37 IPC シリーズを GP として利用したい (WinGP)

この章では、GP-Pro EX で作成したプロジェクトファイルを IPC シリーズで動かし、接続機器と通信するための基本的な説明と、WinGP からアプリケーションを実行するための基本操作について説明します。

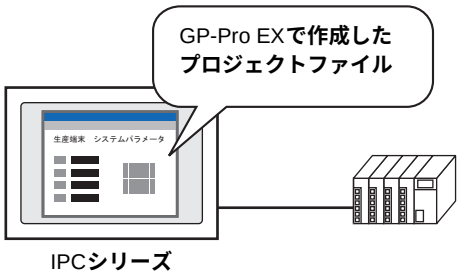
まず「37.4 設定メニュー」（37-32 ページ）をお読みいただき、目的に合った説明ページへ読み進んでください。

37.1 WinGP とは

37.1.1 WinGP とは

■ 概要

WinGP とは (株) デジタル製インダストリアルパネルコンピュータ（以降 IPC と称します）上で GP-Pro EX で作成したプロジェクトファイルを動かし、接続機器と通信できるアプリケーションです。ただし、GP と IPC のハードウェアの違いから、ご利用頂ける機能には相違があります。IPC の豊富なメモリを利用した機能拡張や独自に作成したプログラム (アプリケーション) との連携を行う機能などがあります。

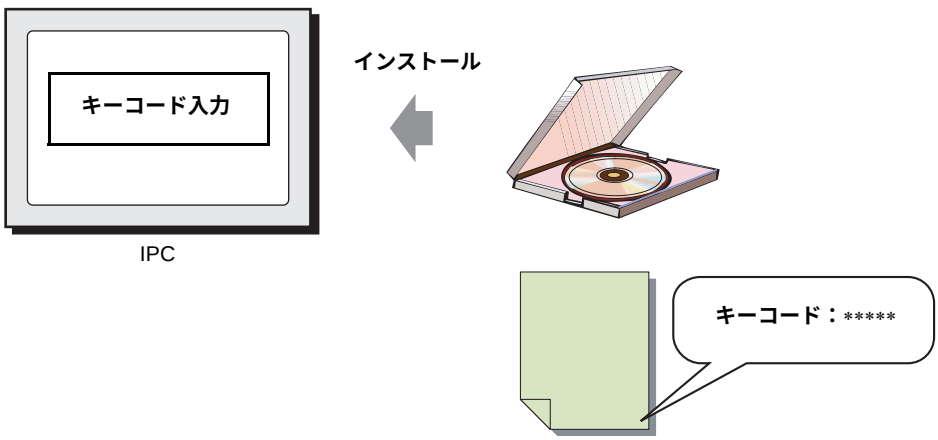


■ ライセンスについて

WinGP をご利用いただくには別途ライセンスをご購入いただく必要があります。
ライセンスをご購入いただくと、[キーコード] の記載された文書が発行されます。

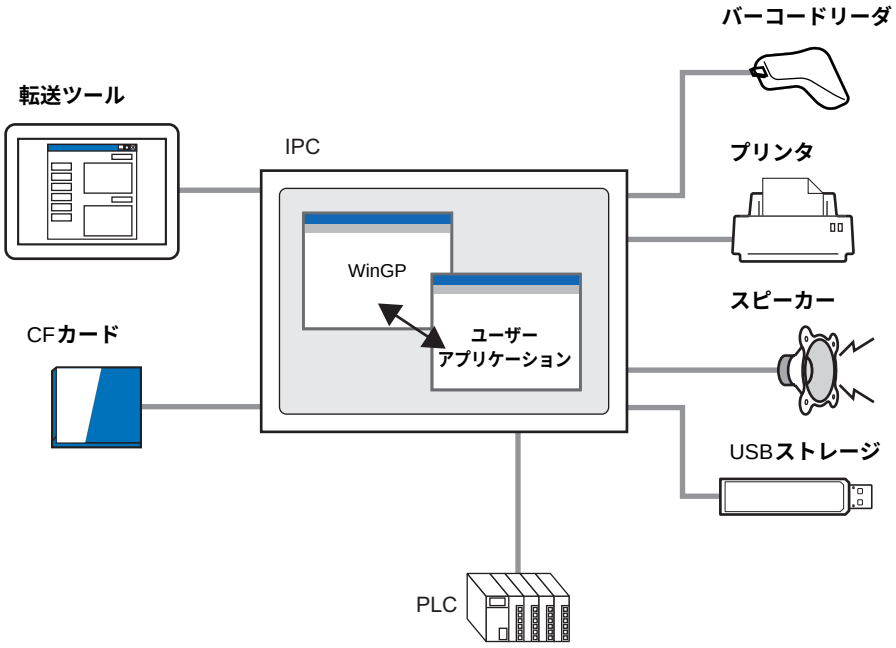
重　要

- **WinGP のインストール時にキーコードが必要です。別売の WinGP ライセンス（型式：EX-WINGP-IPC）を準備してください。**
  インストール手順は下記を参照してください。
  ☞「37.3.2 設定手順 ■ インストール / アンインストール」(37-10 ページ)
- **キーコードを紛失されると、再発行できません。キーコードは大切に保管してください。**



37.1.2 全体構成

WinGP 使用時の接続またはオプション環境は以下のようになります。



37.1.3 GP との相違点

IPC には大容量なメモリ、ストレージがあるため、GP-3500 シリーズと比較すると画面データ、記録データのサイズが以下のように拡張されます。

No.	項目	内容
1	ユーザデータ最大サイズ	8 MB→16 MB に拡張
2	SRAM 最大サイズ	512 KB→5 MB に拡張
3	1 画面最大部品数	384 部品 →1280 部品に拡張
4	1 画面最大デバイス数	1152 点 →3000 点に拡張
5	アラーム履歴記憶件数	768→10000 に拡張
6	アラームメッセージ登録件数	2048→10000 に拡張
7	DRAM 最大サイズ	320 KB→5 MB に拡張

■ WinGP で使用できない機能

WinGP により、IPC 上で GP とほぼ同じ機能が使用できますが、以下の機能は使用できません。

- ブザー音 /AUX 出力
- 2 次元コードリーダの USB 接続
- スクリプトでのプリンタ操作
- 動画録画 / 再生機能
- VM ユニットを使用したビデオ表示
- メモリローダ機能
- モデム転送機能
- バックライト切れ検出
- オフラインモードでの CF カードの初期化
- オフラインモードでのユーザーデータの初期化
- パススルー機能
- システムデータエリアのバックライト OFF、画面表示の ON、OFF 機能

MEMO

- IPC がサポートしている機能については、以下を参照してください。

☞「1.3 機種別サポート機能一覧」(1-5 ページ)

■ WinGP でのみ使用できる機能

機能	機能詳細
スイッチ部品	他のアプリケーションを起動する「アプリケーション起動」スイッチと WinGP を終了する「WinGP 終了」スイッチ が使用できます。
トリガアクション	他のアプリケーションを起動する動作 (EXE 実行動作) と WinGP を終了する動作 (WinGP 終了動作) が使用できます。
スクリプト	他のアプリケーションを起動する関数 (EXE 実行動作) と WinGP を終了する関数 (WinGP 終了動作) が使用できます。
デバイスアクセス API	IPC に接続している接続機器のデバイスを読み書きできる API です。

次のページに続きます。

機能	機能詳細
ハンドリング API	独自で作成したプログラム (アプリケーション) から WinGP の状態取得や、設定変更できる API です。
エラーログ機能	WinGP 通信中に表示されるエラーの内容を保存し、ファイルとして残します。
右クリック時のメニュー	ウィンドウを右クリックして表示されるメニューです。画面切り替えや、オフライン、オンラインモード切り替え、ウィンドウのフルスクリーンモード、最小化、ウィンドウ終了操作をこのメニューで行うことができます。

37.2 動作環境

37.2.1 対応機種

WinGP に対応している表示器は以下の 6 機種です。

■ IPC シリーズ

- PS3651A-T41
- PS3650A-T41
- PS3700A-T41-ASU-P41 (Rev.H 以降)
- PS3710A-T41
- PS3711A-T41-24V
- PS2000B-41 (Pentium III 1GHz) (Rev.M* 以降)

MEMO

- 各対応機種の仕様については、各機種のユーザーズマニュアルでご確認ください。
- 上記に記載のリビジョン以前の機種では、WinGP は起動しません。

◆対応 OS

WinGP に対応している OS は以下のとおりです。

- Windows2000 (Service Pack 3 以上)
- WindowsXP
- WindowsXP Embedded

MEMO

- 日本語 OS 以外の環境で WinGP を動作する場合、WinGP ウィンドウのメニューバー、右クリック時のメニュー、コピーツール、ポップアップメッセージは英語で表示されます。オフラインモードでは [本体設定] ー [メニューとエラー設定] ー [システム言語の設定] に従って表示されます。

37.2.2 対応プロトコル

■ 使用できるプロトコル

重 要

- **対応しているドライバであっても、接続方法によっては動作しない場合があります。必ず「GP-Pro EX 機器接続マニュアル」で接続方法についてご確認ください。**
- **対応しているドライバの最新情報については、(株) デジタルのサポート専用サイト「おたすけ Pro!」(http://www.proface.co.jp/otasuke/) で確認してください。**

WinGP に対応している接続機器ドライバは以下のとおりです。

メーカ	ドライバ名
(株) デジタル	メモリリンク
	汎用イーサネット
三菱電機 (株)	A シリーズ CPU 直結
	A シリーズ イーサネット
	A シリーズ 計算機リンク
	FX シリーズ CPU 直結
	FX シリーズ 計算機リンク
	Q シリーズ CPU 直結
	Q/QnA シリアルコミュニケーション
	Q/QnA シリーズ イーサネット
	QnA シリーズ CPU 直結
	QUTE シリーズ CPU 直結

次のページに続きます。

メーカ	ドライバ名
オムロン (株)	C/CV シリーズ 上位リンク
	CS/CJ シリーズ 上位リンク
	CS/CJ シリーズ イーサネット
	調節器 CompoWay/F
横河電機 (株)	パソコンリンク SIO
	パソコンリンク イーサネット
Siemens AG	SIMATIC S5 CPU 直結
	SIMATIC S7 3964(R)/RK512
	SIMATIC S7 イーサネット
Rockwell Automation	DF1
	EtherNet/IP
Schneider Electric Industries	MODBUS SIO マスタ
	MODBUS TCP マスタ
	Uni-Telway
(株) 安川電機	MEMOBUS SIO
	MEMOBUS イーサネット
	MP シリーズ SIO (拡張)
	MP シリーズ イーサネット (拡張)
(株) キーエンス	KV700/1000 シリーズ CPU 直結
(株) 山武	デジタル調節計 SIO
(株) 日立産機システム	H シリーズ SIO
	H シリーズイーサネット
(株) 明電舎	UNISEQUE シリーズイーサネット
GE Fanuc Automation	Series90 Ethernet
	Series 90-30/70 SNP
	Series 90-30/70 SNP-X
LS 産電 (株)	MASTER-K シリーズ Cnet
	XGT シリーズ FENet
Saia-Burgess Controls Ltd.	Saia S-Bus SIO
シャープ MS (株)	JW シリーズコンピュータリンク SIO
	JW コンピュータリンクイーサネット
ファナック (株)	Power Mate シリーズ
三菱重工業 (株)	DIASYS Netmation MODBUS TCP
松下電工 (株)	FP シリーズ コンピュータリンク SIO
富士電機機器制御 (株)	MICREX-F シリーズ SIO
(株) ジェイテクト	TOYOPUC CMP-LINK Ethernet
	TOYOPUC CMP-LINK SIO
理化工業 (株)	調節計 MODBUS SIO
	温度調節計

37.2.3 モデル環境

本章で、操作や説明をする場合のモデルとなるシステム構成は以下のとおりです。これ以外のシステム構成では、表示や各部の名称が異なることがありますが、同等の機能をもつものと読み替えてください。

■ 標準構成

機材・ソフト	モデルシステムの仕様	備考
OS	Windows®2000	—
接続機器	三菱電機 (株) 製 Q/QnA シリアルコミュニケーションシリーズ	—
IPC	PS-3650A	—

37.2.4 アプリケーションの開発環境

Microsoft® Visual Basic Ver.6.0

Microsoft® Visual C++ Ver.6.0 または Ver.7.0

Microsoft® Visual Studio .NET 2003 以上

37.3 開発の流れ

37.3.1 開発の流れ

IPC でプロジェクトファイルを動かす場合の WinGP のインストール、GP-Pro EX の起動、画面作成、接続機器との接続、運転までの流れを以下に示します。参照先をクリックすると説明ページにジャンプします。

37.3.2 設定手順

■ インストール / アンインストール

重 要

- **対応している** IPC **以外にインストールした場合は動作しません。**
- **インストール前はウィルスチェックソフトを含むすべてのプログラムを終了させてください。**
- Administrator **権限のあるユーザーアカウントでインストールしてください。**

- Windows XP Embedded **をご使用の方へ**

Windows XP Embedded をご利用で、C ドライブにインストールする場合、デフォルトの状態ではライトフィルタ（書き込み禁止設定）の設定がされています。そのため、インストールする前に、Windows XP Embedded の EWFSettingTool.exe で「EWF Disable」を選択し、ライトフィルタ設定を無効にする必要があります。

☞Windows XP Embedded **ユーザーズマニュアル「**3.1 **ライトフィルタの設定手順」**

- Pro-Server EX V1.10 **未満または** Pro-Server with Pro-Studio **をご使用の方へ**

既に Pro-Server EX V1.10 未満または Pro-Server with Pro-Studio がインストールされている IPC には WinGP はインストールできません。アンインストールするか Pro-Server EX V1.10 以上にバージョンアップをしてください。

☞**「**37.10.1 **インストール時の制限事項」(**37-147 **ページ)**

◆インストール手順

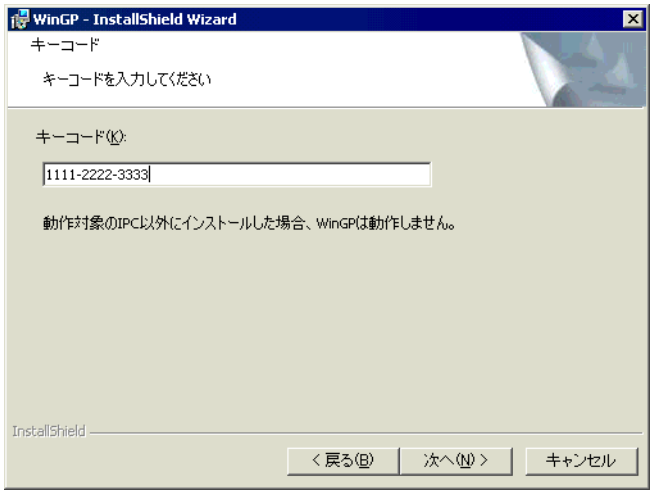
1 IPC（またはパソコン）に GP-Pro EX Ver.2.00 以上の CD-ROM をセットします。

2 以下のようなインストーラ画面が立ち上がりますので [ツールのインストール] を選択します。

3 [WinGP] を選択します。

4 自動的にインストールウィザードが起動しますので、ウィザードに従ってインストールを進めます。

5 途中、キーコードを入力する箇所がありますので、あらかじめ準備しておいた別売のキーコード（型式：EX-WINGP-IPC）を入力します。

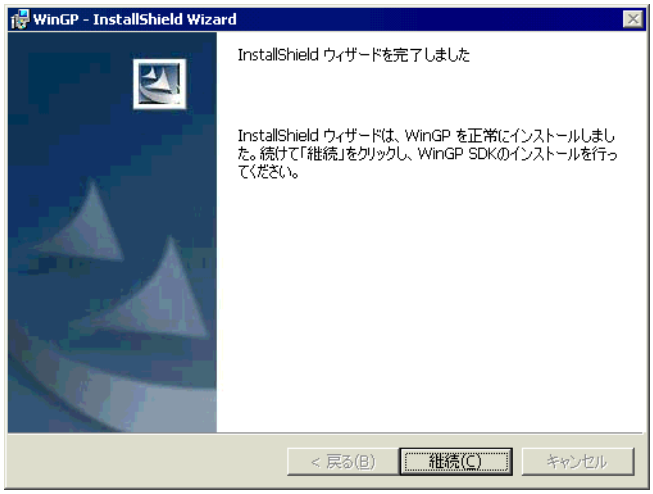


MEMO

- キーコードの詳細については下記を参照してください。

☞「37.1.1 WinGP **とは ■ ライセンスについて**」(37-2 **ページ**)

6 WinGP のインストールが終了したあと続けて WinGP SDK をインストールしますので [継続] をクリックします。



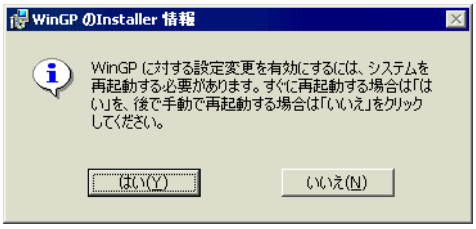
MEMO

- WinGP SDK とは、WinGP と VB.Net,VB または VC などで作成された外部アプリケーションと API で通信するためのソフトウェアです。既に Pro-Server EX V1.10 以上がインストールされている場合は、WinGP SDK のインストールは行われません。この場合は Pro-Server EX V1.10 でデバイスアクセス API が利用できます。WinGP のみインストールされます。インストール時の制限事項については下記を参照してください。

☞「37.10.1 **インストール時の制限事項**」(37-147 **ページ**)

7 以下のメッセージが表示され、インストールが完了します。

8 インストール終了、以下のメッセージが表示されますので [はい] を選択し、再起動します。



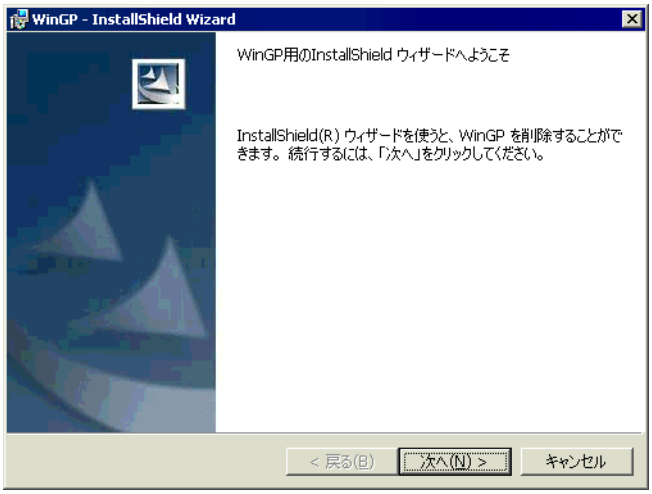
MEMO

- インストール後、WinGP を使用する前に必ず再起動してください。再起動せずに WinGP を起動すると正しく動作しません。

◆アンインストール

2 つアンインストールの方法があります。

- コントロールパネルの [プログラムの追加と削除] からアンインストール
  [スタート] ボタンから [設定 (S)] を選択し、[コントロールパネル (C)] をクリックすると [コントロールパネル] が開くので [プログラムの追加と削除] を選択します。インストールされているアプリケーションの一覧から [GP-Pro EX 2.00 WinGP] を選択して、[削除] をクリックするとアンインストールされます。
- GP-Pro EX の CD-ROM でアンインストール
  GP-Pro EX の CD-ROM をセットすると、アンインストールできます。GP-Pro EX の CD-ROM をセットすると、以下のような画面が表示されるので、[次へ (N)] を選択しウィザードにしたがってアンインストールを行います。



MEMO

- WinGP をアンインストールすると WinGP SDK もアンインストールされます。
- WinGP と Pro-Server EX V1.10 がインストールされているパソコンで、Pro-Server EX V1.10 だけをアンインストールすると API 通信を行うことができません。WinGP を再インストールしてください。

■ GP-Pro EX の起動～プロジェクト作成

GP-Pro EX を起動し、プロジェクトファイルを新規作成します。起動方法は「5.2.2 設定手順」の手順 1 ～ 3 と同様です。

◆表示器、接続機器の選択

MEMO

- 設定内容の詳細は設定ガイドを参照してください。

☞「5.14.2 [新規作成] の設定ガイド」(5-74 ページ)

1 [表示器タイプ] の [シリーズ] で [IPC シリーズ] を選択し、使用する機種を選択します。

2 IPC に接続する機器の [メーカー] と [シリーズ] を選択します。また IPC の COM ポートと接続機器をつなぐ場合は、[ポート] を COM1 ～ COM9 から選択します。

3 [通信設定] をクリックし、通信方式などを設定します。設定方法は「5.2.2 設定手順」の手順 6 ～ 7 と同様です。

◆作画、保存

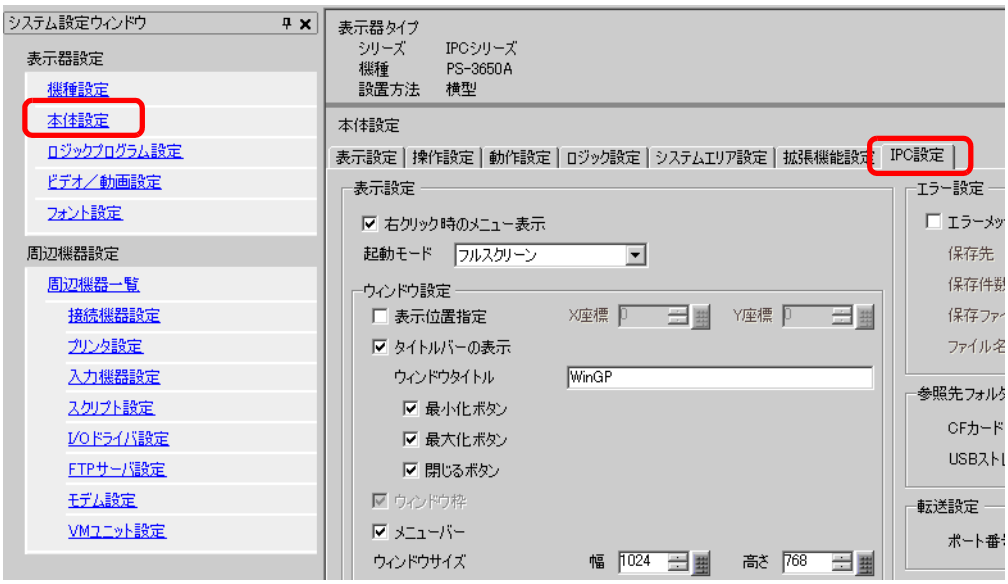
1 作画します。作画方法は「5.2.2 設定手順 ■ 作成・保存する」（5-12 ページ）を参照してください。またアラームなど使用したい機能に応じて、関連する章を参照してください。

重 要

- **GP と IPC のハードウェアの違いから、ご利用頂ける機能には相違があります。WinGP で使用できる機能については下記を参照してください。**

  ☞「37.1.3 GP との相違点」（37-4 ページ）

2 システム設定ウィンドウ [本体設定]-[IPC 設定] を開き、[右クリック時のメニュー表示] にチェックをいれます。チェックした場合、表示器に表示された画面を右クリックすることで、画面切り替えやオフラインモードへの移行が簡単に操作できるようになります。



3 [起動モード] で [ウィンドウ] を選択します。

4 必要に応じて、[ウィンドウ設定] でウィンドウの表示位置を設定したり、タイトルバーの表示 / 非表示を指定します。

5 アラーム機能やサンプリング機能、レシピ（ファイリングデータ転送）機能など、バックアップ SRAM を使用する設定を行っている場合は、[履歴データ保存設定] の [履歴データ格納先] に、バックアップ SRAM の代わりとしてデータを格納するためのフォルダのパスを入力します。

MEMO
- パスを入力しなかった場合は、WinGP のインストール先フォルダ以下の「NAND¥PRJ001¥USER¥SCREEN」に保存されます。

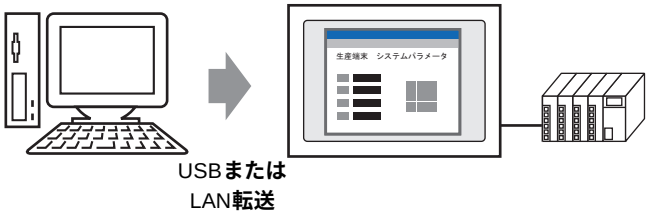
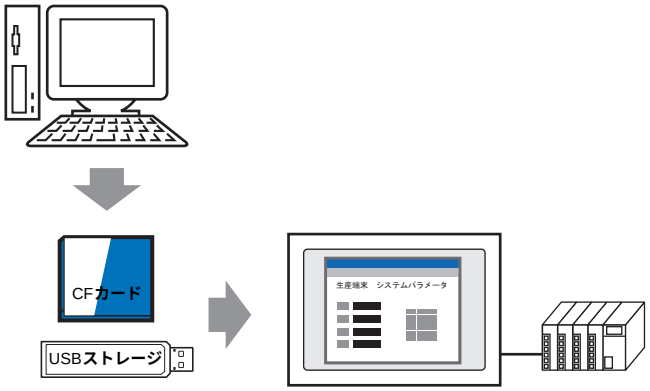
6 [CF カード出力先フォルダ] や [USB ストレージ出力先フォルダ] を設定している場合は、画面転送により出力されるデータの保存先となるフォルダのパスを [参照先フォルダ指定] に入力します。WinGP はここで指定したフォルダ内のデータ（レシピデータなど）を参照して動作します。

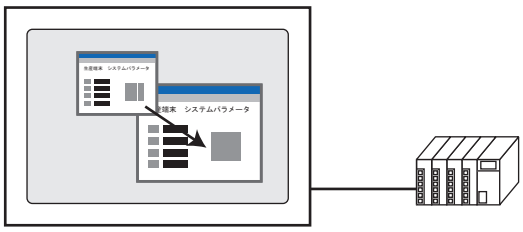
MEMO
- パスを入力しなかった場合は、WinGP のインストール先フォルダ以下の「CFA00」フォルダまたは「USBHD」フォルダに保存されます。
- 参照先フォルダは、出力先フォルダとは異なるフォルダを指定してください。転送先と転送元が同じになると転送エラーが発生します。

7 [プロジェクト (F)] メニューから [名前を付けて保存] を選択します。ファイル名と保存場所を指定して保存します。

■ 転送

IPC 本体にプロジェクトファイルを転送します。別のパソコンで GP-Pro EX の作画を行った後、IPC にプロジェクトファイルを転送する場合と同一 IPC 上に GP-Pro EX と WinGP をインストールする場合では転送方法が異なります。

◆ 別のパソコンで GP-Pro EX の作画を行った後、IPC にプロジェクトファイルを転送する場合	
USB ケーブル /LAN ケーブルで転送する場合  	☞「• USB ケーブル/LAN ケーブルで転送する場合」(37-18 ページ)
CF カードや USB ストレージなどから転送する場合  	☞「• CF カードや USB ストレージから転送する場合」(37-20 ページ)

◆ GP-Pro EX と WinGP が同じ IPC にインストールされている場合	
	☞「◆ GP-Pro EX と WinGP が同じ IPC にインストールされている場合」(37-22 ページ)

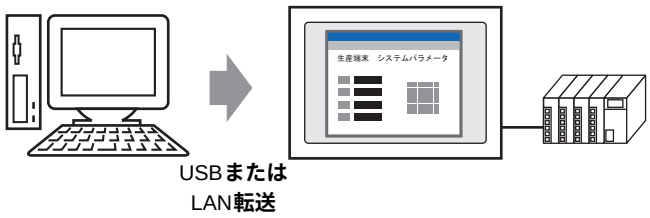
重 要

- Windows XP Embedded をご使用の場合、デフォルトの状態ではライトフィルタ (書き込み禁止設定) が設定されています。そのため、プロジェクトファイルを転送する前に、Windows XP Embedded の EWFSettingTool.exe で「EWF Disable」を選択し、ライトフィルタ設定を無効にしてください。

  ☞ Windows XP Embedded ユーザーズマニュアル「3.1 ライトフィルタの設定手順」

◆別のパソコンで GP-Pro EX の作画を行った後、IPC にプロジェクトファイルを転送する場合

- USB ケーブル /LAN ケーブルで転送する場合



1 [スタート] から [プログラム (P)] ― [Pro-face] ― [WinGP] ― [WinGP] を選択、またはデスクトップの をダブルクリックして WinGP を起動します。

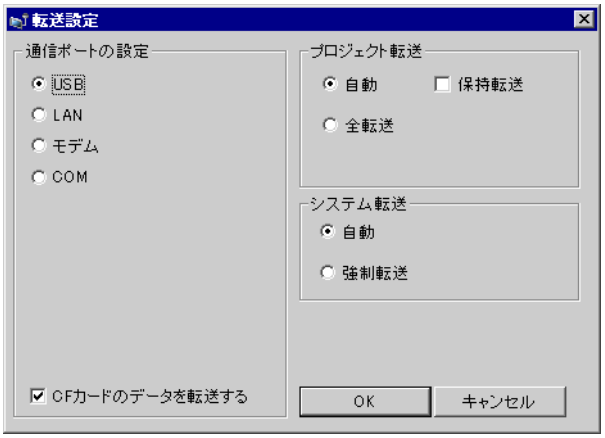
MEMO

- オフライン画面が表示されている状態では転送できません。必ず WinGP はオンラインにしてください。

2 GP-Pro EX の状態ツールバーから [画面転送] をクリックして転送ツールを起動します。

3 [プロジェクト情報] で転送するプロジェクトファイル名などを確認します。違うプロジェクトファイルを転送したい場合は、[プロジェクト選択] ボタンをクリックしてプロジェクトファイルを選択できます。

4 [転送設定情報] で USB または LAN になっているか確認します。[USB] または [LAN] になっていない場合は、[転送設定] ダイアログボックスが表示されるので [通信ポート] の設定で [USB] または [LAN] を選択し、OK をクリックします。



MEMO

- モデムでの転送はできません。

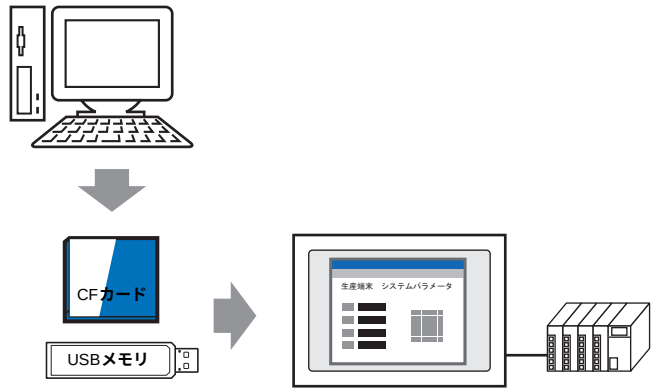
5 [プロジェクト送信] をクリックします。
その後の手順は GP と同じです。下記を参照してください。

☞「33.2 USB **転送ケーブルで転送したい**」(33-5 **ページ**)
☞「33.3 **イーサネット**（LAN）**で転送したい**」(33-12 **ページ**)

MEMO

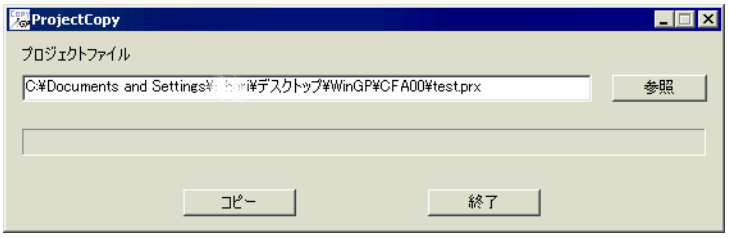
- イーサネット (LAN) で転送する場合は、IPC の [マイネットワーク] のプロパティ ー [ローカルエリア接続のプロパティ] ー [インターネット プロトコル (TCP/IP)] のプロパティで IP アドレスを設定してください。WinGP のオフラインでは IP アドレスを設定できません。

- CF カードや USB ストレージから転送する場合



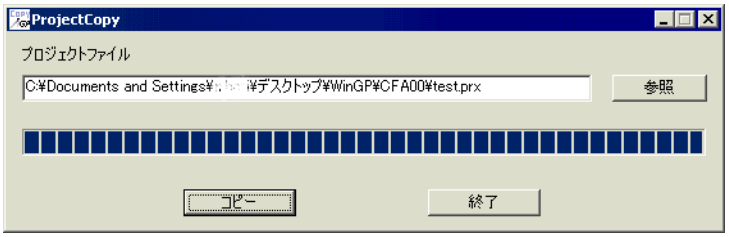
1 WinGP を終了します。WinGP が起動していると転送できません。

2 [スタート] メニューから [プログラム (P)] ― [Pro-face] ― [WinGP] ― [ProjectCopy] の順にクリックし、ProjectCopy（コピーツール）を起動します。



3 [プロジェクトファイル] の 参照 アイコンをクリックし、CF カードや USB ストレージ、デスクトップなどに保存されている GP-Pro EX のプロジェクトファイル (*.prx) を指定します。

4 [コピー] をクリックします。転送中は以下のダイアログボックスが表示されます。

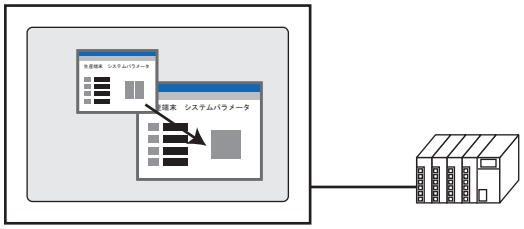


5 コピーが完了すると以下のメッセージが表示されます。

MEMO

- ProjectCopy [コピーツール] は、画面データの送信のみできます。画面データの受信やプロジェクトの全転送はできません。以下の場合は、転送ツールを使用してください。
  - WinGP インストール後、はじめてプロジェクトファイルを転送する場合
  - 接続機器を変更、追加した場合
  - 使用するフォントを変更、追加した場合
  - GP-Pro EX のバージョンアップによりラインタイムシステムやプロトコルドライバがバージョンアップされた後に、プロジェクトファイルを更新した場合
- コピーツールでは WinGP のシステムプログラムを送信できません。WinGP をバージョンアップする場合は転送ツールを使用してください。

◆GP-Pro EX と WinGP が同じ IPC にインストールされている場合



1 [スタート] から [プログラム (P)] ― [Pro-face] ― [WinGP] ― [WinGP] を選択、またはデスクトップの をダブルクリックして WinGP を起動します。

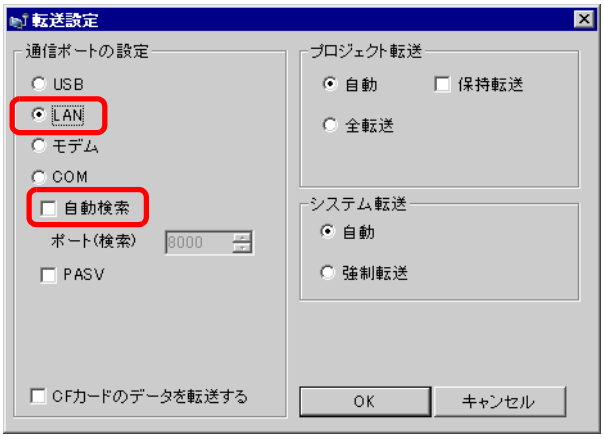
MEMO

- オフライン画面が表示されている状態では転送できません。必ず WinGP はオンラインにしてください。

2 GP-Pro EX の状態ツールバーから [画面転送] をクリックして転送ツールを起動します。

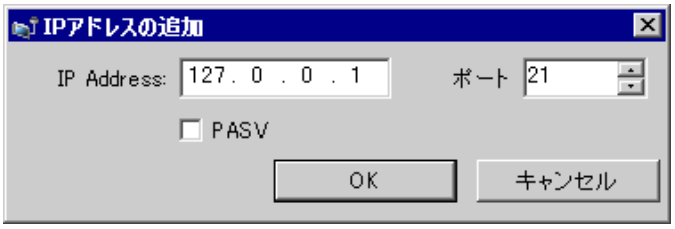
3 [プロジェクト情報] で転送するプロジェクトファイル名などを確認します。違うプロジェクトファイルを転送したい場合は、[プロジェクト選択] ボタンをクリックしてプロジェクトファイルを選択できます。

4 [転送設定] ボタンをクリックします。
[通信ポートの設定] で [LAN] を選択し、[自動検索] チェックボックスを OFF にし、OK をクリックします。



5 [プロジェクト送信] をクリックすると、[本体の選択] ダイアログボックスが開きます。

6 [追加] ボタンを選択し、[IP Address] に [127.0.0.1] と入力し、[OK] をクリックします。

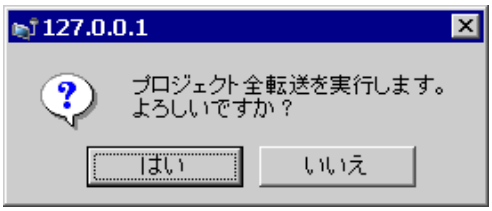


MEMO

- IP アドレス [127.0.0.1] とは、ネットワーク上で現在作業中のコンピュータを表す仮想的なアドレスです。
- [ポート番号] は、システム設定ウィンドウ [本体設定]-[IPC 設定] タブの [転送設定] で指定したポート番号にあわせてください。

7 [IP アドレス] に表示されている [127.0.0.1] のチェックボックスを ON にし、[OK] をクリックします。

8 次のようなダイアログボックスが表示されるので [はい] をクリックします。(同じプロジェクトを再度転送する場合は表示されません)



プロジェクトファイルの転送は [ProjectCopy]（コピーツール）でも転送できます。設定手順は以下を参照してください。

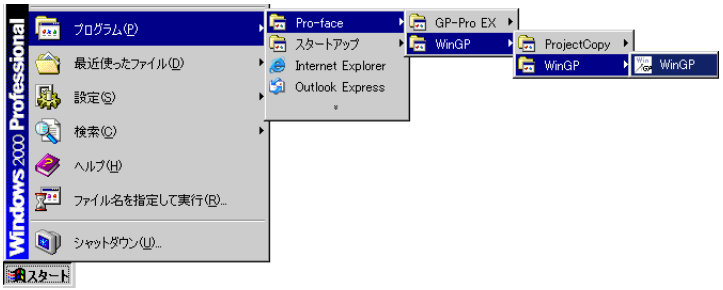
☞「• CF カードや USB ストレージから転送する場合」(37-20 ページ)

■ WinGP の起動・運転

1 IPC と接続機器を接続します。

MEMO
- 通信設定および接続ケーブルなどについては「GP-Pro EX 機器接続マニュアル」を参照してください。

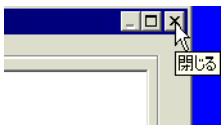
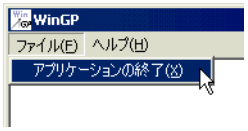
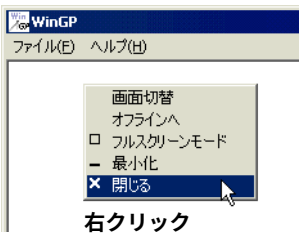
2 [スタート] メニューから [プログラム (P)] ― [Pro-face] ― [WinGP] ― [WinGP] の順にクリックし WinGP を起動します。



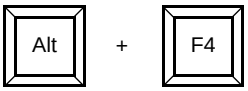
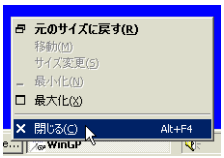
MEMO
- デスクトップ画面上のショートカットをダブルクリックして起動することもできます。

■ WinGP の終了

WinGP を終了します。WinGP は以下の 9 とおりの方法で終了できます。

1	タイトルバーの「閉じる」ボタン	
2	メニューバーの「ファイル」→「アプリケーションの終了」	
3	WinGP 画面で右クリック →「閉じる」	 右クリック

次のページに続きます。

4	スイッチ部品による終了 ☞「◆ スイッチ部品による終了」(37-27 ページ)	
5	D スクリプトによる終了 ☞「◆ D スクリプトによる終了」(37-30 ページ)	
6	トリガアクションによる終了	
7	キーボードの「Alt+F4 キー」を押す	
8	タスクバーの右クリック→「閉じる」	
9	API による終了 ☞「◆ 関数一覧・終了操作」(37-75 ページ)	API 名：StopRuntime()

◆スイッチ部品による終了

WinGP を終了するスイッチを作成します。

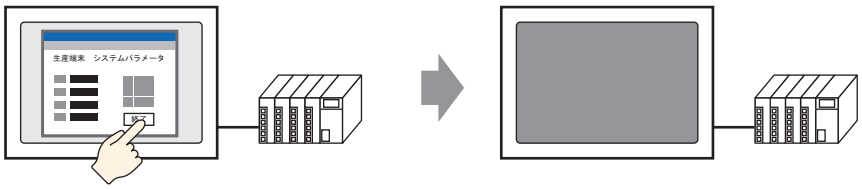
MEMO

- 設定内容の詳細は設定ガイドを参照してください。

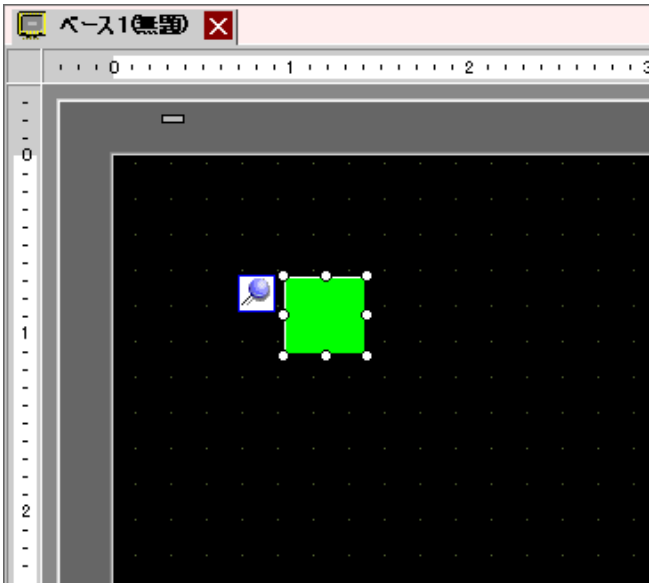
  ☞「11.14 スイッチランプ部品の設定ガイド」（11-42 ページ）

- 部品の配置方法やアドレス・形状・色・銘板の設定方法詳細は、「部品の編集手順」を参照してください。

  ☞「9.6.1 部品の編集手順」（9-37 ページ）



1 [部品 (P)] メニューの [スイッチランプ (C)] から [特殊スイッチ (P)] を選択するか、ツールバーから  をクリックし、画面に配置します。



2 配置したスイッチをダブルクリックすると設定ダイアログボックスが開きます。

3 [形状選択] でスイッチの形状を選択します。

MEMO
- スイッチの形状によっては色が変更できないものがあります。

4 [特殊動作] で [WinGP の終了] を選択します。

MEMO

- [確認ダイアログを表示] のチェックボックスを ON した場合、WinGP 上でスイッチをタッチしたときに以下のダイアログが表示されます。

◆D スクリプトによる終了

MEMO

- 設定内容の詳細は設定ガイドを参照してください。

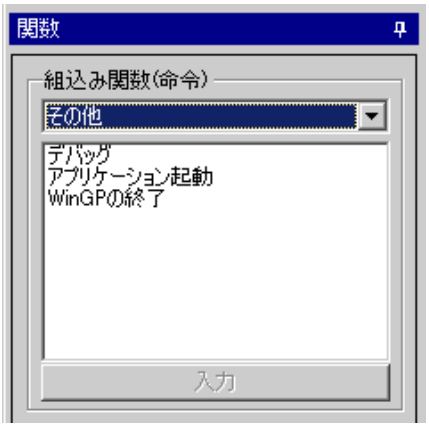
  ☞「20.8.1 Dスクリプト/共通設定[グローバルDスクリプト設定]の設定ガイド」(20-48 ページ)

- [共通設定 (R)] メニューの [グローバル D スクリプト設定 (L)] および [拡張スクリプト設定 (E)] でも同様に WinGP の終了ができます。

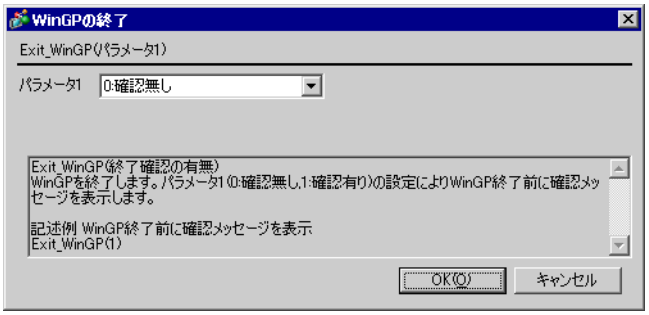
1 [部品 (P)] メニューから [D スクリプト (R)] を選択し、[D スクリプト一覧] ダイアログボックスで [作成] をクリックします。

2 [関数] タブをクリックします。[組み込み関数] (命令) はスクリプトで使用できる命令をクリックするだけで簡単に配置することができます。

3 [組み込み関数 (命令)] のプルダウンメニューから [その他] をクリックします。

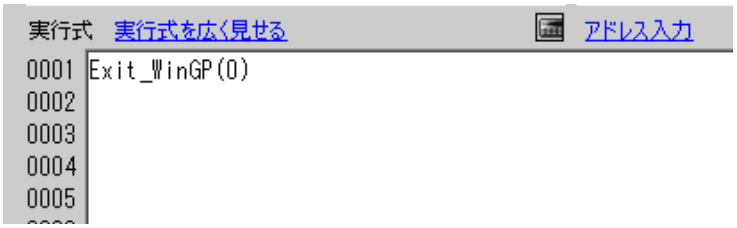


4 [WinGP の終了] をダブルクリックし、以下のダイアログボックスでパラメータの設定を行います。

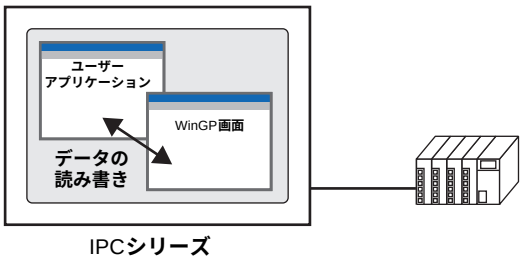


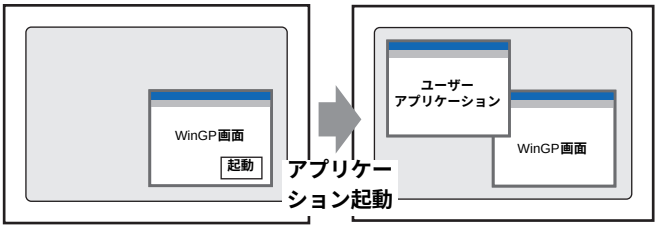
パラメータ 0	0: 確認無し	確認ダイアログは表示されず、直ちに WinGP が終了します。
パラメータ 1	1: 確認有り	WinGP 上で以下のダイアログが表示され、[はい] をクリックすると WinGP を終了します。 

5 [OK] をクリックすると [実行式] に「Exit_WinGP (0) 」または「Exit_WinGP (1) 」が入力されます。



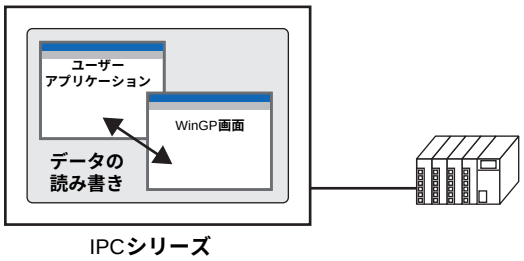
37.4 設定メニュー

ユーザアプリケーションからWinGPの情報取得や操作をしたい	
API を使用し、WinGP とユーザーアプリケーション間でデータの読み書きや操作ができます。  IPCシリーズ	☞ 設定手順（37-34 ページ） ☞ 詳細（37-33 ページ）

WinGPからアプリケーションを実行したい	
WinGP の画面上から他のアプリケーションを実行することができます。  IPCシリーズ　IPCシリーズ	☞ スイッチ起動の設定手順（37-66 ページ） ☞ 詳細（37-65 ページ）

37.5 ユーザアプリケーションから WinGP の情報取得や操作をしたい

37.5.1 詳細



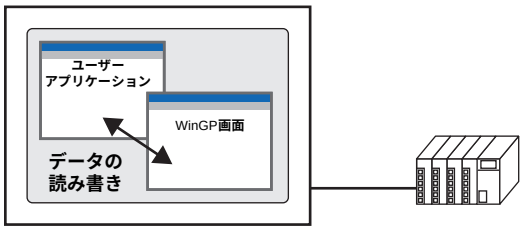
API を使用し、ユーザアプリケーションから WinGP の情報取得や操作ができます。

37.5.2 設定手順

MEMO

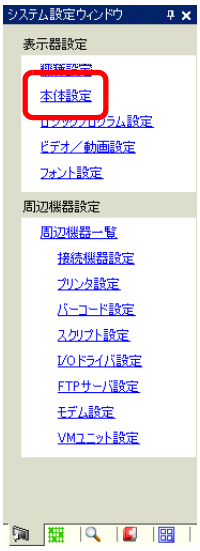
- 設定内容の詳細は設定ガイドを参照してください。

☞「37.9.1 システム設定ウィンドウ [本体設定] ー [IPC 設定] の設定ガイド」(37-136 ページ)

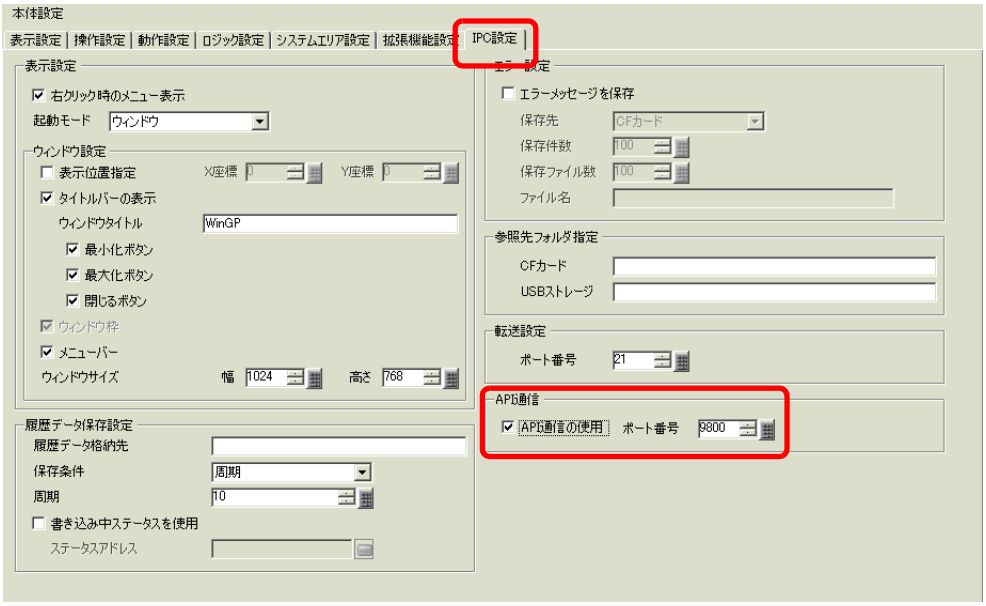


API を使用し、ユーザアプリケーションから WinGP の情報取得や操作ができます。

1 GP-Pro EX のシステム設定ウィンドウで [本体設定] を開きます。



2 [IPC 設定] タブを開き、[API 通信] チェックボックスにチェックをいれ、使用するポート番号を 0 ～ 65535 の間で設定します。[転送設定] の [ポート番号] と重ならない番号を指定してください。



MEMO

- 他の接続機器との通信および FTP 通信とポート番号が重ならないようにしてください。
- ポート番号 8000 ～ 8019 番は転送用ポート番号に指定されているので設定しないでください。

3 プロジェクトファイルを保存し、IPC シリーズへ転送します。

4 WinGP と接続機器の通信を確認します。

5 API を使用するためのプログラミングアプリケーションの設定を行います。

＜ VB.NET でデバイスアクセス API を使用する場合＞

VB.NET でソリューションエクスプローラを開き、[参照設定] で右クリックし、[参照の追加] を選択します。

[参照の追加] ダイアログボックスで [参照] をクリックし、以下のファイルを選択します。
（GP-Pro EX の CD-ROM 内）¥WinGP¥SDK¥Pro-SDK¥DotNet¥BIN¥WinGPAPIDotNet.dll
[開く] をクリックし、[OK] を選択します。
ソースコードの先頭に「Imports ProEasyDotNet」と記述します。

＜ VB6 でデバイスアクセス API を使用する場合＞
VB6 のメニューバーから [プロジェクト] － [標準モジュールの追加] を選択し、以下のモジュールを追加します。
（GP-Pro EX の CD-ROM 内）¥WinGP¥SDK¥Pro-SDK¥VB¥API¥WinGPAPI.bas

＜ VB.NET でハンドリング API を使用する場合＞
VB.NET のメニューバーから [プロジェクト] － [既存項目の追加] を選択し、以下のモジュールを追加します。
（GP-Pro EX の CD-ROM 内）¥WinGP¥SDK¥Pro-SDK¥DotNet¥BIN¥RtCtrlAPI.vb

＜ VB6 でハンドリング API を使用する場合＞
VB6 のメニューバーから [プロジェクト] － [標準モジュールの追加] を選択し、以下のモジュールを追加します。
（GP-Pro EX の CD-ROM 内）¥WinGP¥SDK¥Pro-SDK¥VB¥API¥RtCtrlAPI.bas

6 プログラミングを行います。

MEMO

☞「37.5.3 データの読み書きをするサンプル（デバイスアクセス API）■ サンプル概要」（37-37 ページ）
☞「37.5.4 WinGP の状態を取得 / 設定変更するサンプル（ハンドリング API）■ サンプル概要」（37-52 ページ）

7 IPC シリーズに作成したユーザーアプリケーションをセットアップします。

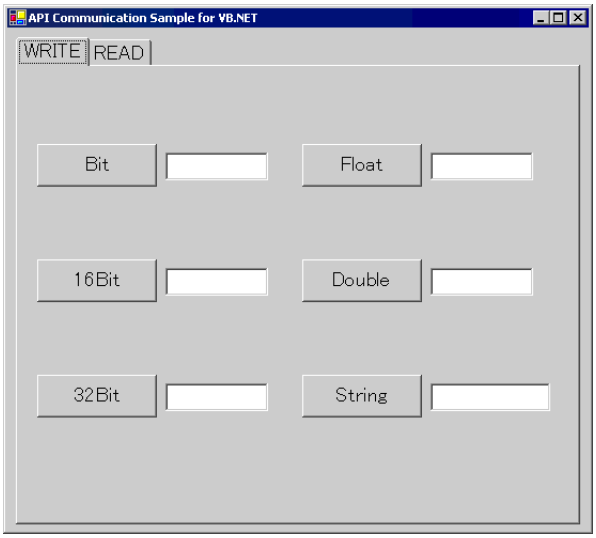
8 WinGP およびユーザーアプリケーションを起動します。

37.5.3 データの読み書きをするサンプル（デバイスアクセス API）

ここでは以下のようなサンプルアプリケーションを例に、API 通信時のプログラムについて説明します。

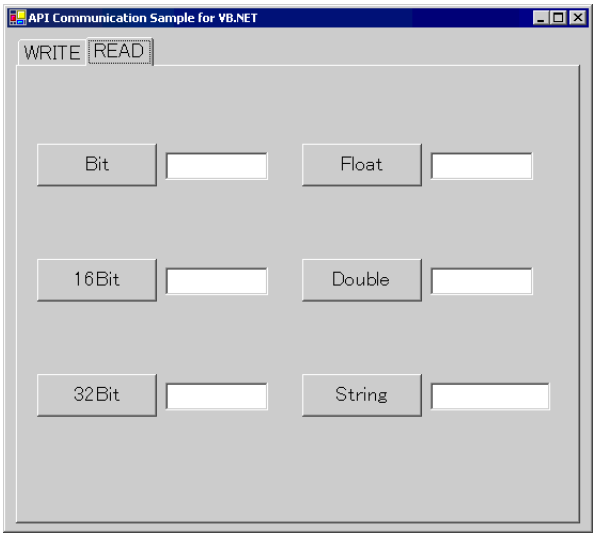
■ サンプル概要

- 書き込み



各ボタンをクリックし、テキストボックスに入力したデータを書き込みます。

- 読み込み



各ボタンをクリックし、テキストボックスにデータを読み込みます。

本サンプルでは例として以下のシンボルを使用しています。

シンボル名	シンボルが参照しているアドレス
Buf_Bit	USR200.00 ビット目
Buf_16	USR201
Buf_32	USR203
Buf_Float	USR207
Buf_Double	USR209
Buf_Str	USR213

■ デバイスアドレスを直接指定する方法

- WinGP に接続機器ドライバを 1 つのみ指定している場合
  WriteDeviceBit("#WinGP", "M100", nDataAry(0), 1)
- WinGP に接続機器ドライバを 1 つ以上指定している場合
  WriteDeviceBit("#WinGP.PLC1", "M100", nDataAry(0), 1)
  ↑ WinGP に接続されている接続機器名です。
- メモリリンクドライバを使用している場合
  WriteDeviceBit("#WinGP.#MEMLINK", "10000", nDataAry(0), 1)
- WinGP の内部デバイスを使用したい場合
  WriteDeviceBit("#WinGP", "USR10000", nDataAry(0), 1)
  WriteDeviceBit("#WinGP", "LS10000", nDataAry(0), 1)
  または
  WriteDeviceBit("#WinGP.#INTERNAL", "USR10000", nDataAry(0), 1)
  WriteDeviceBit("#WinGP.#INTERNAL ", "LS10000", nDataAry(0), 1)

■ VB.Net 2003 プログラム例

サンプルプログラムの場所：（GP-Pro EX の CD-ROM 内）¥WinGP¥SDK¥Pro-SDK¥DotNet¥EasySmpl

```
Imports ProEasyDotNet

Public Class Form1
    Inherits System.Windows.Forms.Form

    #Region " Windows フォームデザイナで生成されたコード "

        Public Sub New()
            MyBase.New()

            ' この呼び出しは Windows フォームデザイナで必要です。
            InitializeComponent()

            ' ProEasy 初期化 .
            Dim iResult As Integer = ProEasy.EasyInit() ' WinGP SDK を最初に一度だけ初期化します。
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
            End If
```

ProEasy**オブジェクトをインポートします。**

InitializeComponent() **呼び出しの後に初期化を追加します。**

```
        End Sub

        ' Form は、コンポーネント一覧に後処理を実行するために dispose をオーバーライドします。
        Protected Overloads Overrides Sub Dispose(ByVal disposing As Boolean)
            If disposing Then
                If Not (components Is Nothing) Then
                    components.Dispose()
                End If
            End If
        MyBase.Dispose(disposing)
        End Sub
```

～中略（以下の Windows フォームデザイナで生成されたコードは省略します）～

```
    #End Region

    Private Sub ReadBit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ReadBit.Click

    End Sub

    Private Sub Read16_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Read16.Click

        Try

            ' 読み込みデータ .
            Dim nDataAry(1) As Short

            ' 読み込み .
            Dim iResult As Integer = ProEasy.ReadDevice16("#WinGP", "Buf_16", nDataAry, 1)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

            Me.Buf_16.Text = CStr(nDataAry(0))

            Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub Read32_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Read32.Click

        Try

            ' 読み込みデータ .
```

ここではGP-Pro EXで設定した"Buf_16"（USR201）というシンボルを使用しています。デバイスアドレスを直接指定することもできます。
☞「■ デバイスアドレスを直接指定する方法」（37-38 ページ）

```
            Dim nDataAry(1) As Integer

            ' 読み込み .
            Dim iResult As Integer = ProEasy.ReadDevice32("#WinGP", "Buf_32", nDataAry, 1)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

            Me.Buf_32.Text = CInt(nDataAry(0))

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub ReadBCD16_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ReadBCD16.Click

        Try
            ' 読み込みデータ .
            Dim nDataAry(1) As Short

            ' 読み込み .
            Dim iResult As Integer = ProEasy.ReadDeviceBCD16("#WinGP", "Buf_BCD16", nDataAry, 1)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

            Me.Buf_BCD16.Text = CShort(nDataAry(0))

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub ReadBCD32_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ReadBCD32.Click

        Try
            ' 読み込みデータ .
            Dim nDataAry(1) As Integer

            ' 読み込み .
            Dim iResult As Integer = ProEasy.ReadDeviceBCD32("#WinGP", "Buf_BCD32", nDataAry, 1)
```

```
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

            Me.Buf_BCD32.Text = CInt(nDataAry(0))

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub ReadFloat_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ReadFloat.Click

        Try
            ' 読み込みデータ .
            Dim nDataAry(1) As Single

            ' 読み込み .
            Dim iResult As Integer = ProEasy.ReadDeviceFloat("#WinGP", "Buf_Float", nDataAry, 1)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

            Me.Buf_Float.Text = CSng(nDataAry(0))

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub ReadDouble_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ReadDouble.Click

        Try
            ' 読み込みデータ .
            Dim nDataAry(1) As Double

            ' 読み込み .
            Dim iResult As Integer = ProEasy.ReadDeviceDouble("#WinGP", "Buf_Double", nDataAry, 1)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
```

```
            End If

            Me.Buf_Double.Text = CDbl(nDataAry(0))

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub ReadStr_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ReadStr.Click

        Try
            ' 読み込みデータ .
            Dim nDataAry As String

            ' 読み込み .
            Dim iResult As Integer = ProEasy.ReadDeviceStr("#WinGP", "Buf_Str", nDataAry, 10)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

            Me.Buf_Str.Text = nDataAry

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub ReadVariant_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ReadVariant.Click

    End Sub

    Private Sub ReadSymbol_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ReadSymbol.Click

    End Sub

    Private Sub WriteBit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
WriteBit.Click

        Try
            ' 書き込みデータ .
            Dim nDataAry(1) As Short
            nDataAry(0) = CShort(Val(Me.WBuf_Bit.Text))
```

```
            ' 書き込み .
            Dim iResult As Integer = ProEasy.WriteDeviceBit("#WinGP", "Buf_16", nDataAry, 1)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub Write16_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Write16.Click

        Try
            ' 書き込みデータ .
            Dim nDataAry(1) As Short
            nDataAry(0) = CShort(Val(Me.WBuf_16.Text))

            ' 書き込み .
            Dim iResult As Integer = ProEasy.WriteDevice16("#WinGP", "Buf_16", nDataAry, 1)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub Write32_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Write32.Click

        Try
            ' 書き込みデータ .
            Dim nDataAry(1) As Integer
            nDataAry(0) = CInt(Val(Me.WBuf_32.Text))

            ' 書き込み .
            Dim iResult As Integer = ProEasy.WriteDevice32("#WinGP", "Buf_32", nDataAry, 1)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
```

```
            MsgBox(sErrMsg)
        End If

    Catch ex As Exception

        MsgBox(ex.Message)

    End Try

End Sub

Private Sub WriteBCD16_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
WriteBCD16.Click

    Try
        ' 書き込みデータ .
        Dim nDataAry(1) As Short
        nDataAry(0) = CShort(Val("&h" + Me.WBuf_BCD16.Text))

        ' 書き込み .
        Dim iResult As Integer = ProEasy.WriteDevice16("#WinGP", "Buf_BCD16", nDataAry, 1)
        If iResult Then
            Dim sErrMsg As String
            ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
            MsgBox(sErrMsg)
        End If

    Catch ex As Exception

        MsgBox(ex.Message)

    End Try

End Sub

Private Sub WriteBCD32_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
WriteBCD32.Click

    Try
        ' 書き込みデータ .
        Dim nDataAry(1) As Integer
        nDataAry(0) = CInt(Val("&h" + Me.WBuf_BCD16.Text))

        ' 書き込み .
        Dim iResult As Integer = ProEasy.WriteDeviceBCD32("#WinGP", "Buf_BCD32", nDataAry, 1)
        If iResult Then
            Dim sErrMsg As String
            ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
            MsgBox(sErrMsg)
        End If

    Catch ex As Exception

        MsgBox(ex.Message)
```

```
        End Try

    End Sub

    Private Sub WriteFloat_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
WriteFloat.Click

        Try
            ' 書き込みデータ .
            Dim nDataAry(1) As Single
            nDataAry(0) = CSng(Val(Me.WBuf_Float.Text))

            ' 書き込み .
            Dim iResult As Integer = ProEasy.WriteDeviceFloat("#WinGP", "Buf_Float", nDataAry, 1)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub WriteDouble_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
WriteDouble.Click

        Try
            ' 書き込みデータ .
            Dim nDataAry(1) As Double
            nDataAry(0) = CDbl(Val(Me.WBuf_Double.Text))

            ' 書き込み .
            Dim iResult As Integer = ProEasy.WriteDeviceDouble("#WinGP", "Buf_Double", nDataAry, 1)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub WriteString_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
WriteString.Click
```

```
        Try
            ' 書き込みデータ .
            Dim nDataAry As String
            nDataAry = Me.WBuf_Str.Text

            ' 書き込み .
            Dim iResult As Integer = ProEasy.WriteDeviceStr("#WinGP", "Buf_Str", nDataAry, 10)
            If iResult Then
                Dim sErrMsg As String
                ProEasy.EasyLoadErrorMessageEx(iResult, sErrMsg)
                MsgBox(sErrMsg)
            End If

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

    End Sub

    Private Sub WriteVariant_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles WriteVariant.Click

        ' VB.NET では Variant 型が廃止され Object 型を使うようになります。
        ' それに伴い WriteDeviceVariant() は
        '                 WriteDeviceEasyObject() に変更されました。

    End Sub

    Private Sub WriteSymbol_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
WriteSymbol.Click

        ' WriteSymbol 系は WriteSymbolVariant() しか見当たらない .

    End Sub

End Class
```

■ VB6 プログラム例

サンプルプログラムの場所：（GP-Pro EX の CD-ROM 内）¥WinGP¥SDK¥Pro-SDK¥VB¥EasySmpl

```
Option Explicit

Private Sub Form_Load()

    Dim iResult As Long

    iResult = EasyInit()
    If iResult Then
        Dim sErrMsg As String
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessageEx(iResult, sErrMsg)
    End If

End Sub

' ------------------------------------------------------------------------------
' WriteDeviceXXX()
' ------------------------------------------------------------------------------

Private Sub WriteBit_Click()

    ' 書き込みデータ .
    Dim nDataAry(1) As Integer
    nDataAry(0) = CInt(Val(Me.WBuf_Bit.Text))

    ' 書き込み .
    Dim iResult As Long
    iResult = WriteDeviceBit("#WinGP", "Buf_Bit", nDataAry(0), 1)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessageEx(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

End Sub
```

ここではGP-Pro EXで設定した"Buf_16"（USR201）というシンボルを使用しています。デバイスアドレスを直接指定することもできます。

☞**「■ デバイスアドレスを直接指定する方法」**（37-38 **ページ**）

```
Private Sub Write16_Click()

    ' 書き込みデータ .
    Dim nDataAry(1) As Integer
    nDataAry(0) = CInt(Val(Me.WBuf_16.Text))

    ' 書き込み .
    Dim iResult As Long
    iResult = WriteDevice16("#WinGP", "Buf_16", nDataAry(0), 1)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessageEx(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

End Sub

Private Sub Write32_Click()

    ' 書き込みデータ .
    Dim nDataAry(1) As Long
    nDataAry(0) = CLng(Val(Me.WBuf_32.Text))

    ' 書き込み .
    Dim iResult As Long
    iResult = WriteDevice32("#WinGP", "Buf_32", nDataAry(0), 1)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessageEx(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

End Sub

Private Sub WriteFloat_Click()

    ' 書き込みデータ .
    Dim nDataAry(1) As Single
    nDataAry(0) = CSng(Val(Me.WBuf_Float.Text))

    ' 書き込み .
    Dim iResult As Long
    iResult = WriteDeviceFloat("#WinGP", "Buf_Float", nDataAry(0), 1)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessageEx(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

End Sub
```

```
Private Sub WriteDouble_Click()

    ' 書き込みデータ .
    Dim nDataAry(1) As Double
    nDataAry(0) = CDbl(Val(Me.WBuf_Double.Text))

    ' 書き込み .
    Dim iResult As Long
    iResult = WriteDeviceDouble("#WinGP", "Buf_Double", nDataAry(0), 1)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessageEx(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

End Sub

Private Sub WriteString_Click()

    ' 書き込みデータ .
    Dim nDataAry As String
    nDataAry = Me.WBuf_Str.Text

    ' 書き込み .
    Dim iResult As Long
    iResult = WriteDeviceStr("#WinGP", "Buf_Str", nDataAry, 10)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessageEx(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

End Sub

' ------------------------------------------------------------------------------
' ReadDeviceXXX()
' ------------------------------------------------------------------------------

Private Sub ReadBit_Click()

    ' 読み込みデータ .
    Dim nDataAry(1) As Integer

    ' 読み込み .
    Dim iResult As Long
    iResult = ReadDeviceBit("#WinGP", "Buf_Bit", nDataAry(0), 1)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessage(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If
```

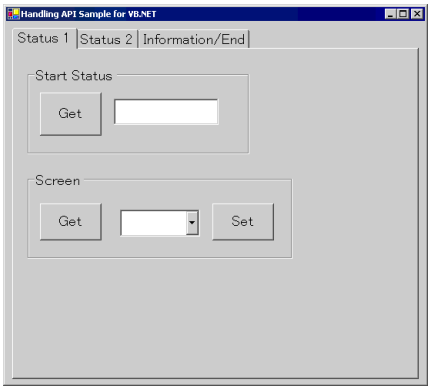
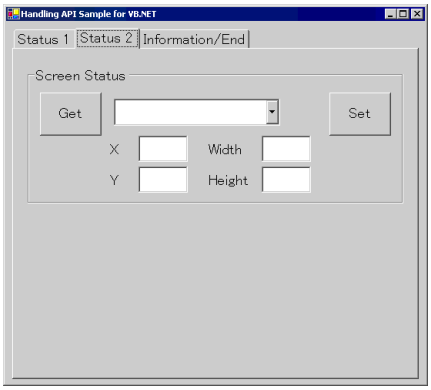
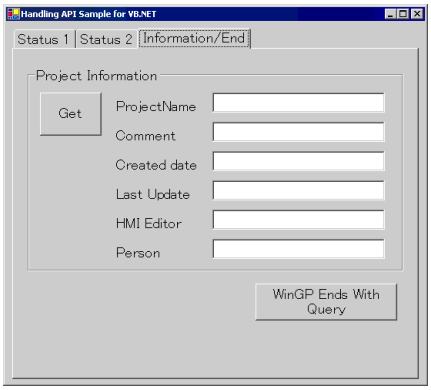
```
    Me.Buf_Bit.Text = CStr(nDataAry(0))

End Sub

Private Sub Read16_Click()

    ' 読み込みデータ .
    Dim nDataAry(1) As Integer

    ' 読み込み .
    Dim iResult As Long
    iResult = ReadDevice16("#WinGP", "Buf_16", nDataAry(0), 1)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessage(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

    Me.Buf_16.Text = CStr(nDataAry(0))

End Sub

Private Sub Read32_Click()

    ' 読み込みデータ .
    Dim nDataAry(1) As Long

    ' 読み込み .
    Dim iResult As Long
    iResult = ReadDevice32("#WinGP", "Buf_32", nDataAry(0), 1)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessage(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

    Me.Buf_32.Text = CStr(nDataAry(0))

End Sub

Private Sub ReadFloat_Click()

    ' 読み込みデータ .
    Dim nDataAry(1) As Single

    ' 読み込み .
    Dim iResult As Long
    iResult = ReadDeviceFloat("#WinGP", "Buf_Float", nDataAry(0), 1)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
```

```
        iMsgResult = EasyLoadErrorMessage(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

    Me.Buf_Float.Text = CStr(nDataAry(0))

End Sub

Private Sub ReadDouble_Click()

    ' 読み込みデータ .
    Dim nDataAry(1) As Double

    ' 読み込み .
    Dim iResult As Long
    iResult = ReadDeviceDouble("#WinGP", "Buf_Double", nDataAry(0), 1)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessage(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

    Me.Buf_Double.Text = CStr(nDataAry(0))

End Sub

Private Sub ReadString_Click()

    ' 読み込みデータ .
    Dim nDataAry As String * 255

    ' 読み込み .
    Dim iResult As Long
    iResult = ReadDeviceStr("#WinGP", "Buf_Str", nDataAry, 10)
    If iResult Then
        Dim sErrMsg As String * 512
        Dim iMsgResult As Long
        iMsgResult = EasyLoadErrorMessage(iResult, sErrMsg)
        MsgBox (sErrMsg)
    End If

    Me.Buf_Str.Text = nDataAry

End Sub
```

37.5.4 WinGP の状態を取得 / 設定変更するサンプル（ハンドリング API）

■ サンプル概要

[Status 1] から [Information/End] タブまでを切り替えて、WinGP の状態を取得したり、設定を変更することができます。

<table>
<tr>
<td>[Status 1] タブ
 
</td>
<td>[Start Status] で [Get] ボタンを押すと、WinGP の起動状態を以下の 6 とおりで表示します。
• 起動中
• オフライン
• オンライン
• 転送モード
• 終了中
• 未動作
[Screen] の [Get] ボタンを押すと、現在、WinGP で表示している画面番号を表示します。また、コンボボックスのリスト内に WinGP で表示可能な画面番号がリスト表示されます。リストから切り替え先の画面番号を選択し、[Set] ボタンを押すと、WinGP で表示されている画面が切り替わります。</td>
</tr>
<tr>
<td>[Status 2] タブ
 
</td>
<td>[Screen State] で [Get] ボタンを押すと、コンボボックス内に WinGP の表示状態を以下の 3 とおりで表示します。
• 最大化（フルスクリーン）
• ウィンドウモード
• 最小化
コンボボックス内の表示を変更し、[Set] ボタンを押すと、表示状態を変更できます。またウィンドウモードの場合のみ X、Y 座標および幅（Width）、高さ（Height）も設定できます。</td>
</tr>
<tr>
<td>[Information/End] タブ
 
</td>
<td>左上の [Start Status] の [Get] ボタンを押すと、WinGP で表示している以下の情報を表示します。
<table>
<tr><td>ProjectName</td><td>プロジェクトファイル名</td></tr>
<tr><td>Comment</td><td>プロジェクトタイトル</td></tr>
<tr><td>Created date</td><td>プロジェクト作成日時</td></tr>
<tr><td>Last Update</td><td>プロジェクト最終更新日時</td></tr>
<tr><td>HMI Editor</td><td>GP-Pro EX バージョン</td></tr>
<tr><td>Person</td><td>作成者</td></tr>
</table>
[WinGP Ends With Query] ボタンを押すと、「終了しますか？」という確認メッセージが表示され「はい」を選択すると WinGP が終了します。</td>
</tr>
</table>

■ VB.NET 2003 プログラム例

サンプルプログラムの場所：（GP-Pro EX の CD-ROM 内）¥WinGP¥SDK¥Pro-SDK¥DotNet¥RtCtrlSmpl

```
Imports System.Runtime.InteropServices

Public Class Form1
    Inherits System.Windows.Forms.Form

    Dim ghWinGP As Int32 = 0 ' API ハンドル .

#Region " Windows フォームデザイナで生成されたコード "

        Public Sub New()
            MyBase.New()

            ' この呼び出しは Windows フォームデザイナで必要です。
            InitializeComponent()

            ' API を初期化します (API).
            Dim nResult As Integer = InitRuntimeAPI()

            ' この段階でハンドルを取得しておきます (API).
            ghWinGP = GetRuntimeHandle(9800)
            If ghWinGP = 0 Then
                MsgBox("(API) ハンドルを取得できませんでした ")
            End If

        End Sub

        ' Form は、コンポーネント一覧に後処理を実行するために dispose をオーバーライドします。
        Protected Overloads Overrides Sub Dispose(ByVal disposing As Boolean)
            If disposing Then
                If Not (components Is Nothing) Then
                    components.Dispose()
                End If
            End If
            CleanupRuntimeAPI()
            MyBase.Dispose(disposing)
        End Sub

        ～中略（以下、Windows フォームデザイナで生成されたコードは省略します）～
#End Region
    ' 5 起動状態取得 .
    Private Sub Bt_GetStartState_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Bt_GetStartState.Click

        Me.Cursor = Cursors.WaitCursor ' カーソルを砂時計に変更 .

        Try

            ' 状態の取得 (API).
            Dim Status As Int32
            Dim RetVal As Int32 = GetRuntimeStartState(ghWinGP, Status)
```

System.Runtime.InteropServicesをインポートします。

InitializeComponent() 呼び出しの後に初期化を追加します。

```
        ' 異常あり ?
        If RetVal <> API_ERROR.E_SUCCESS Then
            MsgBox("Err(" + Str(RetVal).Trim() + "):GetRuntimeStartState()")
        End If

        ' 状態の表示
        Select Case Status
            Case 0
                Me.StartState.Text = " 起動中 "
            Case 1
                Me.StartState.Text = " オンライン "
            Case 2
                Me.StartState.Text = " オフライン "
            Case 3
                Me.StartState.Text = " 転送モード "
            Case 4
                Me.StartState.Text = " 終了中 "
            Case 5
                Me.StartState.Text = " 未動作 "
        End Select

    Catch ex As Exception

        MsgBox(ex.Message)

    End Try

    Me.Cursor = Cursors.Default ' カーソルを元に戻す .

End Sub

Private Sub GetScreenState_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles BT_GetScreenState.Click

    Me.Cursor = Cursors.WaitCursor ' カーソルを砂時計に変更 .

    Try

        ' 状態の取得 .
        Dim Status As Int32
        Dim RetVal As Int32 = GetScreenState(ghWinGP, Status)

        ' 異常あり ?
        If RetVal <> API_ERROR.E_SUCCESS Then
            MsgBox("Err(" + Str(RetVal).Trim() + "):GetScreenState()")
        End If

        ' 状態の表示
        Select Case Status
            Case 0, 1, 2
                Me.ScreenState.SelectedIndex = Status
        End Select
```

```
        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

        Me.Cursor = Cursors.Default ' カーソルを元に戻す .

    End Sub

    Private Sub SetScreenState_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles BT_SetScreenState.Click

        Me.Cursor = Cursors.WaitCursor ' カーソルを砂時計に変更 .

        Try

            ' 設定値の取得 .
            Dim State As Int32 = Me.ScreenState.SelectedIndex
            Dim PosX As Int32 = Val(Me.PosX.Text)
            Dim PosY As Int32 = Val(Me.PosY.Text)
            Dim Width As Int32 = Val(Me.TX_Width.Text)
            Dim Height As Int32 = Val(Me.TX_Height.Text)

            ' 画面状態の設定 .
            Dim RetVal As Int32 = SetScreenState(ghWinGP, State, PosX, PosY, Width, Height)

            ' 異常あり ?
            If RetVal <> API_ERROR.E_SUCCESS Then
                MsgBox("Err(" + Str(RetVal).Trim() + "):SetScreenState()")
            End If

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

        Me.Cursor = Cursors.Default ' カーソルを元に戻す .

    End Sub

    Private Sub GetDispScreen_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles GetDispScreen.Click

        Me.Cursor = Cursors.WaitCursor ' カーソルを砂時計に変更 .

        Dim CurScrNo As Int32 ' 表示中の画面番号 .

        Try

            ' 状態の取得 .
            Dim RetVal As Int32 = GetDisplayScreenNumber(ghWinGP, CurScrNo)
```

```
    ' 異常あり？
    If RetVal <> API_ERROR.E_SUCCESS Then
        MsgBox("Err(" + Str(RetVal).Trim() + "):GetDisplayScreenNumber()")
    End If

Catch ex As Exception

    MsgBox(ex.Message)

End Try

Try

    ' 画面数を取得する .
    Dim ScreenCount As Int32 = 0
    Dim RetVal As Int32 = GetEnumScreenNumberCount(ghWinGP, ScreenCount)

    ' 異常あり？
    If RetVal <> API_ERROR.E_SUCCESS Then
        MsgBox("Err(" + Str(RetVal).Trim() + "):GetEnumScreenNumberCount()")
    End If

    ' 画面番号の取得 .
    If ScreenCount > 0 Then

        ' 画面番号を取得する .
        Dim ScreenNumber(ScreenCount - 1) As Int32
        RetVal = EnumScreenNumber(ghWinGP, ScreenCount, ScreenNumber(0))

        ' 異常あり？
        If RetVal <> API_ERROR.E_SUCCESS Then
            MsgBox("Err(" + Str(RetVal).Trim() + "):EnumScreenNumber()")
        End If

        ' ----- 状態の表示 -----

        ' 一旦全て削除 .
        Me.CB_DispScreen.Items.Clear()

        ' 取得した画面番号を設定 .
        Dim idx As Int32
        For idx = 0 To ScreenNumber.Length - 1
            Me.CB_DispScreen.Items.Add(ScreenNumber(idx))
        Next

        ' 表示中画面番号を表示 .
        For idx = 0 To ScreenNumber.Length - 1
            If CurScrNo = Val(Me.CB_DispScreen.Items(idx)) Then
                Me.CB_DispScreen.SelectedIndex = idx
                Exit For
            End If
        Next

    End If
```

```
        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

        Me.Cursor = Cursors.Default ' カーソルを元に戻す .

    End Sub

    Private Sub SetDispScreen_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles SetDispScreen.Click

        Me.Cursor = Cursors.WaitCursor ' カーソルを砂時計に変更 .

        Try

            ' 画面番号の取得 .
            Dim Screen As Int32
            Screen = Val(Me.CB_DispScreen.Text)

            ' 画面番号の変更 .
            Dim RetVal As Int32 = SetDisplayScreenNumber(ghWinGP, Screen)

            ' 異常あり ?
            If RetVal <> API_ERROR.E_SUCCESS Then
                MsgBox("Err(" + Str(RetVal).Trim() + "):SetDisplayScreenNumber()")
            End If

            ' 正常に変わったかは画面番号を再取得して、設定した値と比較します。
            Dim NowScrNo As Long
            RetVal = GetDisplayScreenNumber(ghWinGP, NowScrNo)
            If RetVal = API_ERROR.E_SUCCESS Then
                If NowScrNo = Screen Then
                    'MsgBox(" 画面は正常に変わりました No=" + Str(NowScrNo))
                End If
            End If

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

        Me.Cursor = Cursors.Default ' カーソルを元に戻す .

    End Sub
```

```
    Private Sub GetProjectInfo_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles GetProjectInfo.Click

        Me.Cursor = Cursors.WaitCursor ' カーソルを砂時計に変更 .

        Try

            ' 取得するパラメタの領域 .
            Dim ProjectFileName As New System.Text.StringBuilder(PROJECTINFO_SIZE.e_FileName)
            Dim ProjectComment As New System.Text.StringBuilder(PROJECTINFO_SIZE.e_Comment)
            Dim ProjectFastTime As New System.Text.StringBuilder(PROJECTINFO_SIZE.e_FastTime)
            Dim ProjectLastTime As New System.Text.StringBuilder(PROJECTINFO_SIZE.e_LastTime)
            Dim ProjectIDownload As New System.Text.StringBuilder(PROJECTINFO_SIZE.e_IDownload)
            Dim HMIEditorVersion As New
System.Text.StringBuilder(PROJECTINFO_SIZE.e_HMIEditorVersion)
            Dim ControlEditorVersion As New
System.Text.StringBuilder(PROJECTINFO_SIZE.e_ControlEditorVersion)
            Dim MakingPerson As New System.Text.StringBuilder(PROJECTINFO_SIZE.e_MakingPerson)

            ' プロジェクト情報取得 .
            Dim RetVal As Int32
            RetVal = GetProjctInformation(ghWinGP, _
                ProjectFileName, _
                ProjectComment, _
                ProjectFastTime, _
                ProjectLastTime, _
                ProjectIDownload, _
                HMIEditorVersion, _
                ControlEditorVersion, _
                MakingPerson)

            ' 異常あり ?
            If RetVal <> API_ERROR.E_SUCCESS Then
                MsgBox("Err(" + Str(RetVal).Trim() + "):GetProjctInformation()")
            End If

            ' 取得した情報の表示
            Me.Prj_File.Text = ProjectFileName.ToString()
            Me.Prj_Comment.Text = ProjectComment.ToString()
            Me.Prj_Date.Text = ProjectFastTime.ToString()
            Me.Prj_LastDate.Text = ProjectLastTime.ToString()
            Me.Prj_HMI.Text = HMIEditorVersion.ToString()
            Me.Prj_Person.Text = MakingPerson.ToString

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

        Me.Cursor = Cursors.Default ' カーソルを元に戻す .

    End Sub
```

```
    ' 13 終了操作 .
    ' 確認ダイアログ付き終了 .
    ' 当然の事ですが、WinGP の終了ダイアログで「終了しません」を選ぶと終了しません。
    ' その時でも戻り値は API_ERROR.E_SUCCESS で返ります。

    Private Sub StopWinGP_Q_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles StopWinGP_Q.Click

        Me.Cursor = Cursors.WaitCursor ' カーソルを砂時計に変更 .

        Try

            ' 終了操作 (API).
            Dim RetVal As Int32 = StopRuntime(ghWinGP, 1)

            ' 異常あり ?
            If RetVal <> API_ERROR.E_SUCCESS Then
                MsgBox("Err(" + Str(RetVal).Trim() + "):StopRuntime()")
            End If

        Catch ex As Exception

            MsgBox(ex.Message)

        End Try

        Me.Cursor = Cursors.Default ' カーソルを元に戻す .

    End Sub

End Class
```

■ VB6 プログラム例

サンプルプログラムの場所：（GP-Pro EX の CD-ROM 内）¥WinGP¥SDK¥Pro-SDK¥VB¥RtCtrlSmpl

MEMO

- このサンプルプログラムの実行ファイルは、日本語、英語以外の OS 環境では正しく動作しません。日本語、英語以外の OS 環境で実行ファイルを動作させるには、それぞれの OS 環境で実行ファイルを作成し直してください。

```
Option Explicit

Private Sub Form_Load()

    ' API を初期化します (API).
    Dim nResult As Long
    nResult = InitRuntimeAPI

    ' この段階でハンドルを取得しておきます (API).
    ghWinGP = GetRuntimeHandle(9800)
    If ghWinGP = 0 Then
        MsgBox ("(API) ハンドルを取得できませんでした ")
    End If

End Sub

Private Sub Bt_GetStartState_Click()

    Screen.MousePointer = vbHourglass

    ' 状態の取得 (API).
    Dim Status As Long
    Dim RetVal As Long
    RetVal = GetRuntimeStartState(ghWinGP, Status)

    ' 異常あり ?
    If RetVal <> CLng(API_ERROR.E_SUCCESS) Then
        MsgBox ("Err(" + Str(RetVal) + "):GetRuntimeStartState()")
    End If

    ' 状態の表示
    Select Case Status
        Case 0
            Me.StartState.Text = " 起動中 "
        Case 1
            Me.StartState.Text = " オンライン "
        Case 2
            Me.StartState.Text = " オフライン "
        Case 3
            Me.StartState.Text = " 転送モード "
        Case 4
            Me.StartState.Text = " 終了中 "
        Case 5
            Me.StartState.Text = " 未動作 "
    End Select
```

```
    Screen.MousePointer = vbDefault

End Sub

Private Sub BT_GetScreenState_Click()

    Screen.MousePointer = vbHourglass

    ' 状態の取得 .
    Dim Status As Long
    Dim RetVal As Long
    RetVal = GetScreenState(ghWinGP, Status)

    ' 異常あり ?
    If RetVal <> API_ERROR.E_SUCCESS Then
        MsgBox ("Err(" + Str(RetVal) + "):GetScreenState()")
    End If

    ' 状態の表示
    Select Case Status
        Case 0, 1, 2
            Me.ScreenState.ListIndex = Status
    End Select

    Screen.MousePointer = vbDefault

End Sub

Private Sub BT_SetScreenState_Click()

    Screen.MousePointer = vbHourglass ' カーソルを砂時計に変更 .

    ' 設定値の取得 .
    Dim State As Long
    Dim PosX As Long
    Dim PosY As Long
    Dim Width As Long
    Dim Height As Long

    State = Me.ScreenState.ListIndex
    PosX = Val(Me.PosX.Text)
    PosY = Val(Me.PosY.Text)
    Width = Val(Me.TX_Width.Text)
    Height = Val(Me.TX_Height.Text)

    ' 画面状態の設定 .
    Dim RetVal As Long
    RetVal = SetScreenState(ghWinGP, State, PosX, PosY, Width, Height)

    ' 異常あり ?
    If RetVal <> API_ERROR.E_SUCCESS Then
        MsgBox ("Err(" + Str(RetVal) + "):SetScreenState()")
    End If
```

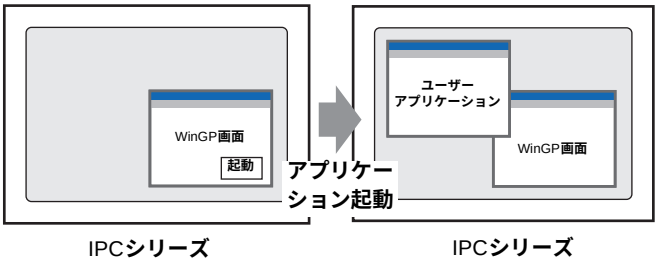
```
    Screen.MousePointer = vbDefault

End Sub

Private Sub GetDispScreen_Click()

    Screen.MousePointer = vbHourglass ' カーソルを砂時計に変更 .

    Dim CurScrNo As Long ' 表示中の画面番号 .

    ' 状態の取得 .
    Dim RetVal As Long
    RetVal = GetDisplayScreenNumber(ghWinGP, CurScrNo)

    ' 異常あり ?
    If RetVal <> API_ERROR.E_SUCCESS Then
        MsgBox ("Err(" + Str(RetVal) + "):GetDisplayScreenNumber()")
    End If

    ' 画面数を取得する .
    Dim ScreenCount As Long
    RetVal = GetEnumScreenNumberCount(ghWinGP, ScreenCount)

    ' 異常あり ?
    If RetVal <> API_ERROR.E_SUCCESS Then
        MsgBox ("Err(" + Str(RetVal) + "):GetEnumScreenNumberCount()")
    End If

    ' 画面番号の取得 .
    If ScreenCount > 0 Then

        ' 画面番号を取得する .
        Dim ScreenNumber() As Long
        ReDim ScreenNumber(ScreenCount - 1) As Long
        RetVal = EnumScreenNumber(ghWinGP, ScreenCount, ScreenNumber(0))

        ' 異常あり ?
        If RetVal <> API_ERROR.E_SUCCESS Then
            MsgBox ("Err(" + Str(RetVal) + "):EnumScreenNumber()")
        End If

        ' ----- 状態の表示 -----

        ' 取得した画面番号を設定 .
        Me.CB_DispScreen.Clear
        Dim idx As Long
        For idx = 0 To ScreenCount - 1
            Me.CB_DispScreen.AddItem (ScreenNumber(idx))
        Next

        ' 表示中画面番号を表示 .
        For idx = 0 To ScreenCount - 1
            If CurScrNo = Val(Me.CB_DispScreen.List(idx)) Then
                Me.CB_DispScreen.ListIndex = idx
```

```
                Exit For
            End If
        Next

    End If

    Screen.MousePointer = vbDefault ' カーソルを元に戻す .

End Sub

Private Sub SetDispScreen_Click()

    Screen.MousePointer = vbHourglass ' カーソルを砂時計に変更 .

    ' 画面番号の取得 .
    Dim ScrNo As Long
    ScrNo = Val(Me.CB_DispScreen.Text)

    ' 画面番号の変更 .
    Dim RetVal As Long
    RetVal = SetDisplayScreenNumber(ghWinGP, ScrNo)

    ' 異常あり ?
    If RetVal <> API_ERROR.E_SUCCESS Then
        MsgBox ("Err(" + Str(RetVal) + "):SetDisplayScreenNumber()")
    End If

    ' 正常に変わったかは画面番号を再取得して、設定した値と比較します。
    Dim NowScrNo As Long
    RetVal = GetDisplayScreenNumber(ghWinGP, NowScrNo)
    If RetVal = API_ERROR.E_SUCCESS Then
        If NowScrNo = ScrNo Then
        ' MsgBox (" 画面は正常に変わりました No=" + Str(NowScrNo))
        End If
    End If

    Screen.MousePointer = vbDefault ' カーソルを元に戻す .

End Sub

Private Sub GetProjectInfo_Click()

    Screen.MousePointer = vbHourglass ' カーソルを砂時計に変更 .

    ' 取得するパラメタの領域 .
    Dim ProjectFileName As String * 256
    Dim ProjectComment As String * 256
    Dim ProjectFastTime As String * 256
    Dim ProjectLastTime As String * 256
    Dim ProjectIDownload As String * 256
    Dim HMIEditorVersion As String * 256
    Dim ControlEditorVersion As String * 256
    Dim MakingPerson As String * 256
```

```
    ' プロジェクト情報取得 .
    Dim RetVal As Long
    RetVal = GetProjctInformation(ghWinGP, _
        ProjectFileName, _
        ProjectComment, _
        ProjectFastTime, _
        ProjectLastTime, _
        ProjectIDownload, _
        HMIEditorVersion, _
        ControlEditorVersion, _
        MakingPerson)

    ' 異常あり ?
    If RetVal <> API_ERROR.E_SUCCESS Then
        MsgBox ("Err(" + Str(RetVal) + "):GetProjctInformation()")
    End If

    ' 取得した情報の表示
    Me.Prj_File.Text = StrConv(ProjectFileName, vbFromUnicode)
    Me.Prj_Comment.Text = StrConv(ProjectComment, vbFromUnicode)
    Me.Prj_Date.Text = StrConv(ProjectFastTime, vbFromUnicode)
    Me.Prj_LastDate.Text = StrConv(ProjectLastTime, vbFromUnicode)
    Me.Prj_HMI.Text = StrConv(HMIEditorVersion, vbFromUnicode)
    Me.Prj_Person.Text = StrConv(MakingPerson, vbFromUnicode)

    Screen.MousePointer = vbDefault ' カーソルを元に戻す .

End Sub

' 13 終了操作 .
' 確認ダイアログ付き終了 .
' 当然の事ですが、WinGP の終了ダイアログで「終了しません」を選ぶと終了しません。
' その時でも戻り値は API_ERROR.E_SUCCESS で返ります。

Private Sub StopWinGP_Q_Click()
    Screen.MousePointer = vbHourglass ' カーソルを砂時計に変更 .

    ' 終了操作 (API).
    Dim RetVal As Long
    RetVal = StopRuntime(ghWinGP, 1)

    ' 異常あり ?
    If RetVal <> API_ERROR.E_SUCCESS Then
        MsgBox ("Err(" + Str(RetVal) + "):StopRuntime()")
    End If

    Screen.MousePointer = vbDefault ' カーソルを元に戻す .

End Sub
```

37.6 WinGP からアプリケーションを実行したい

37.6.1 詳細

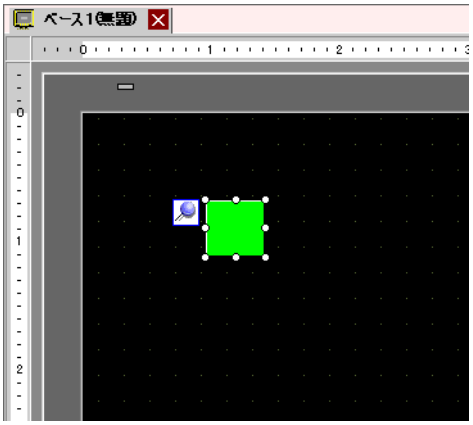


WinGP の画面上から他のアプリケーションを実行することができます。アプリケーションの実行方法は以下の 4 とおりがあります。

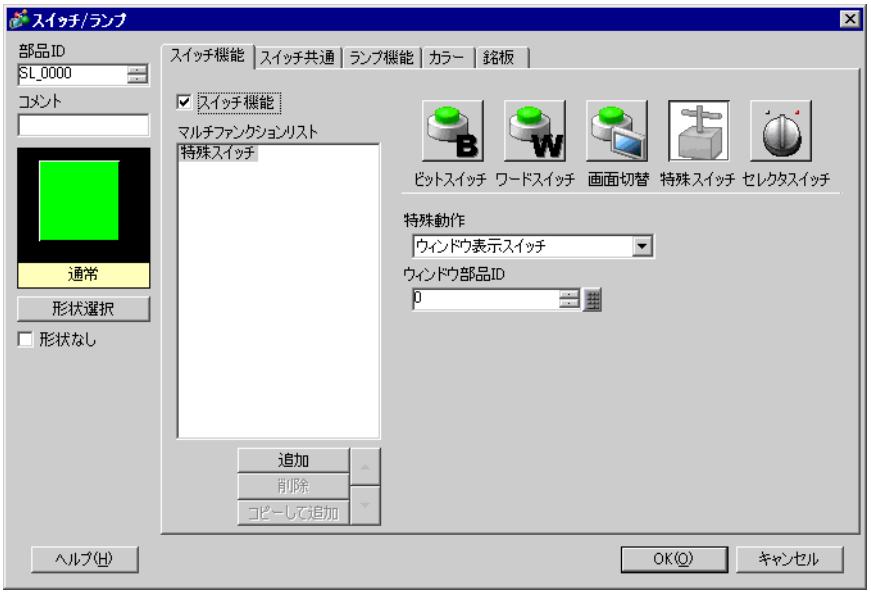
スイッチで起動する	☞「37.6.2 スイッチ起動の設定手順」(37-66 ページ)
D スクリプトで起動する	☞「37.6.3 D スクリプト起動の設定手順」(37-68 ページ)
WinGP のオフライン画面から起動する	☞ [保守 / トラブル解決ガイド]
トリガアクションで起動する	

37.6.2 スイッチ起動の設定手順

1 [部品] メニューの [スイッチランプ] から [特殊スイッチ] を選択するか、ツールバーから をクリックし、画面に配置します。



2 配置したスイッチをダブルクリックすると設定ダイアログボックスが開きます。



3 [形状選択] でスイッチの形状を選択します。

MEMO
- スイッチの形状によっては色が変更できないものがあります。

4 [特殊動作] で [アプリケーション起動] を選択します。

5 [EXE パス] を入力します。

例）C:¥Documents and Settings¥user¥Local Settings¥Temp の sample.exe を実行する場合

指定方法	入力例
フルパス指定	（例）C:¥Documents and Settings¥user¥Local Settings¥Temp¥sample.exe
EXE 名のみ	IPC シリーズの [コントロールパネル]→[システム]→[詳細]→[環境変数] で System 以下の実行ファイルのみ指定できます。 （例）sample.exe （環境変数で Path = C:¥Documents and Settings¥user¥Local Settings¥Temp と指定）
環境変数でパス指定	IPC シリーズの [コントロールパネル]→[システム]→[詳細]→[環境変数] で環境変数の [TEMP] に設定されているフォルダに実行ファイルがある場合のみ環境変数でパス指定できます。
	（例）%TEMP%¥sample.exe （環境変数　TEMP = C:¥Documents and Settings¥user¥Local Settings¥Temp と指定）

6 [パラメータ] で Exe 実行時のオプション（引数）を設定します。[パラメータ] は最大 255 文字以内で設定します。

例）Microsoft Excel のファイルを起動する場合

EXE パス	EXCEL.EXE のパスを指定します。 例）C:¥Program Files¥Microsoft Office¥Office¥EXCEL.EXE
パラメータ	エクセルブック（*.xls）のパスを " " 内に指定します。 例）"C:¥Documents and Settings¥user¥ デスクトップ ¥ 生産管理 .xls"

7 多重起動を禁止する場合は [多重起動の禁止] チェックボックスにチェックを入れ、[ウィンドウタイトル] を入力します。

☞「11.14.4 特殊スイッチ ◆ アプリケーション起動」（11-69 ページ）

37.6.3 D スクリプト起動の設定手順

MEMO

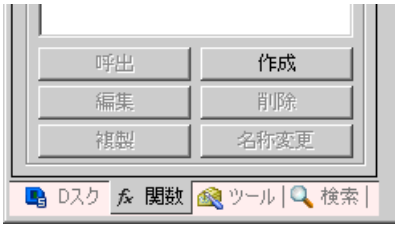
- 設定内容の詳細は設定ガイドを参照してください。

☞「21.7.2 アプリケーション起動」（21-63 ページ）

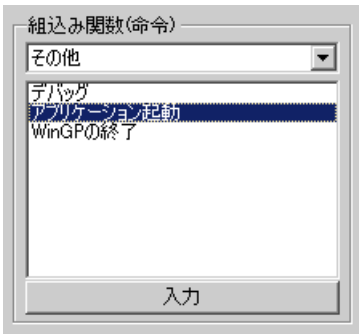
- [共通設定] メニューの [グローバル D スクリプト] および [拡張スクリプト設定] でも同様に EXE の起動ができます。

1 [部品] メニューから [D スクリプト] を選択し、[D スクリプト一覧] ダイアログボックスで [作成] をクリックします。

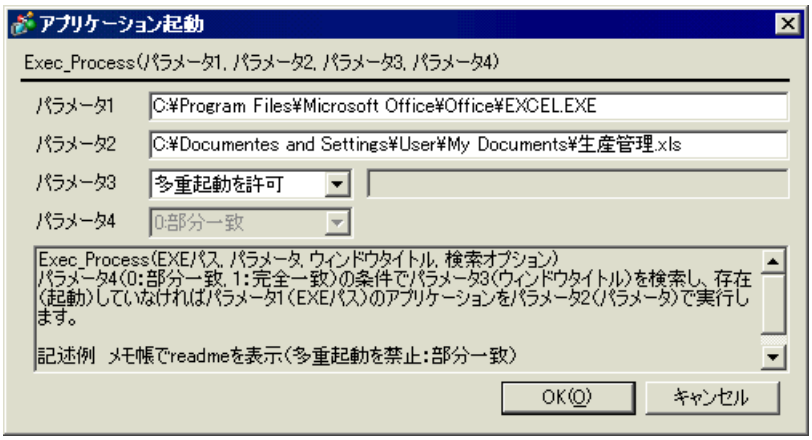
2 [関数] タブをクリックします。[組み込み関数]（命令）はスクリプトで使用できる命令をクリックするだけで簡単に配置することができます。



3 [組み込み関数（命令）] のプルダウンメニューから [その他] をクリックし、[アプリケーション起動] をダブルクリックします。



4 以下のダイアログボックスでパラメータの設定を行います。



パラメータ 1	EXE ファイルのパスを指定します。 ☞「37.6.2 スイッチ起動の設定手順」(37-66 ページ)
パラメータ 2	Exe 実行時のオプション（引数）を設定します。[パラメータ] は最大 255 文字以内で設定します ☞「37.6.2 スイッチ起動の設定手順」(37-66 ページ)
パラメータ 3	[多重起動許可] または [多重起動禁止] を選択します。[多重起動禁止] の場合はウィンドウタイトルを入力します。 ☞「21.7.2 アプリケーション起動」(21-63 ページ)
パラメータ 4	[0: 部分一致] または [1: 完全一致] を選択します。 ☞「21.7.2 アプリケーション起動」(21-63 ページ)

5 [OK] をクリックすると、[実行式] に手順 4 で設定したパラメータが入力されます。

例）

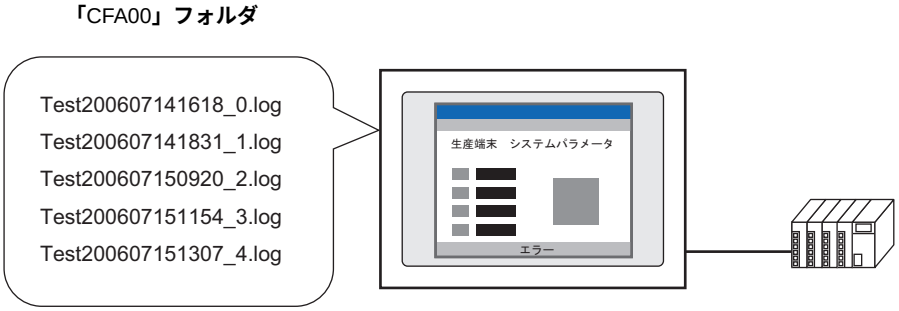
Exec_Process("C:¥Program Files¥Microsoft Office¥Office¥EXCEL.EXE",

"C:¥Documents and Settings¥User¥My Documents¥ 生産管理 .xls","",0)

37.7 WinGP に表示されたエラーメッセージの履歴を残したい

37.7.1 詳細

WinGP 画面上に表示されるシステムエラーやアプリケーションエラーの履歴をログファイルとして残すことができます。エラーが発生するたびに、エラーが発生した日付・時刻・種別（ERROR/WARNING）、エラーメッセージがファイルに書き込まれます。
1 ファイルに最大 1000 件までエラーメッセージが保存できます。



＜エラーログファイルのフォーマット＞
例：「Test200607141618_0.log」のファイルをテキストで開いた場合

日付　　時刻　　種別　エラー内容

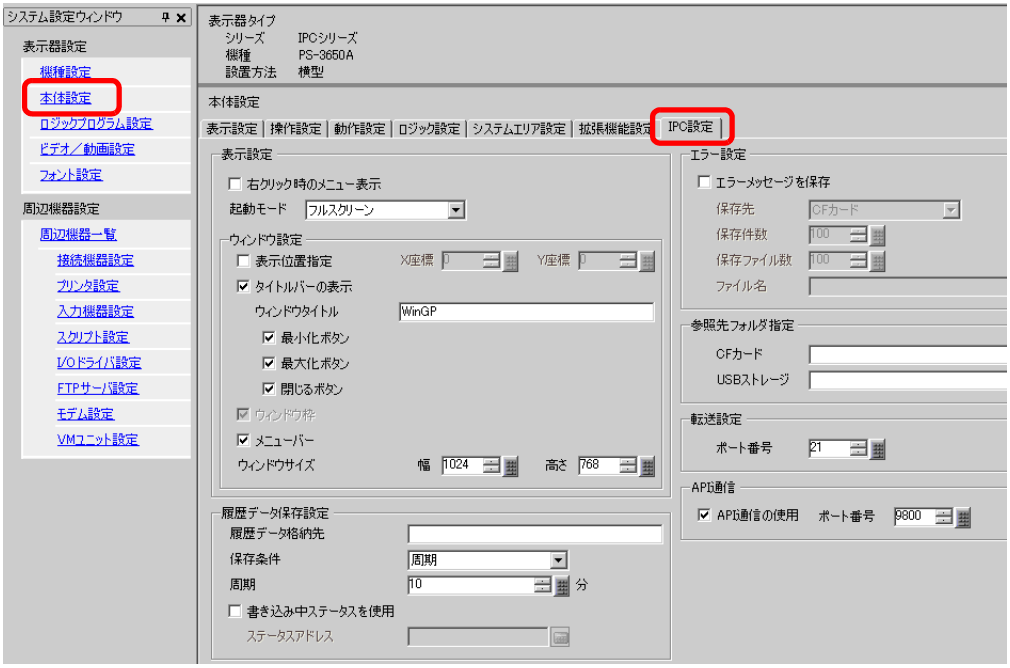
```
2006/07/14,16:18:59.563,ERROR,osKRboot1[c:¥runtime_ デスクトップ ¥win¥power¥src¥pw_main.cpp:831]
2006/07/14,17:26:30.062,WARNING,RHAA070:PLC1: ケーブルが接続されていません ( または接続機器の電源が切れています )

…
```

MEMO

- エラーは 1 件発生するたびにファイルに書き込まれますが、前回保存時から 10 分以内であれば、10 分経過するまで保存は行わず、10 分経過した時点でそれまでに記憶された内容が一度にファイルに書き込まれます。また 10 分以内でも WinGP 終了時には、まだ書き込まれていないエラーがあれば一度に書き込まれます。

37.7.2 設定手順

1 GP-Pro EX のシステム設定ウィンドウで [本体設定] を選択し、[IPC 設定] タブを開きます。



2 [エラーメッセージを保存] にチェックを入れ、[保存先] を指定します。（例：CF カード）

3 [保存件数] に、1 ファイルに保存するエラー件数（例：100）を設定します。
また [保存ファイル数] にはフォルダ内に保存するファイル数（例：5）を設定します。
指定した件数分のエラーがファイルに書き込まれると、自動的に次のファイルが作成されます。指定したファイル数分のログファイルがフォルダ内に作成されると、1 番古いファイルが削除され、新しいログファイルが作成されます。

4 [ファイル名] にファイル名の先頭文字を半角英数字 0 ～ 16 文字で入力します。(例：Test)
ファイル名は以下の形式で付けられます。
（任意のファイル名）[日付時間]_[ID].[拡張子]
日付時間：yyyymmddhhmm
ID：0 から指定した [保存ファイル数] 分の ID が自動的に割り振られます。
拡張子：log
例：2006 年 7 月 14 日　16 時 18 分の場合　Test200607141618_0.log

37.8 API 関数一覧

WinGP で使用できる API には、ハンドリング API とデバイスアクセス API の 2 種類あります。

37.8.1 ハンドリング API

◆概要

独自に作成したプログラム (アプリケーション) から WinGP の状態取得や、設定を変更する機能を使用するための API です。アプリケーションと API の DLL ファイルをリンクさせることにより、WinGP がインストールされている同一の IPC 上でハンドリング API で作成したアプリケーションが動作します。

◆ハンドリング API の DLL ファイル

この API は DLL ファイルで提供されます。DLL ファイル名は RtCtrlAPI.dll で、WINDOWS のフォルダにインストールされます。

◆サポート言語

ハンドリング API で動作するプログラム言語は下記の 5 つになります。

- Visual C++
- Visual Basic 6.0
- VB.NET
- Excel VBA
- C#

◆関数一覧

- WinGP ハンドル取得

通信先の WinGP のハンドルを作成しアプリケーションに返します。
以降の関数では、本関数で取得したハンドルを指定します。

関数名	INT32 GetRuntimeHandle (UINT32 ul_PortNo);
引　数	ul_PortNo ：(i) 通信先 WinGP のある IPC のポート番号
戻り値	WinGP ハンドル

- WinGP ハンドル開放

WinGP ハンドル取得関数で取得したハンドルを開放します。

関数名	bool ReleaseRuntimeHandle (INT32 l_RuntimeHandle);
引　数	l_RuntimeHandle ：(i)WinGP ハンドル
戻り値	true : 成功 / false : 失敗

- **API 初期化**

WinGP 操作・状態取得 API を初期化します。

関数名	bool InitRuntimeAPI (void);
引　数	なし
戻り値	true : 成功 / false : 失敗

- **API 終了**

WinGP 操作・状態取得 API を使い終えるときに後処理をします。

関数名	bool CleanupRuntimeAPI (void);
引　数	なし
戻り値	true : 成功 / false : 失敗

- **起動状態取得**

WinGP の起動状態を取得します。

関数名	INT32 GetRuntimeStartState (INT32 l_RuntimeHandle , INT32 *pl_RuntimeCondition);
引　数	l_RuntimeHandle ：(i) 取得先 WinGP のハンドル *pl_RuntimeCondition ：(o) WinGP 状態 0：STARTING (起動中) 1：START_ONLINE (オンライン) 2：START_OFFLINE (オフライン) 3：START_TRANSFER (転送モード) 4：ENDING (終了中) 5：NOTEXECUTE (動作していない)
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **表示中画面番号取得**

WinGP で表示中の画面番号を WinGP より取得します。

関数名	INT32 GetDisplayScreenNumber (INT32 l_RuntimeHandle , INT32 *pl_DisplayScreenNumber);
引　数	l_RuntimeHandle ：(i) 取得先 WinGP のハンドル pl_DispScreenNumber ：(o) 画面番号 オフラインの場合は画面なし (0) が返る
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **スクリーン状態取得**

WinGP の表示状態を取得します。

関数名	INT32　GetScreenState (INT32　l_RuntimeHandle , INT32　*pl_ScreenState);
引　数	l_RuntimeHandle　：(i) 取得先 WinGP のハンドル pl_ScreenState　：(o) スクリーン状態 0：FULLSCREEN (フルスクリーン) 1：WINDOWSCREEN (ウィンドウスクリーン) 2：MINIMUMSCREEN (最小状態) -1：UNCERTAINTY (不明)
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **言語設定取得**

言語設定番号を返します。

関数名	INT32　GetLanguage (INT32　l_RuntimeHandle, INT32　l_LanguageKind , INT32　*pl_LanguageNumber);
引　数	l_RuntimeHandle　：(i) 取得先 WinGP のハンドル l_LanguageKind　：(i) 言語設定種別 0：SYSTEMLANGUAGE (システム言語設定) 1：USERLANGUAGE (ユーザ言語設定) pl_LanguageNumber　：(o) 言語設定番号 0：SYSTEMLANGUAGE (システム言語設定) 0：日本語 1：英語 1：USERLANGUAGE (ユーザ言語設定)
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **タッチブザー設定取得**

WinGP で設定されているブザー音設定内容を返します。

関数名	INT32　GetTouchBuzzer (INT32　l_RuntimeHandle , INT32　*pl_BuzzerState);
引　数	l_RuntimeHandle　：(i) 取得先 WinGP のハンドル pl_BuzzerState　：(o) ブザー状態 0：BUZZERON (ブザー音なし) 1：BUZZEROFF (ブザー音あり) -1：UNCERTAINTY (不明)
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **プロジェクト情報取得**

WinGP 内のプロジェクト情報を取得します。

関数名	INT32 GetProjctInformation(INT32 l_RuntimeHandle, UINT16 *pus_ProjectFileName , UINT16 *pus_ProjectComment , UINT16 *pus_ProjectFastTime , UINT16 *pus_ProjectLastTime , UINT16 *ps_ProjectIDownload , UINT16 *pus_HMIEditorVersion , UINT16 *pus_ControlEditorVersion , UINT16 *pus_MakingPerson)
引　数	l_RuntimeHandle ：(i) 取得先 WinGP のハンドル ps_ProjectFileName ：(o) プロジェクトファイル名 ps_ProjectComment ：(o) プロジェクトタイトル (コメント) pus_ProjectFastTime ：(o) プロジェクト作成日時 pus_ProjectLastTime ：(o) プロジェクト最終更新日時 ps_ProjectIDownload ：(o) ダウンロード日時 pus_HMIEditorVersion ：(o) HMI エディタバージョン pus_ControlEditorVersion ：(o) CONTROL エディタバージョン pus_MakingPerson ：(o) 作成者名
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **バージョン情報取得**

WinGP バージョンを返します。

関数名	INT32 GetRuntimeVersion(INT32 l_RuntimeHandle, UINT16 *pus_VersionInfo);
引　数	l_RuntimeHandle ：(i) 取得先 WinGP のハンドル pus_VersionInfo ：(o) バージョン情報
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **終了操作**

WinGP に対し終了を要求します。

関数名	INT32 StopRuntime(INT32 l_RuntimeHandle, INT32 l_StopMode);
引　数	l_RuntimeHandle ：(i) 操作する WinGP のハンドル l_StopMode ：(i) 終了モード (未使用) 0：通常終了 1：終了確認ダイアログあり
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **表示画面番号変更**

WinGP の画面番号の変更を要求します。

関数名	INT32　SetDisplayScreenNumber(INT32　l_RuntimeHandle, INT32　l_ScreenNumber);
引　数	l_RuntimeHandle　：(i) 操作する WinGP のハンドル l_ScreenNumber　：(i) 画面番号
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **スクリーン状態変更**

WinGP のスクリーン表示状態を変更します。

関数名	INT32　SetScreenState(INT32　l_RuntimeHandle INT32　l_ScreenState, INT32　l_PosX, INT32　l_PosY, INT32　l_Width, INT32　l_Height);
引　数	l_RuntimeHandle　：(i) 操作する WinGP のハンドル l_ ScreenState　：(i) スクリーン状態 0：FULLSCREEN (フルスクリーン) 1：WINDOWSCREEN (ウィンドウスクリーン) 2：MINIMUMSCREEN (最小状態) l_PosX　：(i) スクリーン座標系での X 座標 (*1) l_PosY　：(i) スクリーン座標系での Y 座標 (*1) l_Width　：(i) ウィンドウスクリーンの幅 (*1) l_Height　：(i) ウィンドウスクリーンの高さ (*1) (*1) 座標、サイズ追加ただしウィンドウスクリーンの場合のみ有効 この引数は第 2 引数の [スクリーン状態] を [WINDOWSCREEN] に設定した場合のみ指定できます。
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **言語設定変更**

WinGP のシステム言語設定 / ユーザ言語設定の言語設定を変更します。
WinGP 再起動後に設定が反映されます。

関数名	INT32 SetLanguage(INT32 l_RuntimeHandle, INT32 l_LanguageKind, INT32 l_LanguageNumber);
引 数	l_RuntimeHandle ：(i) 操作する WinGP のハンドル l_LanguageKind ：(i) 言語設定種別 0：SYSTEMLANGUAGE (システム言語設定) 1：USERLANGUAGE (ユーザ言語設定) l_LanguageNumber ：(i) 言語設定番号
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **タッチブザー設定変更**

WinGP のタッチブザー設定を変更します。

関数名	INT32 SetTouchBuzzer(INT32 l_RuntimeHandle, INT32 l_BuzzerState);
引 数	l_RuntimeHandle ：(i) 操作する WinGP のハンドル l_BuzzerState ：(i) ブザー設定 0：BUZZERON (ブザー音なし) 1：BUZZEROFF (ブザー音あり)
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **表示画面番号列挙数取得**

WinGP で設定可能な画面番号件数を取得します。

関数名	INT32 GetEnumScreenNumberCount(INT32 l_RuntimeHandle, INT32 *l_ScreenNumberCount);
引 数	l_RuntimeHandle ：(i) 操作する WinGP のハンドル l_ScreenNumberCount ：(o) 表示画面件数
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **表示画面番号列挙**

WinGP で設定可能な画面番号を取得し配列に返します。

取得する件数・表示画面は表示画面番号列挙数取得関数で取得した表示画面件数以下にしてください。

関数名	INT32 EnumScreenNumber(INT32 l_RuntimeHandle, INT32 l_ScreenNumberCount, INT32 *pl_ScreenNumbers);
引　数	l_RuntimeHandle ：(i) 操作する WinGP のハンドル l_ScreenNumberCount ：(i) 表示画面件数 l_ScreenNumbers ：(o) 表示画面 (配列で返す)
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **言語番号列挙数取得**

WinGP で設定可能な言語件数を取得します。

関数名	INT32 GetEnumLanguageCount(INT32 l_RuntimeHandle, INT32 l_LanguageKind, INT32 *pl_LanguageCount);
引　数	l_RuntimeHandle ：(i) 操作する WinGP のハンドル l_LanguageKind ：(i) 言語設定種別 0：SYSTEMLANGUAGE (システム言語設定) 1：USERLANGUAGE (ユーザ言語設定) pl_LanguageCount ：(o) 指定可能言語件数
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

- **言語番号列挙**

WinGP 内で設定可能な言語番号を取得します。

関数名	INT32 EnumLanguage(INT32 l_RuntimeHandle, INT32 l_LanguageKind, INT32 l_LanguageCount, INT32 *pl_Languages);
引　数	l_RuntimeHandle ：(i) 操作する WinGP のハンドル l_LanguageKind ：(i) 言語設定種別 0：SYSTEMLANGUAGE (システム言語設定) 1：USERLANGUAGE (ユーザ言語設定) l_LanguageCount ：(i) 指定可能言語件数 pl_LanguageCount ：(o) 指定可能言語
戻り値	ステータス 0： 実行完了 -1： パラメータエラー -2： タイムアウト 1： WinGP が受付けできない状態（終了中など）

37.8.2 デバイスアクセス API

◆概要

独自に作成したプログラム (アプリケーション) から WinGP が通信している接続機器のデバイスや、WinGP 内のデバイスへ読み書きを行う API です。

◆API 通信用の DLL ファイル

この API は DLL ファイルで提供されます。DLL ファイル名は ProEasy.dll で、WINDOWS フォルダにインストールされます。

◆サポート言語

デバイスアクセス API で動作するプログラム言語は下記の 5 つになります。

- Visual C++
- Visual Basic 6.0
- VB.NET
- Excel VBA
- C#

MEMO

- VB.NET または C# では次の API を使用することはできません。
  使用しても動作保証はいたしません。
  - ReadDevice()
  - WriteDevice()
  - ReadSymbol()
  - WriteSymbol()
  - SizeOfSymbol()

◆WinGP SDK でアクセスできるデバイス

WinGP SDK でアクセスできるデバイスは PLC デバイスと USR、LS エリアと GP-Pro EX で登録されたシンボル、ロジック命令の変数のみです。

MEMO

- ロジック命令の構造体変数を使用するには以下のパラメータを使用する必要があります。
  ReadSymbolD/ReadSymbolVariantD/WriteSymbolD/WriteSymbolVariantD を I/F
  ロジック命令の構造体変数を使用する場合の詳細は下記を参照してください。
  ☞「3) ロジック命令の構造体変数によるアクセス時のビットオフセットシンボルの扱い」(37-135 ページ)
- ロジック命令のリアル変数、または R_ デバイスは使用できません。

◆関数一覧

- シングルハンドル系ダイレクトリード API

関数名	ビットデータ
INT WINAPI ReadDeviceBit(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* owData,WORD wCount);	
関数名	16ビットデータ
INT WINAPI ReadDevice16(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* owData,WORD wCount);	
関数名	32ビットデータ
INT WINAPI ReadDevice32(LPCSTR sNodeName,LPCSTR sDeviceName,DWORD* odwData,WORD wCount);	
関数名	16ビットBCDデータ
INT WINAPI ReadDeviceBCD16(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* owData,WORD wCount);	
関数名	32ビットBCDデータ
INT WINAPI ReadDeviceBCD32(LPCSTR sNodeName,LPCSTR sDeviceName,DWORD* odwData,WORD wCount);	
関数名	単精度浮動小数点データ
INT WINAPI ReadDeviceFloat(LPCSTR sNodeName,LPCSTR sDeviceName,FLOAT* oflData,WORD wCount);	
関数名	倍精度浮動小数点データ
INT WINAPI ReadDeviceDouble(LPCSTR sNodeName,LPCSTR sDeviceName,DOUBLE* odbData,WORD wCount);	
関数名	文字列データ
INT WINAPI ReadDeviceStr(LPCSTR sNodeName,LPCSTR sDeviceName,LPSTR psData,WORD wCount);	
関数名	汎用データ
INT WINAPI ReadDevice(LPCSTR sNodeName,LPCSTR sDeviceName,LPVOID pData,WORD wCount,WORD wAppKind);	
関数名	汎用データ (Variant型)
INT WINAPI ReadDeviceVariant(LPCSTR sNodeName,LPCSTR sDeviceName,LPVARIANT pData,WORD wCount,WORD wAppKind);	

- シングルハンドル系ダイレクトライト API

関数名	ビットデータ
INT WINAPI WriteDeviceBit(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* pwData,WORD wCount);	
関数名	16ビットデータ
INT WINAPI WriteDevice16(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* pwData,WORD wCount);	
関数名	32ビットデータ
INT WINAPI WriteDevice32(LPCSTR sNodeName,LPCSTR sDeviceName,DWORD* pdwData,WORD wCount);	
関数名	16ビットBCDデータ
INT WINAPI WriteDeviceBCD16(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* pwData,WORD wCount);	
関数名	32ビットBCDデータ
INT WINAPI WriteDeviceBCD32(LPCSTR sNodeName,LPCSTR sDeviceName,DWORD* pdwData,WORD wCount);	
関数名	単精度浮動小数点データ
INT WINAPI WriteDeviceFloat(LPCSTR sNodeName,LPCSTR sDeviceName,FLOAT* pflData,WORD wCount);	
関数名	倍精度浮動小数点データ
INT WINAPI WriteDeviceDouble(LPCSTR sNodeName,LPCSTR sDeviceName,DOUBLE* pdbData,WORD wCount);	

次のページに続きます。

関数名	文字列データ
INT WINAPI WriteDeviceStr(LPCSTR sNodeName,LPCSTR sDeviceName,LPCSTR psData,WORD wCount);	
関数名	汎用データ
INT WINAPI WriteDevice(LPCSTR sNodeName,LPCSTR sDeviceName,LPVOID pData,WORD wCount,WORD wAppKind);	
関数名	汎用データ (Variant型)
INT WINAPI WriteDeviceVariant(LPCSTR sNodeName,LPCSTR sDeviceName,LPVARIANT pData,WORD wCount,WORD wAppKind);	

- **シングルハンドル系グループシンボルリード** API

関数名	グループシンボル
INT WINAPI ReadSymbol(LPCSTR sNodeName,LPCSTR sSymbolName,LPVOID oReadBufferData);	
関数名	グループシンボル(Variant型)
INT WINAPI ReadSymbolVariant(LPCSTR sNodeName,LPCSTR sSymbolName,LPVARIANT pData);	

- **シングルハンドル系グループシンボルライト** API

関数名	グループシンボル
INT WINAPI WriteSymbolD(LPCSTR sNodeName,LPCSTR sSymbolName,LPVOID pWriteBufferData);	
関数名	グループシンボル(Variant型)
INT WINAPI WriteSymbolVariantD(LPCSTR sNodeName,LPCSTR sSymbolName,LPVARIANT pData);	

- **リード／ライト関数のパラメータ**

＜引数＞

sNodeName: 局名は「#WinGP」固定になります。

sDeviceName: GP-Pro EX で登録されたシンボル名、またはデバイスアドレスを直接記述します。

例 1)　シンボルで指定する場合　　"SWITCH1"

例 2)　デバイスアドレスを直接指定する場合　　"M100"

シンボルで指定する場合、各関数で指定できるデータタイプは次のとおりです。

Function	シンボルのデータタイプ							
	Bit	16Bit		32Bit		Float	Double	String
		符号付き / 符号無し / 16 進	BCD	符号付き / 符号無し / 16 進	BCD			
XXXDeviceBit	○	—	—	—	—	—	—	—
XXXDevice16	—	○	—	—	—	—	—	—
XXXDevice32	—	—	—	○	—	—	—	—
XXXDeviceBCD16	—	—	○	—	—	—	—	—
XXXDeviceBCD32	—	—	—	—	○	—	—	—
XXXDeviceFloat	—	—	—	—	—	○	—	—
XXXDeviceDouble	—	—	—	—	—	—	○	—
XXXDeviceStr	—	—	—	—	—	—	—	○
XXXDevice	○	○	○	○	○	○	○	○

pxxData : Read/Write データへのポインタ

値の読み込み先または書き込み先となるデータへのポインタを指定します。各関数に応じたデータ型のポインタを指定してください。

アクセスするデータの種類	引数のタイプ
ビットデータ	WORD * pwData
16 ビットデータ	WORD * pwData
32 ビットデータ	DWORD * pdwData
16 ビット BCD データ	WORD * pwData
32 ビット BCD データ	DWORD * pdwData
単精度浮動小数点データ	FLOAT * pflData
倍精度浮動小数点データ	DOUBLE * pdbData
文字列データ	LPTSTR psData
汎用データ	LPVOID pData
汎用データ (VB 用)	LPVARIANT pData

wCount : Read/Write データ数

Read/WriteDeviceStr 関数の場合、文字列データのデータ数は 1 バイト単位です。シンボルが 16 ビット幅のデバイスの場合は 2 文字、32 ビット幅のデバイスの場合は 4 文字単位で指定してください。
読み書きできる最大リード数／ライト数は下表の通りです。

アクセスするデータの種類	リード時／ライト時
ビットデータ	255
16 ビットデータ	1020
32 ビットデータ	510
16 ビット BCD データ	1020
32 ビット BCD データ	510
単精度浮動小数点データ	510
倍精度浮動小数点データ	255
文字列データ	1020 文字（半角）

wAppKind : データタイプ値

データタイプ値の指定方法は値を直接指定する方法と、定義名で指定する方法があります。詳細は以下を参照してください。
☞「37.8.2 デバイスアクセス API ◆ データ型について」(37-103 ページ)

MEMO

- Read/WriteDevice 関数は、データタイプをパラメータで指定するので、動的にデータタイプを変更できます。

＜戻り値＞
正常終了 : 0
以上終了 : エラーコード

＜補足＞
Read/WriteDeviceBit 関数を使用する場合
pwData には、wCount 数分だけ D0 ビットから詰めて格納します。
例 : wCount が 20 の場合

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
PwData	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
PwData+1	*	*	*	*	*	*	*	*	*	*	*	*	20	19	18	17

連続する複数ビットデータを扱う場合は、Read/WriteDeviceBit より Read/WriteDevice16 や Read/WriteDevice32 で 16/32 ビット単位の Read/Write の方が効率的です。
「*」には不定な値が入ります。アプリケーションプログラムでマスクしてください。

Read/WriteDeviceBCD16/32 関数を使用する場合
接続機器内部で、データを BCD として扱っている場合は、これらの関数を使用します。ただし、この関数と受け渡しするデータ（pxxData の内容）は、BCD ではなくバイナリデータとなります。（「WinGP SDK」内部で BCD 変換を行っています。）負の数は扱えません。

関数名	10進表現	16進表現
Read/WriteDeviceBCD16	0 ～ 9999	0000 ～ 270F
Read/WriteDeviceBCD32	0 ～ 99999999	00000000 ～ 05F5E0FF

文字列データ関数を使用する場合
文字列データを受け取る変数は、受け取れるだけの十分なデータ領域を確保してください。

- SRAM 内データアクセス API

関数名	SRAM バックアップデータの読み出し

SRAM 内にある下記データを読み出し、パソコン内のファイルとして保存します。
保存されるファイル形式はファイリングデータの場合はバイナリ形式のファイル、それ以外のデータの場合は CSV 形式のファイルです。

INT WINAPI EasyBackupDataRead(LPCSTR sSaveFileName,LPCSTR sNodeName,INT iBackupDataType,INT iSaveMode);

引数

sSaveFileName: （In）読み出したデータの保存先ファイルのファイルパス（文字列のポインタ）

sNodeName: （In）読み出すデータ元の参加局名（文字列のポインタ）
局名は「#WinGP」固定になります。

iSaveMode: (In) 保存方法
0: 新規（同名のファイルがすでにある場合は、そのファイルを一旦削除し、上書きします。）
1: 追加（ファイルの最後に追記する、まだそのファイルが無い場合は、新規に作成します。）
上記以外 : 予約

iBackupDataType: (In) 読み出すデータの種類

値	データ種類
0x0001	ファイリングデータ
0x0002	サンプリンググループ番号 1 のサンプリングデータ
0x0003	サンプリンググループ番号 1 以外の全てのサンプリンググループのデータ
0x0004	
0x0005	アラームブロック 1
0x0006	アラームブロック 2
0x0007	アラームブロック 3
0x0008	アラームブロック 4
0x0009	アラームブロック 5
0x000A	アラームブロック 6
0x000B	アラームブロック 7
0x000C	アラームブロック 8
上記以外	（予約）

データの種類がアラームブロック 1 ～ 8 の場合、1 アラームブロック内には GP-Pro EX の設定により最大アクティブデータ、ヒストリデータ、ログデータの 3 種類が格納されていますが、この API では、以下の優先順位で有効なデータを持っているか確認し、あればそれを対象とします。
(1) アラームヒストリ
(2) アラームログ
(3) アラームアクティブ
どれも有効でなければエラーとなります。

戻り値

正常終了 :0
異常終了 : エラーコード

関数名	SRAM バックアップデータの拡張読み出し

SRAM 内にある下記データを読み出し、パソコン内のファイルとして保存します。
保存されるファイル形式は　ファイリングデータの場合はバイナリ形式のファイル、それ以外のデータの場合は CSV 形式のファイルです。
EasyBackupDataRead() にくらべ、バックアップデータでとれないデータにアクセスすることができます。

INT WINAPI EasyBackupDataReadEx(LPCSTR sSaveFileName, LPCSTR sNodeName, INT iBackupDataType, INT iSaveMode, INT iNumber = 0 , INT iStringTable = 0x0000);

引数

sSaveFileName: （In）読み出したデータの保存先ファイルのファイルパス（文字列のポインタ）

sNodeName: （In）読み出すデータ元の参加局名（文字列のポインタ）
局名は「#WinGP」固定になります。

iSaveMode: (In) 保存方法
0: 新規（同名のファイルがすでにある場合は、そのファイルを一旦削除し、上書きします。）
1: 追加（ファイルの最後に追記する、まだそのファイルが無い場合は、新規に作成します。）
上記以外 : 予約

iBackupDataType: (In) 読み出すデータの種類

値	データの種類
0x0001	ファイリングデータ
0x0002	サンプリンググループ番号 1 のサンプリングデータ
0x0003	サンプリンググループ番号 1 以外の全てのサンプリンググループのデータ
0x0004	
0x0005	アラームブロック 1 アラームの種類は iNumber で指定します
0x0006	アラームブロック 2 アラームの種類は iNumber で指定します
0x0007	アラームブロック 3 アラームの種類は iNumber で指定します
0x0008	アラームブロック 4 アラームの種類は iNumber で指定します
0x0009	アラームブロック 5 アラームの種類は iNumber で指定します
0x000A	アラームブロック 6 アラームの種類は iNumber で指定します
0x000B	アラームブロック 7 アラームの種類は iNumber で指定します
0x000C	アラームブロック 8 アラームの種類は iNumber で指定します
0x8002	特定のグループ番号のサンプリンググループ グループ番号は iNumber で指定します

戻り値

正常終了 :0
異常終了 : エラーコード

次のページに続きます。

iNumber: iBackupDataType の値により下表の値を入れてください。

iBackupDataType の値	内容
0x0005 から 0x000C	アラームデータの種類にはアクティブ、ヒストリ、ログの 3 種類がありますが、どれを対象にするかを指定します。 iNumber の値 / 内容 0: 以下の優先順位でアラームブロックが有効なデータを持っているか確認し、あればそれを対象とします。 (1) アラームヒストリ (2) アラームログ (3) アラームアクティブ どれも有効でなければエラーとなります。 1: アラームアクティブを対象とします。 2: アラームヒストリを対象とします。 3: アラームログを対象とします。 もし対象としたデータの種類を iBackupDataType で指定されたアラームブロックが持っていない場合はエラーとなります。
0x8002	読み出す対象のサンプリンググループのグループ番号 1 から 64 の値
上記以外	（予約）

iStringTable: (In) 予約
常に 0 を指定してください。

関数名	SRAM バックアップデータの書き込み

指定されたバイナリ形式のファイリングデータを、SRAM 内に書き込みます。

INT WINAPI EasyBackupDataWrite(LPCSTR sSourceFileName,LPCSTR sNodeName,INT iBackupDataType);

引数

sSourceFileName: (In) 書き込むバイナリ型式のファイリングデータファイルのファイルパス（文字列のポインタ）
sNodeName: (In) 書き込み先の参加局名（文字列のポインタ）
局名は「#WinGP」固定になります。
iBackupDataType: (In)「1」固定（ファイリングデータを意味します）

戻り値

正常終了 :0
異常終了 : エラーコード

- **システム系** API

関数名	メッセージ処理の抑制
多くの WinGP SDK API の関数は処理に時間がかかる場合、関数の中で Windows のメッセージを処理していますが、この Windows のメッセージ処理を行うか、抑制するかを指定することができます。 抑制にすると、関数実行中は Windows のメッセージをメッセージキューに蓄積したまま処理しません。 その結果、関数処理中にアイコンがクリックされ、関数の二重呼び出しを行ってしまうようなことが起こらなくなります。 ただし、この場合、「アイコンがクリックされた」というメッセージだけではなく、全ての Windows のメッセージ処理が抑制され、タイマーやウィンドウの再描画など、重要なメッセージの処理が行われませんのでご注意ください。 処理を行うか抑制するかは、WinGP SDK のハンドルごとに指定できます、デフォルトは「処理する」になっています。 INT EasySetWaitType(DWORD dwMode);	
引数 dwMode: (In)1 を指定するとメッセージの処理を行います。 2 を指定するとメッセージの処理を抑制します。	**戻り値** 正常終了 :0 異常終了 : エラーコード

関数名	メッセージ処理方法の取得
WinGP SDK API コール中のメッセージ処理方法が、現在どのモードになっているかを取得する関数です。 INT EasyGetWaitType();	
引数	**戻り値** 1: メッセージの処理を行います。 2: メッセージの処理を抑制します。

関数名	エラーコードの文字列変換

WinGP SDK の各種 API が返したエラーコードをエラーメッセージに変換します。
EasyLoadErrorMessage() は、メッセージとしてマルチバイト文字列（ASCII）を返します。
EasyLoadErrorMessageW() は、メッセージとしてワイド文字列（UNICODE）を返します。

BOOL WINAPI EasyLoadErrorMessage(INT iErrorCode,LPSTR osErrorMessage);
BOOL WINAPI EasyLoadErrorMessageW(INT iErrorCode,LPWSTR owsErrorMessage);

引数

iErrorCode:	（In）WinGP SDK の関数が返したエラーコード
osErrorMessage:	（Out）変換された文字列（マルチバイト文字列）を格納する領域へのポインタ（512 バイト以上を確保しコールしてください。）
owsErrorMessage:	（Out）変換された文字列（ワイド文字列）を格納する領域へのポインタ（1024 バイト以上を確保しコールしてください。）

戻り値
正常終了 :0 以外
文字列変換に失敗（使用されていないエラーコードなど）:0

特記事項
- この API は Pro-Server with Studio との互換性のためにあります。
- EasyLoadErrorMessageEx() の方が、より詳しいエラーメッセージに変換しますので、そちらをご利用ください。

関数名	エラーコードの文字列変換（状況情報付き）

WinGP SDK の各種 API が返したエラーコードをエラーメッセージに変換します。
このとき、可能であれば、エラーが発生した状況や情報を付加してエラーメッセージを返します。
EasyLoadErrorMessage() は、指定されたエラーコードに対して常に同じエラーメッセージを返しますが、EasyLoadErrorMessageEx() は、通信相手名やエラーの発生場所など、エラーの発生した状況により、より詳しいエラー情報を返します。同じエラーコードでも場合によっては違うエラーメッセージを返します。
EasyLoadErrorMessageEx(),EasyLoadErrorMessageExM() は、メッセージとしてマルチバイト文字列（ASCII）を返します。
EasyLoadErrorMessageExW(),EasyLoadErrorMessageExWM() は、メッセージとしてワイド文字列（UNICODE）を返します。

BOOL WINAPI EasyLoadErrorMessageEx(INT iErrorCode,LPSTR osErrorMessage);
BOOL WINAPI EasyLoadErrorMessageExW(INT iErrorCode,LPWSTR owsErrorMessage);

引数

iErrorCode :	（In）WinGP SDK の関数が返したエラーコード
osErrorMessage :	（Out）変換された文字列（マルチバイト文字列）を格納する領域へのポインタ（1024 バイト以上を確保しコールしてください。）
owsErrorMessage :	（Out）変換された文字列（ワイド文字列）を格納する領域へのポインタ（2048 バイト以上を確保しコールしてください。）

戻り値
正常終了 :0 以外
文字列変換に失敗（使用されていないエラーコードなど）:0

特記事項
- EasyLoadErrorMessage() は、WinGP SDK の API をコールして、その API がエラーコードを返した場合、そのエラーコードをメッセージに変換するために利用される場合を想定しています。
- WinGP SDK は、ハンドルごとにエラーの状況情報を 1 つ分しか覚えていません。そのため、エラーの発生元となった API と EasyLoadErrorMessage() の間に、別の API をコールするとエラーの状況情報が書き換えられてしまうため、EasyLoadErrorMessage() はエラーの状況情報を返しません。

- **その他** API

関数名	IPC 時間 DWORD 型読み出し
指定された局の現在時刻を数値（DWORD 型）として取得する関数です。ただし、LS2048 から 6 ワードに時刻が保存されているものに対してのみ有効な関数です。 DWORD WINAPI EasyGetGPTime(LPCSTR sNodeName, DWORD* odwTime);	
引数 sNodeName: 局名は「#WinGP」固定になります。 odwTime: 取得された時刻（時刻は DWORD 型（実体は ANSI で定義されている time_t 型）で取得）	**戻り値** 正常終了 :0 異常終了 : エラーコード
特記事項	

関数名	IPC 時間 VARIANT 型読み出し
指定された局の現在時刻を数値（Variant 型）として取得する関数です。ただし、LS2048 から 6 ワードに時刻が保存されているものに対してのみ有効な関数です。 DWORD WINAPI EasyGetGPTimeVariant(LPCSTR sNodeName, LPVARIANT ovTime);	
引数 sNodeName: 局名は「#WinGP」固定になります。 ovTime: 取得された時刻（取得する時刻は VARIANT 型（内部処理形式は Date）で取得）	**戻り値** 正常終了 :0 異常終了 : エラーコード
特記事項	

関数名	IPC 時間 STRING 型読み出し
指定された局の現在時刻を、文字列（LPTSTR 型）として取得する関数です。ただし、LS2048 から 6 ワードに時刻が保存されているものに対してのみ有効な関数です。 DWORD WINAPI EasyGetGPTimeString(LPCSTR sNodeName, LPCSTR sFormat, LPSTR osTime);	
引数 sNodeName: 局名は「#WinGP」固定になります。 pFormat: 文字列として取得する時刻の書式設定文字列。パーセント記号（%）に続く書式指定コードが、＜特記事項＞に示すものに置き換えられます。 なお、それ以外の文字は、変更されないでそのまま表示されます。 osTime: 文字列として取得した時刻（ただし、取得される文字列長＋ 1（NULL）以上の領域が確保されていない場合、予期せぬメモリ領域の破壊が発生します。そのため、予想される文字列長＋ 1（NULL）以上の領域を確保する必要があります。確保されていない場合、動作保証はいたしません。）	**戻り値** 正常終了 :0 異常終了 : エラーコード

次のページに続きます。

特記事項

パーセント記号（%）に続く書式指定コードが、次表に示すものに置き換えられます。なお、それ以外の文字は、変更されないでそのまま表示されます。例えば、実際の時刻が 2006/1/2 12:34:56 の場合に「%Y_%M %S」と指定すると、文字列は「2006_34 56」となります。

書式指定コード	フォルダ
%a	曜日の省略名（※ 2）
%A	曜日の正式名（※ 2）
%b	月の省略名（※ 2）
%B	月の正式名（※ 2）
%c	ロケールに応じた日付と時間の表現
%#c	ロケールに応じた日付と時間の長い表現
%d	10 進数で表す月の日付（01 ～ 31）（※ 1）
%H	24 時間表記の時間（00 ～ 23）（※ 1）
%I	12 時間表記の時間（01 ～ 12）（※ 1）
%j	10 進数で表す年頭からの日数（001 ～ 366）（※ 1）
%m	10 進数で表す月（01 ～ 12）（※ 1）
%M	10 進数で表す（00 ～ 59）（※ 1）
%p	現在のロケール AM/PM（※ 2）
%S	10 進数で表す秒（00 ～ 59）（※ 1）
%U	10 進数で表す週の通し番号。日曜日を週の最初に日とする（00 ～ 53）（※ 1）
%w	10 進数で表す曜日。日曜日を 0 とする（0 ～ 6）（※ 1）
%W	10 進数で表す週の通し番号。月曜日を週の最初の日とする（00 ～ 53）（※ 1）
%x	現在のロケールの日付表示
%#x	現在のロケールに応じた長い日付表示
%X	現在のロケールの時刻表示（※ 2）
%y	10 進数で表す西暦の下 2 桁（00 ～ 99）（※ 1）
%Y	10 進数で表す 4 桁の西暦（※ 1）
%z、%Z	時間帯の名前またはその省略名。時間帯がわからない場合には文字を入れない（※ 2）
%%	パーセント記号（※ 2）

※ 1：d、H、I、j、m、M、S、U、w、W、y、Y の前に # をつける（例えば、%#d）と、先行ゼロがあれば削除されます。（例えば、05 の場合は 5 となります。）

※ 2：a、A、b、B、p、X、z、Z、% の前に # をつける（例えば、%#a）と、# は無視されます。

関数名	IPC **時間** STRING VARIANT **型読み出し**

指定された局の現在時刻を文字列（Variant 型）として取得する関数です。ただし、LS2048 から 6 ワードに時刻が保存されているものに対してのみ有効な関数です。

DWORD WINAPI EasyGetGPTimeStringVariant(LPCSTR sNodeName, LPCSTR sFormat, LPVARIANT ovTime);

引数

sNodeName:	局名は「#WinGP」固定になります。
pFormat:	文字列として取得する時刻の書式設定文字列。パーセント記号（%）に続く書式指定コードが、以下に示すものに置き換えられます。なお、それ以外の文字は、変更されないでそのまま表示されます。（詳細については、「IPC 時間 STRING 型読み出し関数」の＜特記事項＞を参照してください。）
ovTime:	文字列として取得した時刻（取得する時刻は VARIANT 型（内部処理形式は BSTR）で取得）

戻り値

正常終了 :0

異常終了 : エラーコード

関数名	参加局ステータス読み出し

接続されている機器（IPC）状態を取得することができます。また、レスポンスタイムアウト値を可変に設定できるため、接続確認にも使用できます。

INT WINAPI GetNodeProperty(LPCSTR sNodeName,DWORD dwTimeLimit,LPSTR osGPType,LPSTRosSystemVersion,LPSTR osComVersion,LPSTR osECOMVersion);

引数

sNodeName: 局名は「#WinGP」固定になります。
dwTimeLimit: （In）レスポンスタイムアウト設定値
（0 を設定した場合はデフォルト値 3000msec）
設定範囲は 1 ～ 2,147,483,647、単位 msec

以下のエリアに対象局の情報が返されます。
それぞれ 32 バイト以上のエリアを確保してください。
osGPType: （Out）機種コード
osSystemVersion: （Out）システムバージョン
osComVersion: （Out）PLC プロトコルドライバのバージョン
空白です。
osECOMVersion: （Out）2Way ドライバのバージョン
空白です。

戻り値

正常終了 :0
異常終了 : エラーコード

関数名	シンボル／グループのバイトサイズを求める

デバイスシンボルやグループシンボルにアクセスするために必要なバッファの合計バイト数を求めます。

INT WINAPI SizeOfSymbol(LPCSTR sNodeName,LPCSTR sSymbolName,INT* oiByteSize);

引数

sNodeName: 局名は「#WinGP」固定になります。
sSymbolName: （In）求めたいデバイスシンボル名もしくはグループシンボル名
oiByteSize: （Out）求めたいバイトサイズ

戻り値

正常終了 :0
異常終了 : エラーコード

特記事項

sSymbolName にはデバイスシンボル、非配列グループ、配列グループ全体の全体、配列グループの 1 要素分を指定することができます。

関数名	グループのメンバー数を求める

指定されたグループシンボルもしくはシンボルシート内のメンバー（シンボルとグループの合計）数を求めます。

INT WINAPI GetCountOfSymbolMember(LPCSTR sNodeName,LPCSTR sSymbolName,INT* oiCountOfMember);

引数

sNodeName: 局名は「#WinGP」固定になります。
sSymbolName: （In）求めたいグループシンボル名またはシンボルシート名
oiCountOfMember:（Out）求めたいメンバー数

戻り値

正常終了 :0
異常終了 : エラーコード

特記事項

指定されたグループシンボル内にグループシンボルがある場合、内側のグループシンボルに複数のデバイスシンボルがいくつあっても、1 メンバーとカウントされます。

関数名	シンボル／グループ／シンボルシートの定義情報を求める

指定されたデバイスシンボル／グループシンボル／シンボルシートの定義情報（データ型やデータ数など）を求めます。

INT WINAPI GetSymbolInformation(LPCSTR sNodeName,LPCSTR sSymbolName,INT iMaxCountOfSymbolMember,LPSTR osSymbolSheetName,SymbolInformation* oSymbolInformation,INT* oiGotCountOfSymbolMember);

引数

sNodeName: 局名は「#WinGP」固定になります。
sSymbolName: （In）シンボル／グループ名／シート名
iMaxCountOfSymbolMember: （In）求める情報の最大数 1 以上の値を指定してください。用意した oSymbolInformation の個数を指定します。
osSymbolSheetName: （Out）sSymbolName が属しているシンボルシートの名前を返します。66 バイト以上のワークを用意してください。
oSymbolInformation: （Out）求めた詳細情報を配列の形で返します。iMaxCountOfSymbolMember で指定した個数分をワークとして用意してください。
oiGotCountOfSymbolMember: （Out）実際に oSymbolInformation に返した情報数を返します。

戻り値

正常終了 :0
異常終了 : エラーコード

特記事項

• SymbolInformation の構造

```
struct SymbolInformation
{
  WORD m_wAppKind;// データタイプ　シンボルの場合は 1 ～ 12,
グループの場合は 0x8000
  WORD m_wDataCount; // データ数
  DWORD m_dwSizeOf; // アクセスするのに必要なバッファのバイト数
  char m_sSymbolName[64+1];// シンボルもしくはグループの名前
  char m_bDummy1[3];// 予約
  char m_sDeviceAddress[256+1]; // デバイスアドレス（グループの場合は空白です )
  char m_bDummy2[3];// 予約
};
```

oSymbolInformation に求めた情報が、SymbolInformation 構造の配列で返りますが、1 個目には sSymbolName で指定されたシンボルもしくはグループ、シートの情報がセットされます。
2 個目以降には、sSymbolName がグループの場合は、グループのメンバーの情報がセットされます。
sSymbolName がシートの場合は、シート全体の情報がセットされます。
sSymbolName がシンボルの場合には、2 個目以降はありません。

対象のシンボルがビットオフセットシンボルの場合、以下の点について注意してください。

(1) ビットオフセットシンボルを情報元のシンボルとして直接指定した場合（sSymbolName にビットオフセットシンボルを直接指定した場合）は、oSymbolInformation の 1 個目の SymbolInformation の m_dwSizeOf には、ビットシンボルをアクセスするためのバイト数 2 がセットされます。
情報元が 1 シンボルなので、oSymbolInformation の 2 個目以降はありません。

(2) 情報元のシンボルとしてはグループシンボルを指定し、そのグループ内にビットオフセットシンボルがある場合、oSymbolInformation の 2 個目以降の m_dwSizeOf は、グループアクセス時のメンバーとしてアクセスサイズなので 0 がセットされます。

• メンバー数が不明な場合は、GetCountOfSymbolMember() でメンバー数を求め、その値＋ 1 の分の SymbolInformation をワークとして用意し、この関数を呼び出してください。

- CF カード関係 API

関数名	CF カードステータス読み出し

接続されている機器（IPC）の CF カード接続状態を取得できます。

INT WINAPI EasyIsCFCard(LPCSTR sNodeName) ;

引数

sNodeName: 局名は「#WinGP」固定になります。
ネットワークプロジェクトに登録されている必要があります。）

戻り値

関数の戻り値	ステータス
0x00000000	正常
0x10000001	CF カードなし、または CF カードスロットのカバーが開いている（CF カード有無は関係なし）
0x10000002	
0x10000004	CF カード異常を検出
0x10000008	
その他	カード関連以外のエラー

関数名	CF カード内ファイル一覧読み出し（任意フォルダ名）

IPC に挿入されている CF カード内にあるファイル一覧をパラメータで渡されたファイルに出力します。ファイル一覧を取得したいフォルダを任意に指定できます。

INT WINAPI EasyGetListInCfCard(LPCSTR sNodeName, LPCSTR sDirectory, INT* oiCount, LPCSTR sSaveFileName) ;

引数

sNodeName:	局名は「#WinGP」固定になります。
sDirectory:	取得するフォルダ名（すべて大文字）
oiCount:	読み出したファイルの数
sSaveFileName:	読み出したディレクトリ情報の格納先ファイル名。なお、指定したファイル内には、stEasyDirInfo 型の配列に格納されたデータが、pioCount で返された個数分、バイナリデータとして格納されます。なお、ファイル名、ファイルの拡張子はすべて大文字として保存されます。

```
struct stEasyDirInfo {
   BYTE bFileName[8+1];// ファイル名 ( 最後は 0 で完結 )
BYTE bExt[3+1];// ファイルの拡張子 ( 最後は 0 で完結 )
BYTE bDummy[3];// ダミー
DWORD dwFileSize;// ファイルのサイズ
BYTE bFileTimeStamp[8+1];// ファイルのタイムスタンプ ( 最後は 0 で完結 )
BYTE bDummy2[3];// ダミー 2
} ;
```

戻り値

正常終了：0
異常終了：エラーコード

特記事項

「bFileTimeStamp」の補足として、8 バイトのうち、上位 4 バイトが MS-DOS 形式の時刻を、下位 4 バイトが MS-DOS 形式の日付を 16 進文字列として表しています。
なお、MS-DOS 形式の日付、時刻のフォーマットは以下のとおりです。
(例：20C42C22 の場合、2C22 が MS-DOS の日付を 16 進で表記したもの、20C4 が MS-DOS の時刻を 16 進で表したものとなるため、2002/1/2 4:6:8 を表すことになります。)

ビット	内容
0 ～ 4	年月日の日（1 ～ 31）。
5 ～ 8	年月日の月（1=1 月、2=2 月、～ 12=12 月）。
9 ～ 15	年月日の年。ただし、1980 年からの経過年数で指定します。これらのビットが表す値に 1980 を足すと、実際の年が得られます。

MS-DOS 形式の時刻を指定します。この日付は、次の形式で 1 個の 16 ビット値にパックされています。

ビット	内容
0 ～ 4	時分秒の秒を 2 で割った値です（0 ～ 29）。
5 ～ 10	時分秒の分（0 ～ 59）。
11 ～ 15	時分秒の時（24 時間制で 0 ～ 23）。

関数名	CF カード内ファイル一覧読み出し（タイプ指定）

IPC に挿入されている CF カード内にあるファイル一覧をパラメータで渡されたファイルに出力します。読み出すファイル一覧は “ sDirectory ” で指定するディレクトリに限ります。

INT WINAPI EasyGetListInCard(LPCSTR sNodeName, LPCSTR sDirectory, INT* oiCount, LPCSTR sSaveFileName);

引数

sNodeName:	局名は「#WinGP」固定になります。
sDirector:	取得するディレクトリ名 (全て大文字)。なお、以下のディレクトリのみサポートします。 LOG（ロギングデータ） TREND（トレンドデータ） ALARM（アラームデータ） CAPTURE（キャプチャデータ） FILE（ファイリングデータ）
oiCount:	読み出したファイルの数

sSaveFileName:	読み出したディレクトリ情報の格納先ファイル名。なお、指定したファイル内には、stEasyDirInfo 型の配列に格納されたデータが、pioCount で返された個数分、バイナリデータとして格納されます。なお、ファイル名、ファイルの拡張子はすべて大文字として保存されます。
	struct stEasyDirInfo {
	BYTE bFileName[8+1];// ファイル名 (最後は 0 で完結)
	BYTE bExt[3+1];// ファイルの拡張子 (最後は 0 で完結)
	BYTE bDummy[3];// ダミー
	DWORD dwFileSize;// ファイルのサイズ
	BYTE bFileTimeStamp[8+1];// ファイルのタイムスタンプ (最後は 0 で完結)
	BYTE bDummy2[3];// ダミー 2
	} ;

戻り値

正常終了：0
異常終了：エラーコード

関数名	CF カードファイル読み出し（任意ファイル名指定）

CF カードに存在する指定されたファイルの内容を読み出す関数です。読み出すファイルを任意に指定できます。

INT WINAPI EasyFileReadInCfCard(LPCSTR sNodeName, LPCSTR sFolderName, LPCSTR sFileName, LPCSTR pWriteFileName, DWORD* odwFileSize);

引数

sNodeName:	局名は「#WinGP」固定になります。
sFolderName:	読み出す CF カード内のファイルのフォルダ名（最大文字数半角 32 文字）
sFileName:	読み出す CF カード内のファイル名（最大 8.3 形式の文字列）
pWriteFileName:	読み出した CF ファイルの保存先ファイル名（フルパス）
odwFileSize:	読み出した CF ファイルのファイルサイズ

戻り値

正常終了：0
異常終了：エラーコード

関数名	CF カード内ファイル読み出し（タイプ指定）

CF カードに存在する指定されたファイルの内容を読み出す関数です。読み出すファイルは“ pReadFileType ”で指定するファイルの種類に限ります。

INT WINAPI EasyFileReadCard(LPCSTR sNodeName, LPCSTR pReadFileType, WORD wReadFileNo, LPCSTR sWriteFileName, DWORD* odwFileSize);

引数

sNodeName: 局名は「#WinGP」固定になります。
pReadFileType: 読み出す CF カード内のファイルの種類（<特記事項>参照）
wReadFileNo: 読み出す CF カード内のファイル番号
sWriteFileName: 読み出した CF ファイルの保存先ファイル名（フルパス）
odwFileSize: 読み出した CF ファイルのファイルサイズ

戻り値

正常終了：0
異常終了：エラーコード

特記事項

サポートしているファイルの種類は以下の表のとおりです。CF カードの指定フォルダ内に保存されているもののみ、読み込み可能です。

■ ファイルの種類

データ種別	ファイルの種類	対象フォルダ
ファイリングデータ	ZF または F	FILE
CSV データ	ZR	FILE
イメージ画面	ZI または I	DATA
サウンドデータ	ZO または O	DATA
GP-Pro EX 専用折れ線グラフデータ（互換用）	ZT	TREND
GP-Pro EX 専用データサンプリングのデータ（互換用）	ZS	TREND
アラーム 1	Z1 または ZA	ALARM
アラーム 2	Z2 または ZH	ALARM
アラーム 3	Z3 または ZG	ALARM
アラーム 4 ～ 8	Z4 ～ Z8	ALARM
『GP-Pro EX』専用ロギングデータ（互換用）	ZL	LOG
キャプチャデータ	CP	CAPTURE
サンプリング 1 ～ 64	ZS1 ～ ZS64	SAMP01 ～ SAMP64

関数名	CF カードファイル書き込み（任意ファイル名指定）

指定されたファイルを CF カードへ書き込む関数です。書き込むファイルを任意に指定できます。

INT WINAPI EasyFileWriteInCfCard(LPCSTR sNodeName, LPCSTR pReadFileName, LPCSTR sFolderName,LPCSTR sFileName);

引数		戻り値
sNodeName:	局名は「#WinGP」固定になります。	正常終了：0
pReadFileName:	CF カードへの書き込み元となるファイル名（フルパス）	異常終了：エラーコード
sFolderName:	CF カード内へ書き込むファイルのフォルダ名（最大文字数半角 32 文字）	
sFileName:	CF カード内へ書き込むファイル名（最大 8.3 形式の文字列）	

関数名	CF カードファイル書き込み（タイプ指定）

指定されたファイルを CF カードへ書き込む関数です。書き込むファイルは “ pWriteFileType ” で指定するファイルの種類に限ります。

INT WINAPI EasyFileWriteCard(LPCSTR sNodeName, LPCSTR pReadFileName, LPCSTR sWriteFileType, WORD wWriteFileNo);

引数		戻り値
sNodeName:	局名は「#WinGP」固定になります。	正常終了：0
pReadFileName:	CF カードへの書き込み元となるファイル名（フルパス）	異常終了：エラーコード
sWriteFileType:	CF カード内へ書き込むファイルの種類（CF カードファイル読み出し関数（タイプ指定）の＜特記事項＞を参照）	
wWriteFileNo:	CF カード内へ書き込むファイルの番号	

関数名	CF カードファイル削除（任意ファイル指定）

CF カードに存在する指定されたファイルを削除する関数です。削除するファイルを任意に指定できます。

INT WINAPI EasyFileDeleteInCfCard(LPCSTR sNodeName, LPCSTR sFolderName, LPCSTR sFileName) ;

引数		戻り値
sNodeName:	局名は「#WinGP」固定になります。	正常終了：0
sFolderName:	削除する CF カード内のファイルのフォルダ名（最大文字数半角 32 文字)	異常終了：エラーコード
sFileName:	削除する CF カード内のファイル名（最大 8.3 形式の文字列）	

次のページに続きます。

特記事項

■ サポートするファイルの種類

データ種別	ファイルの種類	対象フォルダ
ファイリングデータ	ZF または F	FILE
CSV データ	ZR	FILE
イメージ画面	ZI または I	DATA
サウンドデータ	ZO または O	DATA
GP-Pro EX 専用折れ線グラフデータ（互換用）	ZT	TREND
GP-Pro EX 専用データサンプリングのデータ（互換用）	ZS	TREND
アラーム 1	Z1 または ZA	ALARM
アラーム 2	Z2 または ZH	ALARM
アラーム 3	Z3 または ZG	ALARM
アラーム 4 ～ 8	Z4 ～ Z8	ALARM
GP-Pro EX 専用ロギングデータ（互換用）	ZL	LOG
キャプチャデータ	CP	CAPTURE
サンプリング 1 ～ 64	ZS1 ～ ZS64	SAMP01 ～ SAMP64

関数名	CF カードファイル名変更

CF カードに存在する指定されたファイルを名称変更する関数です。

INT WINAPI EasyFileRenameInCfCard(LPCSTR sNodeName, LPCSTR sFolderName, LPCSTR sFileName, LPCSTR sFileRename) ;

引数

sNodeName:	局名は「#WinGP」固定になります。
sFolderName:	CF カード内の名前変更するファイルのフォルダ名（最大文字数半角 32 文字）
sFileName:	CF カード内の名前変更されるファイル名（最大 8.3 形式の文字列）
sFileRename:	名前変更後のファイル名（最大 8.3 形式の文字列）

戻り値

正常終了：0
異常終了：エラーコード

関数名	CF カードファイル削除（タイプ指定）

CF カードに存在する指定されたファイルを削除する関数です。削除するファイルは “ pDeleteFileType ” で指定するファイルの種類に限ります。

INT WINAPI EasyFileDeleteCard(LPCSTR sNodeName, LPCSTR pDeleteFileType, WORD wDeleteFileNo);

引数

sNodeName: 局名は「#WinGP」固定になります。
pDeleteFileType: 削除する CF カード内のファイルの種類（<特記事項>参照）
wDeleteFileNo: 削除する CF カード内のファイル番号

戻り値

正常終了：0
異常終了：エラーコード

特記事項

存在しないファイルに対してこの関数を実行した場合は、エラーにならず正常終了します。
サポートしているファイルの種類は以下の表のとおりです。CF カードの指定フォルダ内に保存されているもののみ、読み込み可能です。

■サポートするファイルの種類

データ種別	ファイルの種類	対象フォルダ
ファイリングデータ	ZF	FILE
CSV データ	ZR	FILE
イメージ画面	ZI	DATA
サウンドデータ	ZO	DATA
折れ線データ	ZT	TREND
サンプリング	ZS	TREND
アラーム 4 ～ 8	Z4 ～ Z8	ARAM
ロギングデータ	ZL	LOG
アラームログ	ZG	ALARM
アラームヒストリ	ZH	ALARM
アラームアクティブ	ZA	ALARM
画面データバックアップ	ZC	MRM
画面キャプチャ	CP	CAPTURE

関数名	CF カード空き容量取得
指定された参加局に接続されている CF カード内の空き容量を取得する関数です。 INT WINAPI EasyGetCfFreeSpace(LPCSTR sNodeName, INT* oiUnallocated);	
引数 sNodeName: 局名は「#WinGP」固定になります。 oiUnallocated: CF カード内の空き容量（バイト単位として取得される）	**戻り値** 正常終了：0 異常終了：エラーコード
特記事項	

関数名	FTP パッシブモード設定
CF カードへのアクセスは FTP プロトコルで通信を行います。 WinGP SDK の FTP プロトコルは通常モードとパッシブポートの 2 モードをサポートしています。 この API はそのモードを設定します。 INT WINAPI EasyFileSetPassiveMode(INT iPassive);	
引数 iPassive: （In）0：通常モード 0 以外 : パッシブモード WinGP SDK 初期化時は「通常モード」です。	**戻り値** 正常終了：0 異常終了：エラーコード
特記事項	

- **キューイングアクセス制御** API

関数名	デバイスリード要求のキューイング開始
この API コール後、ExecuteQueuingAccess() がコールされるまでデバイスリード系の要求をキューイングします。 キューイングは、WinGP SDK ハンドル単位で行われます。 INT WINAPI BeginQueuingRead();	
引数	**戻り値** 正常終了：0 異常終了：エラーコード
特記事項 • BeginQueuingRead() がコールされてから ExecuteQueuingAccess() がコールされるまで、デバイスライト系の API をコールしないでください。以後のキャッシュリード、ダイレクトリードの命令はキューイングされます。ただし、キャッシュリードとダイレクトリードの命令を混在させることはできません。 • キューイング中の命令を破棄する場合は、CancelQueuingAccess() をコールしてください。 • キューイングできる最大命令数は 1500 件、データの最大バイト数は 1MByte 以内です。	

関数名	デバイスライト要求のキューイング開始
この API コール後、ExecuteQueuingAccess() がコールされるまで、デバイスリード系の要求をキューイングします。 キューイングは WinGP SDK ハンドル単位で行われます。 INT WINAPI BeginQueuingWrite();	
引数	**戻り値** 正常終了：0 異常終了：エラーコード
特記事項 • BeginQueuingWrite() がコールされてから ExecuteQueuingAccess() がコールされるまで、デバイスリード系の API をコールしないでください。以後のキャッシュライト、ダイレクトライトの命令はキューイングされます。ただし、キャッシュライトとダイレクトライトの命令を混在させることはできません。 • キューイング中の命令を破棄する場合は、CancelQueuingAccess() をコールしてください。 • キューイングできる最大命令数は 1500 件、データの最大バイト数は 1MByte 以内です。	

関数名	キューイングしているデバイスリード／ライト要求の実行
キューイングしているデバイスリード／ライト要求に従い、実際にデバイスデータへアクセスします。 INT WINAPI ExecuteQueuingAccess();	
引数	**戻り値** 正常終了：0 異常終了：エラーコード
特記事項 • ExecuteQueuingAccess() は、すべてのデバイスへのアクセスが成功すると成功を返し、一つでも失敗するとアクセスエラーを返します。デバイスごとにアクセスの成否が知りたい場合は、IsQueuingAcceessSucceeded() をコールし確認します。 • キューイングアクセスにアクションを登録することはできません。	

関数名	キューイングしているデバイスリード／ライト要求の破棄
キューイングしているデバイスリード／ライト要求を破棄します。 INT WINAPI CancelQueuingAccess();	
引数	**戻り値** 正常終了：0 異常終了：エラーコード
特記事項 BeginQueuingWrite() または BeginQueuingRead() をコール後、ExecuteQueuingAccess() をコールするまでは、デバイスへのアクセスの要求はキューイングされます。 もし、何らかの理由で、キューイングされている要求が不要になった場合は、この API をコールしてください。キューイングしている要求を破棄し、キューイング動作を終了します。	

関数名	キューイングしているデバイスリード／ライト要求の破棄
ExecuteQueuingAccess() をコールしたあと、ここのデバイスアクセスへの要求の成否を求めます。 INT WINAPI IsQueuingAccessSucceeded(INT iIndex);	
引数 iIndex : （In）確認したい要求の番号 BeginQueuingWrite() または BeginQueuingRead() をコールして、ExecuteQueuingAccess() をコールするまでに、デバイスアクセスの要求をキューイングするために、デバイスアクセス系の API を何回かコールします。ただし、実際のデバイスアクセスの結果は、ExecuteQueuingAccess() 実行後でないと分かりません。 デバイスアクセスの結果を知りたい場合、ExecuteQueuingAccess() 実行後、その知りたいデバイスを要求した順番（0 からの番号）で指定してください。	**戻り値** XX ：エラーコード 0 ：指定された番号のデバイスアクセスは成功しています。
特記事項 （例） BeginQueuingWrite(); WriteDevice16("Node1","LS100",Data,10); WriteDevice16("Node1","LS200",Data,10); WriteDevice16("Node1","LS300",Data,10); ExecuteQueuingAccess() 上記の登録で、"Node1" の "LS200" へのアクセスが成功しかたどうかは、IsQueuingAccessSucceeded(1) で確認します。 0 が返ればアクセスは成功しています。	

◆データ型について

- API でデータタイプを指定する場合、または答えとして受け取る場合のデータタイプの基本形

定義名	10 進値	16 進値	データの意味
EASY_AppKind_Bit	1	0x0001	ビットデータ
EASY_AppKind_SignedWord	2	0x0002	16 ビット符号付きデータ
EASY_AppKind_UnsignedWord	3	0x0003	16 ビット符号無しデータ
EASY_AppKind_HexWord	4	0x0004	16 ビット 16 進データ
EASY_AppKind_BCDWord	5	0x0005	16 ビット BCD データ
EASY_AppKind_SignedDWord	6	0x0006	32 ビット符号付きデータ
EASY_AppKind_UnsignedDWord	7	0x0007	32 ビット符号無しデータ
EASY_AppKind_HexDWord	8	0x0008	32 ビット 16 進データ
EASY_AppKind_BCDDWord	9	0x0009	32 ビット BCD データ
EASY_AppKind_Float	10	0x000A	単精度浮動小数点データ
EASY_AppKind_Real	11	0x000B	倍精度浮動小数点データ
EASY_AppKind_Str	12	0x000C	文字列データ

- 特殊な場合に利用可能なデータタイプ

定義名	10 進値	16 進値	データの意味
EASY_AppKind_NULL	0	0x0000	Default（現在の記載している内容を書く） デバイスアドレスとしてシンボルを利用可能な API でそのシンボルに定義づけされているデータ型をデータ型として利用することを示します。
EASY_AppKind_BOOL	513	0x0201	BOOL（現在の記載している内容を書く） Bit データを 1 ビット単位で VARIANT 型の BOOL として扱います。

◆接続機器名の指定

GP-Pro EX ではデバイスを指定する場合、シンボル名を指定すると接続機器まで指定したことになりますが、デバイスアクセス API では接続機器名を別途指定する必要があります。
例）ReadDevice 16("#WinGP","PLC.1 バルブ ", Data,10);

◆デバイス長

物理的に 16 ビット幅のデバイスに対し 32 ビットアクセスした場合の動作について

WinGP では、物理的に 16 ビット幅のデバイスに対し、32 ビットのシンボルを割り付け、そのシンボルでアクセスした場合や、データ型を直接 32 ビット型としてアクセスした場合など、16 ビット幅のデバイスに対し 32 ビットデバイスとして扱うことができます。
その場合、連続する 2 つの 16 ビット幅のデバイスを 1 デバイスとして扱います。

◆シンボルのインデックス指定（16 ビットの場合）

デバイスアクセス API のデバイス名でのみ、シンボルのインデックス指定ができます。シンボルのインデックス指定とは下記のとおり、シンボル名の後ろに [] で数値を指定する方法です。意味は、そのシンボルのデータ型で「数値」で指定されている個数分、シンボル名で表されるデバイスから進めたデバイスを指します。

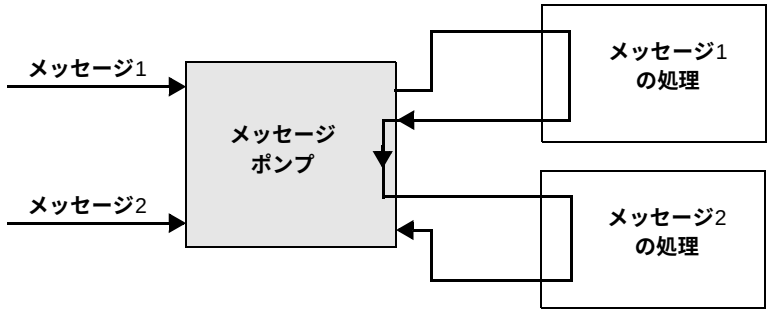
(シンボル名)[数値]

例) バルブ [2]
バルブというシンボルが D100 で 16 ビット符号ありと指定されていた場合、D102 を指すことになります。また、D100 で 32 ビット符号なしと指定されていた場合は、D104 を指すことになります。

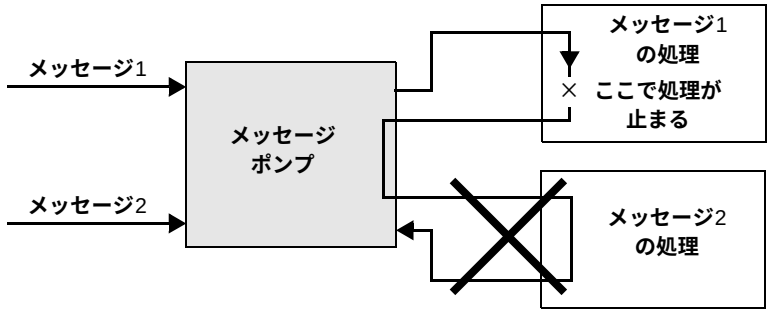
◆Windows のメッセージ処理について

多くの Windows のプログラムは、「アイコンがクリックされた」、「マウスが動かされた」、「キーが押された」などのイベントに応じて、ダイアログを表示したり、音をならしたりする、イベント駆動型 (イベントドリブン) のプログラムになっています。
Windows はイベントが発生すると、そのイベントの種類を示すメッセージをアプリケーションに送ります。
アプリケーションはメッセージを受け取ることでイベントが発生したことを確認し、それぞれの処理を行います。
本書では Windows からメッセージを順番に受け取り、個々の処理へ分岐する部分 (VB なら DoEvents, に相当し、VC なら GetMessage() と DispatchMessage() を行う部分) をメッセージポンプと呼ぶことにします。
メッセージポンプは VC や VB で通常プログラムすると、VC や VB のフレームワークに隠蔽されていますが、このメッセージポンプが上手く動作しないと、Windows のアプリケーションは意図しない動作を行うアプリケーションになります。
例えば、あるメッセージを処理するルーチンが処理に時間がかかり復帰しない場合、その間に発生したイベントをアプリケーションは Windows から受け取ることができないため、そのイベントの処理ができません。

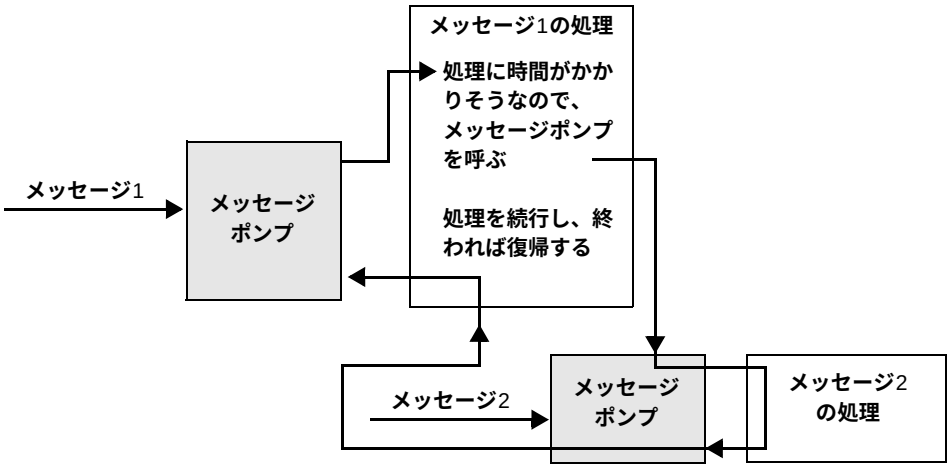
例) メッセージ 1、メッセージ 2 の順番でメッセージが Windows から送られる場合、メッセージポンプはメッセージ 1 を取り出し、メッセージ 1 用のサブルーチンを呼び出します。
そして、そこから復帰してくると次のメッセージ (メッセージ 2) を取り出し、メッセージ 2 用のサブルーチンを呼び出します。



この時、メッセージ 1 の処理が長い時間かかる場合、メッセージポンプへ復帰できないので、メッセージポンプ 2 の処理ができません。



このような場合に、メッセージポンプを強制的に動作させてください。(VB なら DoEvents, VC なら GetMessage() と DispatchMessage() を呼ぶ)



Windows のアプリケーションはアプリケーションが上手くメッセージポンプを動かすことを前提に作られた OS です。WinGP SDK は例に示したような事が起こらないように、時間のかかる処理の場合、関数内でメッセージポンプを動かしています。

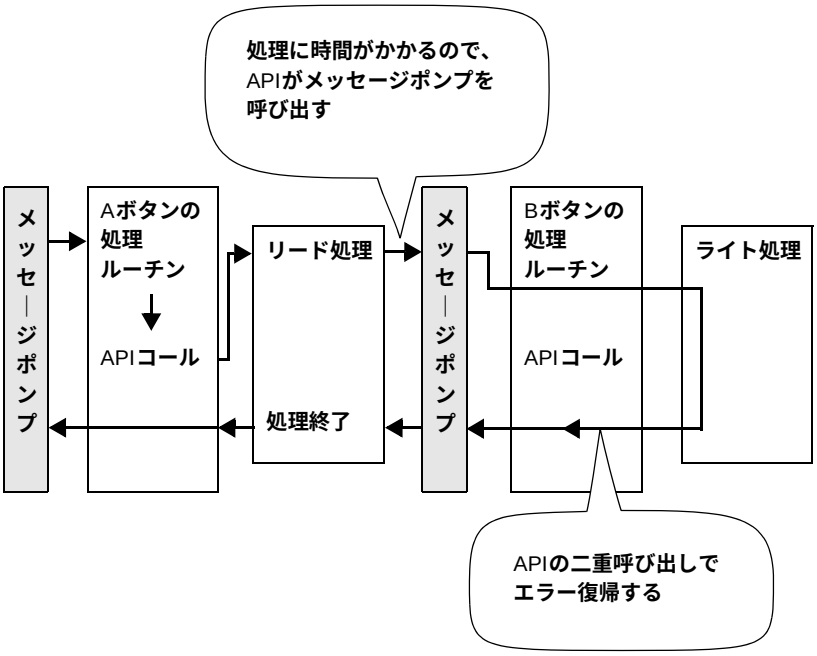
◆API 二重読み出しの禁止

- API 二重読み出し

WinGP SDK は、1 つのデバイスアクセス API を呼び出し中にさらに別のデバイスアクセス API を呼び出すことを (二重呼び出し) 禁止しています。しかし、デバイスアクセス API は、API 内でメッセージポンプを動かしていますので、イベントが発生すればユーザープログラムが動き出します。メッセージの処理ルーチンの中で、API を呼び出すと二重呼び出しが発生することがあります。
二重呼び出しになる事例を以下に示します。

(1) 2 つのボタンを押すことによる二重呼び出し

2 つのボタン A と B があり、A が押されるとデバイスのリード API を呼び出し、B が押されるとデバイスのライト API を呼び出すとします。
この場合、A のボタンを押しデバイスのリード API を呼び出している最中に B のボタンを押すとデバイスのライト API が呼ばれ、その時点で API の二重呼び出しとなり、エラーとなります。



(2) タイマーによる二重呼び出し

Windows のプログラムで周期的な処理を行う場合、よくタイマーイベントを利用しますが、タイマーイベントを利用するプログラムでは、気を付けてプログラムしないと、API を二重に呼び出してしまうことがあります。

1) 1 秒に一回、周期的にデバイスリード API を呼び出しデバイスをリードし、それを表示する
2) あるボタンが押されるとデバイスライト API を呼び出し、デバイスに値を書き込む

このようなプログラムは、以下のようなタイミングの場合、エラーになります。

・1) のタイマーイベントが発生してリード中に、2) のボタンを押し、2) の処理が動き出したとき
・2) のライト中にタイマーイベントが発生し、1) のリードを行うとき

- **API の二重呼び出しの回避策**

API の二重呼び出しの回避策には、以下のような方法があります。

(1) ユーザープログラムで API の二重呼び出しを行わないように、アルゴリズムを改良する。
例えば、
- タイマー処理ルーチンおよびボタンの処理ルーチンの先頭で、必ずタイマーのキャンセルを行う。
- 1 つのボタンが押されて処理をしている間は、そのボタンや別のボタンが押されても無視するようにする。

(2) API 内でメッセージ処理をしないようにする
EasySetWaitType() を引数 2 でコールする。ただし、この場合二重呼び出しの元となるメッセージ以外もメッセージについて処理されないので、アプリケーションが意図しない動作を行うなど他の問題が発生することがあります。

◆VB で文字列をリードする場合

VB で文字列をリードするには以下のような、2 種類の方法があります。

(1) VB で ReadDeviceStr を利用し文字列をリードする場合
この場合、事前にリードした文字列の格納先のサイズを指定 (固定) する必要があります。

```
Public Sub Sample1()
    Dim strData As String * 10      ' 読み込むサイズを指定しているので正しい指定方法
    'Dim strData As String          ' 文字列のサイズを指定していないので誤った指定方法

    Dim IErr As Long

    IErr  = ReadDeviceStr("ReadDeviceStrD", "ReadDeviceVariantD", strData, 10)
    If IErr <> 0 Then
        MsgBox "Read Error = " & IErr
    Else
        MsgBox "Read String = " & strData
    End If

End Sub
```

(2) VB で ReadDeviceVariant を利用し文字列をリードする場合
事前にリードした文字列の格納先のサイズを指定しない場合は Variant 型を利用します。

```
Public Sub Sample2()

    Dim IErr As Long
    Dim vrData As Variant           ' 読み込んだデータの格納先に Variant 型を指定します
    IErr = ReadDeviceVariant("GP1", "LS100", vrData, 10, EASY_AppKind_Str)
    If IErr <> 0 Then
        MsgBox "Read Error = " & IErr
    Else
        MsgBox "Read String = " & vrData
    End If
End Sub
```

ここで気を付ける点として、WinGP SDK は文字列の完結に NULL を利用しています。そのため、上記の方法で取得した文字列に文字列完結の NULL があれば、文字列を縮める必要があります。
文字列を NULL まで縮めるサンプル関数を示します。

```
Public Function TrimNull(strData As String) As String
    Dim i As Integer
    i = InStr(1, strData, Chr$(0), vbBinaryCompare)
    If 0 < i Then
        TrimNull = Left(strData, i - 1)
    Else
        TrimNull = strData
    End If
End Function
```

◆エラーコード一覧

「戻り値」で確認できるエラーコードです。

MEMO
- エラーメッセージ中の「Pro-Server」や「Pro-Studio」は、「WinGP SDK」と読み替えてください。

- REAA で始まるエラー

エラーコード	エラーメッセージ	原因と対処方法
0xC0A10010 REAA016 -1063190512 3231776784	xx **ポート (番号** :xx) **を使用することができませんでした** (xx: **ポート名 / 番号**)	xx ポート (番号 :xx) を使用することができませんでした。システムポート番号が既に使用されている可能性があります。
0xC0A10011 REAA017 -1063190511 3231776785	**書込み禁止エリアにアクセスしました** (xx) (xx: **デバイス名**)	D スクリプトまたはネットワーク越しで書き込み禁止エリア（LS0000 ～ LS0019、LS2032 ～ LS2095、LS9000 ～ LS9999）に書き込みすることはできません。
0xC0A10012 REAA018 -1063190510 3231776786	**アドレス範囲外のデバイスにアクセスしました** (xx) (xx: **デバイス名**)	範囲外のデバイスにアクセスしました。
0xC0A10015 REAA021 -1063190507 3231776789	**不正な** ID(**局、機器、デバイス**) **が指定されました**	不正な ID が指定されました。存在しないデバイスにアクセスしています。
0xC0A10016 REAA022 -1063190506 3231776790	**不正な** ID(**局、機器、デバイス**) **が指定されました**	
0xC0A1001A REAA026 -1063190502 3231776794	**不正または未設定のデバイスアドレスがあります**	不正なデバイスが指定されました。存在しないデバイスにアクセスしています。
0xC0A1001B REAA027 -1063190501 3231776795	**不正または未設定のデバイスアドレスがあります**	
0xC0A1001C REAA028 -1063190500 3231776796	**不正または未設定のデバイスアドレスがあります**	

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

- RYAA で始まるエラー

エラーコード	エラーメッセージ	原因と対処方法
0xC0AF0001 RYAA001 -1062273023 3232694273	**指定された共有メモリは既にあります。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0AF0002 RYAA002 -1062273022 3232694274	**指定された共有メモリはありません。**	
0xC0AF0003 RYAA003 -1062273021 3232694275	**共有メモリは既にありますが、その共有メモリは指定されたサイズ分はありません。**	他のアプリケーションを終了するか OS を再起動してください。
0xC0AF0004 RYAA004 -1062273020 3232694276	**メモリやリソース不足で共有メモリが作成できません。**	
0xC0AF0005 RYAA005 -1062273019 3232694277	**既に実行中または終了処理中のため、**TdasEngine **を開始できませんでした。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0AF0006 RYAA006 -1062273018 3232694278	**既に停止中または終了処理中のため、**TdasEngine **を停止できませんでした。**	
0xC0AF0007 RYAA007 -1062273017 3232694279	TdasEngine **に処理の登録ができませんでした。**	
0xC0AF0008 RYAA008 -1062273016 3232694280	TdaInfo **は小サービスの状態遷移中であるため、状態遷移できません。**	
0xC0AF0009 RYAA009 -1062273015 3232694281	**相手局に存在しない接続機器名** (xx) **が指定されました。** (xx: **接続機器名**)	
0xC0AF000A RYAA010 -1062273014 3232694282	**小サービスの状態が不正なため、処理を実行できません。**	
0xC0AF000B RYAA011 -1062273013 3232694283	**小サービスが稼動中でないため、処理を実行できません。**	

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0AF000C RYAA012 -1062273012 3232694284	**小サービスがエラー停止中のため、処理を実行できません。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0AF000D RYAA013 -1062273011 3232694285	**サポートされていない小サービスの I/F がコールされました。**	
0xC0AF0010 RYAA016 -1062273008 3232694288	**メモリ不足のため、アイテム登録できませんでした。**	他のアプリケーションを終了するか OS を再起動してください。
0xC0AF0011 RYAA017 -1062273007 3232694289	**アイテムが登録されていないデバイスにアクセスしました。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0AF0012 RYAA018 -1062273006 3232694290	**範囲外のアイテムにアクセスしました。**	範囲外のデバイスにアクセスしました。
0xC0AF0013 RYAA019 -1062273005 3232694291	**指定されたクラスタ内に不正なアイテムが指定されているため、クラスタ登録に失敗しました。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0AF0014 RYAA020 -1062273004 3232694292	**指定されたデータ型は有効なデータ型ではありません。**	
0xC0AF0015 RYAA021 -1062273003 3232694293	**不正なアクセスタイプが指定されました。**	
0xC0AF0016 RYAA022 -1062273002 3232694294	**不正なデータ種別が指定されました。**	
0xC0AF0017 RYAA023 -1062273001 3232694295	**演算書込みで指定されたデータ点数が多すぎます** (xx **以下にしてください**)。 (xx: **データ点数**)	

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0AF0018 RYAA024 -1062273000 3232694296	**演算書込みで下限値を下回りました。**	範囲外の値を書込みしようとしました。範囲内の値を書き込むように設定を変更してください。
0xC0AF0019 RYAA025 -1062272999 3232694297	**演算書込みで上限値を越えました。**	
0xC0AF001A RYAA026 -1062272998 3232694298	**メモリ不足のため、ネットワーク先に処理要求できませんでした。**	他のアプリケーションを終了するか OS を再起動してください。
0xC0AF001B RYAA027 -1062272997 3232694299	**指定のグループが見つかりませんでした。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0AF001C RYAA028 -1062272996 3232694300	**比較した 2 つのアクセスチケットの局、機器、デバイスのいずれかが異なっています。**	
0xC0AF001D RYAA029 -1062272995 3232694301	**指定されたアクセスチケットが自局ではありません。**	
0xC0AF001E RYAA030 -1062272994 3232694302	**メモリ不足のため、キャッシュ登録できませんでした。**	他のアプリケーションを終了するか OS を再起動してください。
0xC0AF0020 RYAA032 -1062272992 3232694304	**ブロック型ではないアクセスチケットでブロックアクセスされました。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0AF0021 RYAA033 -1062272991 3232694305	**処理対象となる小サービスが見つかりませんでした。**	
0xC0AF0022 RYAA034 -1062272990 3232694306	**デバイスへの一括アクセスサイズの上限を超えました。**	デバイスへの一括読み込み／書き込みのバッファサイズは最大 10K バイトです。それ以下になるように設定してください。
0xC0AF0023 RYAA035 -1062272989 3232694307	**異なるネットワーク プロジェクトが使用されています。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0AF0030 RYAA048 -1062272976 3232694320	**相手局との通信中にエラーが発生しました。相手局と正しくネットワーク接続されていることを確認してください。**	LAN カードの設定に問題がないかを確認してください。
0xC0AF0031 RYAA049 -1062272975 3232694321	**相手局から指定時間内に応答がありませんでした。相手局と正しくネットワーク接続されていることを確認してください。**	
0xC0AF0032 RYAA050 -1062272974 3232694322	**相手局から指定時間内に応答がありませんでした。相手局と正しくネットワーク接続されていることを確認してください。**	
0xC0AF0033 RYAA051 -1062272973 3232694323	**自局または相手局が終了したため、相手局との通信が停止しました。**	WinGP 局をオンラインにしてください。
0xC0AF0040 RYAA064 -1062272960 3232694336	**デバイス読出しに失敗しました。**	不正または未設定のデバイスアドレスに読み込みを行った可能性があります。正しいデバイスアドレスを指定してください。
0xC0AF0041 RYAA065 -1062272959 3232694337	**デバイス書込みに失敗しました。**	
0xC0AF0045 RYAA069 -1062272955 3232694341	**指定された要求は未サポートです。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0AF0046 RYAA070 -1062272954 3232694342	**指定された要求は未サポートです。**	
0xC0AF0050 RYAA080 -1062272944 3232694352	**ネットワーク プロジェクト ファイルのプロジェクト** ID **が異なります (異なる** NPJ **が使用されています)。**	
0xC0AF0051 RYAA081 -1062272943 3232694353	**ネットワーク プロジェクト ファイルに必要なデータが存在しませんでした。**	
0xC0AF0052 RYAA082 -1062272942 3232694354	**ネットワーク プロジェクト ファイルが壊れています。**	

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0AF0053 RYAA083 -1062272941 3232694355	ネットワーク プロジェクト ファイルが存在しませんでした。	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0AF0067 RYAA103 -1062272921 3232694375	GP オンラインが終了したので、処理を中断しました。	WinGP オンラインが終了したので、実行中の処理を中断しました。処理を完了させるには、WinGP をオンラインにして、処理を再度実行してください。

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

- SAAA で始まるエラー

エラーコード	エラーメッセージ	原因と対処方法
0xC0B00001 SAAA001 -1062207487 3232759809	システムエラー	パソコンを再起動してください。それでも発生する場合は再インストールを行ってください。
0xC0B00002 SAAA002 -1062207486 3232759810	OS のリソースかメモリが不足しているため処理できません。	
0xC0B00003 SAAA003 -1062207485 3232759811	Pro-Server EX からの処理結果が返信されるまで、新たな処理はできません。	
0xC0B00004 SAAA004 -1062207484 3232759812	Pro-Server EX が終了したので、処理を中断しました。	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0B00005 SAAA005 -1062207483 3232759813	Pro-Server EX が処理の途中で終了したため、処理を中断しました	
0xC0B00006 SAAA006 -1062207482 3232759814	Pro-Server EX が先に終了しているため、処理できません。	

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0B00007 SAAA007 -1062207481 3232759815	指定されたコネクターは既に登録されています。アプリケーションは既に実行中です。	パソコンを再起動してください。それでも発生する場合は再インストールを行ってください。
0xC0B00008 SAAA008 -1062207480 3232759816	OLE の関数でエラーが発生しました。データの変換ができません。	
0xC0B0000A SAAA010 -1062207478 3232759818	Pro-Server EX がまだ起動されていないため、リソースを参照できません。	
0xC0B0000B SAAA011 -1062207477 3232759819	Pro-Server EX がまだ起動されていないため、システムに対して処理の依頼を要求できません。	
0xC0B0000C SAAA012 -1062207476 3232759820	システムが壊れています。処理できません。	
0xC0B00011 SAAA017 -1062207471 3232759825	xx ファイルをアクセス中にエラーが発生しました。ファイルがロック (共有) されているか、壊れています。 (xx: ファイル名)	パソコンを再起動してください。それでも発生する場合は再インストールを行ってください。
0xC0B00012 SAAA018 -1062207470 3232759826	コネクター数が多すぎ登録できません。	
0xC0B00029 SAAA041 -1062207447 3232759849	PRX ファイルから接続機器情報の取得に失敗しました。	画面プロジェクトファイルが破損している可能性があります。 GP-Pro EX で強制転送後に WinGP と WinGP SDK を再起動してください。
0xC0B0002A SAAA042 -1062207446 3232759850	PRX ファイルからシンボル情報の取得に失敗しました。	
0xC0B0002B SAAA043 -1062207445 3232759851	PRX ファイルからデバイスアドレスの取得に失敗しました。	
0xC0B0002C SAAA044 -1062207444 3232759852	PRX ファイルから設定情報の取得に失敗しました。	

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0B0002D SAAA045 -1062207443 3232759853	**一時ファイルの作成に失敗しました。**	テンポラリ ファイル作成フォルダの空き容量が少ない可能性があります。ドライブの空き容量を確認し、少ない場合は空き容量を増やしてください。
0xC0B0002E SAAA046 -1062207442 3232759854	PRX **ファイルを開けません。**	画面プロジェクトファイルが破損している可能性があります。 GP-Pro EX で強制転送後に WinGP と WinGP SDK を再起動してください。
0xC0B0002F SAAA047 -1062207441 3232759855	**一時ファイルの削除に失敗しました。**	再度実行してください。
0xC0B00030 SAAA048 -1062207440 3232759856	**指定された** PRX **ファイルにエラーがあります。**	画面プロジェクトファイルが破損している可能性があります。 GP-Pro EX で強制転送後に WinGP と WinGP SDK を再起動してください。
0xC0B00031 SAAA049 -1062207439 3232759857	PRX **ファイル内に必要なデータがありません。**	
0xC0B00032 SAAA050 -1062207438 3232759858	**指定されたファイルは** PRX **ファイルではありません。**	
0xC0B00062 SAAA098 -1062207390 3232759906	**ネットワークプロジェクトファイルが壊れていて、読込みができません。指定したファイルが正しいネットワークプロジェクトファイルであるかを確認してください。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0B00063 SAAA099 -1062207389 3232759907	**ネットワークプロジェクトファイルに書込めません。**	ディスクの空きエリアが十分かどうか、ハードディスクに問題がないかなどを確認してください。
0xC0B00064 SAAA100 -1062207388 3232759908	**ネットワークプロジェクトファイルでないか古いバージョンのネットワークプロジェクトファイルです。データを読込めません。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0B00065 SAAA101 -1062207387 3232759909	(xx) **において、指定された接続機器はありません。削除されたか名前を変更された可能性があります。設定内容を見直してください。** (xx: **参加局名**)	
0xC0B00066 SAAA102 -1062207386 3232759910	**指定された参加局** (xx) **は登録されていません。矛盾があります。設定内容を見直してください。** (xx: **参加局名**)	

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0B00067 SAAA103 -1062207385 3232759911	**指定された参加局情報が不正です。参加局の情報がありません。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0B00068 SAAA104 -1062207384 3232759912	**参加局** (XX) **のシステム先頭デバイスの設定にエラーがあります。デバイスの指定を確認してください（**XX: **参加局名）**	
0xC0B00069 SAAA105 -1062207383 3232759913	(xx:xx) **はデバイスもしくはシンボルとして不正です。解析できません。** (xx: **デバイス / シンボル名**)	存在しないデバイス / シンボル名が指定されています。正しいデバイス / シンボル名に変更してください。それでも解決しない場合は、画面プロジェクトファイルが破損している可能性があります。GP-Pro EX で強制転送後に WinGP と WinGP SDK を再起動してください。
0xC0B0006C SAAA108 -1062207380 3232759916	**ネットワーク設定が壊れています。**	ネットワーク設定を見直してください。
0xC0B00078 SAAA120 -1062207368 3232759928	(**シンボルシート：**xx **シンボル：**xx **アドレス：**xx) **はデバイスアドレスとして不正です。**(xx: **シンボルシート名、**xx: **シンボル名、**xx: **アドレス**)	画面プロジェクトファイルが破損している可能性があります。 GP-Pro EX で強制転送後に WinGP と WinGP SDK を再起動してください。
0xC0B0007C SAAA124 -1062207364 3232759932	(**シンボルシート：**%s **シンボル：**%s **アドレス：**%s) **は有効なデバイス範囲を越えています。**(xx: **シンボルシート名、**xx: **シンボル名、**xx: **アドレス**)	画面プロジェクトファイルが破損している可能性があります。 GP-Pro EX で強制転送後に WinGP と WinGP SDK を再起動してください。
0xC0B00082 SAAA130 -1062207358 3232759938	**指定された** (xx) **局はネットワークプロジェクトに登録されていません。（**xx**：）参加局名**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0B00083 SAAA131 -1062207357 3232759939	**指定された** (xx) **局は** GP **シリーズ局ではありません。（**xx**：）参加局名**	
0xC0B00084 SAAA132 -1062207356 3232759940	**指定された** (xx) **局の接続機器はサポートしていません。（**xx**：）参加局名**	

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0B00095 SAAA149 -1062207339 3232759957	(シンボルシート：xx シンボル：xx デバイス個数：xx) はデバイス個数の範囲を超えています。(有効範囲：xx 〜 xx)	GP-Pro EX のシンボル画面で、登録しているシンボル数を減らしてください。
0xC0B00096 SAAA150 -1062207338 3232759958	(シンボルシート：xx グループ：xx) は最大行数を超えています。行数を減らしてください。(xx 行以下)	
0xC0B0009C SAAA156 -1062207332 3232759964	1 シンボルシート内でシンボル名もしくはグループ名が重複しています。(シンボルシート：xx　名前１：xx 名前 2：xx)	GP-Pro EX のシンボル画面で、登録しているシンボル名を変更してください。
0xC0B0009D SAAA157 -1062207331 3232759965	(参加局：xx) の機器ドライバはサポートしていません (必要な機器ドライバがインストールされていません)。 (xx: 参加局名)	接続機器の差分インストールをおこなってください。
0xC0B000A9 SAAA169 -1062207319 3232759977	(xx：xx) 指定されたデバイスもしくはシンボルは有効なデバイス範囲を越えています。 (xx: デバイス、xx：番号)	範囲外のデバイスにアクセスしました。
0xC0B000E0 SAAA224 -1062207264 3232760032	警告：異なるシンボルシート内にシンボル名もしくはグループ名が重複しています。重複している名前を利用するには、シート名を指定してください。 (xx：既に存在するシンボルシート名、xx：同一のシンボル名が属しているシンボルシート名、xx：同一のシンボル名)	名前が重複しないように、GP-Pro EX のシンボル設定画面で名前を変更してください。
0xC0B000E1 SAAA225 -1062207263 3232760033	警告：シンボルシート名とシンボルもしくはグループ名が重複しています。重複している名前を利用するには、シート名を指定してください。 (xx：同一のシンボル名が属しているシンボルシート名、xx：同一のシンボル名)	
0xC0B000E4 SAAA228 -1062207260 3232760036	警告：配列変数 (xx) は要素数が多すぎて、WinGP 用 API 通信では配列全体にアクセスすることはできません。WinGP 用 API 通信でアクセスできる配列要素は先頭から XX 個までです。 シンボル名、xx：配列要素数	①　GP-Pro EX で配列を複数個に分割して登録することを検討してください。 ②　分割できない場合は Pro-Server EX では GP-Pro EX のプロジェクトファイルをネットワークプロジェクトにインポートする時、一度にアクセスできる個数を超えるような配列変数は自動的に分割して複数のシンボルとして登録する機能があります。 WinGP SDK ではなく、Pro-Server EX の利用を検討してください。

*　1 行目：エラーコード
　　2 行目：統一エラーコード
　　3 行目：10 進符号付きエラーコード
　　4 行目：10 進符号なしエラーコード

- SAAF で始まるエラー

エラーコード	エラーメッセージ	原因と対処方法
0xC0B00201 SAAF001 -1062206975 3232760321	TCP/IP が初期化できません。	Windows の［コントロールパネル］-［ネットワーク接続］から接続設定がされているか確認し、かつ、その接続設定のプロパティで TCP/IP プロトコルがインストールされているか確認してください。WinGP SDK は TCP/IP プロトコルがインストールされていないと動作しません。
0xC0B00203 SAAF003 -1062206973 3232760323	このパソコンには有効な IP アドレスが割り付けられていません。パソコンの TCP/IP の環境を確認してください。	LAN カードが正しく動作しているか確認してください。 LAN ケーブルも確認してください。
0xC0B00204 SAAF004 -1062206972 3232760324	PLCInfo.xml ファイルをロードできません。	プロトコルドライバをアップデートしてください。それでも解決しない場合は、WinGP SDK を再インストールしてください。
0xC0B00205 SAAF005 -1062206971 3232760325	Editor Driver がロードできません。	
0xC0B00206 SAAF006 -1062206970 3232760326	Active X I/F でエラーが発生しました。	OS が対象バージョンか確認してください。 パソコンを再起動しても現象が発生する場合は WinGP SDK を再インストールしてください。
0xC0B00207 SAAF007 -1062206969 3232760327	Pro-Server EX 用の DLL、EXE 間でバージョン不一致のため実行できません。xx プログラムを強制終了します。 (xx: プログラム名)	1 台のパソコン内に複数の異なるバージョンの Pro-Server EX または WinGP SDK の DLL がインストールされていないか確認してください。1 台のパソコンには 1 バージョンの Pro-Server EX または WinGP SDK のみインストールできます。
0xC0B00209 SAAF009 -1062206967 3232760329	ファイル Core.ID が見つかりません。	パソコンを再起動してください。それでも解決しない場合は、WinGP SDK を再インストールしてください。
0xC0B0020B SAAF011 -1062206965 3232760331	ProNet.dll が正しくインストールされていません。	
0xC0B0020C SAAF012 -1062206964 3232760332	Pro-Server EX を起動できません。WinGP 局 EX や Pro-Server EX を利用する全てのアプリケーションを一旦終了してから再度試みてください。	WinGP SDK を利用するアプリケーションを正常終了していないため、WinGP SDK が起動できない可能性があります。WinGP SDK および、それを利用しているアプリケーションを全て一旦終了し、再度試みてください。
0xC0B00211 SAAF017 -1062206959 3232760337	新しい API ではサポートしていません。	ご使用の API は利用できません。別の方法を検討してください。

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0B00212 SAAF018 -1062206958 3232760338	**指定された文字列はデバイスアドレスとして不正です。**	アドレスの指定方法を再確認してください。 接続機器や参加局が変更になっていないかを確認してください。 対応する接続機器のドライバがインストールされているかを確認してください。
0xC0B00213 SAAF019 -1062206957 3232760339	**指定されたデバイスはビットアクセス以外はサポートしていません。**	アクセス対象のデバイスおよびアクセス方法を確認してください。
0xC0B00214 SAAF020 -1062206956 3232760340	**指定された機器ドライバはサポートしていません（必要な機器ドライバがインストールされてません）。**	接続機器の差分インストールをおこなってください。
0xC0B00215 SAAF021 -1062206955 3232760341	**パラメータの値が不正です。**	API の引数を確認してください。
0xC0B00216 SAAF022 -1062206954 3232760342	**デバイス番号が範囲外です。**	デバイス番号を確認してください。
0xC0B00217 SAAF023 -1062206953 3232760343	**指定されたデバイスは存在しません。**	接続機器の設定、システム先頭エリアの設定が正しいかを確認してください。
0xC0B00218 SAAF024 -1062206952 3232760344	**指定されたグループシンボルは存在しません。**	グループシンボルの指定が正しいかを確認してください。
0xC0B0021A SAAF026 -1062206950 3232760346	**キューイングアクセスに、リードアクセスとライトアクセス、もしくは、キャッシュアクセスとダイレクトアクセスを混在させることはできません。**	キューイングの開始から実際のアクセス実行までの間に、異なるアクセス方法がないかを確認してください。 異なるアクセス方法が必要な場合は、キューイングアクセスを分けてください。
0xC0B0021D SAAF029 -1062206947 3232760349	**指定された局はネットワークプロジェクトに登録されていません。**	参加局の指定を確認してください。
0xC0B0021F SAAF031 -1062206945 3232760351	**API が 2 重に呼び出されました。指定された Pro-Server EX 用アクセスハンドルは既に実行中です。**	同時に API をコールしないように EasySetWaitType() の使用を検討してください。
0xC0B00220 SAAF032 -1062206944 3232760352	**データの型変換で変換元と変換先のデータの型が変換可能な型ではありません。**	Variant 型の内容を確認してください。

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0B00221 SAAF033 -1062206943 3232760353	未サポートのバックアップデータの種類が指定されました。	データの種類の指定を確認してください。
0xC0B00222 SAAF034 -1062206942 3232760354	SRAM バックアップデータファイルの作成もしくはオープンに失敗しました。	パソコンの保存先ファイルやフォルダの指定、空き容量やファイルへのアクセス権などを確認してください。
0xC0B00223 SAAF035 -1062206941 3232760355	バックアップデータの書込みもしくは読込みでファイルのアクセスに失敗しました。	SRAM バックアップデータの書き込み、または読み込みで指定するファイルへのアクセスでエラーが発生しました。パソコンの空き容量やファイルへのアクセス権など確認して再度実行してください。
0xC0B00224 SAAF036 -1062206940 3232760356	SRAM バックアップデータの書込みで、指定されたファイルサイズが大きすぎます。96Kbyte 以下にしてください。	SRAM バックアップデータの書き込みで指定するファイルが正しいかを確認してください。また、ファイルサイズが 96K バイト以下のものを指定してください。
0xC0B00225 SAAF037 -1062206939 3232760357	数値の指定が不正です。数値を正しく指定してください。	数値として有効な文字列かを確認してください。
0xC0B00226 SAAF038 -1062206938 3232760358	データ点数の指定が 0 か範囲を超えています。	データ点数を確認してください。
0xC0B00227 SAAF039 -1062206937 3232760359	最大アクセス先数が多すぎます (1500 箇所以内にしてください)。	分割してアクセスすることを検討してください。
0xC0B00228 SAAF040 -1062206936 3232760360	アクセスするデータの合計バッファサイズが大きすぎます（1M バイト以内にしてください)。	
0xC0B00230 SAAF048 -1062206928 3232760368	Pro-Server EX が起動できません。	パソコンを再起動してください。それでも解決しない場合は、WinGP SDK を再インストールしてください。
0xC0B00238 SAAF056 -1062206920 3232760376	GP3000 シリーズ /WinGP に対してロギングデータの読出しはおこなえません。	WinGP 局に対してロギングデータの読み出しを実行しないように設定を変更してください。
0xC0B00239 SAAF057 -1062206919 3232760377	GP3000 シリーズ /WinGP に対してトレンドデータの読出しはおこなえません。	WinGP 局に対してトレンドデータの読み出しを実行しないように設定を変更してください。

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0B00240 SAAF064 -1062206912 3232760384	**指定された** Pro-Server EX **用アクセスハンドルは有効なものではありません。**	パソコンを再起動してください。それでも発生する場合は再インストールを行ってください。
0xC0B00241 SAAF065 -1062206911 3232760385	**未サポートのコマンドのため、処理を継続できません。**	パソコンを再起動してください。それでも解決しない場合は、WinGP SDK を再インストールしてください。
0xC0B00242 SAAF066 -1062206910 3232760386	Pro-Server EX **は停止したため、処理できません。**	WinGP SDK の終了は全てのアプリケーション終了後に行ってください。
0xC0B00243 SAAF067 -1062206909 3232760387	**サーバーからの処理結果待ちで、先にアプリケーションが終了しようとしました。**	もし、WM_QUIT を受信したくないのであれば、マルチハンドル系の API を EasySetWaitTypeM(2) でご利用ください。
0xC0B00244 SAAF068 -1062206908 3232760388	**ファイル名が** 256 **文字を超えています。**256 **文字以内にしてください。**	ファイル名の指定を確認してください。
0xC0B00245 SAAF069 -1062206907 3232760389	**キューイングアクセスの登録が開始されていません。**	プログラムのシーケンスを確認してください。
0xC0B00246 SAAF070 -1062206906 3232760390	**キューイングアクセスの実際のアクセスが実施されていません。**	
0xC0B00247 SAAF071 -1062206905 3232760391	**指定された番号へのデバイスアクセスは失敗しています。**	ケーブルや接続機器の動作環境を確認してください。
0xC0B00248 SAAF072 -1062206904 3232760392	**指定された番号のデバイスアクセスは登録されていません。事前登録したアクセス件数と番号を確認してください。**	プログラムのシーケンスを確認してください。
0xC0B0024C SAAF076 -1062206900 3232760396	**指定されたグループ番号はサンプリングデータのグループ番号の範囲を超えています。**	API の引数を見直してください。
0xC0B0024D SAAF077 -1062206899 3232760397	**キューイングアクセスに読込みと書込みを混在させることはできません。**	プログラムのシーケンスを確認してください。

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0B00250 SAAF080 -1062206896 3232760400	**語句がありません。**	API の引数を見直してください。
0xC0B00251 SAAF081 -1062206895 3232760401	**不正な名前か語句です。名前か語句の中に使用できない文字があります。**	
0xC0B00252 SAAF082 -1062206894 3232760402	**指定された局はネットワークプロジェクトに登録されていません。**	API の引数を確認してください。
0xC0B00253 SAAF083 -1062206893 3232760403	**指定された接続機器は登録されていません。**	
0xC0B00254 SAAF084 -1062206892 3232760404	**配列のインデックスの指定エラーです。**	配列の指定方法を確認してください。
0xC0B00255 SAAF085 -1062206891 3232760405	**指定されたデバイスは未定義なシンボルか不正なアドレスです。**	デバイスアドレスの指定方法を確認してください。
0xC0B00256 SAAF086 -1062206890 3232760406	**シンボル名が不正かグループの入れ子指定が深すぎます。**	
0xC0B00257 SAAF087 -1062206889 3232760407	**文字列型のシンボルに対してのインデックス指定はできません。**	
0xC0B00258 SAAF088 -1062206888 3232760408	**配列のインデックスが大きすぎます。**	
0xC0B00259 SAAF089 -1062206887 3232760409	**デバイス指定としてグループシンボルを利用できないものに対し、グループシンボルが指定されました。**	
0xC0B0025A SAAF090 -1062206886 3232760410	**デバイス指定にグループシンボルを指定してください。**	

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0B0025B SAAF091 -1062206885 3232760411	**シンボルシート名が不正か、指定された機器では利用できません。**	デバイスアドレスの指定方法を確認してください。
0xC0B0025C SAAF092 -1062206884 3232760412	**接続機器名が 2 重に指定されています。**	致命的エラーが発生しました。 GP-Pro EX で強制転送後に、WinGP と WinGP SDK を再起動してください。
0xC0B0025D SAAF093 -1062206883 3232760413	**指定されたシンボルのデータ型は指定のデータ型と異なるため利用できません。**	シンボルのデータ型と指定されたデータ型と異なるため利用できないません。シンボル名、もしくはデータ型を確認してください。
0xC0B0025E SAAF094 -1062206882 3232760414	**オプション指定文字列の解析に失敗しました。**	API の引数を確認してください。
0xC0B00262 SAAF098 -1062206878 3232760418	**ファイルの読出しに失敗しました。**	CF カードフォルダ内に指定されたファイルが存在するか確認してください。存在する場合はファイルのアクセス権を確認してください。
0xC0B00263 SAAF099 -1062206877 3232760419	**ファイルの書込みに失敗しました。**	書き込み先のアクセス権を確認してください。アクセス権に問題がない場合は、CF カードの空き容量が少なくなっている可能性があるため、CF カードの空き容量を確認してください。
0xC0B00264 SAAF100 -1062206876 3232760420	**指定されたファイルが見つかりません。**	指定したファイルが存在するか確認してください。
0xC0B00265 SAAF101 -1062206875 3232760421	**ファイルの削除に失敗しました。**	CF カードフォルダ内に指定されたファイルが存在するか確認してください。存在する場合はファイルのアクセス権を確認してください。
0xC0B00266 SAAF102 -1062206874 3232760422	**ファイル名の変更に失敗しました。**	CF カードフォルダ内に指定されたファイルが存在するか確認してください。存在する場合はファイルのアクセス権、変更後のファイル名に使用禁止文字を使用していないかを確認してください。
0xC0B00267 SAAF103 -1062206873 3232760423	**ファイルリスト保存ファイルを開くことができません。**	保存先フォルダのアクセス権を確認してください。アクセス権に問題がない場合はドライブの容量が少なくなっている可能性があるため、ドライブの空き容量を確認してください。
0xC0B00269 SAAF105 -1062206871 3232760425	**ファイル名が入力されていません。**	ファイル名を入力してください。

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード	エラーメッセージ	原因と対処方法
0xC0B0026A SAAF106 -1062206870 3232760426	ファイルパスが長すぎます。	ファイルパスを短くしてください。
0xC0B0026C SAAF108 -1062206868 3232760428	GP3000 シリーズ局への接続がリセットされました。	WinGP 局の電源が切れていないか、ケーブルが抜けていないかを確認し、再度実行してください。
0xC0B0026D SAAF109 -1062206867 3232760429	相手局からの応答がありません。	
0xC0B0026E SAAF110 -1062206866 3232760430	処理中に接続が中断されたため、処理が完了しませんでした。	
0xC0B0026F SAAF111 -1062206865 3232760431	指定された局名が存在しないため接続できません。	#WinGP の局名を使用してください。
0xC0B00272 SAAF114 -1062206862 3232760434	パラメータの値が不正です。	引数を見直し、正しい値を設定してください。
0xC0B00273 SAAF115 -1062206861 3232760435	CF カード内のファイル一覧取得に失敗しました。	指定したファイルタイプが正しいかを確認してください。また、保存先フォルダのアクセス権を確認してください。アクセス権に問題がない場合はドライブの容量が少なくなっている可能性があるため、ドライブの空き容量を確認してください。
0xC0B00274 SAAF116 -1062206860 3232760448	GP3000 シリーズ局 /WinGP 局に接続できませんでした。	WinGP 局がビジー状態の可能性があります。少し時間がたってから再度実行してください。 または、転送ツールで WinGP 局と接続している場合は、転送ツールを終了してから、再度実行してください。
0xC0B002A6 SAAF166 -1062206810 3232760486	SRAM バックアップデータの読出しは既に実行されているため、現在処理できません。	もう一度 SRAM バックアップデータの読み出しを実行してください。
0xC0B002A7 SAAF167 -1062206809 3232760487	SRAM バックアップデータの読出しで、引数に誤りがあります。	正しい引数を使用して、SRAM バックアップデータの読み出しを実行してください。
0xC0B002A8 SAAF168 -1062206808 3232760488	保存ファイルの書込みに失敗しました。	パソコンのハードディスク空き容量が少ない場合には、空き容量を増やした後に再度実行してください。またはパソコンを再起動して再度実行してください。

* 1 行目：エラーコード
2 行目：統一エラーコード
3 行目：10 進符号付きエラーコード
4 行目：10 進符号なしエラーコード

エラーコード		メッセージ
10 進	16 進	
9300	2454h	ネットワークプロジェクトファイルがありません。
9301 ： 9329	2455h ： 2471h	予約
9330	2472h	リソース不足で実行できません。プログラムを強制終了します。
9331	2473h	システムリソースがデットロックしました。プログラムを強制終了します。
9332	2474h	システムエラー
9333	2475h	プログラム間のバージョン不一致のため実行できません。プログラムを強制終了します。
9334 ： 9339	2476h ： 247Bh	予約
9340	247Ch	<%s> ファイルをアクセス中にエラーが発生しました。
9341	247Dh	Pro-Server を使用するアプリケーション数が多すぎます。
9342	247Eh	OS のリソース不足（メモリー不足）です。
9343	247Fh	設定されたコネクターは別のアプリケーションで使用されています。
9344	2480h	Pro-Server が未だ起動されていません。データを参照できませんでした。
9345	2481h	Pro-Server が終了されています。データを参照できませんでした。
9346	2482h	Pro-Server が終了されています。処理を継続できません。
9347	2483h	Pro-Server が未だ起動されていません。処理を継続できません。
9348	2484h	Pro-Server を起動できませんでした。
9349	2485h	Pro-Studio を起動できませんでした。
9350	2486h	未サポートのコマンドです。処理を継続できません。
9351	2487h	ネットワークプロジェクトファイルのロードに失敗しました。
9352	2488h	設定された局名は既に登録されています。
9353	2489h	設定された局名は登録されていません。
9354	248Ah	未サポートのバックアップデータの種類が指定されました。
9355	248Bh	ファイルの書込みに失敗しました。
9356	248Ch	SRAM バックアップデータ保存用ファイルを作成できませんでした。
9357	248Dh	設定された局名は登録されていません。
9358	248Eh	Pro-Server は既に起動中です。二重起動できません。
9359	248Fh	予約
9360	2490h	'%s' の設定がありません。
9361	2491h	'%s' に 0 は設定できません。
9362	2492h	'%s' は次のように設定してください。“xxx.xxx.xxx.xxx xxx” には 0 〜 255 の値を設定してください。
9363	2493h	'%s' は不正な値です。
9364	2494h	'%s' として使用できない文字が含まれています。
9365	2495h	'%s' が設定されていません。
9366	2496h	サーバーからの処理結果が返信されるまで、新たな処理はできません。
9367	2497h	サーバーからの処理結果待ち状態で、アプリケーションが終了しようとしました。

次のページに続きます。

エラーコード		メッセージ
10 進	16 進	
9368	2498h	読み出し可能なモードではないため実行できません。再ログオンしてください。
9369	2499h	書き込み可能なモードではないため実行できません。再ログオンしてください。
9370	249Ah	設計者モードではないため実行できません。再ログオンしてください。
9371	249Bh	指定された番号は、登録されていません。
9372 ： 9375	249Ch ： 249Fh	予約
9376	24A0h	ファイル (Core.ID) が壊れています。
9377 ： 9389	24A1h ： 24ADh	予約
9390	24AEh	指定されたハンドルのモードが EASY_TB_STATUS_NOW あるいは EASY_TB_STATUS_LAST_READ です。\n モードを WASY_TB_STATUS_PAST あるいは EASY_TB_STATUS_INDEX に設定してから実行してください。
9391	24AFh	指定された LS エリアを開けませんでした。
9392	24B0h	指定された LS エリアは開かれていません。
9393	24B1h	CF カード内のファイル一覧取得に失敗しました。
9394	24B2h	CF カード内のファイル読み出しに失敗しました。
9395	24B3h	CF カード内のファイル書き込みに失敗しました。
9396	24B4h	CF カードが挿入されていません。
9397	24B5h	CF カードが初期化されていません。
9398	24B6h	CF カードに異常があります。
9399	24B7h	指定されたファイル名にアクセスできません。
9400	24B8h	ProEasy.DLL の関数が二重に呼び出されました。PfnApiEasy.DLL の関数は既に実行中です。
9401	24B9h	指定された Pro-Server 用アクセスハンドルは有効ではありません。
9402	24BAh	Pro-Server は停止したため、処理できません。
9403	24BBh	OLE の関数でエラーが発生しました、データのコンバートが出来ません。
9404	24BCh	データの型変数で変換元のデータに有効なデータがありません／足りません。
9405	24BDh	データの型変数で変換元と変換先のデータ型が変換可能な型ではありません。
9406	24BEh	指定された引数が有効な値ではありません。
9407	24BFh	タイムバーを作成できませんでした。
9408	24C0h	シンボル名は登録されていません。
9409	24C1h	配信シートが開けません。
9410	24C2h	指定されたタイムバーはすでにロックされています。
9411	24C3h	指定されたハンドルはすでにリンクされています。
9412	24C4h	指定されたハンドルはリンクされていません。
9413	24C5h	指定されたハンドルはデータベースとリンクしていません。
9414	24C6h	指定されたハンドルは現在ロック中です。解除してから実行して下さい。
9415	24C7h	引数がまちがっています。

次のページに続きます。

エラーコード		メッセージ
10 進	16 進	
9416	24C8h	Date 型あるいは Date 型と互換性がある型にして下さい。
9417	24C9h	指定された時刻は範囲外の値です。
9418	24CAh	不正な引数が設定されています。
9419	24CBh	指定されたハンドルに対するデータベースは閉じられています。
9420	24CCh	データベースへの書き込みは許可されていません。
9421	24CDh	アクションコンテンツの INI ファイル ('%s') がオープンできません。
9422	24CEh	アクションコンテンツの INI ファイル ('%s') の '%s' が解析できません。
9423	24CFh	アクション '%s' はネットワークプロジェクトの中にインストールされていないアクションコンテンツを使用しています。
9424	24D0h	アクションが多すぎて登録できません。
9425	24D1h	指定されたアクションは既に登録されています。
9426	24D2h	アクション '%s' が使用するアクションコンテンツが起動できません。 指定されたアクションは登録されていません。
9427	24D3h	ActiveX I/F でエラーが発生しました。
9428	24D4h	指定されたアクションはレジストリに登録されました。
9429 ： 9449	24D5h ： 24E9h	予約
9450	24EAh	データが指定されていません。
9451	24EBh	指定された局名は登録されていません。
9452	24ECh	デバイス種別の指定が不正です。
9453	24EDh	局名とデバイス名を区切る ’！’ が有りません。
9454	24EEh	有効なデバイス名の指定が有りません。
9455	24EFh	有効なデバイスが指定されていないので処理できません。
9456	24F0h	32 ビットデバイスへのワードアクセスは出来ません。
9457	24F1h	デバイスのアクセス範囲が有効範囲外です。
9458	24F2h	見かけ上のデバイス数の指定が不正です。
9459	24F3h	見かけ上のデバイス点数の指定が 0 か範囲を超えています。
9460	24F4h	指定されたシンボルを有効なデバイスに変更できません。
9461	24F5h	数値指定エラー、値を正しく指定してください。
9462	24F6h	ライフタイムの指定が不正です。
9463	24F7h	ビット位置の指定が不正です。
9464 ： 9469	24F8h ： 24FDh	予約
9470	24FEh	指定された局に接続できません。
9471	24FFh	指定された局は Windows 搭載コンピュータのため、処理を実行できません。
9472	2500h	画面キャプチャデータの JPEG 保存に失敗しました。
9473	2501h	画面キャプチャ機能をサポートしていません。
9474	2502h	キャプチャ許可フラグが ON になっていません。
9475	2503h	CF カード内の空き容量の取得に失敗しました。
9476	2504h	データ転送機能をサポートしていません。
9477	2505h	ProNet.dll が正しくインストールされていません。

次のページに続きます。

エラーコード		メッセージ
10 進	16 進	
9478	2506h	2WayDriver のバージョンが 4.50 未満のため実行できません。
9479	2507h	予約
9480	2508h	CF カード内のファイル削除に失敗しました。
9481	2509h	CF カード内のファイル名変更に失敗しました。
9482	250Ah	ファイルが 256 文字を超えています。256 文字以内にしてください。
9483 ： 9499	250Bh ： 251Bh	予約
9500	251Ch	Pro-Server スケジュール管理スレッド初期化エラー
9501	251Dh	Pro-Server LAN 管理スレッド初期化エラー
9502	251Eh	Pro-Server タイマー管理スレッド初期化エラー
9503	251Fh	Pro-Server DDE 制御スレッド初期化エラー
9504	2520h	Pro-Server API 制御スレッド初期化エラー
9505	2521h	Pro-Server API パラメータエラー
9506	2522h	レスポンスタイムアウト
9507	2523h	Pro-Server が LAN の初期化に失敗しました。
9508	2524h	データがありません。
9509	2525h	無効なデバイスです。
9510	2526h	無効なアドレスです。
9511	2527h	アドレスが範囲外です。
9512	2528h	データタイプエラー
9513	2529h	伝文エラー
9514	252Ah	Pro-Server のキャッシュ機能が初期化できません。
9515	252Bh	データベースを利用中のためネットワークプロジェクトをロードできません。
9516 ： 9559	252Ch ： 2557h	予約
9560	2558h	システムエラー (DLL のロードに失敗しました。)
9561	2559h	システムエラー (DLL のバージョンが古い可能性があります。)
9562	255Ah	システムエラー
9563	255Bh	指定されたプロパティ ID は定義されていません (バージョンが古い可能性があります。)
9564	255Ch	数値として不正な文字が指定されています。
9565	255Dh	文字数が多すぎます。
9566	255Eh	数値が大きすぎます。
9567	255Fh	システムエラー (COM が起動できません)
9568	2560h	システムエラー (GP-Viewer のランタイムを起動できませんでした)
9569	2561h	ファイルを開くことができませんでした。
9570	2562h	ファイルの読み込みに失敗しました。
9571	2563h	ファイルの書き込みに失敗しました。
9572	2564h	ファイル構造が不正です (タグがありません)
9573	2565h	ファイル構造が不正です (終了タグがありません)

次のページに続きます。

エラーコード		メッセージ
10 進	16 進	
9574	2566h	ファイル構造が不正です (予定外のエンドタグがあります)
9575	2567h	シグネチャーが一致しない。
9576	2568h	サポートしていないパラメータがあります。
9577	2569h	ファイルの最後に達しました。
9578	256Ah	構造がおかしい。
9579	256Bh	メモリー不足のため処理できません。
9580	256Ch	デバイス名が解析できません。
9581	256Dh	DB 名が指定されていません。
9582	256Eh	DB にアクセスできません。
9583	256Fh	DB は他のプログラム (データビュー等) が既にロック (編集) しているため編集できません。
9584	2570h	局名かデバイス名が設定されていません。
9585	2571h	DB がクローズされていいて使用できません (NPJ をセーブ・ロードすると自動的に使用中の DB は一旦クローズされます。)
9586	2572h	データベースが壊れている可能性があります。
9587	2573h	データは蓄積されていません。
9588	2574h	指定された時間のデータを見つけることができなかった。
9589	2575h	ポーリングするための設定がされていない。
9590	2576h	データベースはオープンされていません (または既に閉じられました)
9591	2577h	既にポーリングは開始されています。
9592	2578h	最新の蓄積日時より古い日時のレコードを書き込もうとしています。
9593	2579h	指定されたレコードは削除されています。
9594	257Ah	指定されたファイルサイズを超えています。
9595	257Bh	指定されたファイル番号は存在しません。
9596 ： 9599	257Ch ： 257Fh	予約
9600	2580h	GP 内の資源が足りなくなり処理できません。
9601 ： 9619	2581h ： 2593h	予約
9620	2594h	ネットワークプロジェクトのアイテムが二重登録です (ネットワークプロジェクトファイルが壊れています)
9621 ： 9639	2595h ： 25A7h	予約
9640	25A8h	ネットワークプロジェクトファイルに登録されていない配信データを受信しました。
9641	25A9h	配信先局でデータの書き込みに失敗しました。
9642 ： 9659	25AAh ： 25BBh	予約
9660	25BCh	データの読み出しに失敗しました。
9661	25BDh	デバイスのリードでアクセス範囲異常です。

次のページに続きます。

エラーコード		メッセージ
10 進	16 進	
9662 ： 9669	25BEh ： 25C5h	予約
9670	25C6h	デバイスのライトでアクセス範囲異常です。
9671 ： 9699	25C7h ： 25E3h	予約
9700	25E4h	存在しない配信情報に対するファーストトリガ成立コマンドを受信しました。
9701 ： 9709	25E5h ： 25EDh	予約
9710	25EEh	存在しない配信情報に対するセカンドトリガ成立コマンドを受信しました。
9711 ： 9729	25EFh ： 2601h	予約
9730	2602h	GP がビジーです。画面転送中か、他の PC との間で SRAM バックアップデータ保存を実行中です。
9731	2603h	SRAM バックアップデータ読み出し異常です。（アイテム ID が前回と違います。）
9732	2604h	SRAM バックアップデータ読み出し異常です。（データ種別が前回と違います。)
9733	2605h	SRAM バックアップデータ読み出し異常です。（ブロック番号が前回と違います。)
9734	2606h	SRAM バックアップデータ読み出し異常です。（要求データ数が 0 か、前回と違います。)
9735 ： 9739	2607h ： 260Bh	予約
9740	260Ch	GP がビジーです。画面転送中か、他の PC との間で SRAM バックアップデータ保存を実行中です。
9741	260Dh	SRAM バックアップデータ書き込み異常です。（アイテム ID が前回と違います。）
9742	260Eh	SRAM バックアップデータ書き込み異常です。（データ種別が前回と違います。)
9743	260Fh	SRAM バックアップデータ書き込み異常です。（ブロック番号が前回と違います。)
9744	2610h	SRAM バックアップデータ書き込み異常です。（要求データ数が 0 か、前回と違います。)
9745 ： 9749	2611h ： 2615h	予約
9750	2616h	コマンド異常。
9751	2617h	CF カードのアクセスに失敗しました。
9752	2618h	CF カードユニットがありません。

次のページに続きます。

エラーコード		メッセージ
10 進	16 進	
9753 ： 9779	2619h ： 2633h	予約
9780	2634h	書き込みで PLC との通信エラーが発生しました。[詳細コード %02x:%04x]
9781	2635h	設定された SRAM バックアップデータが GP にはありません。
9782	2636h	GP の SRAM バックアップデータが異常です。[詳細コード %04x]
9783	2637h	新アラームブロックをサポートしていません。
9784 ： 9789	2638h ： 263Dh	予約
9790	263Eh	リモートアクセス権がありません。(リモート接続されていません)
9800	2648h	パラメータ・エラーが発生しました。
9801	2649h	配信データ数が許容範囲を超えました。
9802	264Ah	ファイル作成時にエラーが発生しました。
9803	264Bh	EXCEL シート作成時にエラーが発生しました。
9804	264Ch	ファイル書き込み中にエラーが発生しました。
9805	264Dh	ファイルオープン時にエラーが発生しました。
9806	264Eh	読み取り専用ファイルのため終了しました。
9807	264Fh	印刷時にエラーが発生しました。
9808	2650h	保存先フォルダのアクセス権がありません。
9809	2651h	予約
9810	2652h	メッセージテーブルファイルが見つかりません。
9811	2653h	メッセージテーブルファイルが開けません。
9812	2654h	メッセージテーブルファイル内に指定のシートがありません。
9813	2655h	メッセージテーブルファイルが正しくありません。
9814	2656h	該当する有効なコードがありません。
9815	2657h	POP 認証中にエラーが発生しました。詳細はログビューアを参照してください。
9816	2658h	メールが送信できませんでした。詳細はログビューアを参照してください。
9817	2659h	一部のメールが送信できませんでした。詳細はログビューアを参照してください。
9818 9819	265Ah 265Bh	予約
9820	265Ch	指定のデータベースが見つかりません。
9821	265Dh	指定のテーブルが見つかりません。または、指定のテーブルにレコードが存在しません。
9822	265Eh	指定のフィールド名が見つかりません。
9823	265Fh	指定対象データは見つかりませんでした。
9824	2660h	フィールドのデータが不正です。
9825	2661h	認証に失敗しました。
9826	2662h	データベースアクセス中にエラーが発生しました。
9827	2663h	ProServer ハンドルの取得に失敗しました。
9828	2664h	文字データがありませんでした。

次のページに続きます。

エラーコード		メッセージ
10 進	16 進	
9829 : 9839	2665h : 266Fh	予約
9840	2670h	アクションレポートシートのテンプレートが開けないか、シートが追加できません。
9841	2671h	EXCEL の起動に失敗しました。
9842	2672h	テンプレートブックが開けません。
9843	2673h	アクションシステムエラー
9844	2674h	出力ブックを保存できません。
9845	2675h	指定されたテンプレートシート (%s) はテンプレート内にありません。
9846	2676h	シートの追加に失敗しました。
9847	2677h	コマンド (%s) が解釈できないため実行できません。
9848	2678h	印刷に失敗しました。
9849	2679h	指定されたデータの種類はサポートしていません。
9850	267Ah	Pro-Sever のバージョンが古いため実行できません。
9851	267Bh	アクションレポートシートが壊れています。
9852	267Ch	指定されたグループはありません。
9853	267Dh	画像を貼り付ける事ができません。
9854	267Eh	ファイルヘッダーが壊れています、リードできません。
9855	267Fh	指定された CSV ファイル (%s) がオープンしません。
9856	2680h	書き込みエリアのサイズが小さすぎます。
9857	2681h	テンポラリーファイルが作製できません、もしくは読めません。
9858	2682h	GP/GLC 内に有効なファイルは一つもありません。
9859	2683h	指定されたデータ型はサポートしていません。
9860	2684h	ファイル名が長すぎて出力ブックが作れません。
9861	2685h	マクロの実行でエラーが発生しました、詳細はログビューアを参照してください。
9862	2686h	GP の画面キャプチャデータの保存に失敗しました。
9863	2687h	許可フラグが ON になっているか確認してください。
9864	2688h	保存ファイル名の指定が異常です。
9865	2689h	CF カード内に指定されたファイルがありませんでした。
9866	268Ah	ブラウザ・アプリケーションが所定のフォルダにありません。ブラウザに表示することができません。
9870	268Eh	バイナリファイルのダウンロード中にエラーが発生しました。
9871	268Fh	バイナリファイルの読み出しに失敗しました。
9872	2690h	バイナリファイルのオープン時にエラーが発生しました。
9873	2691h	バイナリファイルの解析に失敗しました。
9874	2692h	EXCEL ファイルへの書き込み中にエラーが発生しました。
9875	2693h	CSV ファイルへの書き込み中にエラーが発生しました。
9876	2694h	バイナリファイルの生成に失敗しました。
9877	2695h	指定されたファイルが存在しません。
9878	2696h	EXCEL ファイルからバイナリファイルの変換に失敗しました。
9879	2697h	CSV ファイルからバイナリファイルの変換に失敗しました。

次のページに続きます。

エラーコード		メッセージ
10 進	16 進	
9880	2698h	配信されたデータが範囲外です。
9881	2699h	GP ログデータのアップロードに失敗しました。
9882	269Ah	バックアップするデータがありません。
9883	269Bh	データが 1 シートに収まりません。
9884	269Ch	Microsoft Excel がマシン上に存在しません。
9885	269Dh	パラメータの内容が不正です。
9886	269Eh	データの書き込みに失敗しました。
9887	269Fh	CSV ファイルの読込みに失敗しました。
9888	26A0h	不要なファイルを削除する際にエラーが発生しました。
9889	26A1h	アクションが失敗に終わりました。
9891	26A3h	ACCESS ファイルに一致するデータがありません。
9892	26A4h	コマンド・エラー
9893	26A5h	ACCESS データへのアップロードに失敗しました。
9894	26A6h	指定されたテーブルを開くことができません。

37.8.3 ビットデータのアクセスについて

WinGP SDK は、ビットデバイスへアクセスする場合、そのビットデータを扱う方法として 3 種類の方法を提供しています。

1) 16 ビット単位での扱い：ビットデバイスに対し、16 ビット単位のビット列として扱う方法です。
ビットデータは D0 ビットから指定された個数分、右詰で格納 / 使用されます。
データバッファは指定個数が 1 個でも、16 ビット分必要となります。また、指定個数に 16 ビット単位で必要になります。

（例）20 個のビットデバイスを指定した場合のデータバッファの格納順序

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
*	*	*	*	*	*	*	*	*	*	*	*	20	19	18	17

＜適用 API ＞
ReadDeviceBit/WriteDeviceBit()
ReadDevice/WriteDevice(),ReadDeviceVariant/WriteDeviceVariant() で、データ型に 1（EASY_AppKind_Bit）を指定した場合
ReadSymbol/WriteSymbol() で、ビットシンボルやビットシンボルを含むグループを指定した場合

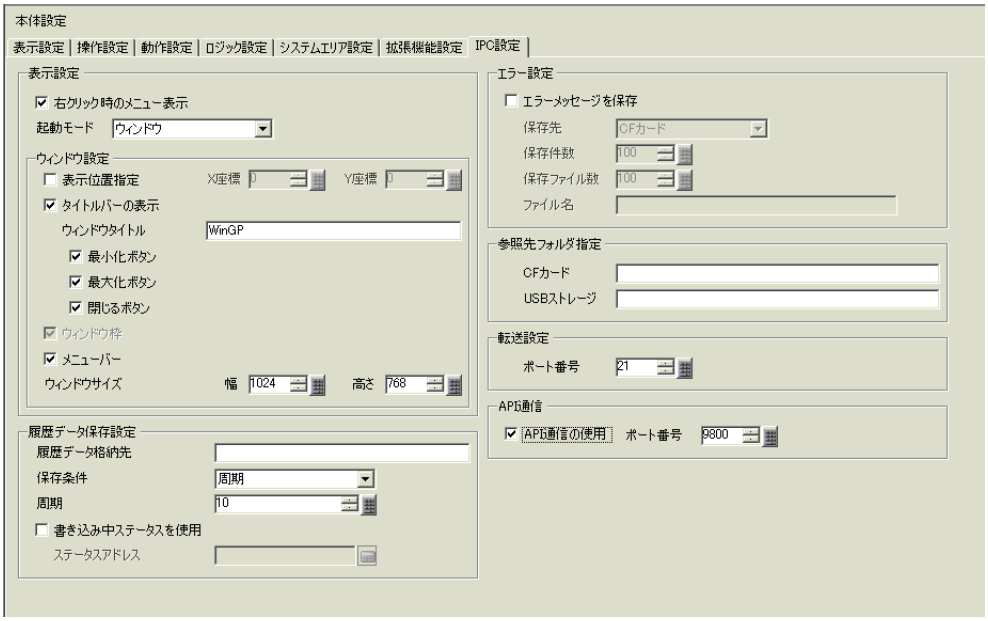
2) Variant の BOOL 単位での扱い：1 ビットを Variant の BOOL データとして扱う方法です。
データバッファは 1 ビットが 1 Variant の BOOL 型となり、指定個数分の BOOL 型の配列として扱います。

＜適用 API ＞
ReadDeviceVariant/WriteDeviceVariant() で、データ型に 0x201 (EASY_AppKind_BOOL) を指定した場合
ReadSymbolVariant/WriteSymbolVarinat() で、ビットシンボルやビットシンボルを含むグループを指定した場合

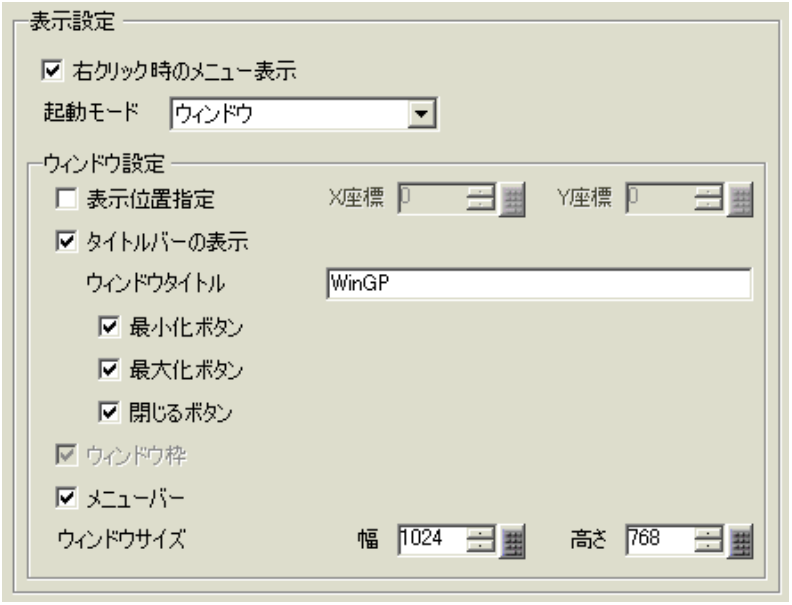
3) ロジック命令の構造体変数によるアクセス時のビットオフセットシンボルの扱い
ビットオフセットシンボルを直接指定してデバイスへアクセスした場合、そのデータバッファは、上記で説明した “ 16 ビット単位での扱い ” か “ Variant の BOOL 単位での扱い ” になります。
ただし、グループシンボル内にビットオフセットシンボルがあり、ロジック命令の構造体変数でデバイスへアクセスした場合、データバッファ内にそのビットオフセットシンボル用のデータ領域は確保されません。
ビットオフセットシンボルはそのシンボルが単体で存在することはなく、必ず、その親となるワードシンボルがあります。データ領域としてはその親の分が確保されるので、ビットオフセットシンボルの分はその親の分の一部を利用してください。

37.9 設定ガイド

37.9.1 システム設定ウィンドウ [本体設定] ー [IPC 設定] の設定ガイド



■ 表示設定

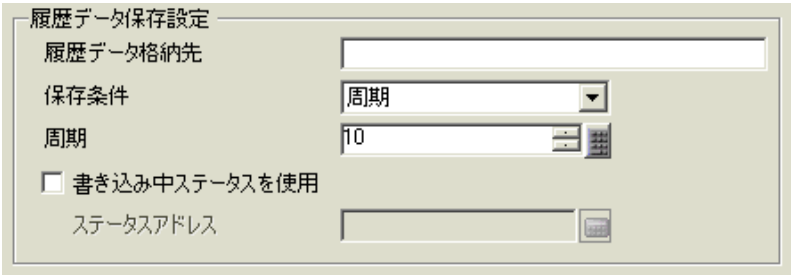


設定項目	設定内容
右クリック時のメニュー表示	WinGP 上でウィンドウを右クリックしてメニューを表示するかどうかを設定します。 「37.9.2 ウィンドウフレームの設定ガイド ■ 右クリック時のメニュー」（37-144 ページ）

次のページに続きます。

設定項目			設定内容
起動モード			[WinGP] 起動時に表示するウィンドウ画面のサイズを [ウィンドウ][フルスクリーン] から選択します。[ウィンドウ] を選択した場合は、指定された画面サイズでウィンドウが表示されます。[フルスクリーン] を選択した場合は、画面サイズに関係なく IPC のフルスクリーンで拡大され表示されます。
ウィンドウ設定			WinGP 起動時のウィンドウ表示位置を指定するかどうかを設定します。表示位置は X 座標、Y 座標から設定します。
	表示位置指定		• X 座標 0 ～選択機種の最大解像度 (横)–1 • Y 座標 0 ～選択機種の最大解像度 (横)–1
	タイトルバーの表示		ウィンドウフレームにタイトルバーを表示するかどうかを設定します。 ☞「37.9.2 ウィンドウフレームの設定ガイド」(37-143 ページ)
		ウィンドウタイトル	タイトルバーに表示するウィンドウタイトル名を半角英数字 63 文字以内で設定します。
		最小化ボタン	ウィンドウ最小化ボタンを表示するかどうかを設定します。
		最大化ボタン	ウィンドウ最大化ボタンを表示するかどうかを設定します。
		閉じる	ウィンドウを閉じるボタンを表示するかどうかを設定します。
	ウィンドウ枠		ウィンドウ枠表示を行うかどうかを設定します。 MEMO • [タイトルバーの表示] が設定されている場合は [ウィンドウ枠] は常に表示され、チェックが入っている状態です。
	メニューバー		ウィンドウフレームにメニューバーを表示するかどうかを設定します。
	ウィンドウサイズ		ウィンドウのサイズを [幅]、[高さ] で設定します。[幅]、[高さ] は 0 ～選択機種の最大解像度で設定します。 MEMO • PS-2000B をご利用の場合は 0 ～ 1024 で設定します。

■ 履歴データ保存設定



設定項目	設定内容
履歴データ格納先	バックアップ SRAM の代わりとなるバックアップデータ保存場所を半角英数 255 文字以内でフルパス (ドライブ名、フォルダ名) 指定します。何も設定されていない場合は初期値として WinGP インストール先フォルダ以下の「NAND¥PRJ001¥USER¥SCREEN」に保存されます。
保存条件	バックアップを実行する条件を [周期]、[ビット ON]、[ビット変化] から選択します。 • **周期** [周期] で設定された時間ごとに、バックアップが実行されます。 • **ビット** ON [コントロールビットアドレス] で指定されたビットが ON になったときにバックアップが実行されます。ただし、前回保存タイミングから 1 分以上経過している場合のみ保存します。 • **ビット変化** [コントロールビットアドレス] で指定されたビットが ON になったときにバックアップが実行されます。ただし、前回保存タイミングから 1 分以上経過している場合のみ保存します。
周期	[バックアップトリガ] で [周期] を選択している場合にバックアップを繰り返し実行する周期を 1 ～ 60 分で設定します。
コントロールビットアドレス	[バックアップトリガ] で [ビット ON] または [ビット変化] を選択している場合にバックアップの実行をコントロールするアドレスを設定します。
書き込み中ステータスを使用	バックアップデータの書き込み状況を示すビットアドレスを使用するかどうかを設定します。
ステータスアドレス	バックアップデータの書き込み状況をここで設定したビットアドレスの ON、OFF で表します。 • ON データの書き込み中です • OFF データの書き込みは行っていません。

ビット	名称	ビット ON 条件	ビット OFF 条件
0	書込中	ファイル書き込み開始時	ファイル書き込み終了
1	書込エラー	書き込み失敗時	書き込み開始時

■ エラー設定

設定項目	設定内容
エラーメッセージを保存	[WinGP] ウィンドウに表示される、システムエラー、アプリケーションエラーを保存するかどうかを設定します。 **MEMO** • 前回保存時から 10 分以内であれば、頻繁に書き込みアクセスするのを防ぐため、エラーログファイルを保存せず前回保存時から 10 分経過後に保存します。その場合は 10 分間で記録した内容をすべてエラーログファイルに書き込みます。 • 同じエラーが連続で発生した場合でもすべて保存されます。 • エラーログ機能動作中に、IPC の時刻を変更するとエラーログが時刻の経過どおりに保存されません。
保存先	エラーログファイルの保存先を [CF カード]、[USB ストレージ] から選択します。 **MEMO** • [CF カード] または [USB ストレージ] を選択すると指定された保存先フォルダに [LOG] フォルダを作成し、その中にエラーログファイルが作成されます。
保存件数	１つのエラーログファイルに保存するエラーメッセージの件数を 1 ～ 1000 で設定します。
保存ファイル数	エラーログファイルを保存するファイル数を 0 ～ 1024 で設定します。 **MEMO** • [保存ファイル数] が 0 に設定されている場合は [CF カード] または [USB ストレージ] の容量制限までは制限なく保存されます。 • エラーログファイルが設定した [保存件数] になるまでは、常に最新の日付のエラーログファイルに記録を追加します。 ただし、日付と時間の設定を変更した場合、現在の設定より日付や時間が進んだり遅れたりしたエラーログファイルが作成されることがあります。この場合、日付が最新でも記録が追加されず、[保存件数] にならなくても残る可能性があります。 • エラーメッセージの発生が [エラー設定] の [保存ファイル数] を超えた場合は 1 番古いファイルを削除し、新しいファイルが追加されます。

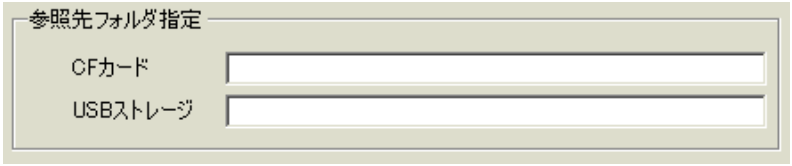
次のページに続きます。

設定項目	設定内容
ファイル名	エラーログファイルのファイル名の接頭語を半角英数 0 ～ 16 文字以内で設定します。 以下の形式でファイル名が設定されます。 [接頭語][日付時分]_[ID].[拡張子] （例） [接頭語] : Test [保存日付時分]: 2006 年 7 月 14 日 16 時 18 分 [ID] : 0（0 ～のシリアル番号） 同じ時間に複数ファイルが作成された場合、 何個目のファイルかを区別する番号です。 [拡張子] : log （固定文字） ファイル名 : Test200607141618_0.log **MEMO** • ファイル名が設定されていない場合は [保存日付時分] と [ID] のみでファイル名が設定されます。

■ 参照先フォルダ指定

[プロジェクト (F)] メニューの [プロパティ (I)] ー [出力先フォルダ (C)] で指定している [CF カード出力先フォルダ] や [USB ストレージ出力先フォルダ] 内のデータの保存先となるフォルダを指定します。

(IPC シリーズ以外の機種では、画面転送により表示器に装着している CF カードや USB ストレージに保存されますが、IPC シリーズではここで指定したフォルダが CF カードや USB ストレージの代わりとなります。)



設定項目	設定内容
CF カード	CF カードの代わりに使用するフォルダをフルパスで指定します。パスは半角、全角とも 239 文字以内で設定します。指定しなかった場合は WinGP のインストール先フォルダ以下の「CFA00」フォルダに保存されます。
USB ストレージ	USB ストレージ (USB メモリなど) の代わりに使用するフォルダをフルパスで指定します。パスは半角、全角とも 239 文字以内で設定します。指定しなかった場合は WinGP のインストール先フォルダ以下の「USBHD」フォルダに保存されます。

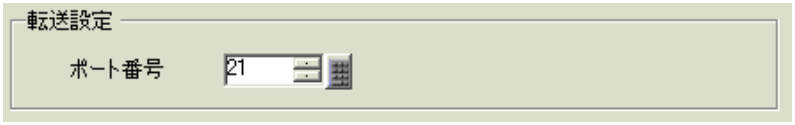
MEMO

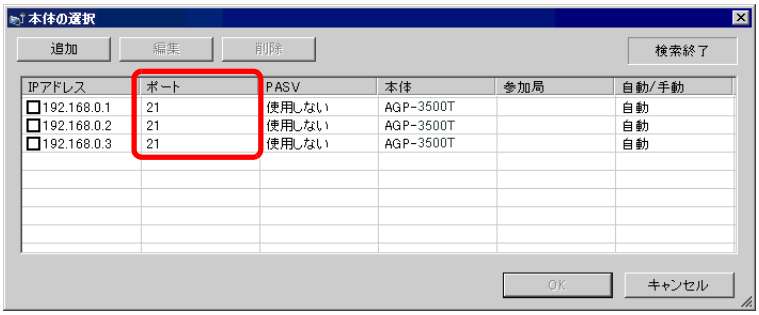
- CF カードや USB ストレージの代わりに使用するフォルダはネットワーク上にあっても設定できますが、接続先の環境（OS や言語設定など）によってはファイル名の表示が正しくできない場合があります。

重 要

- **IPC の OS が Windows XP Embedded を使用している場合、IPC の設定によりシステムのドライブ (C ドライブ) にライトフィルタ（書き込み禁止）することができます。指定した参照先フォルダが C ドライブの場合、ライトフィルタの設定が有効になっていると、ファイルを書き込むことができません。必ずライトフィルタの設定されていないドライブを参照先フォルダに設定してください。**
- **参照先フォルダは、出力先フォルダとは異なるフォルダを指定してください。転送先と転送元が同じになると転送エラーが発生します。**

■ 転送設定



設定項目	設定内容
ポート番号	転送用に使用するポート番号を 0 ～ 65535 で設定します。ポート番号を変更した場合は、プロジェクトを LAN 転送する際のポート番号を、ここで指定した番号にあわせる必要があります。 

MEMO

- 転送ツールで使用するポート番号を忘れた場合はオフラインモードの [WinGP 設定] ― [転送設定] で確認できます。

■ API 通信

設定項目	設定内容
API 通信の使用	API 通信（ハンドリング API またはデバイスアクセス API）を使用するかどうかを設定します。
ポート番号	API 通信に使用するポート番号を 0 ～ 65535 で設定します。8000 ～ 8019 番以外で、[転送設定] の [ポート番号] と重ならない番号を設定してください。 **MEMO** • [周辺機器設定] で他の接続機器が使用しているポート番号を確認し、重複しないように設定してください。

37.9.2 ウィンドウフレームの設定ガイド

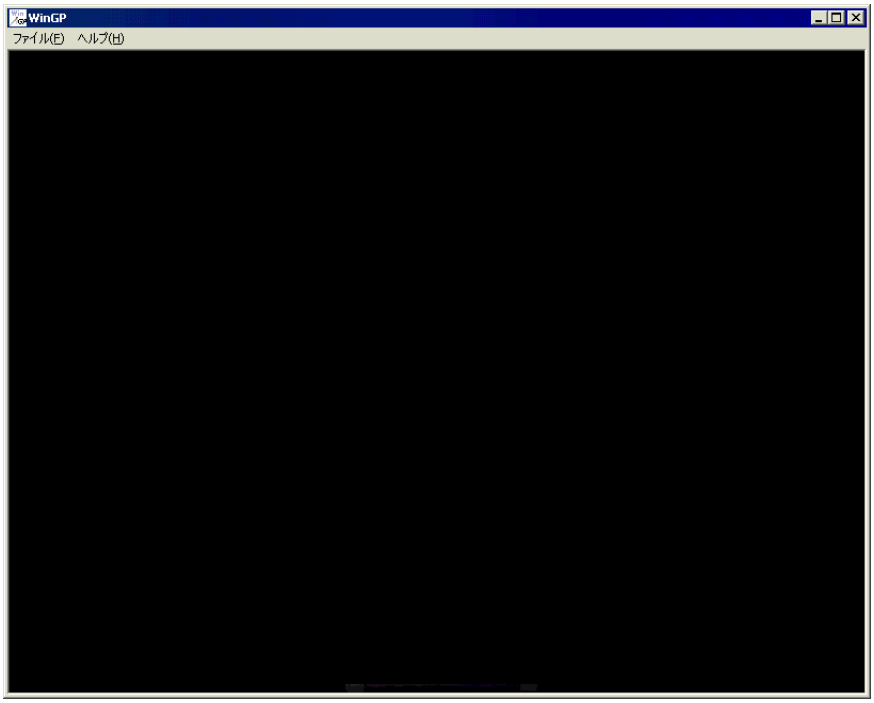
■ ウィンドウフレーム

表示器本体に表示される WinGP のウィンドウについてご紹介します。

MEMO

- 表示設定については以下を参照してください。

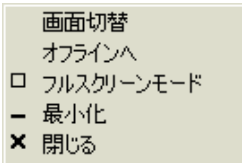
☞「37.9.1 システム設定ウィンドウ[本体設定]－[IPC設定]の設定ガイド ■ 表示設定」(37-136 ページ)



設定項目	設定内容
タイトルバー	ウィンドウタイトルや、ウィンドウ最小化、最大化、閉じるボタンを表示します。ウィンドウタイトルは [システム設定ウィンドウ] の [IPC 設定] で設定したタイトルが表示されます。タイトルが設定されていない場合は空白が表示されます。
最小化ボタン	ウィンドウを隠し、タスクバー内にアイコンを表示します。
最大化ボタン	ウィンドウをフルスクリーンモードに変更します。
閉じるボタン	WinGP を終了します。
メニューバー	• **ヘルプ** [バージョン情報] を表示します。 • **ファイル** WinGP を終了する [アプリケーションの終了] を表示します。
ウィンドウ枠	カーソルをウィンドウ枠にあてドラッグ＆ドロップするとウィンドウサイズを変更できます。初期設定のサイズより小さく変更した場合は、スクロールバーが表示されます。

■ 右クリック時のメニュー

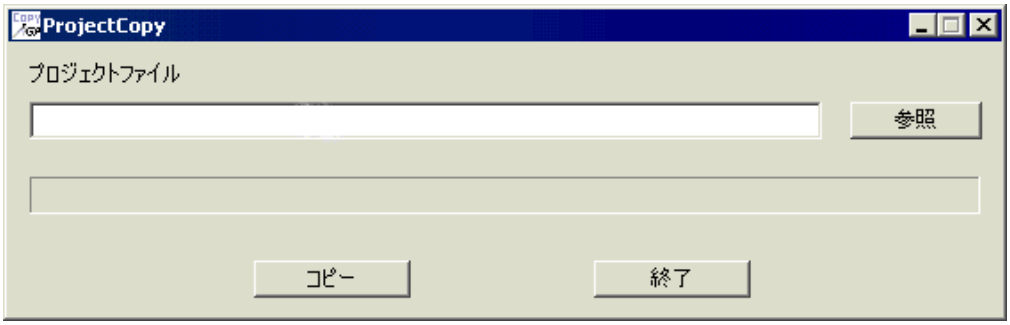
WinGP ウィンドウフレーム上で右クリックした時またはキーボードの をクリックすると表示されるメニューです。GP-Pro EX のシステム設定ウィンドウ [本体設定]-[IPC 設定] で、[右クリック時のメニュー表示] を設定している場合に表示されます。

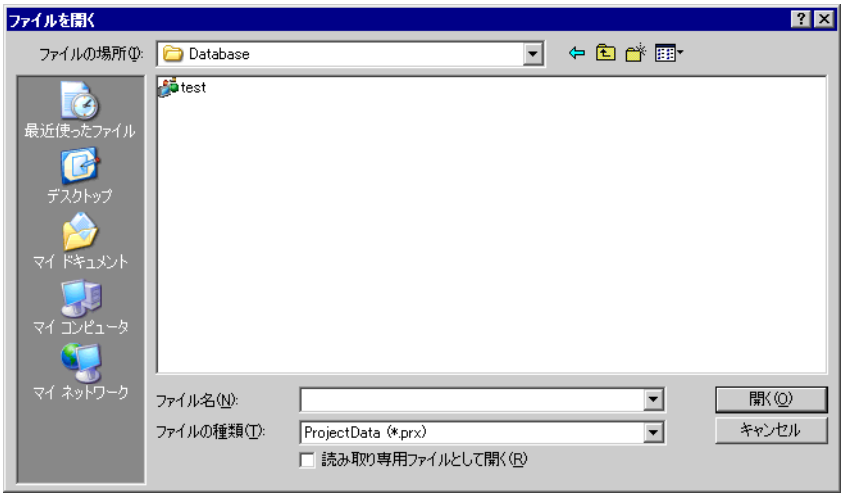


設定項目		設定内容
画面切替		[画面切替] を選択すると下記のような [表示画面の選択] ダイアログボックスが表示され、表示画面を切り替えることができます。  **MEMO** • オフラインモードの場合はこの項目はメニューに表示されません。
	画面番号	切り替える表示画面の番号を 1 ～ 9999 で設定します。 **MEMO** • プロジェクトファイルに設定されていない画面の画面番号を設定しても画面は開きません。
	番号	画面の番号を表示します。
	タイトル	画面のタイトルを表示します。
	開く	[画面番号] または画面番号一覧から選択した画面を開きます。
	閉じる	[表示画面の選択] ダイアログボックスを表示します。
オフラインへ (オンラインへ)		オフラインモードへ移行します。またオフラインモードを表示している場合はオンライン画面へ移行します。
フルスクリーンモード		画面をフルスクリーンで表示します。 **MEMO** • [フルスクリーンモード] で表示されている場合は、[ウィンドウ] が表示されウィンドウを元のサイズに戻します。 • IPC の右上と左下をタッチして表示される [システムメニュー] の [リセット] でも [フルスクリーンモード] のサイズを元に戻すことができます。
最小化		ウィンドウを隠し、タスクバー内にアイコンを表示します。
閉じる		WinGP を終了します。

37.9.3 ProjectCopy（コピーツール）の設定ガイド

[スタート] ― [すべてのプログラム] ― [Pro-face] ― [WinGP] ― [ProjectCopy] を選択すると以下のようなダイアログボックスが表示され、プロジェクトファイルの画面データのみを WinGP にコピーすることができます。



設定項目	設定内容
プロジェクトファイル	コピーするプロジェクトファイルのパスを入力または表示します。
参照	コピーするプロジェクトファイルの場所を指定します。 
コピー	コピーを開始します。
終了	ProjectCopy を終了します。

37.10 制限事項

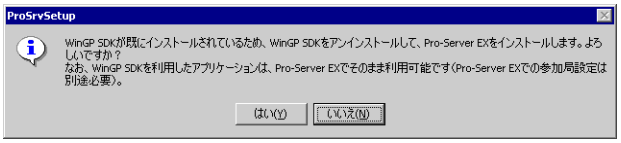
- WinGP は二重起動できません。
- IPC で 1 画面上の部品数が 1280 個を超えた場合、警告のメッセージが表示されます。画面に配置している部品の数を減らしてください。メッセージが表示されても部品の配置、転送はできます。
- IPC で 1 画面上のアドレス数が 3000 個を超えた場合、警告のメッセージが表示されます。画面に配置しているアドレスの数を減らしてください。メッセージが表示されても、アドレスの設定、転送はできます。
- IPC で多数の部品が配置されている場合、[表示器変更] で他のシリーズ へ変換すると部品数、アドレス数の制限値が変化するため保存時に警告が表示されます。
- 機種変更で設定されているアラーム履歴数やワード監視数が変換後の機種の制限値を超えている場合、表示器変更時にエラーが表示されますが、機種変更できます。
- 履歴記憶数は 8 区分まで設定できます。
- ブロック 1 ～ 8 までのビット監視・ワード監視両方の登録数の合計は最大 10000 個までです。
- GP-Pro EX の設定で SRAM を使用する総容量が 5MB を超えた場合はエラーチェック時に警告が表示され正しくサンプリング、アラーム機能が動作しません。プロジェクトファイルの保存や転送ができるのは 5MB までのデータです。
- OS のシャットダウン操作を行わずに IPC の電源を OFF した場合、WinGP 終了時のバックアップファイルの保存ができないため、記録内容が前回の保存が動作した時の内容になります。
  バッテリバックアップ機能を搭載した IPC では、電源断時にスタンバイモード（レジューム）信号が送られるため、WinGP ではこれを受けてバックアップファイルを保存します。
- タッチブザーの音設定は PC ランタイムが独自で鳴らすブザー音の設定の機能であり、IPC のタッチパネル本体のタッチブザー音設定とは異なります。IPC のタッチパネル本体のタッチブザー音と PC ランタイムのブザー音の両方を有と設定すると PC ランタイムの画面をタッチしたときに 2 回ブザーがなりますので、IPC のタッチパネル本体のタッチブザー音を有に設定している場合は PC ランタイムのタッチブザーを無に設定して下さい。
- システム設定ウィンドウの [スクリプト設定] － [通信設定] － [フロー制御] を無しに設定している場合、[SIO ポート操作] のステータス [EXIT_SIO_STAT] で送信エラーの検出はできません。
- 特殊スイッチや、トリガアクション、スクリプトの特殊動作で [アプリケーション起動] の [多重起動禁止] の設定をしていても [ウィンドウタイトル] が入力されていない場合は、多重起動が実行されます。
- 特殊スイッチや、トリガアクション、スクリプトの特殊動作で [アプリケーション起動] の多重起動を禁止したい [ウィンドウのタイトル] は完全に一致するウィンドウタイトルを入力してください。
- IPC 以外の機種で特殊スイッチや、トリガアクション、スクリプトの特殊動作の [アプリケーション起動]、[WinGP の終了] が設定されているプロジェクトファイルは GP に転送できますが、GP 上では動作しません。

37.10.1 インストール時の制限事項

- WinGP をインストールした先が半角英数 200 文字以上になる場合はシミュレーションを起動した際に「インストール先のパスが 200 文字を超えているため、起動できません」というエラーが表示され、正常に動作しません。WinGP を半角英数 200 文字以内のパスをインストール先として再インストールしてください。
- 対応 OS 以外の OS にインストールしようとするとエラーメッセージが表示され、インストールできません。
- インストールは Windows の Administrator 権限を持つアカウントでログオンし、インストールしてください。
- WinGP はインストール先を別フォルダ指定するなどして、複数インストールすることはできません。WinGP がインストールされている IPC に再度インストール CD を挿入するとアンインストールが実行されます。
- WinGP は修復インストールできません。修復する場合は WinGP をアンインストールし、再インストールを行ってください。
- WinGP はパソコンにインストールできますが動作はしません。
- Pro-Server with Pro-Studio for Windows または Pro-Server EX をインストールしている IPC に WinGP をインストールする場合は、インストール状況により、WinGP をインストールできない場合があります。各インストール状態は以下のとおりです。

インストール状態	WinGP のインストール
Pro-Server with Pro-Studio for Windows が既にインストールされている	以下のようなエラーメッセージが表示され、WinGP をインストールできません。Pro-Server with Pro-Studio をアンインストールしてから WinGP をインストールしてください。 
Pro-Server EX Ver1.10 未満が既にインストールされている	以下のようなエラーメッセージが表示され、WinGP をインストールできません。Pro-Server EX をアンインストールするか、V1.10 以上にバージョンアップしてから WinGP をインストールしてください。 
Pro-Server EX Ver1.10 以上が既にインストールされている	WinGP のインストールができます。（WinGP SDK はインストールされません）
Pro-Server with Pro-Studio for Windows、Pro-Server EX 共にインストールされていない	WinGP のインストールができます。（WinGP SDK が自動的にインストールされます）

- WinGP をインストールしている IPC に Pro-Server with Pro-Studio for Windows または Pro-Server EX をインストールする場合は、WinGP が正常に動作しない場合があります。各動作は以下のとおりです。

インストールする S/W	動作
Pro-Server with Pro-Studio for Windows	Pro-Server with Pro-Studio for Windows、WinGP 共に動作しなくなります。その場合は、一旦両方をアンインストールしてください。WinGP がインストールされている IPC には Pro-Server with Pro-Studio for Windows をインストールしないでください。
Pro-Server EX Ver1.10 未満	Pro-Server EX Ver1.10 未満のインストーラ起動直後に以下のようなエラーメッセージが表示され、インストールを行うことができません。Pro-Server EX はインストールされていなくてもエラーメッセージでは以下のように表示されます。 
Pro-Server EX Ver1.10 以上	Pro-Server EX Ver1.10 以上のインストーラ起動直後に以下のようなエラーメッセージが表示されます。[はい] を選択した場合、WinGP SDK がアンインストールされ、その後 Pro-Server EX Ver1.10 のインストールが行われます。  また Pro-Server EX Ver1.10 のインストール途中にインストールを中断した場合は、WinGP を再インストールしてください。 **MEMO** • WinGP をインストールすると GP-Pro EX のインストール先に SDK というフォルダで WinGP SDK がインストールされ、Pro-Server EX でユーザアプリケーション作成時に指定したパスとは異なりますが、パスを変更することなく Pro-Server EX で作成したユーザアプリケーションも使用できます。 • WinGP をインストールした後に Pro-Server EX をインストールし、Pro-Server EX をアンインストールすると WinGP SDK が使用できません。

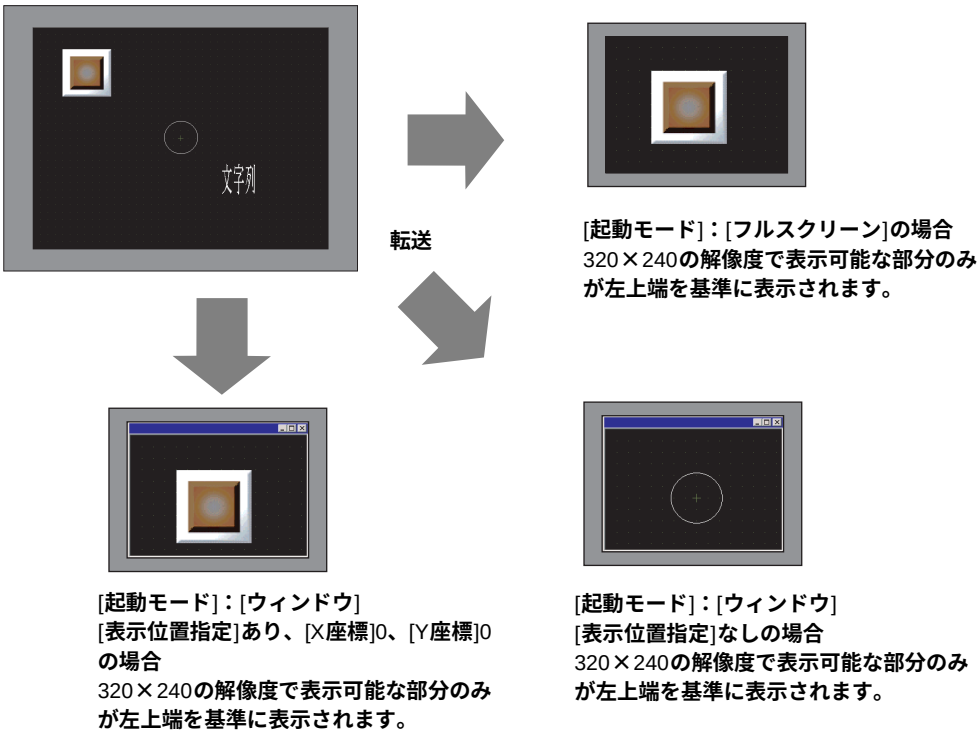
- インストール後、WinGP を使用する前に必ず再起動してください。再起動せずに WinGP を起動すると正しく動作しません。

37.10.2 ウィンドウフレームの制限事項

- 画面解像度 (画面サイズ) の異なる IPC へは転送できますが、解像度が小さくなる場合は正しく表示されません。

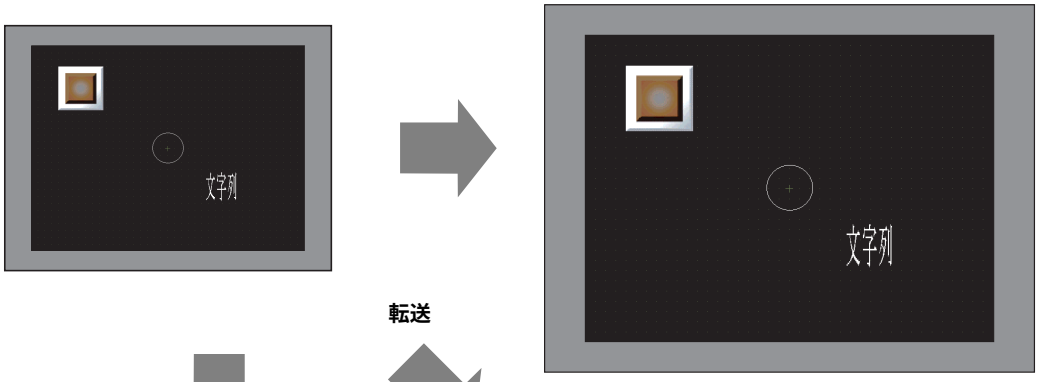
例 1)
IPC：画面サイズ (800 × 600) で作画し、320 × 240 の IPC へ転送した場合

作画データ

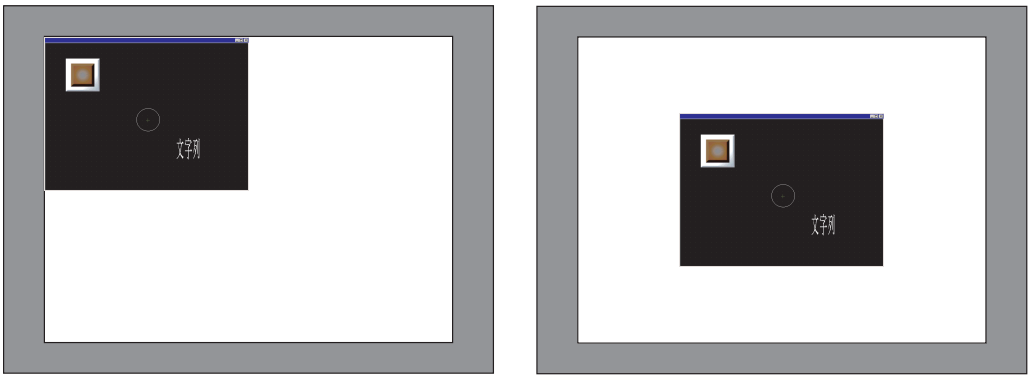


例 2)
IPC：画面サイズ (800 × 600) で作画し、1600 × 1200 の IPC へ転送した場合

作画データ



[起動モード]：[フルスクリーン]の場合
800×600の解像度で、1600×1200に拡大されて表示されます。



[起動モード]：[ウィンドウ]
[表示位置指定]あり、[X座標]0、[Y座標]0の場合
800×600の解像度で左上端を基準に表示されます。

[起動モード]：[ウィンドウ]
[表示位置指定]なしの場合
800×600の解像度でセンタリングして表示されます。

- 画面サイズいっぱいの解像度で画面データを作成した場合、ウィンドウ枠の一部が画面からはみ出します。
  画面からはみ出さないようにするためには、「タイトルバーの表示」、「ウィンドウ枠」、「メニューバー」を表示しない設定にするか、またはフルスクリーンで表示してください。タイトルバーを表示しない場合やフルスクリーンにした場合は、タイトルバー上の終了ボタンが表示されませんのでご注意ください。
- IPC 間で機種変更を行った場合、システム設定ウィンドウの [IPC 設定][表示設定] は機種変更前の設定を保持します。ただし、[表示位置] の X 座標、Y 座標は初期値 “0” に、「ウィンドウサイズ」は変換先の IPC の機種に関係なく、XGA (1024 × 768) のサイズに初期化されます。

37.10.3 Windows XP Embedded 使用時の制限事項

- Windows XP Embedded ではシステムドライブをライトフィルタする機能があり、ライトフィルタ中は、システムドライブにファイル書き込みできないため、ファイル書き込み先フォルダをライトフィルタされていないドライブに設定する必要があります。このため、ファイル書き込み先フォルダを設定により変更できます。

37.10.4 API 通信の制限事項

- Windows XP SP2 にて API 通信を利用する場合、Windows Update にて次のパッチを必ず当ててください。「Windows XP Service Pack 2 用更新プログラム (KB884020)」

■ ハンドリング API の制限事項

- ハンドリング API の文字情報はすべて Unicode で扱います。API でバージョン情報とプロジェクト情報を読み出す場合は、Unicode で読み出されるので他の文字コード (ASCII など) で使用されたい場合は変換してください。
- ハンドリング API は TCP/IP の設定がない IPC 上では使用できません。必ずネットワーク設定に TCP/IP プロトコルがインストールされていることをご確認ください。

■ デバイスアクセス API の制限事項

- デバイスアクセス API を使用する場合は先に WinGP を起動してください。WinGP が起動していない状態でデバイスアクセス API を利用するとエラーになります。また WinGP 終了後にデバイスアクセス API を起動するとタイムアウトエラーが発生します。
- ユーザーアプリケーションで API 通信実行中に IPC をスタンバイ状態にしないでください。
  必ずデバイスアクセス API の実行が完了してからスタンバイ状態になるよう、ユーザーアプリケーションで制御してください。
- Pro-Server EX のバージョンアップによりプロトコルが追加される場合は、GP-Pro EX でアップデートしたプロトコルのモジュールを WinGP SDK がインストールされている IPC でインストールする必要があります。
- ReadSymbolD(),ReadSymbolVariantD(),WriteSymbolD(),WriteSymbolVariantD() の API で下記の配列要素数を超える配列変数は使用できません。

配列変数の種類	WinGP API 通信でアクセス可能な最大要素数
ビット変数	255
整数変数	510
フロート変数	510

- Pro-Server EX V1.10 をインストールした場合は Pro-Server EX の制御を別途実施しなければならなくなります。
- デバイスアクセス API は TCP/IP の設定がない IPC 上では使用できません。必ずネットワーク設定に TCP/IP プロトコルがインストールされていることをご確認ください。
- デバイスアクセス API でアクセスしている途中で WinGP を終了した場合は API からのリターンがすべてエラーとなります。
- Visual C++ Ver.6 で C:¥Program files¥Pro-face¥WinGP¥SDK¥VC¥Public¥ProEasy.h または Pro-Studio の [プログラミング補助]-[VC：宣言文] でクリップボード経由で作成したヘッダをコンパイルすると LPVARIANT が未定義のエラーになることがあります。LPVARIANT は afxdisp.h の中で定義されていますので、これを include していないと未定義エラーになります。これを回避するには、通常は stdafx.h の中で #include <afxdisp.h> と定義するようにしてください。

37.10.5 転送時の制限事項

- モデム転送や COM ポートを使用した転送はできません。
- WinGP は起動後の初期化処理中に、必要なファイルに異常がある場合 (破損もしくは消失)、再転送要求を促す画面を表示します。
- 機種が異なる IPC へ転送すると異なる機種であることを示すエラーダイアログボックスが表示され、転送できません。異なる機種のプロジェクトファイルを転送する場合は、エディタで機種変換を行った上で転送してください。
- [ProjectCopy]（コピーツール）は WinGP が使用するファイルを更新するため、WinGP を終了させておく必要があります。WinGP 動作中にコピー操作をした場合はエラーメッセージが表示され、コピーが実行されません。
- OS が Windows XP Embedded である場合、IPC のツールにより、システムのドライブ (C ドライブ) にライトフィルタを設定することができます。WinGP のインストール先が C ドライブであった場合、ライトフィルタを有効に設定していると、WinGP のシステムファイルや画面データを書き変えることができません。転送前にライトフィルタを無効に設定してから、転送を開始してください。
- WinGP では転送ツールで使用するポート番号を変更できるため、変更後のポート番号を忘れてしまうと転送ツールから LAN 転送できません。

■ [ProjectCopy]（コピーツール）を使用した場合の制限事項

- ProjectCopy [コピーツール] は、画面データの送信のみできます。画面データの受信やプロジェクトの全転送はできません。以下の場合は、転送ツールを使用してください。
  - WinGP インストール後、はじめてプロジェクトファイルを転送する場合
  - 接続機器を変更、追加した場合
  - 使用するフォントを変更、追加した場合
  - GP-Pro EX のバージョンアップによりラインタイムシステムやプロトコルドライバがバージョンアップされた後に、プロジェクトファイルを更新した場合
- コピーツールでは WinGP のシステムプログラムを送信できません。WinGP をバージョンアップする場合は転送ツールを使用してください。

37.10.6 エラーログの制限事項

- エラーログ機能が書き込みしようとしているタイミングで、エラーログファイルが開かれている場合は、ファイルへの書き込みができません。
- エラーメッセージの発生が [エラー設定] の [保存ファイル数] を超えた場合は 1 番古いファイルを削除し、新しいファイルが追加されます。
- 前回保存時から 10 分以内であれば、頻繁に書き込みアクセスするのを防ぐため、エラーログを保存せず前回保存時から 10 分経過後に保存します。その場合は 10 分間で記録した内容をすべてエラーログファイルに書き込みます。