンピュータ・コミュニケーション・ユニット(CCU)
上記ユニットの中で対象となるMEWNET-W/Pリンクユニットが何番目にあたるかによって、ルートNo.が決定されます。CPUに近い方から、ルート1、ルート2、ルート3となります。

ポートNo.毎にルートを変更したい場合は、必ずルートNo.指定の前にポートNo.の指定を行ってください。

ルート指定をクリアすることができるのは、FP2, FP2SH, FP10SH(Ver.3.0以上)です。

このプロパティはET-LANユニットを使用されている場合は、お使いになれません。
(ルート指定が無効となります)

使用例
```
'ポートNo.1のルートNo.を1に設定する
object.Port = 1
object.Route = mewRoute1
```

NetworkTypeプロパティ

通信するネットワークのタイプを取得します。
デザイン時に指定します。実行時には参照のみ可能です。

構文
object.NetworkType

NetworkType プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。

定数	値	内容
mewC_NET	1	RS232C(C-NET)のネットワークを使用しています。(既定値)
mewEthernetLocal	5	Ethernetのネットワークを使用しています。
mewMODEM	6	モデムを使用しています。

解説
ネットワークタイプを取得します。

使用例

```
'現状のネットワークタイプを参照する
Dim lngReturn As Long
lngReturn = object.NetworkType
```

Languageプロパティ

通信のエラーメッセージ等を表示する言語情報を取得します。
デザイン時に指定します。実行時には参照のみ可能です。

構文
object.Language

Language プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。

定数	値	内容
mewJapanese	0	日本語（既定値）
mewEnglish	1	英語
mewChineseSimplified	2	中国語
mewKorean	4	韓国語
mewSpanish	5	スペイン語
mewItalian	6	イタリア語
mewGermany	7	ドイツ語
mewFrench	8	フランス語

解説
言語情報を取得します。

使用例
```
'現状の言語情報を参照する
Dim lngReturn As Long
lngReturn = object.Language
```

ProgressBarプロパティ

通信の進行状況をプログレスバーで表示するかどうかを設定します。値の取得も可能です。
デザイン時にも使用できます。

構文
object.ProgressBar[= *Boolean*]

ProgressBar プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。
Boolean	プログレスバーを表示するかどうかを設定します。省略可能です。

設定値
引数*Boolean* の設定値は次のとおりです。

値	内容
True	プログレスバーを表示する(既定値)
False	プログレスバーを表示しない

解説
プログレスバー表示・非表示の設定に使用します。表示に設定している場合でも、短時間の通信で終了するような場合には表示されないことがあります。

使用例
```
'プログレスバーを非表示に指定する
object.ProgressBar = False
```

Bankプロパティ

ファイルレジスタにアクセスするときのバンクNo.を設定します。値の取得も可能です。
デザイン時には指定できません。

構文
object.Bank[= *Long*]

Bank プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。
Long	バンクNo.を指定します。省略可能です。

設定値
引数*Long* の設定値は次のとおりです。

定数	値	内容
mewBankNone	−1	バンク指定なし(既定値)
mewBank0	0	バンクNo.0
mewBank1	1	バンクNo.1
mewBank2	2	バンクNo.2

解説
現状、ファイルレジスタがバンク切替できるのは、FP2SHのみです。それ以外の機種では、mewBankNone(=−1)を指定してください。また、FP2SHのファイルレジスタにアクセスする場合、バンク指定無しが指定されたときには、バンクNo.0のファイルレジスタにアクセスします。

使用例
```
'バンク無しを設定する
object.Bank = mewBankNone
```

TaskNameプロパティ

通信タスクの名称が設定できます。値の取得も可能です。
デザイン時には使用できません。実行時に設定します。

構文
object.TaskName[= *String*]

TaskName プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。
String	指定したい通信タスク名称をを指定します。省略可能です。

解説
特に設定する必要はありません。(既定値は CommX1になっています。)
このプロパティを使用しない場合、通信タスクの名称は、本オブジェクトの名称になっています。

使用例

```
'タスク名称を"MyProgram"に設定する
object.TaskName = "MyProgram"
```

LCCommandプロパティ

通信するMEWTOCOL階層指定コマンドが設定できます。値の取得も可能です。
デザイン時には使用できません。実行時に設定します。

構文
object.LCCommand[= *String*]

LCCommand プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。
String	指定したい階層指定コマンドを指定します。省略可能です。

解説
特に設定する必要はありません。
このプロパティを使用しない場合、階層指定コマンドは無効になっています。

使用例

```
' MEWTOCOL階層指定コマンド "ルート1局番11"を設定する
object.LCCommand = "%AA#LC00030700011102**"
```

注).MEWTOCOLコマンドの詳細は別紙マニュアルを参照して下さい。

ErrorDescriptionプロパティ

通信のエラーメッセージを取得します。
デザイン時には使用できません。実行時に参照のみ可能です。

構文
object.ErrorDescription

ErrorDescription プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。

解説
通信エラーが発生した場合、エラー内容の文字列が格納されます。

使用例

```
'エラーメッセージを表示する
MsgBox object.ErrorDescription
```

ErrorNumberプロパティ

通信のエラーNo.を取得します。
デザイン時には使用できません。実行時には参照のみ可能です。

構文
object.ErrorNunmer

ErrorNumber プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。

解説
通信エラーが発生した場合、エラーNo.が格納されます。
エラーの内容を取得したい場合は、ErrorDescriptionプロパティを使用してください。

使用例

```
'エラーNo.を取得する
Dim lngReturn As long
lngReturn = object.ErrorNumber
```

ThreadNumberプロパティ

イーサネット通信使用時、単独の通信プロセスを使用されたい場合、このプロパティに1を設定します。

構文
object.ThreadNumber[= *Integer*]

ThreadNumberプロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。
Integer	通信プロセスを新規作成するかどうかを設定します。

設定値
引数*Integer*の設定値は次のとおりです。

値	内容
0	現在、起動している通信プロセス上で通信を実行します。
1	新規に通信プロセスを生成します。

解説
イーサネット通信使用時、多系列のPLCネットワークに対して通信する場合、他系列PLCネットワークエラーの影響を受けたくないとき、利用します。
但し、作成されるアプリケーションも、CommXに対してマルチタスクにてアクセスしないと未使用の時と同じ動作になります。

使用例

```
'新規通信プロセスを生成します
object.ThreadNumber= 1
object.PortOpen(・・・)
```

TelephoneNumberプロパティ

モデムを介してPLCと通信する際の相手先電話番号を設定します。値の取得も可能です。
デザイン時には使用できません。実行時に設定します。

構文
object.TelephoneNumber[= *String*]

TelephoneNumber プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。
String	指定したい相手先電話番号を指定します。省略可能です。

解説
このプロパティは、必ずポートをオープンする前に設定する必要があります。
(既定値は ""になっています。)

使用例

```
'COMポート1のモデムを介して、012-345-6789に電話をかける
object.TelephoneNumber = "012-345-6789"
object.PortOpen(1)
```

ReceiveIntervalプロパティ

受信監視間隔をミリ秒単位で設定します。
値の取得も可能です。
デザイン時には使用できません。実行時に設定します。

構文
object.ReceiveInterval[= *Long*]

ReceiveInterval プロパティの構文は、次の指定項目から構成されます。

指定項目	内容
object	オブジェクトの名前を指定します。必ず指定します。
Long	モデムの受信監視間隔をミリ秒(ms)単位でを指定します。省略可能です。

解説
遠隔地にあるモデムからの送信に関して、コンピュータ側で受信監視処理を行うときの監視間隔をミリ秒(ms)単位で指定します。
設定しない場合、既定値は10000ms(10秒)に設定されています。

使用例

```
'モデムの受信監視間隔を30秒に設定する
object.ReceiveInterval = 30000
```

メソッド一覧

[通信関数]

■一般仕様

PortOpen

PortClose

AreaRead

AreaReadCSV

AreaReadEx

AreaWrite

AreaWriteCSV

AreaWriteEx

MonitorRead

MonitorReadCSV

PlcStatusRead

PlcModeChange

ShowParameter

GeneralCommand

ExtendRead

ExtendWrite

■モデム仕様

ReceivePortOpen

ReceivePortClose

[変換関数]

CnvBinToDec

CnvDecToBin

CnvOctToDec

CnvDecToOct

CnvHexToDec

CnvDecToHex

CnvLongToSingle

CnvSingleToLong

PortOpenメソッド

通信を開始します。

構文

lngReturn = object.**ProtOpen**(intPort, intStationNumber)
PortOpen メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intPort	ポートNo.を指定します。必ず指定します。
intStationNumber	PLCの局番を指定します。省略可能です。省略した場合の値は自局(局番0)になります。

設定値

引数 intPort の設定値は次のとおりです。

ネットワーク	値の範囲	内容
RS232C(C-NET)	1～5	使用するCOMポートを指定します。 複数のCOMポートをオープン場合は、オープンするCOMポート分この指定が必要です。
Ethernet	1～64	■ ET-LANユニットを使用する場合 通信する相手先局番をポートNo.として指定します。 複数台のET-LANユニットと通信する場合などは、通信する相手先局分、この指定が必要です。 ■ ET-LANユニットを使用せず(RS232C <-> Ethernet変換ユニットを使用して)、ルート指定でリンクユニットの局番を使用する場合。 ポートNo.として必ず1を指定します。
モデム	1～5	使用するCOMポートを指定します。 複数のCOMポートを使用してモデムと接続する場合は、オープンするCOMポート分、この指定が必要です。

戻り値

戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
その他	通信エラー

解説

このメソッドは通信を開始するときに必ず必要です。

使用例

```
'COMポート1で、PLCと接続する。
Dim lngReturn As Long
lngReturn = object.PortOpen(1)

'ET-LANを使用して、相手先局番4と5の2台に接続する。
Dim lngReturn As Long
lngReturn = object.PortOpen(4)
lngReturn = object.PortOpen(5)
```

PortCloseメソッド

通信を停止します。

構文
lngReturn = object.ProtClose(intPort)
PortClose メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intPort	ポートNo.を指定します。必ず指定します。

設定値
引数 intPort の設定値は次のとおりです。

ネットワーク	値の範囲	内容
RS232C(C-NET)	1～5	使用するCOMポートを指定します。 複数のCOMポートをオープン場合は、オープンするCOMポート分この指定が必要です。
Ethernet	1～64	■ ET-LANユニットを使用する場合 通信する相手先局番をポートNo.として指定します。 複数台のET-LANユニットと通信する場合などは、通信する相手先局分、この指定が必要です。 ■ ET-LANユニットを使用せず(RS232C Ethernet変換ユニットを使用して)、ルート指定でリンクユニットの局番を使用する場合。 ポートNo.として必ず1を指定します。
モデム	1～5	使用するCOMポートを指定します。 複数のCOMポートを使用してモデムと接続する場合は、オープンするCOMポート分、この指定が必要です。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
その他	通信エラー

解説
このメソッドは通信を停止するときに必要です。

使用例

```
'COMポート1で、PLCと接続していた通信を停止する。
Dim lngReturn As Long
lngReturn = object.PortClose(1)

'ET-LANを使用して、相手先局番4と局番5の2台に接続していた通信を停止する。
Dim lngReturn As Long
lngReturn = object.PortClose(4)
lngReturn = object.PortClose(5)
```

AreaReadメソッド

連続したデバイスエリアの読込みを、1パケット118バイトで行います。全PLCで使用できます。

構文

lngReturn = object.**AreaRead**(intStationNumber, strAreaName, strAddress, lngCount, intData())
AreaRead メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
strAreaName	読込むデバイスの種類を指定します。必ず指定します。 ワードデバイス : "WX", "WY", "WR", "WL", "DT", "SV", "EV", "LD", "LD" ビットデバイス : "X", "Y", "R", "L", "T", "C" ICカード : "IC"
strAddress	読込むデバイスの先頭アドレスを指定します。必ず指定します。 例) ワードデバイス : "20", "100" ビットデバイス : "10", "2A", "901B"
lngCount	読込むデバイスの数(ビットデバイス:ビット数、ワードデバイス:ワード数)を指定します。必ず指定します。
intData()	読込んだデータを格納するエリアを指定します。必ず指定します。

戻り値

戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
-3	デバイス範囲エラー
その他	通信エラー

解説

このメソッドは1パケット118バイトで通信します。従って、広範囲の連続エリアを読込む場合には、自動的に複数回にわけて通信を行います。1回の通信で完結できるワード数は以下のとおりです。

- ワードデバイス指定時 : 24ワード(ファイルレジスタをバンク指定して読込む場合は、22ワード)
- ビットデバイス指定時 : 384ビット
- ICカード指定時 : 24ワード

使用例

```
'局番4のワードデバイス(DT1000～DT1999)のデータを読込む
Dim lngReturn As long
Dim intData (0 To 999) As Integer

lngReturn = object. AreaRead (4, "DT", "1000", 1000, intData())
'DT1000に10が格納されていれば、intData(0)の値は10になります。


'局番4のビットデバイス(R10～R1F)のデータを読込む
Dim lngReturn As long
Dim intData (0 To 15) As Integer

lngReturn = object. AreaRead (4, "R", "10", 16, intData())
'R15が ONであれば、intData(5)の値は1になります。


'局番4のICカード(IC1000～IC1FFF)のデータを読込む
Dim intData (0 To 4095) As Integer
lngReturn = object.AreaRead(4, "IC", "1000", 4096, intData())
'IC1010に100が格納されていれば、intData(16)の値は100になります。
```

AreaReadCSVメソッド

連続したデバイスエリアの読込みを行います。全PLCで使用できます。

構文
lngReturn = object.**AreaReadCSV**(intStationNumber, strAreaName, strAddress, lngCount, strRegister, intExMode)
AreaReadCSV メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
strAreaName	読込むデバイスの種類を指定します。必ず指定します。 ワードデバイス　："WX", "WY", "WR", "WL", "DT", "SV", "EV", "LD", "LD" ビットデバイス　："X", "Y", "R", "L", "T", "C" ＩＣカード　："IC"
strAddress	読込むデバイスの先頭アドレスを指定します。必ず指定します。 例）ワードデバイス　："20", "100" ビットデバイス　："10", "2A", "901B"
lngCount	読込むデバイスの数（ビットデバイス：ビット数、ワードデバイス：ワード数）を指定します。 必ず指定します。
strRegister	読込んだデータを格納するエリアを指定します。カンマ区切りで格納します。必ず指定します。
intExMode	パケットモード 0:118バイト 1:2048バイト

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
-3	デバイス範囲エラー
その他	通信エラー

解説
このメソッドはパケットモードにより通信バイト数が変わります。
0の場合は１パケット１１８バイトで通信します。従って、広範囲の連続エリアを読込む場合には、自動的に複数回にわけて通信を行います。１回の通信で完結できるワード数は以下のとおりです。
・ワードデバイス指定時　：２４ワード（ファイルレジスタをバンク指定して読込む場合は、２２ワード）
・ビットデバイス指定時　：３８４ビット
・ＩＣカード指定時　：２４ワード

1の場合は１パケット２０４８バイトで通信します。従って、広範囲の連続エリアを読込む場合には、自動的に複数回にわけて通信を行います。１回の通信で完結できるワード数は以下のとおりです。
・ワードデバイス指定時　：５０６ワード（ファイルレジスタをバンク指定して読込む場合は、５０４ワード）
・ビットデバイス指定時　：８０９６ビット
・ＩＣカード指定時　：５０６ワード

使用例

```
'局番4のワードデバイス(DT1000～DT1999)のデータを読込む
Dim lngReturn As Long
Dim strData As String

lngReturn = object.AreaReadCSV(4, "DT", "1000", 1000, strData, 0)
'DT1000に10が格納されていれば、0番目の値は10になります。


'局番4のビットデバイス(R10～R1F)のデータを読込む
Dim lngReturn As Long
Dim strData As String

lngReturn = object.AreaReadCSV(4, "R", "10", 16, strData, 0)
```

'R15が ONであれば、5番目の値は1になります。

'局番4のICカード(IC1000～IC1FFF)のデータを読込む
Dim strData As String

lngReturn = *object*.**AreaReadCSV**(4, "IC", "1000", 4096, strData, 0)
'IC1010に100が格納されていれば、16番目の値は100になります。

AreaReadExメソッド

連続したデバイスエリアの読込みを、1パケット2048バイトで行います。使用できるPLCに制限があります。
使用できるPLCは以下のとおりです。
FP3(Ver. 4.5以上), FP10S(Ver. 1.5以上), FP10SH, FP2, FP2SH

構文
lngReturn = object.AreaReadEx(intStationNumber, strAreaName, strAddress, lngCount, intData())
AreaReadEx メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
strAreaName	読込むデバイスの種類を指定します。必ず指定します。 ワードデバイス : "WX", "WY", "WR", "WL", "DT", "SV", "EV", "LD", "LD" ビットデバイス : "X", "Y", "R", "L", "T", "C" ICカード : "IC"
strAddress	読込むデバイスの先頭アドレスを指定します。必ず指定します。 例) ワードデバイス : "20", "100" ビットデバイス : "10", "2A", "901B"
lngCount	読込むデバイスの数(ビットデバイス:ビット数、ワードデバイス:ワード数)を指定します。必ず指定します。
intData()	読込んだデータを格納するエリアを指定します。必ず指定します。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
-3	デバイス範囲エラー
その他	通信エラー

解説
このメソッドは1パケット2048バイトで通信します。従って、広範囲の連続エリアを読込む場合には、自動的に複数回にわけて通信を行います。1回の通信で完結できるワード数は以下のとおりです。

- ワードデバイス指定時 : 506ワード(ファイルレジスタをバンク指定して読込む場合は、504ワード)
- ビットデバイス指定時 : 8096ビット
- ICカード指定時 : 506ワード

使用例

```
'局番4のワードデバイス(DT1000～DT1999)のデータを読込む
Dim lngReturn As long
Dim intData (0 To 999) As Integer

lngReturn = object. AreaReadEx (4, "DT", "1000", 1000, intData())
'DT1000に10が格納されていれば、intData(0)の値は10になります。


'局番4のビットデバイス(R10～R1F)のデータを読込む
Dim lngReturn As long
Dim intData (0 To 15) As Integer

lngReturn = object. AreaReadEx (4, "R", "10", 16, intData())
'R15が ONであれば、intData(5)の値は1になります。


'局番4のICカード(IC1000～IC1FFF)のデータを読込む
Dim lngReturn As long
Dim intData (0 To 4095) As Integer

lngReturn = object.AreaReadEx(4, "IC", "1000", 4096, intData())
'IC1010に100が格納されていれば、intData(16)の値は100になります。
```

AreaWriteメソッド

連続したデバイスエリアの書込みを、1パケット118バイトで行います。全PLCで使用できます。

構文
lngReturn = object.AreaWrite(intStationNumber, strAreaName, strAddress, lngCount, intData())
AreaWrite メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
strAreaName	書込むデバイスの種類を指定します。必ず指定します。 ワードデバイス : "WX", "WY", "WR", "WL", "DT", "SV", "EV", "LD", "LD" ビットデバイス : "X", "Y", "R", "L", "T", "C" ICカード : "IC"
strAddress	書込むデバイスの先頭アドレスを指定します。必ず指定します。 例) ワードデバイス : "20", "100" ビットデバイス : "10", "2A", "901B"
lngCount	書込むデバイスの数(ビットデバイス:ビット数、ワードデバイス:ワード数)を指定します。必ず指定します。
intData()	書込むデータを格納したエリアを指定します。 ビットデバイスをONする場合は1を、OFFする場合は0を指定してください。 必ず指定します。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
-3	デバイス範囲エラー
その他	通信エラー

解説
このメソッドは1パケット118バイトで通信します。従って、広範囲の連続エリアを書込む場合には、自動的に複数回にわけて通信を行います。1回の通信で完結できるワード数は以下のとおりです。

- ワードデバイス指定時 : 24ワード(ファイルレジスタをバンク指定して読込む場合は、22ワード)
- ビットデバイス指定時 : 384ビット
- ICカード指定時 : 24ワード

使用例

```
'局番4のワードデバイス(DT1000～DT1999)にデータを書込む
Dim lngReturn As long
Dim intData (0 To 999) As Integer

IntData(0) = 10 'DT1000に10を書込む
IntData(1) = 20 'DT1001に20を書込む

lngReturn = object. AreaWrite (4, "DT", "1000", 1000, intData())
'DT1000に10が、DT1001に20が、格納されます。


'局番4のビットデバイス(R1A)をONする
Dim lngReturn As long
Dim intData (0) As Integer

IntData(0) = 1
lngReturn = object. AreaWrite(4, "R", "1A", 1, intData())
'R1Aが ONになります。


'局番4のICカード(IC1000～IC1FFF)のデータを書込む
Dim lngReturn As long
Dim intData (0 To 4095) As Integer
IntData(0) = 10 'IC1000に10を書込む
IntData(1) = 20 'IC1001に20を書込む
```

```
lngReturn = object.AreaWrite(4, "IC", "1000", 4096, intData())
'IC1000に10が、IC1001に20が、格納されます。
```

AreaWriteCSVメソッド

連続したデバイスエリアの書込みを行います。全PLCで使用できます。

構文

lngReturn = object.AreaWriteCSV(intStationNumber, strAreaName, strAddress, lngCount, strRegister, intExMode)

AreaWriteCSV メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
strAreaName	書込むデバイスの種類を指定します。必ず指定します。 ワードデバイス　："WX", "WY", "WR", "WL", "DT", "SV", "EV", "LD", "LD" ビットデバイス　："X", "Y", "R", "L", "T", "C" ＩＣカード　："IC"
strAddress	書込むデバイスの先頭アドレスを指定します。必ず指定します。 例）ワードデバイス　："20", "100" ビットデバイス　："10", "2A", "901B"
lngCount	書込むデバイスの数（ビットデバイス：ビット数、ワードデバイス：ワード数）を指定します。必ず指定します。
strRegister	書込むデータを格納したエリアを指定します。カンマ区切りで格納します。 ビットデバイスをＯＮする場合は１を、ＯＦＦする場合は０を指定してください。 必ず指定します。
intExMode	パケットモード 0:118バイト 1:2048バイト

戻り値

戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
-3	デバイス範囲エラー
その他	通信エラー

解説

このメソッドはパケットモードにより通信バイト数が変わります。

0の場合は１パケット１１８バイトで通信します。従って、広範囲の連続エリアを読込む場合には、自動的に複数回にわけて通信を行います。１回の通信で完結できるワード数は以下のとおりです。

・ワードデバイス指定時　：２４ワード（ファイルレジスタをバンク指定して読込む場合は、２２ワード）
・ビットデバイス指定時　：３８４ビット
・ＩＣカード指定時　：２４ワード

1の場合は１パケット２０４８バイトで通信します。従って、広範囲の連続エリアを読込む場合には、自動的に複数回にわけて通信を行います。１回の通信で完結できるワード数は以下のとおりです。

・ワードデバイス指定時　：５０６ワード（ファイルレジスタをバンク指定して読込む場合は、５０４ワード）
・ビットデバイス指定時　：８０９６ビット
・ＩＣカード指定時　：５０６ワード

使用例

```
'局番4のワードデバイス(DT1000～DT1999)にデータを書込む
Dim lngReturn As Long
Dim strData As String

'DT1000に10を、DT1001に20を書込む
strData = "10,20,0,0,0,0,0,・・・"

lngReturn = object.AreaWriteCSV(4, "DT", "1000", 1000, strData, 0)
'DT1000に10が、DT1001に20が、格納されます。


'局番4のビットデバイス(R1A)をONする
```

```
Dim lngReturn As Long
Dim strData As String

strData = "1"
lngReturn = object.AreaWriteCSV(4, "R", "1A", 1, strData, 0)
'R1Aが ONになります。


'局番4のICカード(IC1000～IC1FFF)のデータを書込む
Dim lngReturn As Long
Dim strData As String

'IC1000に10を、IC1001に20を書込む
strData = "10,20,0,0,0,0,0,・・・"

lngReturn = object.AreaWriteCSV(4, "IC", "1000", 4096, strData, 0)
'IC1000に10が、IC1001に20が、格納されます。
```

AreaWriteExメソッド

連続したデバイスエリアの書込みを、1パケット2048バイトで行います。使用できるPLCに制限があります。
使用できるPLCは以下のとおりです。
FP3(Ver. 4.5以上), FP10S(Ver. 1.5以上), FP10SH, FP2, FP2SH

構文
lngReturn = object.AreaWriteEx(intStationNumber, strAreaName, strAddress, lngCount, intData())
AreaWriteEx メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
strAreaName	書込むデバイスの種類を指定します。必ず指定します。 ワードデバイス : "WX", "WY", "WR", "WL", "DT", "SV", "EV", "LD", "LD" ビットデバイス : "X", "Y", "R", "L", "T", "C" ICカード : "IC"
strAddress	書込むデバイスの先頭アドレスを指定します。必ず指定します。 例) ワードデバイス : "20", "100" ビットデバイス : "10", "2A", "901B"
lngCount	書込むデバイスの数(ビットデバイス:ビット数、ワードデバイス:ワード数)を指定します。必ず指定します。
intData()	書込むデータを格納したエリアを指定します。 ビットデバイスをONする場合は1を、OFFする場合は0を指定してください。 必ず指定します。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
-3	デバイス範囲エラー
その他	通信エラー

解説
このメソッドは1パケット2048バイトで通信します。従って、広範囲の連続エリアを書込む場合には、自動的に複数回にわけて通信を行います。1回の通信で完結できるワード数は以下のとおりです。

- ワードデバイス指定時 : 506ワード(ファイルレジスタをバンク指定して読込む場合は、504ワード)
- ビットデバイス指定時 : 8096ビット
- ICカード指定時 : 506ワード

使用例

```
'局番4のワードデバイス(DT1000～DT1999)にデータを書込む
Dim lngReturn As long
Dim intData (0 To 999) As Integer

IntData(0) = 10 'DT1000に10を書込む
IntData(1) = 20 'DT1001に20を書込む

lngReturn = object.AreaWriteEX (4, "DT", "1000", 1000, intData())
'DT1000に10が、DT1001に20が、格納されます。


'局番4のビットデバイス(R1A)をONする
Dim lngReturn As long
Dim intData (0) As Integer

IntData(0) = 1
lngReturn = object. AreaWriteEX (4, "R", "1A", 1, intData())
'R1Aが ONになります。


'局番4のICカード(IC1000～IC1FFF)のデータを書込む
Dim lngReturn As long
Dim intData (0 To 4095) As Integer
```

```
IntData(0) = 10 'IC1000に10を書込む
IntData(1) = 20 'IC1001に20を書込む

lngReturn = object. AreaWriteEX (4, "IC", "1000", 4096, intData())
'IC1000に10が、IC1001に20が、格納されます。
```

MonitorReadメソッド

異なる種類のデバイスや連続していないデバイスエリアの読込みを行います。全PLCで使用できます。
ワードデバイス、ビットデバイス各々80種類以内のデバイスが指定できます。
(この関数でICカードにはアクセスできません。)

構文
lngReturn = object.**MonitorRead**(intStationNumber, lngBitCount, strBitAreaName(), strBitAddress(), intBitRegister(), lngWordCount, strWordAreaName(), strWordAddress(), lngWordBank(), intWordRegister())
MonitorRead メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
lngBitCount	読込むビットデバイスのビット数を指定します。必ず指定します。(*1)
strBitAreaName	読込むビットデバイスの種類を指定します。必ず指定します。(*2)
strBitAddress	読込むビットデバイスのアドレスを指定します。必ず指定します。
intBitRegister	読込んだビットデバイスの値が格納されます。必ず指定します。
lngWordCount	読込むワードデバイスのワード数を指定します。必ず指定します。(*1)
strWordAreaName	読込むワードデバイスの種類を指定します。必ず指定します。(*2)
strWordAddress	読込むワードデバイスのアドレスを指定します。必ず指定します。
lngWordBank	読込むワードデバイスのバンクを指定します。必ず指定します。(*3)
intWordRegister	読込んだワードデバイスの値が格納されます。必ず指定します。

(*1) 読込むデバイスが無いときは0を指定します。
(*2) 読込むデバイスの種類は以下の中から選択してください。
ワードデバイス : "WX", "WY", "WR", "WL", "DT", "SV", "EV", "LD", "LD"
ビットデバイス : "X", "Y", "R", "L", "T", "C"
(*3) FP2SHのバンク0、1、2のファイルレジスタ読込むときには、各々0, 1, 2を指定してください。
それ以外は、-1を指定してください。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
その他	通信エラー

解説
ワードデバイス、ビットデバイス各々個別で使用する変数を宣言する必要があります。
このメソッドは1パケット118バイトで通信します。従って、多数のデバイスを読込む場合には、自動的に複数回にわけて通信を行います。1回の通信で完結できるワード数は以下のとおりです。
・ワードデバイス : 16ワード(ファイルレジスタをバンク指定して読込む場合は、22ワード)
・ビットデバイス : 80ビット
・ICカード : 読込めません

使用例

```
'局番4のビットデバイス(L105, R12C)とワードデバイス(DT100, WL222)のデータを読込む
Dim lngReturn As Long
Dim lngBitCount As Long
Dim strBitAreaName(0 To 1) As String
Dim strBitAddress(0 To 1) As String
Dim intBitRegister(0 To 1) As Integer
Dim lngWordCount As Long
Dim strWordAreaName(0 To 1) As String
Dim strWordAddress(0 To 1) As String
Dim lngWordBank(0 To 1) As Long
Dim intWordRegister(0 To 1) As Integer

'ビットデバイス(L105, R12C)を登録します
lngBitCount = 2
strBitAreaName(0) = "L"
strBitAddress(0) = "105"
strBitAreaName(1) = "R"
strBitAddress(1) = "12C"
```

```
'ワードデバイス(DT100, WL222)を登録します
lngWordCount = 2
strWordAreaName(0) = "DT"
strWordAddress(0) = "100"
lngWordBank(0) = -1
strWordAreaName(1) = "WL"
strWordAddress(1) = "222"
lngWordBank(1) = -1

lngReturn = object.MonitorRead(4, _
      lngBitCount, strBitAreaName(), strBitAddress(), intBitRegister(), _
      lngWordCount, strWordAreaName(), strWordAddress(), lngWordBank(), intWordRegister())

'intBitRegister(0) にL105のデータが
'intBitRegister(1) にR12Cのデータが
'intWordRegister(0) にDT100のデータが
'intWordRegister(1) にWL222のデータが格納されます。
```

MonitorReadCSVメソッド

異なる種類のデバイスや連続していないデバイスエリアの読込みを行います。全PLCで使用できます。
ワードデバイス、ビットデバイス各々８０種類以内のデバイスが指定できます。
（この関数でＩＣカードにはアクセスできません。）

構文
lngReturn = object.**MonitorReadCSV**(intStationNumber, lngBitCount, strBitAreaName, strBitAddress, strBitRegister, lngWordCount, strWordAreaName, strWordAddress, strWordBank, strWordRegister)
MonitorRead メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
lngBitCount	読込むビットデバイスのビット数を指定します。必ず指定します。(*1)
strBitAreaName	読込むビットデバイスの種類を指定します。カンマ区切りで格納します。必ず指定します。(*2)
strBitAddress	読込むビットデバイスのアドレスを指定します。カンマ区切りで格納します。必ず指定します。
strBitRegister	読込んだビットデバイスの値が格納されます。カンマ区切りで格納します。必ず指定します。
lngWordCount	読込むワードデバイスのワード数を指定します。必ず指定します。(*1)
strWordAreaName	読込むワードデバイスの種類を指定します。カンマ区切りで格納します。必ず指定します。(*2)
strWordAddress	読込むワードデバイスのアドレスを指定します。カンマ区切りで格納します。必ず指定します。
strWordBank	読込むワードデバイスのバンクを指定します。カンマ区切りで格納します。必ず指定します。(*3)
strWordRegister	読込んだワードデバイスの値が格納されます。カンマ区切りで格納します。必ず指定します。

(*1) 読込むデバイスが無いときは0を指定します。
(*2) 読込むデバイスの種類は以下の中から選択してください。
ワードデバイス　："WX", "WY", "WR", "WL", "DT", "SV", "EV", "LD", "LD"
ビットデバイス　："X", "Y", "R", "L", "T", "C"
(*3) FP2SHのバンク０、１、２のファイルレジスタ読込むときには、各々0, 1, 2を指定してください。
それ以外は、-1を指定してください。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
その他	通信エラー

解説
ワードデバイス、ビットデバイス各々個別で使用する変数を宣言する必要があります。
このメソッドは１パケット１１８バイトで通信します。従って、多数のデバイスを読込む場合には、自動的に複数回にわけて通信を行います。１回の通信で完結できるワード数は以下のとおりです。
・ワードデバイス　：１６ワード（ファイルレジスタをバンク指定して読込む場合は、２２ワード）
・ビットデバイス　：８０ビット
・ＩＣカード　：読込めません

使用例

```
'局番4のビットデバイス(L105, R12C)とワードデバイス(DT100, WL222)のデータを読込む
Dim lngReturn As Long
Dim lngBitCount As Long
Dim strBitAreaName As String
Dim strBitAddress As String
Dim strBitRegister As String
Dim lngWordCount As Long
Dim strWordAreaName As String
```

```
Dim strWordAddress As String
Dim strWordBank As String
Dim strWordRegister As String

'ビットデバイス(L105, R12C)を登録します
lngBitCount = 2
strBitAddress = "105,12C"

'ワードデバイス(DT100, WL222)を登録します
lngWordCount = 2
strWordAreaName = "DT,WL"
strWordAddress = "100,222"
strWordBank = "-1,-1"

lngReturn = object.MonitorReadCSV(4, _
      lngBitCount, strBitAreaName, strBitAddress, strBitRegister, _
      lngWordCount, strWordAreaName, strWordAddress, strWordBank, strWordRegister)

'strBitRegister  0番目にL105のデータが
'strBitRegister  1番目にR12Cのデータが
'strWordRegister 0番目にDT100のデータが
'strWordRegister 1番目にWL222のデータが格納されます。
```

PlcStatusReadメソッド

PLCの状態(ステータス)を読込むことができます。全PLCで使用できます。

構文

lngReturn = object.**PLCStatusRead**(intStationNumber, intRunFlag, intPlcError)
AreaRead メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
intRunFlag	PLCの動作モードを取得します。必ず指定します。 動作モードによって各々以下の値が格納されます。 PROGモード :0 RUNモード :1 REMOTE - PROG.モード :10 REMOTE - RUNモード :11
intPlcError	PLCのｴﾗｰ状態を取得します。必ず指定します。 PLC正常時 :0 エラー発生時 :エラーコード(0以外) 0以外の値が格納されているときは、アクセスされるPLCマニュアルのエラーコード一覧の章を参照してください。(PLCの種類により内容が異なります。)

戻り値

戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
その他	通信エラー

解説

PLCの状態を読込みたいときに使用します。上記に説明されている各内容は、PLCの種類によってことなりますので、アクセスされるPLCのマニュアルをご参照の上、ご使用ください。

使用例

```
'局番0のPLCの動作モードを読込む
Dim lngReturn as long
Dim intRunFlg As Integer
Dim intPlcError As Integer
lngReturn = object. PlcStatusRead( 0, intRunFlag, intPlcError)

'PLCの動作モードを表示
Select Case intRunFlg
    Case 0
        MsgBox "PROG"
    Case 1
        MsgBox "RUN"
    Case 10
        MsgBox "REMOTE - PROG"
    Case 11
        MsgBox "REMOTE - RUN"
End Select

'エラーコードを表示
Select Case intPLCError
    Case 0
        MsgBox "エラーはありません！"
    Case Else
        MsgBox "E" & intPLCError
End Select
```

PlcModeChangeメソッド

PLCのRUNモード <-> PROG.モードを切り替えます。
(PLCがリモートモードになっていなければ、切り替えることはできません。)

構文
lngReturn = object.**PlcModeChange**(intStationNumber, intRunFlg)
PlcModeChange メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
intRunFlg	変更後の動作モードを指定します。必ず指定します。 PROG. => RUN :1 RUN . => PROG. : 0

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
-4	PLCがリモートモードになっていない
その他	通信エラー

解説
先にPLCの動作モードがリモートになっているかを確認した後に、このメソッドを使用するようにしてください。
(PLCがリモートモードになっていなければ、切り替えることはできません。PLCがリモートモードになっているかどうかは、PlcStatusReadメソッドで判別できます。)
また、ET-LANユニットを使用してEthernet通信している場合は、PLCをPROG.モードにした状態で通信を終了(ProtClose)すると、その後PLCを手動にてPROG.=>RUNに切り替えない限り、二度と接続できなくなります。注意してください。

使用例
```
'局番0のPLCをPROG.からRUNに切り替える
Dim lngReturn as long
Dim intRunFlg As Integer
Dim intPlcError As Integer
lngReturn = object.PlcStatusRead (0, intRunFlg, intPlcError)
'PLCがリモートモードなら RUNへ変更
If intRunFlg >= 10 then
    lngReturn = object.PlcModeChange (0, 1)
End If
'PLCはRUNモードに変更されます。
```

ShowParameterメソッド

通信設定ダイアログを表示します。通信設定を表示して変更するときには、通信している全てのポートをクローズする必要があります。

構文
lngReturn = object.**ShowParameter**

戻り値
戻り値の値は次のとおりです。

戻り値	内容
vbOk	OKボタンが押された
vbCancel	キャンセルボタンが押された
vbAbort	中止ボタンが押された
その他	エラー

解説
通信設定を表示して変更するときには、通信している全てのポートをクローズする必要があります。
変更した通信設定の内容は、アプリケーションを再起動しないと反映されません。

使用例

```
'COMポート1の通信設定を変更する
Dim lngReturn as long

lngReturn = object.PortClose(1)
lngReturn = object.ShowParameter
lngReturn = object.PortOpen(1)
```

GeneralCommandメソッド

通信コマンドの送受信を行います。ATコマンドなどMEWTOCOLコマンド以外の通信でも使用できます。

構文
lngReturn = object.GeneralCommand(strSend, intReceiveLength, strReceive)
GeneralCommand メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
strSend	送信するデータを格納したエリアを指定します。 送信するデータが無い場合は受信のみ行います。必ず指定します。
intReceiveLength	受信したデータを格納するエリアの大きさを指定します。必ず指定します。
strReceive	受信したデータを格納するエリアを指定します。必ず指定します。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
その他	通信エラー

解説
ＰＬＣ以外の機器と通信を行いたい時に使用します。バイナリ通信以外のときにこのメソッドを使用するようにしてください。
各機器の種類によって通信手順がことなりますので、アクセスされる機器のマニュアルをご参照の上、ご使用ください。

使用例

```
'MODEMを初期化します
Dim lngReturn as Long
Dim strSend As String
Dim strRecive As String

strSend = "ATV1E0S0=1S2=43"
lngReturn = object.GeneralCommand(strSend, 255, strRecive)
'初期化に成功すればOKが受信されます。
```

ExtendReadメソッド

FP Σ の拡張メモリユニットのデータを読み込みます。

構文
lngReturn = object.**ExtendRead**(intStationNumber, intSlot, bytBank(), intNumber(), intCount, intData())
ExtendRead メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
intSlot	スロットNo.(0-3)を指定します。必ず指定します。
bytBank()	バンク読込みフラグ　1：読み込み。 0から255バンクの256配列が必要です。必ず指定します。
intNumber()	読込みスタートアドレス(0-1023)。 0から255バンクの256配列が必要です。必ず指定します。
lngCount()	読込むワードの数を指定します。0から255バンクの256配列が必要です。必ず指定します。
intData()	読込んだデータを格納するエリアを指定します。読込む全バンクのワード総合計分のエリアを用意します。必ず指定します。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
-3	デバイス範囲エラー
その他	通信エラー

解説
このメソッドはFP Σ の拡張メモリの1ユニット内256バンクのデータを一括で読込みます。
読込む必要のないバンクに関しては、読み飛ばす事(bytBank[バンクナンバー]に0をセット)ができます。
また、バンク単位で読込むエリアを変えることも下記パラメータにより可能です。
- 読込み開始アドレス　：　intNumber[バンクナンバー]
- 読込みワード数　：　intCount[バンクナンバー]

使用例

```
'スロット0から255バンクのアドレス0から10ワードのデータを読み込むます。
Dim lreturn As Long
Dim i As Integer
Dim intNo As Integer                        'PLC局番
Dim intSlotNo As Integer                    'スロットNo
Dim bytBank(255) As Byte                    'バンク読込みフラグ
Dim intNumber(255) As Integer               '読込みスタートアドレス
Dim intCount(255) As Integer                '読込みカウント
Dim intData(3000) As Integer                '書込みデータ

    intNo = 0                               'PLC局番に0をｾｯﾄする
    intSlotNo = 0                           'スロットNoに0をセットする
    '各バンクエリアに読込みデータをセットする
    For i = 0 To 255
        bytBank(i) = 1                      '読込みフラグをONする
        intNumber(i) = 0                    '読込み開始アドレスに0をセットする
        intCount(i) = 10                    '読込みワード数をセットする
    Next i
    lreturn = object.ExtendRead(0, 0, bytBank, intNumber, intCount, intData)
```

ExtendWriteメソッド

FP Σ の拡張メモリユニットのデータを書込みます。

構文
lngReturn = object.ExtendWriteEx(intStationNumber, intslot, bytBank(), intNumber(), intCount, intData())
ExtendWrite メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	PLCの局番を指定します。必ず指定します。
intSlot	スロットNo.(0-3)を指定します。必ず指定します。
bytBank()	バンク書込みフラグ。1:書込み。 0から255の256配列を用意します。必ず指定します。
intNumber()	書込みスタートアドレス(0-1023)。0から255バンクの256配列を用意します。 必ず指定します。
intCount()	書込むワードの数を指定します。0から255バンクの256配列が必要です。必ず指定します。
intData()	書込むデータを格納したエリアを指定します。 書込む全バンクのデータを連続で格納したエリアを指定します。 必ず指定します。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
-3	デバイス範囲エラー
その他	通信エラー

解説
このメソッドはFP Σ の拡張メモリの1ユニット内256バンクのデータを一括で読込みます。
書込む必要のないバンクに関しては、書き飛ばす事(bytBank[バンクナンバー]に0をセット)ができます。
また、バンク単位で書込むエリアを変えることも下記パラメータにより可能です。
- 書込む開始アドレス : intNumber[バンクナンバー]
- 書込むワード数 : intCount[バンクナンバー]

使用例

```
'スロット0から255バンクのアドレス0に10ワードのデータを書込みます。
Dim lreturn As Long
Dim i As Integer
Dim intNo As Integer                          'PLC局番
Dim intSlotNo As Integer                      'スロットNo
Dim bytBank(255) As Byte                      'バンク書込みフラグ
Dim intNumber(255) As Integer                 '書込みスタートアドレス
Dim intCount(255) As Integer                  '書込みカウント
Dim intData(3000) As Integer                  '書込みデータ

    intNo = 0                                 'PLC局番に0をセットする
    intSlotNo = 0                             'スロットNoに0をセットする
    '各バンクエリアに書込みデータをセットする
    For i = 0 To 255
        bytBank(i) = 1                        '書込みフラグをONする
        intNumber(i) = 0                      '書込み開始アドレスに0をセットする
        intCount(i) = 10                      '書込みワード数をセットする
    Next i
    '書込みエリアにデータをセットする
    For i = 0 To 2559
        intData(i) = i + 1
    Next i
    lreturn = object.ExtendWrite(0, 0, bytBank, intNumber, intCount, intData)
```

ReceivePortOpenメソッド

遠隔地にあるモデムからの送信に関して、コンピュータ側で受信監視処理を開始します。

構文
lngReturn = object.**ReceiveProtOpen**(intPort)
ReceivePortOpen メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intPort	受信するポートNo.を指定します。必ず指定します。

設定値
引数 intPort の設定値は次のとおりです。

ネットワーク	値の範囲	内容
モデム	1～5	使用するCOMポートを指定します。 複数のCOMポートを使用して遠隔地からのモデムの受信を監視する場合は、オープンするCOMポート分、この指定が必要です。 他のネットワークでは、使用できません。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
-3	パラメータエラー
その他	通信エラー

解説
このメソッドはモデムからの受信を開始するときに必ず必要です。

使用例

```
'COMポート2で、モデムからの受信を監視する。
Dim lngReturn As long
lngReturn = object.ReceivePortOpen(2)
```

ReceivePortCloseメソッド

遠隔地にあるモデムからの送信に関して、コンピュータ側で受信監視処理を停止します。

構文
lngReturn = object.ReceiveProtClose(intPort)
ReceivePortClose メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
intPort	受信監視を停止するポートNo.を指定します。必ず指定します。

設定値
引数 intPort の設定値は次のとおりです。

ネットワーク	値の範囲	内容
モデム	1～5	受信監視を停止するCOMポートを指定します。 他のネットワークでは、使用できません。

戻り値
戻り値の値は次のとおりです。

戻り値	内容
0	正常終了
-1	致命的エラー
-2	キーユニット装着エラー
その他	通信エラー

解説
このメソッドはモデムからの受信監視を停止するときに必要です。本指定がなくても、アプリケーションを終了すると全てのポートはクローズされます。

使用例

```
        'COMポート2で、モデムと接続していた通信を停止する。
        Dim lngReturn As long
    lngReturn = object.ReceivePortClose(1)
```

CnvBinToDecメソッド

2進数文字列を10進整数に変換します

構文
lngDec = object.CnvBinToDec(strBin, intByteSize)
CnvBinToDec メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
strBin	変換する2進数文字列("1"と"0"の文字列)を指定します。必ず指定します。
intByteSize	変換するバイト数を指定します。必ず指定します。 下位から変換されます。(下位は、上記指定文字列の右側です。) 1バイト(右から 8文字分):1 1ワード(右から16文字分):2 2ワード(右から32文字分):4

戻り値
戻り値は 10進整数です。

使用例

```
'2進数[10101010010101010000111111110000] の1バイトを10進数に変換する
Text1.Text = object.CnvBinToDec("10101010010101010000111111110000", 1)
'テキストボックスには[240]と表示されます。(下位(右から)8ビット(文字)のみが変換対象)

'2進数[10101010010101010000111111110000] の1ワードを10進数に変換する
Text1.Text = object.CnvBinToDec("10101010010101010000111111110000", 2)
'テキストボックスには[4080]と表示されます。(下位(右から)16ビット(文字)のみが変換対象)

'2進数[10101010010101010000111111110000] の2ワードを10進数に変換する
Text1.Text = object.CnvBinToDec("10101010010101010000111111110000", 4)
'テキストボックスには[-1437265936]と表示されます。(下位(右から)32ビット(文字)のみが変換
対象)
```

CnvDecToBinメソッド

１０進整数を 2進数文字列に変換します

構文
srtBin = object.CnvDecToBin(lngDec, intByteSize)
CnvDecToBin メソッドの構文は、次の指定項目から構成されます。

指定項目 内容

lngDec	変換する１０進整数を指定します。必ず指定します。
intByteSize	2進数に変換後、格納するバイト数を指定します。必ず指定します。 下位から格納されます。(下位は、上記指定文字列の右側です。) 1バイト(右から　8文字分):1 1ワード(右から１６文字分):2 2ワード(右から３２文字分):4

戻り値
戻り値は　2進数文字列です。

使用例

```
'１０進数[240]を２進数に変換し、その１バイトを格納する
Text1.Text = object.CnvDecToBin( 240, 1)
'テキストボックスには[11110000]と表示されます。
'(変換後、下位(右から)8ビット(文字)のみを取得)

'１０進数[4080]を２進数に変換し、その１ワード分を格納する
Text1.Text = object.CnvDecToBin( 4080, 2)
'テキストボックスには[0000111111110000]と表示されます。
'(変換後、下位(右から)16ビット(文字)のみを取得)

'１０進数[-1437265936]を２進数に変換し、その２ワード分を格納する
Text1.Text = object.CnvDecToBin( -1437265936, 4)
'テキストボックスには[10101010101010100000111111110000]と表示されます。
'(変換後、下位(右から)32ビット(文字)のみを取得)
```

CnvOctToDecメソッド

8進数文字列を10進整数に変換します

構文

lngDec = object.**CnvOctToDec**(strOct, intByteSize)
CnvOctToDec メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
strOct	変換する8進数文字列を指定します。必ず指定します。
intByteSize	変換するバイト数を指定します。必ず指定します。 下位から変換されます。 1バイト:1 1ワード:2 2ワード:4

戻り値

戻り値は　10進整数です。

使用例

```
'8進数[377] の1バイトを10進数に変換する
Text1.Text = object.CnvBinToDec("377", 1)
'テキストボックスには[255]と表示されます。

'8進数[177400] の1ワードを10進数に変換する
Text1.Text = object.CnvBinToDec("177400", 2)
'テキストボックスには[-256]と表示されます。

'8進数[37700177400] の2ワードを10進数に変換する
Text1.Text = object.CnvBinToDec("37700177400", 4)
'テキストボックスには[-16711936]と表示されます。
```

CnvDecToOctメソッド

１０進整数を ８進数文字列に変換します

構文
srtOct = object.CnvDecToOct(lngDec, intByteSize)
CnvDecToOct メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
lngDec	変換する１０進整数を指定します。必ず指定します。
intByteSize	８進数に変換後、格納するバイト数を指定します。必ず指定します。 下位から格納されます。 1バイト:1 1ワード:2 2ワード:4

戻り値
戻り値は ８進数文字列です。

使用例

```
'１０進数[255]を８進数に変換し、その１バイトを格納する
Text1.Text = object.CnvDecToOct( 255, 1)
'テキストボックスには[377]と表示されます。

'１０進数[-256]を８進数に変換し、その１ワード分を格納する
Text1.Text = object.CnvDecToOct( -256, 2)
'テキストボックスには[177400]と表示されます。

'１０進数[-16711936]を８進数に変換し、その２ワード分を格納する
Text1.Text = object.CnvDecToOct( -16711936, 4)
'テキストボックスには[37700177400]と表示されます。
```

CnvHexToDecメソッド

１６進数文字列を　１０進整数に変換します

構文
lngDec = object.CnvHexToDec(strHex, intByteSize)
CnvHexToDec メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
strHex	変換する１６進数文字列を指定します。必ず指定します。
intByteSize	変換するバイト数を指定します。必ず指定します。 下位から変換されます。（下位は、上記指定文字列の右側です。） 1バイト（右から２文字分）：1 1ワード（右から４文字分）：2 2ワード（右から８文字分）：4

戻り値
戻り値は　10進整数です。

使用例

```
'１６進数[12345678] の1バイトを１０進数に変換する
Text1.Text = object.CnvHexToDec("12345678", 1)
'テキストボックスには[120]と表示されます。(右から2文字のみが変換対象)

'１６進数[12345678] の1ワードを１０進数に変換する
Text1.Text = object.CnvHexToDec("12345678", 2)
'テキストボックスには[22136]と表示されます。(右から4文字のみが変換対象)

'１６進数[12345678] の2ワードを１０進数に変換する
Text1.Text = object.CnvHexToDec("12345678", 4)
'テキストボックスには[305419896]と表示されます。(右から)8文字のみが変換対象)
```

CnvDecToHexメソッド

１０進整数を １６進数文字列に変換します

構文
srtHex = object.**CnvDecToHex**(lngDec, intByteSize)
CnvDecToHex メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
lngDec	変換する１０進整数を指定します。必ず指定します。
intByteSize	１６進数に変換後、格納するバイト数を指定します。必ず指定します。 下位から格納されます。（下位は、上記指定文字列の右側です。） 1バイト（右から２文字分）：1 1ワード（右から４文字分）：2 2ワード（右から８文字分）：4

戻り値
戻り値は　１６進数文字列です。

使用例

```
'１０進数[305419896]を１６進数に変換し、その１バイトを格納する
Text1.Text = object.CnvDecToHex( 305419896, 1)
'テキストボックスには[78]と表示されます。
'(変換後、右から2文字のみを取得)

'１０進数[305419896]を１６進数に変換し、その１ワード分を格納する
Text1.Text = object.CnvDecToHex ( 305419896, 2)
'テキストボックスには[5678]と表示されます。
'(変換後、右から4文字のみを取得)

'１０進数[305419896]を１６進数に変換し、その２ワード分を格納する
Text1.Text = object.CnvDecToHex ( 305419896, 4)
'テキストボックスには[12345678]と表示されます。
'(変換後、右から8文字のみを取得)
```

CnvLongToSingleメソッド

長整数型を　単精度浮動小数点型に変換します

構文
sngData = object.CnvLongToSingle(lngData)
CnvLongToSingle メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
lngData	変換する長整数を指定します。必ず指定します。

戻り値
戻り値は　単精度浮動小数点型です。

使用例

```
'長整数[100] を単精度浮動小数点型に変換する
Text1.Text = object.CnvLongToSingle(100)
'テキストボックスには[1.401298E-43]と表示されます。
```

CnvSingleToLongメソッド

単精度浮動小数点型を 長整数型に変換します

構文
lngData = object.CnvSingleToLong(sngData)
CnvSingleToLong メソッドの構文は、次の指定項目から構成されます。

指定項目	内容
sngData	変換する単精度浮動小数点型を指定します。必ず指定します。

戻り値
戻り値は 長整数型です。

使用例

```
'単精度浮動小数点型[1.401298E-43]を長整数型に変換する
Text1.Text = object. CnvSingleToLong( 1.401298E-43)
'テキストボックスには[100]と表示されます。
```

イベント一覧

■モデム仕様

OnReceive

OnReceiveイベント

遠隔地にあるモデムからの送信に関して、コンピュータ側で受信したときに、このイベントが発生します。

構文

Private Sub Object_OnReceive(intStationNumber As Integer)
OnReceive イベントの構文は、次の指定項目から構成されます。

指定項目	内容
intStationNumber	接続されたPLCの局番として0が格納されます。

解説

このイベントは、遠隔地にあるモデムからの送信があり、コンピュータ側で受信したときに、発生します。
従って、先に、ReceivePortOpenメソッドで、モデム受信用のポートを必ずオープンしてください。
オープンされていなければ、このイベントは発生しません。
また、このイベント内での処理が終了すると、現在接続中のモデムとの通信が切断され、他の遠隔地からの受信待ちになります。(ポートはオープンしています。)
PLC内のデータを読込む処理等は、必ずこのイベント内に記述してください。

使用例

```
'遠隔地にあるモデムからの送信を受信したときに、データレジスタ100～109(DT100-109)を読込む
Private Sub Object_OnReceive( intStationNumber As Integer)
        Dim lngReturn as long
        Dim intData(0 to 10) As Integer
        lngReturn = object.AreaRead( intStationNumber, "DT", "100", 10, intData())
        'DT100に10が格納されていれば、intData(0)は10になります。
End Sub
```