

38 | WinGP (GP Operations on IPC Series)

This chapter covers how to run GP-Pro EX projects on IPC Series machines, connect to devices/PLCs, and run other applications from WinGP. You can also run WinGP on PC/AT compatible machines.

Please start by reading "38.4 Settings Menu" (page 38-34) and then turn to the corresponding page.

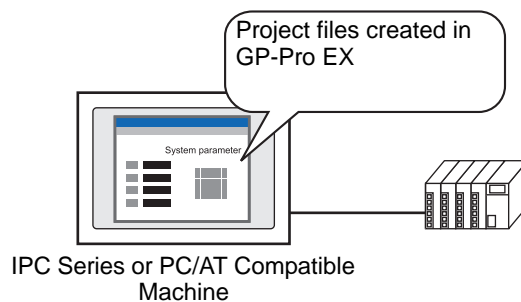
38.1	What is WinGP?	38-2
38.2	Operating Environment	38-7
38.3	Development Process	38-12
38.4	Settings Menu	38-34
38.5	Retrieve WinGP information or Operate WinGP from user application.....	38-35
38.6	Running the Application from WinGP.....	38-73
38.7	Allocating the switch feature to the function key	38-79
38.8	Keep History of Error Messages Displayed in WinGP	38-87
38.9	API Function List	38-89
38.10	Settings Guide.....	38-164
38.11	Restrictions	38-175

38.1 What is WinGP?

38.1.1 What is WinGP?

■ Summary

WinGP is an application that can run GP-Pro EX project files on Digital's Industrial Panel Computers (IPC) or PC/AT-compatible machines and communicate with connected devices/PLCs. However, because GP and IPC or PC/AT compatible machines are different types of hardware, there are differences in the functions that IPC or PC/AT compatible machines can use. There are features that fully utilize the extra memory capacity on the IPC or PC/AT compatible machines, and there are applications that have been developed specifically for IPC or PC/AT compatible machines.



■ License

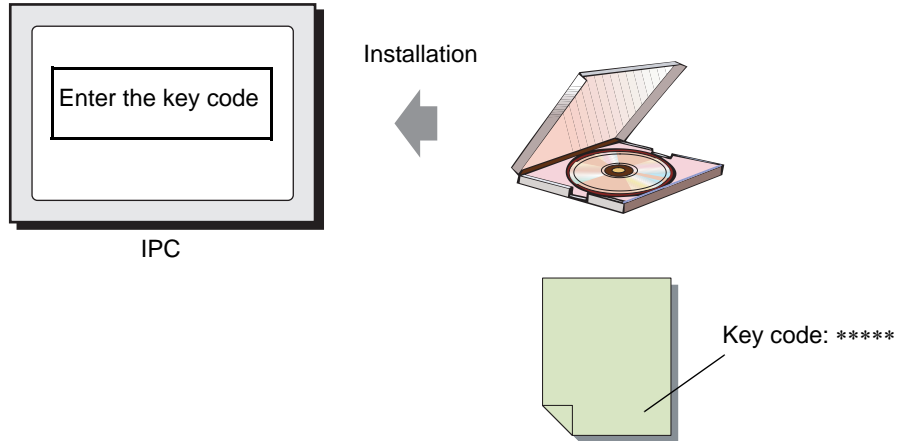
To use the WinGP, you need to purchase the license separately

When you purchase the license, a document with the [Key code] will be issued.

IMPORTANT

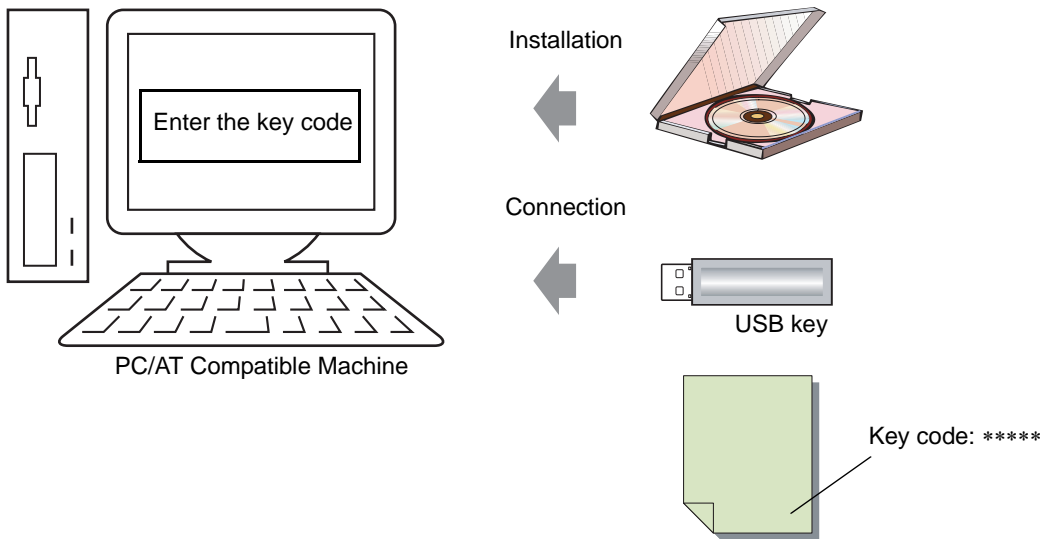
- To install WinGP, the key code is required. Please purchase the WinGP license separately.
(Type: EX-WINGP-IPC)
See below for the installation procedure.
☞ "38.3.2 Setup Procedure ■ Installation/Uninstall" (page 38-13)
- The key code cannot be reissued if lost. Please keep it at hand.

◆ IPC



◆ PC/AT Compatible Machine

When you purchase a license for PC/AT compatible machines, you are provided with a [USB Key] and a document with the [Key code].

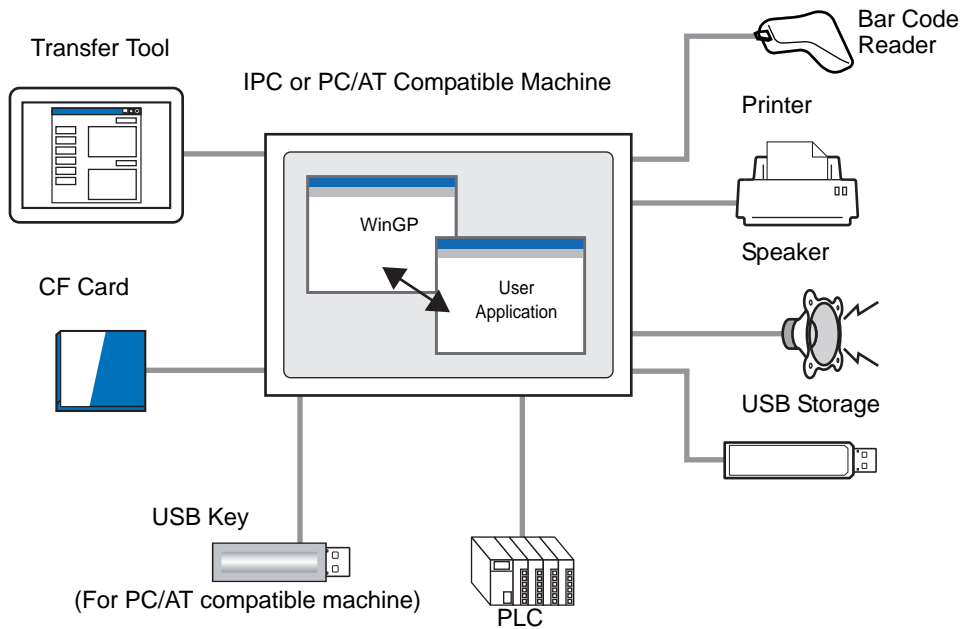


NOTE

- WinGP stops when you remove the USB key. Please leave the USB key inserted while working with WinGP.

38.1.2 Full Configuration

The following figure shows the connections and option environment for using WinGP.



38.1.3 Differences between IPC and GP

Since the IPC has a larger memory and storage area, the size of screen data and record data can be expanded as shown below, unlike the GP-3500 series.

Model	Function	Description
1	Maximum user data size	8MB → 16MB
2	Maximum SRAM size	512KB → 5MB
3	Maximum number of parts per screen	384 parts → 1280 parts
4	Maximum number of devices per screen	1152 parts → 3000 parts
5	Number of alarms saved in the history	768 → 10000
6	The number of registered alarm messages	2048 → 10000
7	Maximum DRAM size	320KB → 5MB

■ Features not available in WinGP

Most WinGP functions are available on IPC or PC/AT compatible machines, except for the following features:

- Buzzer/AUX output
- USB connection for two-dimensional code reader
- Printer operation using scripts
- Movie record/play feature
- Video display on the VM unit
- Memory loader feature
- Modem transfer feature
- Detect Backlight Burnout
- CF Card initialization in offline mode
- User data initialization in offline mode
- Pass-through feature
- The backlight OFF, screen display ON and OFF features of the system data area, set time
- Logic Program
- Logic monitor
- Address Monitor
- I/O Driver
- FTP Server Connection
- Web Server
- Ladder Monitor

NOTE

- The following information describes functions supported by IPC or PC/AT compatible machines.

☞ "1.3 Supported Features" (page 1-5)

■ **Features available only in WinGP**

Feature	Feature Details
Switch Parts	The [Start application] switch to start other applications and the [Exit WinGP] switch to exit WinGP are available.
Trigger Action	Start other applications (EXE operation). Exit WinGP (Exit WinGP operation).
Script	Start other applications (EXE operation). Exit WinGP (Exit WinGP operation).
Device Access API	API that can read and write to a device that is connected to the IPC or PC/AT.
Handling API	The API obtains the WinGP state from the third party software tools and changes the settings.
Error log feature	Saves the error summary displayed during WinGP communication in a file.
Right-click menu	To show this menu, right-click in the window. You can switch screens and modes between offline and online, maximize the window to full screen, and minimize and exit the window from this menu.

38.2 Operating Environment

38.2.1 Compatible Models

The following IPC and PC/AT compatible machines support WinGP.

NOTE

- To check the specifications for each supporting model, see the IPC series user manual.
 - WinGP will only start on the models listed below.
-

■ PS Series

When running WinGP, use one of the following operating systems.

Windows® 2000 (Service Pack 3 or later)

Windows® XP

Windows® XP Embedded

- PS3451A-T41-24V
- PS3450A-T41-24V
- PS3450A-T41
- PS3651A-T41
- PS3650A-T41
- PS3700A-T41-ASU-P41 (Rev.H or later)
- PS3710A-T41
- PS3710A-T41-PA1
- PS3711A-T41-24V
- PS2000B-41 (Pentium III 1GHz) (Rev.M* or later)

■ PL Series

When running WinGP, use one of the following operating systems.

Windows® 2000

Windows® XP

Windows® XP Embedded

- APL3000-BD
- APL3000-BA
- APL3600-TA
- APL3600-TD
- APL3600-KA
- APL3600-KD
- APL3700-TA
- APL3700-TD
- APL3700-KA
- APL3700-KD
- APL3900-TA
- APL3900-TD

■ PC/AT

WinGP Operating Environment Requirement

	Required Specifications	Remarks
CPU	1GHz or faster	
Operating System	Windows® 2000 (Service Pack 3 or later) Windows® XP (Home Edition/ Professional) Windows Vista® (Ultimate / Home Premium / Home Basic / Business / Enterprise) 32bit	
Resolution	SVGA 800x600 or above	SXGA is recommended.
Memory	512 MB or more	1 GB or more is recommended.
Hard Disk Space	200 MB or more	This capacity is required for installation.

NOTE

- When WinGP operates in a non-Japanese operating system environment, the WinGP window menu bar, right-click menu, copy tool, and popup messages are all displayed in English. In the offline mode, they are displayed in the system language selected in [Display Unit], [Menu and Error Settings], [System Language].

38.2.2 Supported Protocols

■ Available Protocols

IMPORTANT

- Even though a driver supports the WinGP, the WinGP may not operate due to connection methods. Please refer to "GP-Pro EX Device/PLC Connection Manual" for the connections.
- Please check the latest information about supported drivers at the Pro-face support site, Otasuke Pro! <http://www.pro-face.com/otasuke/>)

The following Device/PLC drivers support the WinGP.

Maker	Driver name
Digital Electronics Corporation of Japan	Memory Link
	General-purpose Ethernet
Mitsubishi Electronics Corporation	A Series CPU Direct
	A Series Ethernet
	A Series Calculator Link
	FX Series CPU Direct
	FX Series Calculator Link
	Q series CPU Direct
	Q/QnA Serial Communication
	Q/QnA Series Ethernet
	QnA Series CPU Direct
	QUTE Series CPU Direct
OMRON Corporation	C/CV Series Upper Link
	CS/CJ Series Upper Link
	CS/CJ Series Ethernet
	Adjuster CompoWay/F
Yokogawa Electric Corporation	PC link SIO
	PC link Ethernet
Siemens AG	SIMATIC S5 CPU Direct
	SIMATIC S7 3964(R)/RK512
	SIMATIC S7 Ethernet
Rockwell Automation	DF1
	EtherNet/IP
Schneider Electric Industries	MODBUS SIO Master
	MODBUS TCP Master
	MODBUS Slave
	Uni-Telway
Yaskawa Electric Corporation	MEMOBUS SIO
	MEMOBUS Ethernet
	MP Series SIO (Expanded)
	MP Series Ethernet (Expanded)

Continued

Maker	Driver name
KEYENCE Corporation	KV-700/1000 Series CPU Direct
	KV-700/1000 Series Ethernet
Yamatake Corporation	Digital Controller SIO
Hitachi Industrial Equipment Systems Co., Ltd.	H series SIO
	H series Ethernet
Hitachi, Ltd.	S10V Series Ethernet
	S10 series SIO
Meidensha Corporation., Ltd.	UNISEQUE series Ethernet
GE Fanuc Automation	Series90 Ethernet
	Series 90-30/70 SNP
	Series 90-30/70 SNP-X
LS Industrial Systems Co., Ltd.	MASTER-K Series Cnet
	XGT Series FENet
Saia-Burgess Controls Ltd.	Saia S-Bus SIO
Sharp MS Corporation	JW Series Computer Link SIO
	JW Series Computer Link Ethernet
FANUC Ltd.	Power Mate Series
Mitsubishi Heavy Industries, Ltd.	DIASYS Netmation MODBUS TCP
	UP/V
Matsushita Electric Works, Ltd.	FP series PC link SIO
Fuji Electric FA Components & Systems Co., Ltd.	MICREX-F series SIO
	MICREX-SX Series SIO
	MICREX-SX Series Ethernet
JTEKT Corporation	TOYOPUC CMP-LINK Ethernet
	TOYOPUC CMP-LINK SIO
RKC Instrument Inc.	Controller MODBUS SIO
	Temperature controller
Toshiba Corporation	Computer Link SIO
	Computer Link Ethernet
Toshiba Machine Co., Ltd.	PROVISOR TC200
Shinko Technos Co., Ltd.	Controller SIO
Koyo Electronics Industries Co., Ltd.	KOSTAC/DL Series CCM SIO
	KOSTAC/DL Series MODBUS TCP
IAI Corporation	ROBO Cylinder MODBUS SIO
FATEK AUTOMATION Corporation	FB Series SIO
CHINO Corporation	Controller MODBUS SIO
Modbus-IDA	General-purpose MODBUS RTU SIO Master
Hyundai Heavy Industries Co., Ltd.	Hi4 Robot

38.2.3 Model Environment

In this section, the following system configuration is used as a model to explain the operations and features. In other system configurations, the display and part names may differ. If so, replace the names with those with similar features used in your system configuration.

■ Standard Configuration

Hardware/Software	Model system specifications	Remarks
Operating System	Windows® 2000	-
Device/PLC	Q/QnA serial communication series manufactured by Mitsubishi Electric Corporation	-
IPC or PC/AT Compatible Machine	PS-3650A	-

38.2.4 Application Development Environment

Microsoft® Visual Basic Ver.6.0

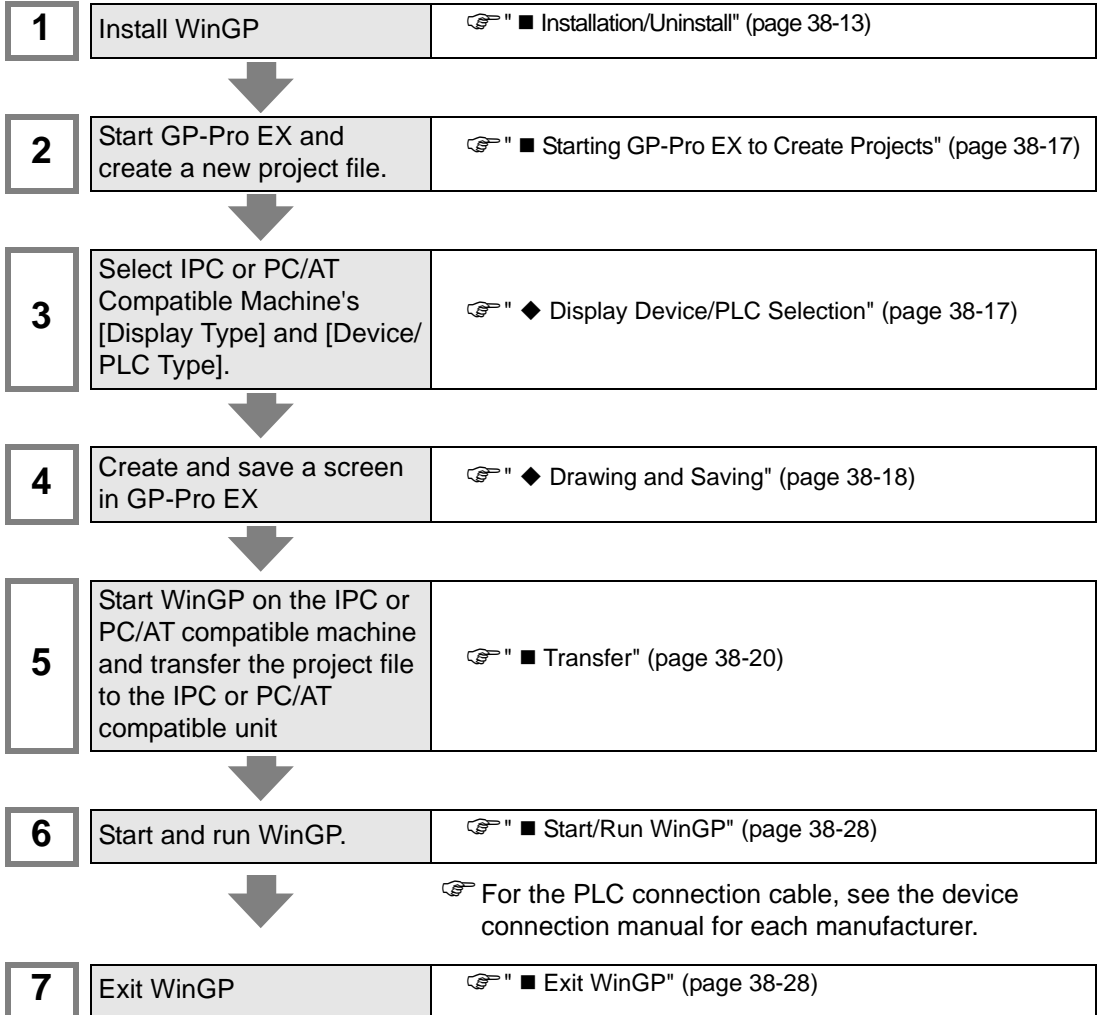
Microsoft® Visual C++ Ver.6.0 or Ver.7.0

Microsoft® Visual Studio .NET 2003 or later

38.3 Development Process

38.3.1 Development Process

The following figure shows the process flow, from installing WinGP, starting GP-Pro EX, creating screens, and connecting to the device/PLC, to running project files on the IPC or PC/AT compatible machine. Click the link to view the page explaining each process.



38.3.2 Setup Procedure

■ Installation/Uninstall

IMPORTANT

- WinGP will not operate if installed on an unsupported IPC or PC/AT compatible machine.
 - Exit all programs including virus check software.
 - Use a user account with administrator authority for installation.
-

- Windows XP Embedded Users

Windows XP Embedded has a write protection setting. To install WinGP on your C drive, you need to disable the write protection filter setting. Using EWFSettingTool.exe, select "EWF Disable" to disable the setting before installation.

☞ Windows XP Embedded users manual "3.1 Write filter setting process"

- Pro-Server EX Version Before V1.10 or Pro-Server with Pro-Studio Users

You cannot install WinGP on an IPC in which Pro-Server EX before V.1.10 or Pro-Server with Pro-Studio is installed. If an earlier version of Pro-EX exists, either uninstall or update Pro-Server EX to V1.10 or later.

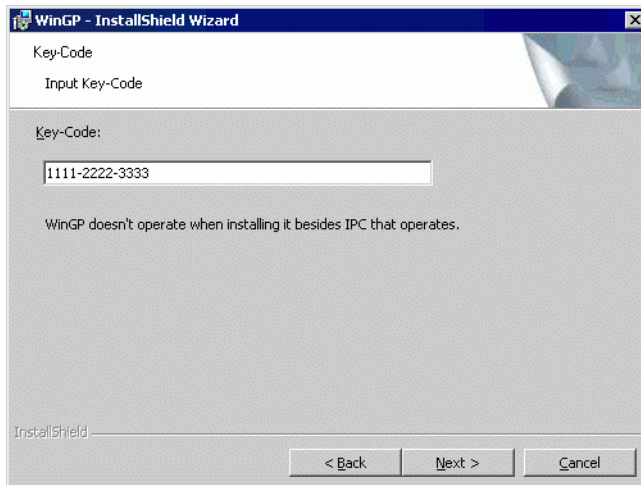
☞ "38.11.1 Restrictions On Installation" (page 38-177)

◆ **Installation Procedure**

- 1 Insert the GP-Pro EX Installation CD (Disk 2) into the CD drive of the IPC or PC/AT compatible machine.
- 2 In the installation set up, click [WinGP].



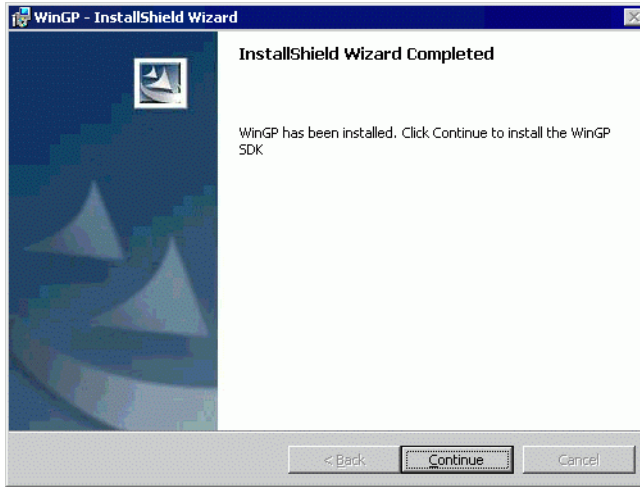
- 3 Follow the instructions in the installation wizard to complete installation.
- 4 During installation, you are asked to enter the key code. Enter your separately purchased key code (enter: EX-WINGP-IPC).



NOTE

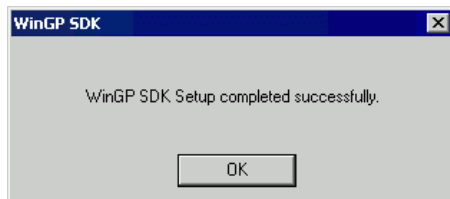
- For details on key codes, see below.
☞ "38.1.1 What is WinGP? ■ License" (page 38-2)

5 After WinGP is installed, install WinGP SDK sequentially. Click [Continue].

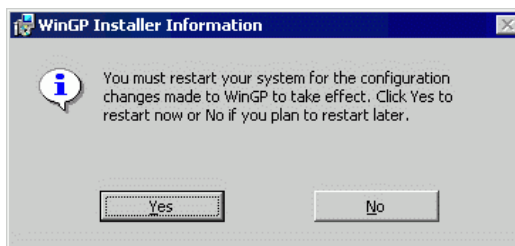


-
- NOTE**
- WinGP SDK is a software for communicating with external applications created on the WinGP and VB .NET, VB or VC using API. If Pro-Server EX V1.10 or later is already installed, WinGP SDK will not install. In this case, the device access API is available on Pro-Server EX V1.10. Only WinGP will be installed. For restrictions on installation, see below.
☞ "38.11.1 Restrictions On Installation" (page 38-177)
-

6 The following message appears. Click OK to complete the installation.



7 Once the installation is complete, the following message appears. Select [Yes] and restart the IPC (or PC).

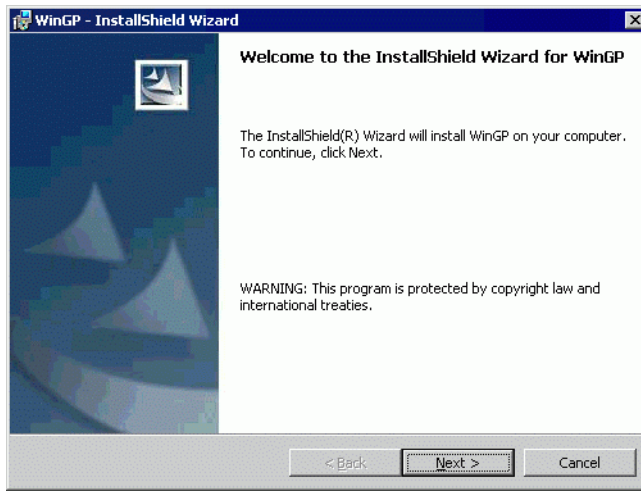


-
- NOTE**
- After the installation, restart the IPC before using WinGP. The WinGP will not operate properly without restarting the IPC.
-

◆ Uninstall

There are two ways to uninstall.

- Uninstall using Windows Control Panel, [Add/Remove Programs]
From the task bar, click [Start], point to [Settings] and then click [Control Panel]. In the [Control Panel], select [Add/Remove Programs]. In the list of installed applications, select [GP-Pro EX 2.00 WinGP] and click [Remove] to uninstall.
- Uninstall WinGP using GP-Pro EX CD-ROM.
Insert the GP-Pro EX CD-ROM to uninstall. When the GP-Pro EX CD-ROM is inserted, the following screen appears; click [Next (N)] and follow the wizard to uninstall WinGP.



NOTE

- WinGP SDK is uninstalled together with WinGP.
 - Uninstalling Pro-Server EX V1.10 from the PC with WinGP and Pro-Server EX V1.10, API communication is disabled. Please re-install WinGP.
-

■ Starting GP-Pro EX to Create Projects

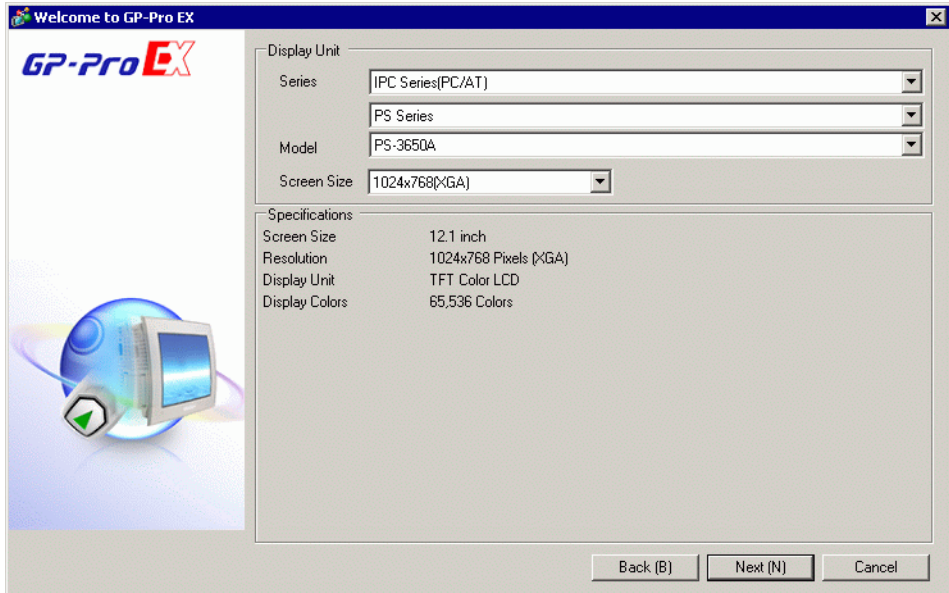
Start GP-Pro EX to create a new project file. The start process is the same as steps 1 to 3 in "5.2.2 Setup Procedure".

◆ Display Device/PLC Selection

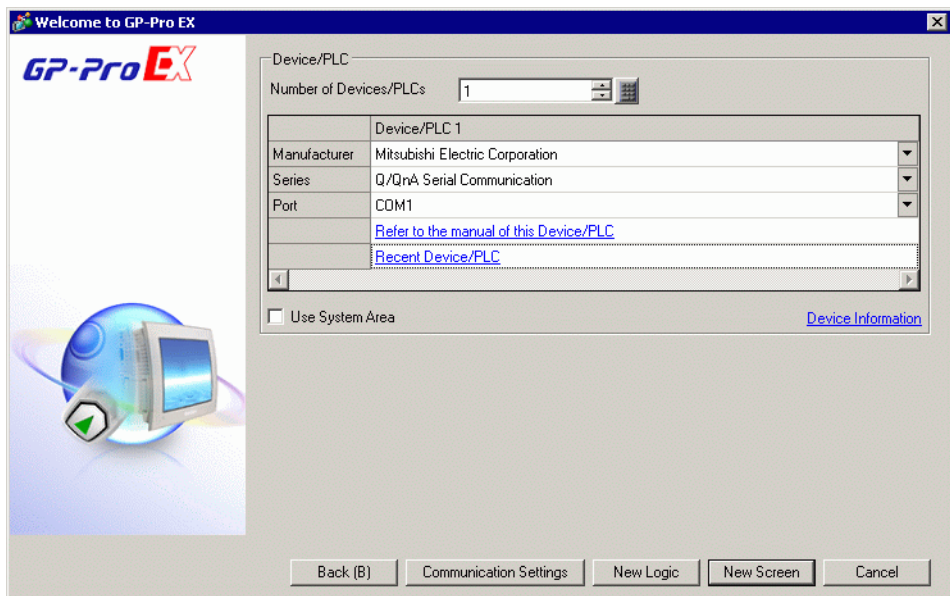
NOTE

- Please refer to the Settings Guide for details.
 ☞ "5.17.2 [New] Settings Guide" (page 5-102)

1 In [Display Unit], [Series], select [IPC Series (PC/AT)] and select the type you use.



2 Select the device/PLC [Maker] and [Series] that you are connecting to the IPC. If connecting the device/PLC to the IPC COM port, for the [Port] select COM1 to COM9.



3 Click [Communication Settings] to set up the communication format and other settings. The set up process is the same as steps 6 to 7 in "5.2.2 Setup Procedure".

◆ **Drawing and Saving**

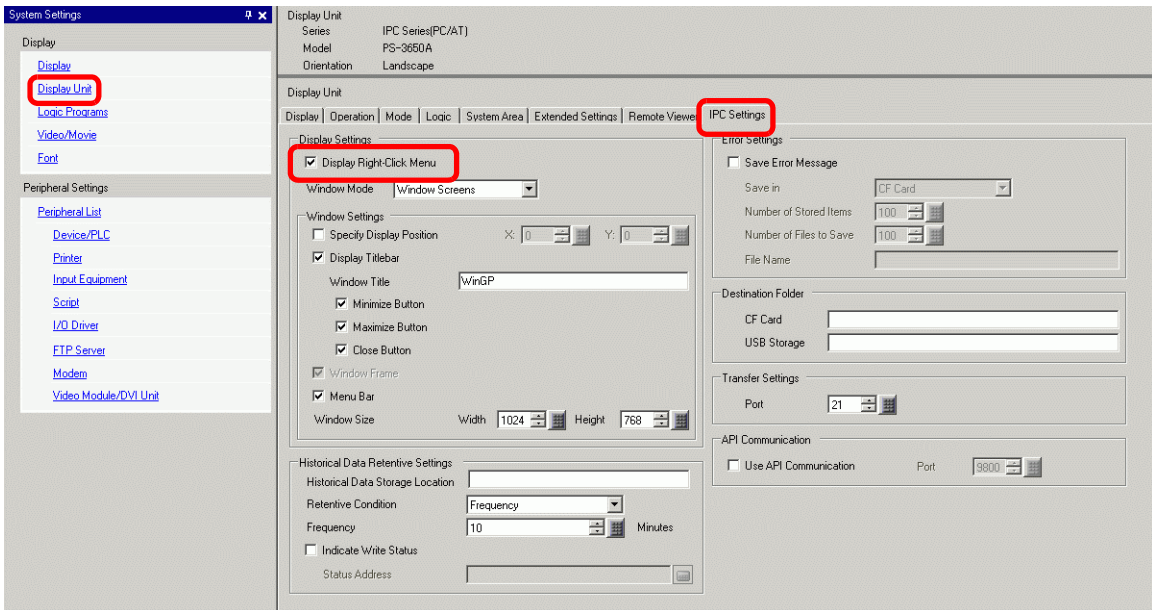
1 Draws pictures. For drawing methods, refer to "5.2.2 Setup Procedure ■ Creating/Saving" (page 5-15). You can also refer to chapters related to particular features, such as alarms.

IMPORTANT

- Since the GP, IPC, and PC/AT compatible machines are different pieces of hardware, the available features are different. For features available in WinGP, see below.

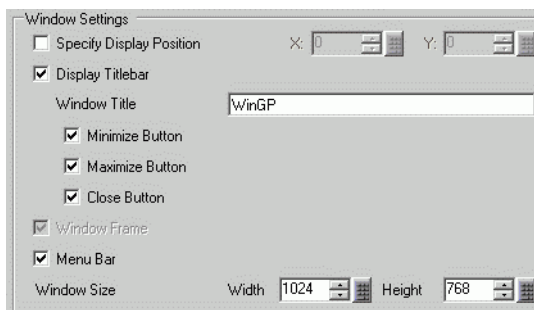
☞ "38.1.3 Differences between IPC and GP" (page 38-4)

2 In the System Settings window, select [Display Unit] and click the [IPC Settings] tab. Select the [Display Right-Click Menu] check box. When selected, the right-click menu enables you to change screens or go to Offline mode.



3 From [Window Mode], select [Window Screens].

4 As required, in the [Window Settings], define the window display position and show or hide the window titlebar.



5 When using the backup SRAM to store alarms, sampling data, or recipe files, in the [Historical Data Retentive Settings] area's [Historical Data Storage Location] field, type the folder path that will be used to emulate the SRAM backup function.

NOTE

- When you do not input a path, it is saved to the following WinGP installation folder: "NAND\PRJ001\USER\SCREEN"
-

6 When setting up destination folders in the [Destination Folder] area, in the [CF Card Folder] or [USB Storage Destination Folder] fields, type the path where data will be output, relative to the Screen Transfer destination. WinGP references the data (such as Recipes) in the folder defined here.

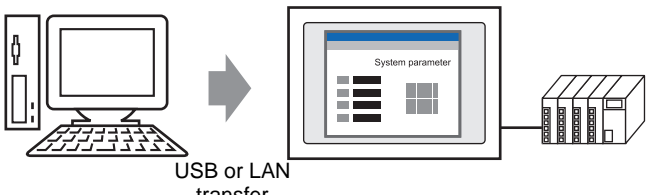
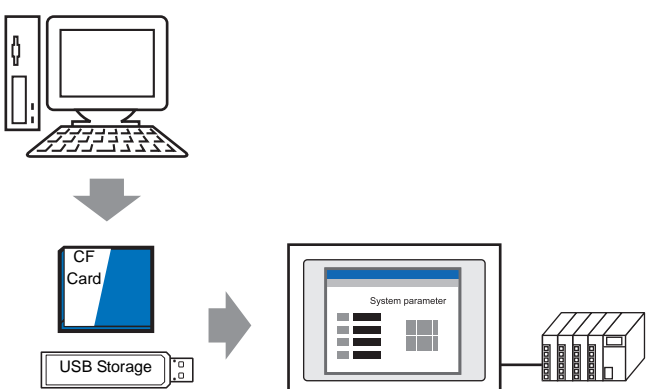
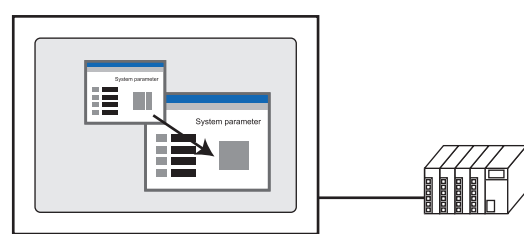
NOTE

- If you do not input a path, it is saved to one of the following WinGP installation folders: "CFA00" or "USBHD".
 - Define a Destination Folder that is different from the [CF Card Folder] or [USB Storage Destination Folder]. Otherwise, an error will occur.
-

7 From the [Project (F)] menu, select [Save as (A)]. Define the location and file name for the project.

■ Transfer

Transfers a project to the IPC or PC/AT compatible machine. The transfer operation is different when you create a GP-Pro EX project on a PC and then transfer it to a different machine, and when you create the GP-Pro EX project on the same machine as GP-Pro EX and WinGP.

◆ Transferring a Project to a IPC or PC/AT Compatible Machine	
<p>Transferring with USB cable/Ethernet cable</p>  <p style="text-align: center;">USB or LAN transfer</p>	<p>☞ "• Transferring with USB cable/Ethernet cable" (page 38-21)</p>
<p>Transferring from a CF Card or USB storage</p> 	<p>☞ "• Transferring from CF Card or USB storage" (page 38-23)</p>
◆ Installing GP-Pro EX and WinGP on the same IPC or PC/AT Compatible Machine	
	<p>☞ " ◆ Installing GP-Pro EX and WinGP on the same IPC or PC/AT Compatible Machine" (page 38-25)</p>

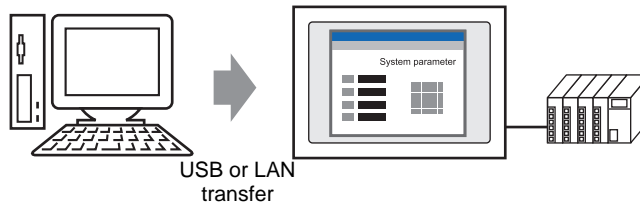
IMPORTANT


- When using Windows XP Embedded, a write protection is set at default. Thus, you need to disable the write protection filter setting before transferring a project file. Select "EWF Disable" from EWFSettingTool.exe in Windows XP Embedded.

☞ Windows XP Embedded users manual "3.1 Write filter setting process"

◆ **Transferring a Project to a IPC or PC/AT Compatible Machine**

- Transferring with USB cable/Ethernet cable

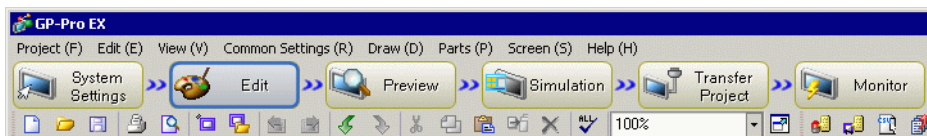


1 From the [Start] menu, point to [Programs], [Pro-face], [WinGP], or double-click  on the desktop to start WinGP.



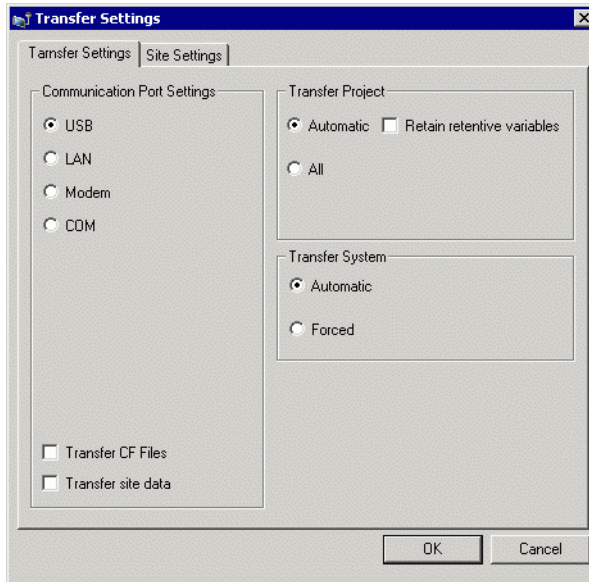
NOTE • You cannot transfer files when the Screen Offline message is displayed. Be sure WinGP is online.

2 On the GP-Pro EX state tool bar, click [Transfer Project] to launch the transfer tool.



3 Confirm the project details in [Project Information]. To transfer a different project file, click the [Select Project] button and select the project file.

- 4 In the [Transfer Settings] area, confirm that the USB or LAN option is selected. If neither [USB] nor [LAN] is selected, display the [Transfer Settings] dialog box and in the [Communication Port Settings], select either [USB] or [LAN] and click [OK].



NOTE

- Modem transfer is not available.
-

- 5 Click [Send Project].

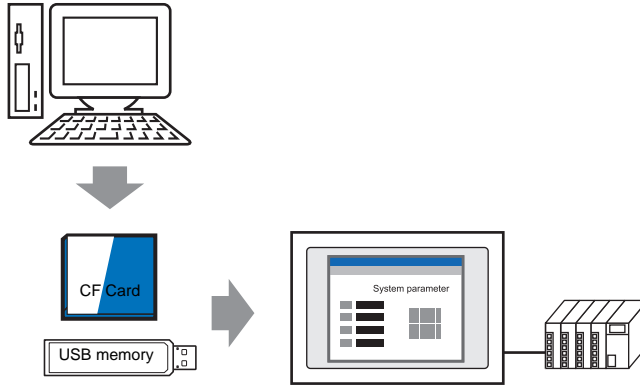
The following process is the same as the GP. See below.

- ☞ "33.2 Transferring Project Files via USB Transfer Cable" (page 33-5)
- ☞ "33.3 Transferring Project Files via Ethernet (LAN)" (page 33-12)

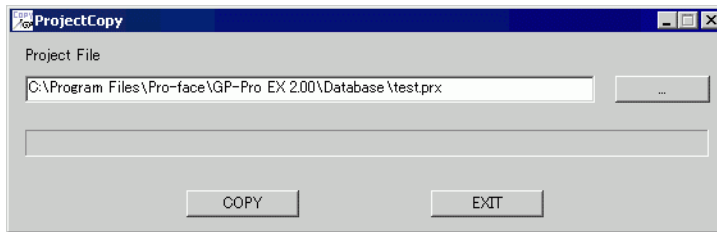
NOTE

- When transferring via Ethernet (LAN), make sure you set up the IPC or PC/AT compatible machine's IP address. On the Desktop, right-click My Network and select Properties. In the [Network Connections] dialog box, right-click [Local Area Connection] and define the IP address in [Internet Protocol (TCP/IP)]. You cannot define the IP address in the WinGP offline menu.
-

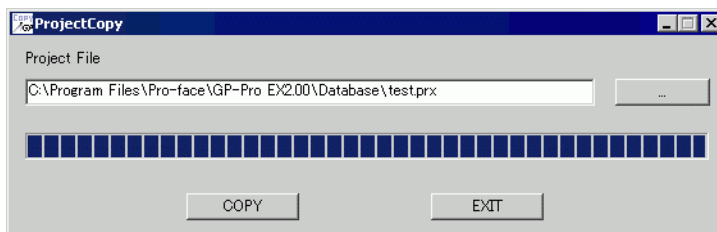
- Transferring from CF Card or USB storage



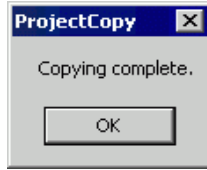
- 1 Exit WinGP. You cannot transfer projects while WinGP is running.
- 2 On the [Start] menu, click [Programs], point to [Pro-face], [WinGP], and then click [Project Copy] to launch the project copy tool.



- 3 [Click the [Project File] **Browse** icon, and specify the GP-Pro EX project file (*.prx) stored in the CF Card, USB storage, or desktop.
- 4 [Click [Copy]. The following dialog box appears during transfer.



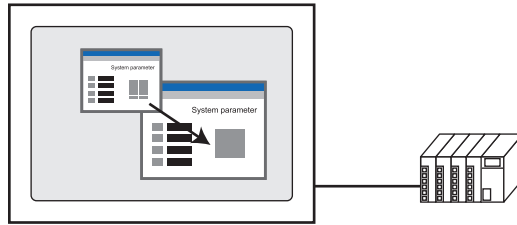
5 When copying is complete, the following message appears. Click [OK].




NOTE

- Only screen data transfer is available using Project Copy [Copy Tool]. Receiving screen data or full transfer of project is not available. In the following cases, please use the Transfer Tool.
 - The first time you transfer the project after installing WinGP
 - Change or add a Device/PLC
 - Change or add a font
 - After upgrading GP-Pro EX, the run-time system or protocol driver is updated and you update the project.
 - You cannot send the WinGP system program using the Copy Tool. Please use the Transfer Tool when you upgrade WinGP.
-

◆ Installing GP-Pro EX and WinGP on the same IPC or PC/AT Compatible Machine



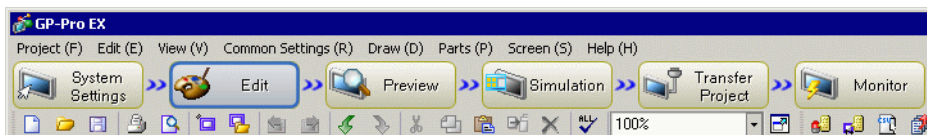
1 From the [Start] menu, point to [Programs], [Pro-face], [WinGP], and then click [WinGP]. Or double-click  on the desktop.



NOTE

- You cannot transfer files when the Screen Offline message is displayed. Be sure WinGP is online.

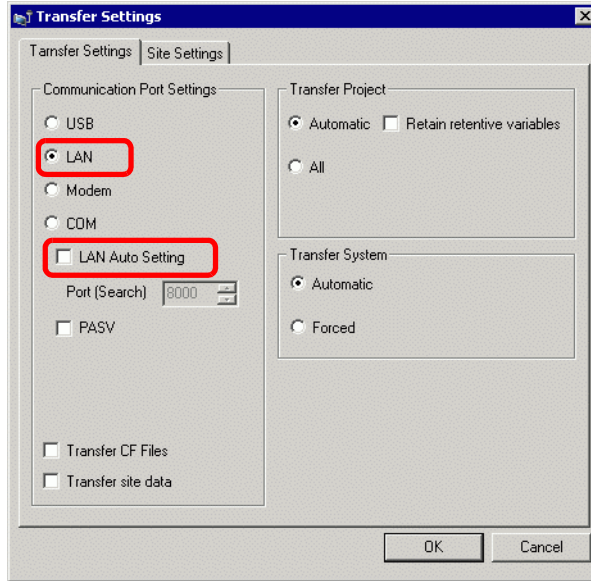
2 On the GP-Pro EX state tool bar, click [Transfer Project] to launch the transfer tool.



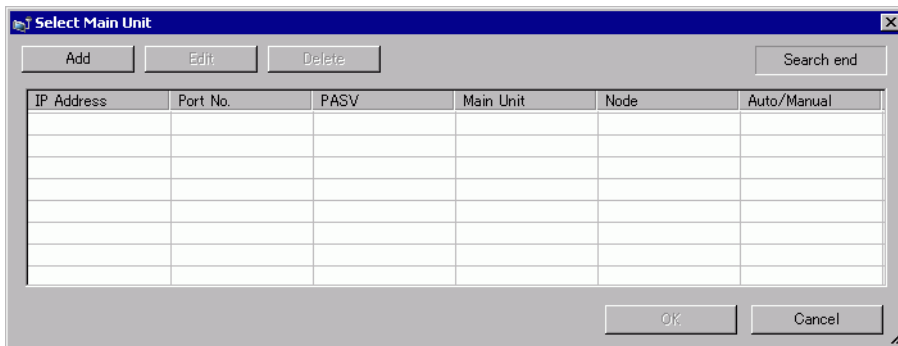
3 Confirm the project details in [Project Information]. To transfer a different project file, click the [Select Project] button and select the project file.

4 Click the [Transfer Settings] button.

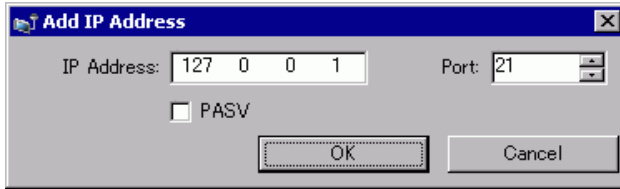
In [Communication Port Settings], select the [LAN] option. Clear the [Automatic] check box and click [OK].



5 Click [Send a Project]. The [Select Display Unit] dialog box appears.



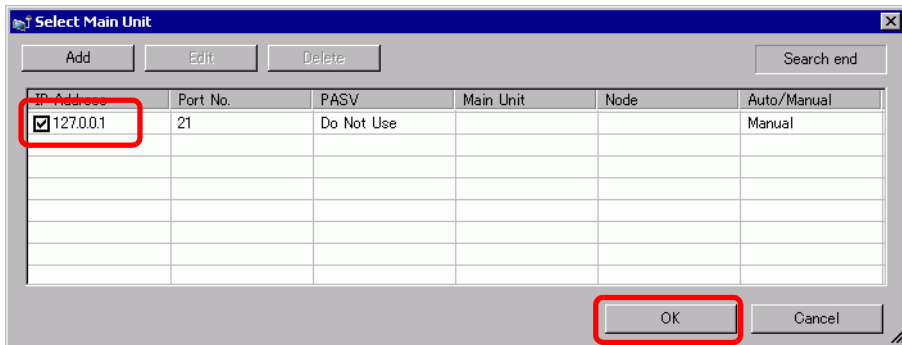
6 Click [Add]. Enter [127.0.0.1] in [IP Address] and click [OK].



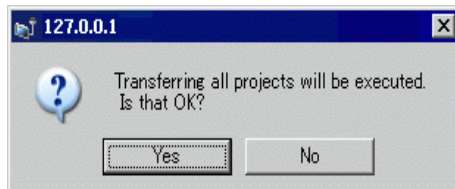
NOTE

- The IP address [127.0.0.1] is a virtual address that indicates the computer you are currently using on the network.
- Make sure the [Port] number matches the port number defined in the [System Settings] window [Display Unit] page, [IPC Settings] tab [Transfer Settings] area.

7 Select the [127.0.0.1] check box displayed in [IP Address] and click [OK].



8 When the following dialog box appears, click [Yes]. (The dialog box will not appear if you transfer the same project again.)



Project file transfer is available using [Project Copy] (Copy Tool). For information on the setup procedure, see the following.

- ☞ "• Transferring from CF Card or USB storage" (page 38-23)

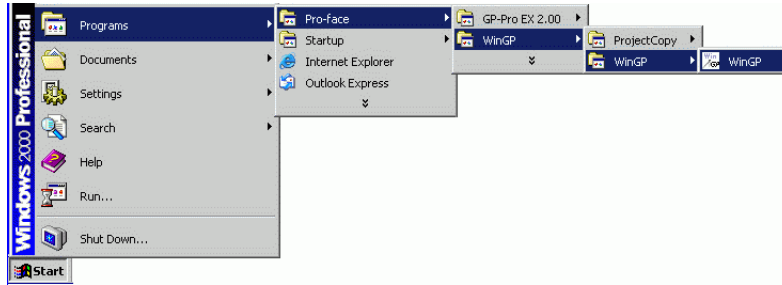
■ Start/Run WinGP

1 Connect the IPC or PC/AT compatible machine to the device/PLC.

NOTE

- Please refer to the "GP-Pro EX Device/PLC Connection Manual" about communication settings and connection cables.

2 On the [Start] menu, from [Programs], point to [Pro-face], [WinGP], and then click [WinGP].


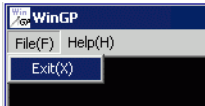
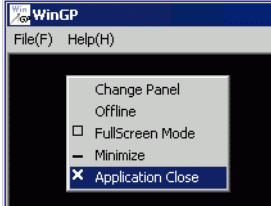


NOTE


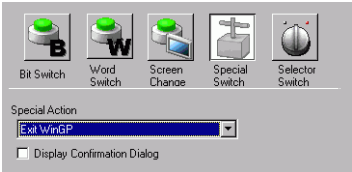

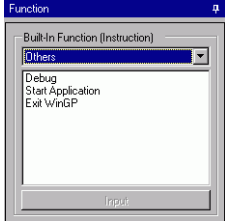
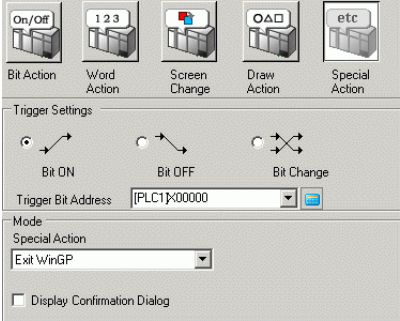

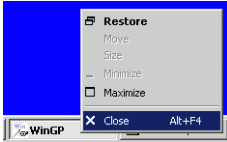

- Alternatively, double-click the shortcut on the desktop screen.

■ Exit WinGP

Exit WinGP. The following shows nine ways to exit WinGP.

1	Click the [Close] button on the title bar.	 <p>A close-up of the top-right corner of a Windows window. The title bar is blue and contains three buttons: minimize, maximize, and close. A mouse cursor is clicking the close button, which is labeled 'Close'.</p>
2	From the [File] menu, select [Exit].	 <p>A screenshot of the WinGP application window. The 'File(F)' menu is open, showing 'Exit(X)' as the selected option.</p>
3	Right-click [Close Application] on the WinGP screen.	 <p>A screenshot of the WinGP application window with a context menu open over the main area. The menu items are: Change Panel, Offline, FullScreen Mode (with a checkbox), Minimize, and Application Close (with a mouse cursor over it).</p> <p style="text-align: center;">Right-click</p>

Continued

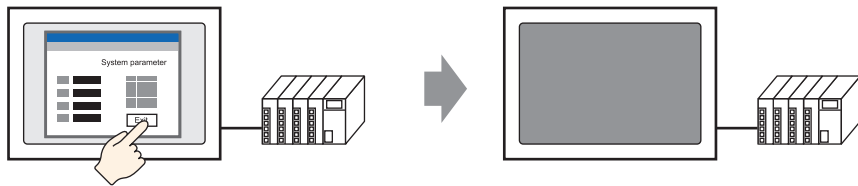
<p>4</p>	<p>Use the switch part to exit.  " ◆ Use switch parts to exit" (page 38-30)</p>	
<p>5</p>	<p>Use D-Script to exit.  " ◆ Use D-Script to exit" (page 38-32)</p>	
<p>6</p>	<p>Use the trigger action to exit.</p>	
<p>7</p>	<p>Press "Alt+F4" on the keyboard.</p>	
<p>8</p>	<p>Right-click the task bar and click [Close].</p>	
<p>9</p>	<p>Use API to exit.  " • Exit Operation ◆ Function List" (page 38-94)</p>	<p>API name: StopRuntime()</p>


◆ **Use switch parts to exit**

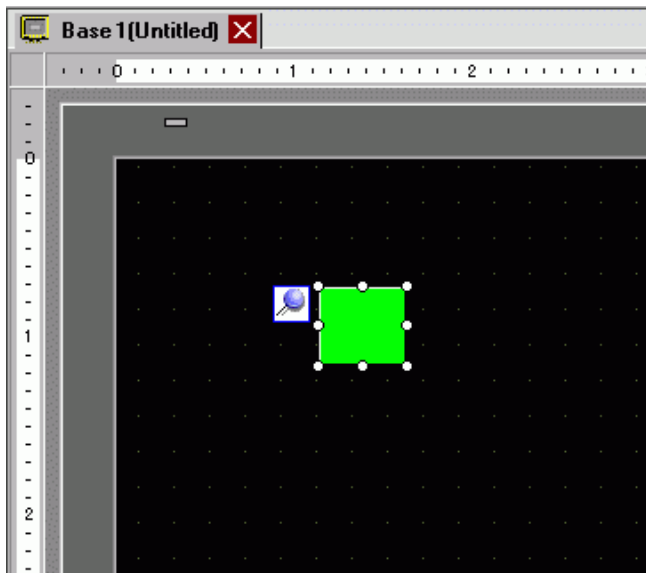
Create a switch to exit WinGP.

NOTE

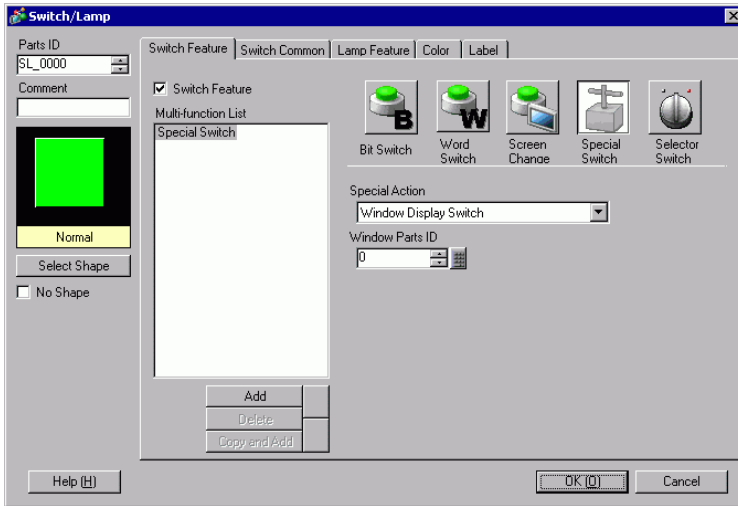
- Please refer to the Settings Guide for details.
 ☞ "10.15 Switch Lamp Parts Settings Guide" (page 10-48)
- For details of the part placement method and the address, shape, color, and label setting method, refer to the "Part Editing Procedure".
 ☞ "8.6.1 Editing Parts" (page 8-44)



- 1 On the [Part (P)] menu, point to [Switch Lamp (C)], and then click [Special Switch (P)], or click  on the tool bar to place the switch.



2 Double-clicking the Switch part opens the Settings dialog box.

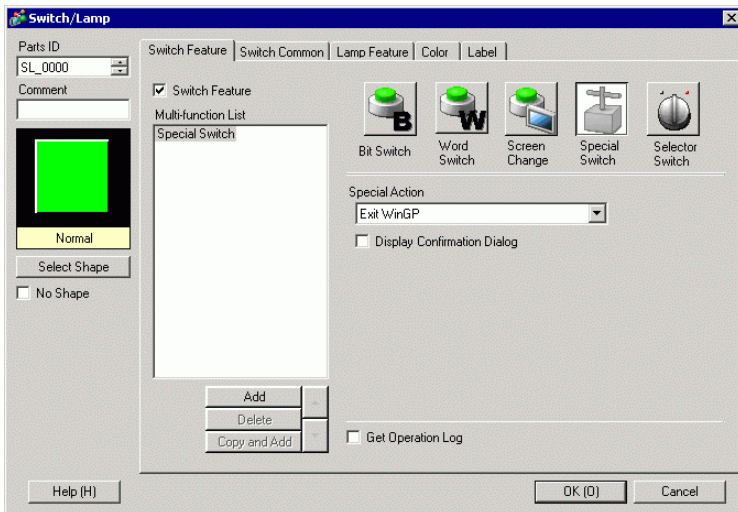


3 In [Select Shape], select the Switch shape.

NOTE

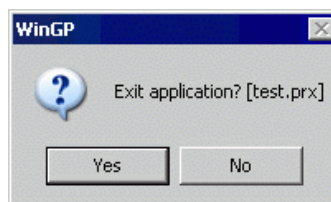
- Some switch shapes do not allow you to change the color.

4 In [Special Action], select [Exit WinGP].



NOTE

- If you select the [Confirm] check box, the following message appears when you touch the switch on the WinGP.

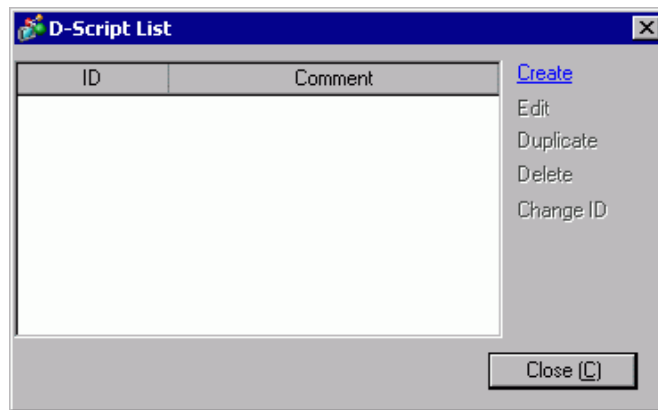


◆ Use D-Script to exit

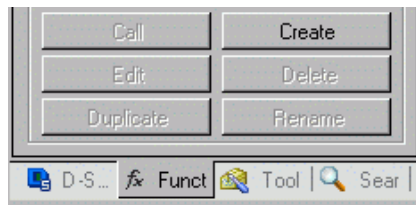
NOTE

- Please refer to the Settings Guide for details.
 ☞ "21.9.1 D-Script/Common [Global D-Script] Settings Guide" (page 21-54)
- On the [Common Settings (R)] menu, you can also select [Global D-Script (L)] or [Extended Script (E)] to exit WinGP.

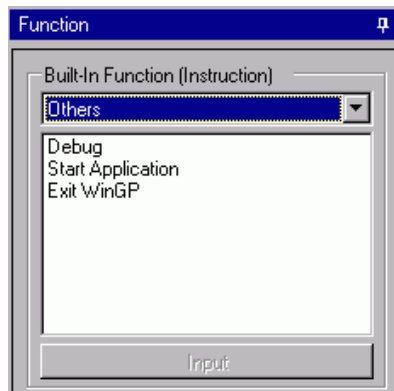
- 1 On the [Parts (P)] menu, select [D-Script (R)] and in the [D-Script list] dialog box Click [Create].



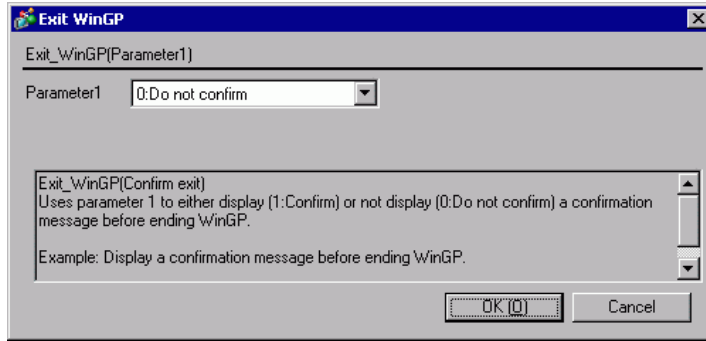
- 2 Click the [Function] tab. Simply click the instruction available to the script to easily place the [Built-In Function (Instruction)].

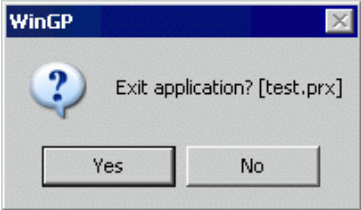


- 3 On the [Built-In Function (Instruction)] pull-down menu, click [Others].

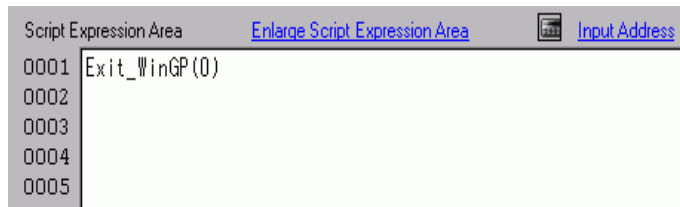


4 Double-click [Exit WinGP] and configure the parameter settings in the dialog box below.

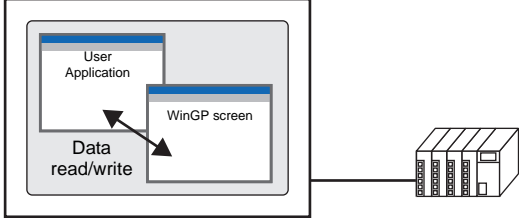
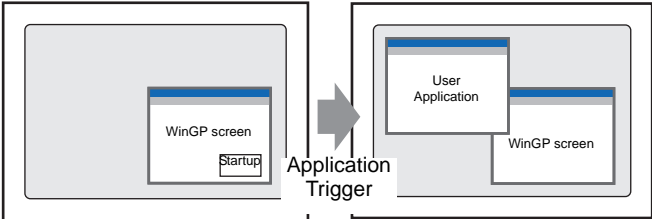
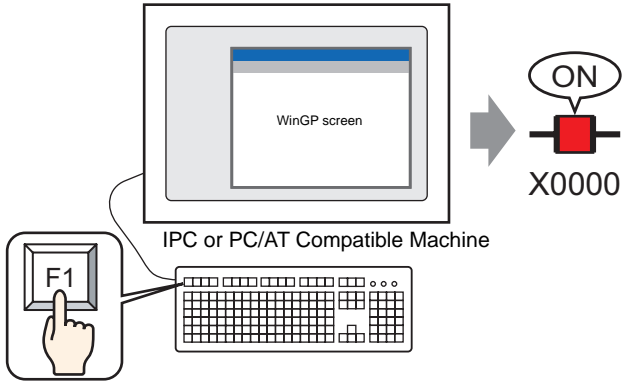


Parameter 0	0: Do not confirm	The confirmation dialog box does not appear and WinGP exits immediately.
Parameter 1	1: Confirm	The following dialog box appears in the WinGP. Click [Yes] to exit WinGP. 

5 Click [OK (O)] to view "Exit_WinGP (0)" or "Exit_WinGP (1)" in [Script Expression Area].

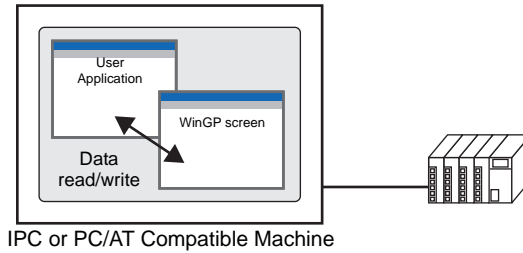


38.4 Settings Menu

Retrieve WinGP information or Operate WinGP from user application	
<p>API allows the operations such as Read/Write between the WinGP and user applications.</p>  <p>IPC or PC/AT Compatible Machine</p>	<ul style="list-style-type: none"> ☞ Setup Procedure (page 38-36) ☞ Details (page 38-35)
Running the Application from WinGP	
<p>On the WinGP screen, you can execute other applications.</p>  <p>IPC or PC/AT Compatible Machine IPC or PC/AT Compatible Machine</p>	<ul style="list-style-type: none"> ☞ Switch Startup Settings (page 38-74) ☞ Details (page 38-73)
Allocating the switch feature to the function key	
<p>Press the function key on the keypad while WinGP is running to operate the switch feature.</p>  <p>IPC or PC/AT Compatible Machine</p>	<ul style="list-style-type: none"> ☞ Setup Procedure (page 38-80) ☞ Details (page 38-79)

38.5 Retrieve WinGP information or Operate WinGP from user application

38.5.1 Details

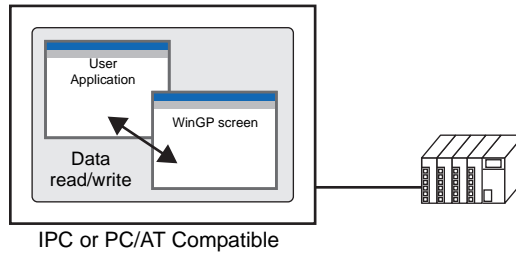


API allows to retrieve WinGP information or operate WinGP from user applications.

38.5.2 Setup Procedure

NOTE

- Please refer to the Settings Guide for details.
☞ "38.10.1 System Settings [Display Unit Settings] [IPC Settings] Settings Guide" (page 38-164)

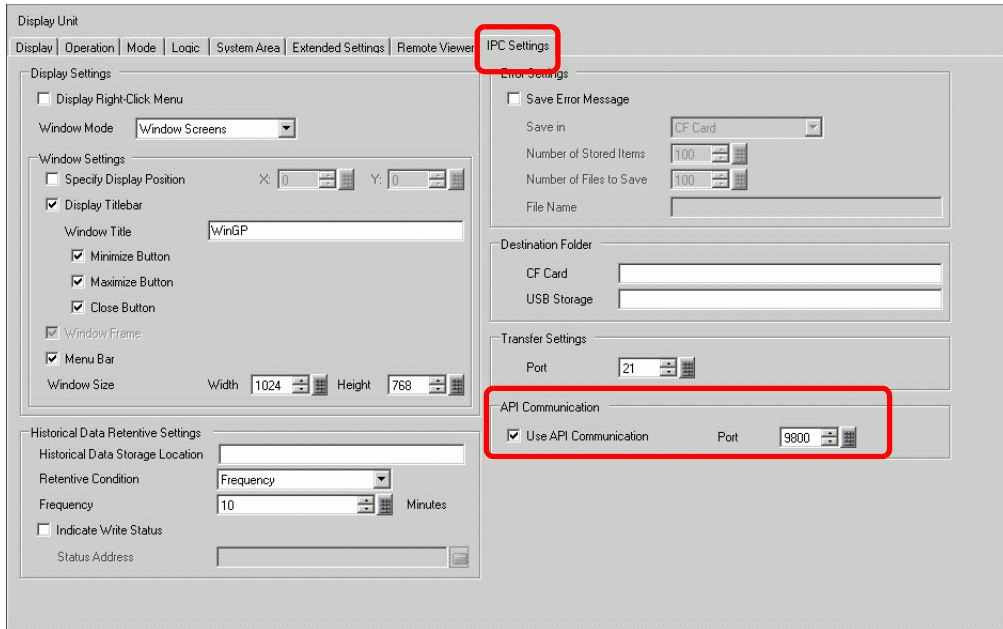


API allows to retrieve WinGP information or operate WinGP from user applications.

- 1 In the GP-Pro EX [System Settings] window, click [Display Unit].



- Open the [IPC Settings] tab and select the [API Communication] check box to specify the port to enable from 0 to 65535. Define a value that differs from the [Transfer Settings] [Port] number.



NOTE

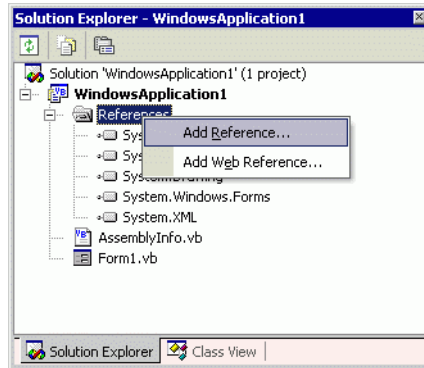
- Be sure not to use the same port as that for communication with the other device/PLC or for FTP communication.
- Please do not use port numbers 8000 to 8019, which are designated as the port numbers for transfer.

- Save the Project file, and transfer it to the IPC or PC/AT compatible machine.
- Acknowledge the communication between WinGP and the device/PLC.

5 Set up a programming application to use API.

When using the device access API in VB.NET

Open the solution explorer in VB.NET and right-click [Browse Settings] to select [Add Browse].



Click [Browse] in the [Add Browse] dialog box and select the following file.

(In GP-Pro EX CD-ROM)\WinGP\SDK\Pro-SDK\DotNet\BIN\WinGPAPIDotNet.dll

Click [Open] and select [OK].

At the top of the source code, enter "Imports ProEasyDotNet."

When using device access API in VB6

From VB6 menu bar, select [Project] -[Add Standard Module] and add the following module.

(In GP-Pro EX CD-ROM) \WinGP\SDK\Pro-SDK\VB\API\WinGPAPI.bas

When using handling API in VB.NET

From VB.NET Menu Bar, select [Project]-[Add Existing Item] and add the following module.

(In GP-Pro EX CD-ROM) \WinGP\SDK\Pro-SDK\DotNet\BIN\RtCtrlAPI.vb

When using handling API in VB6

From VB6 menu bar, select [Project] -[Add Standard Module] and add the following module.

(In GP-Pro EX CD-ROM)\WinGP\SDK\Pro-SDK\VB\API\RtCtrlAPI.bas

6 Execute programming.

NOTE

☞ "38.5.3 Samples of Read/Write data (device access API) ■ Sample Summary" (page 38-39)

☞ "38.5.4 Sample to retrieve the WinGP status and change the settings (Handling API) ■ Sample Summary" (page 38-57)

7 Setup the user application created on the IPC or PC/AT compatible machine.

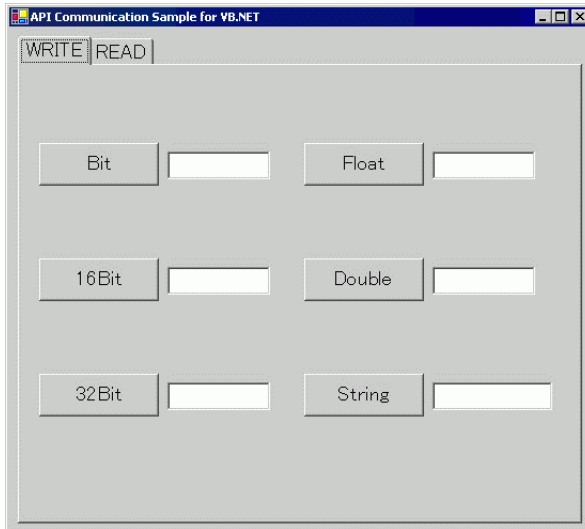
8 Start WinGP and the user application.

38.5.3 Samples of Read/Write data (device access API)

This section explains the program for API communication using the sample application as shown below.

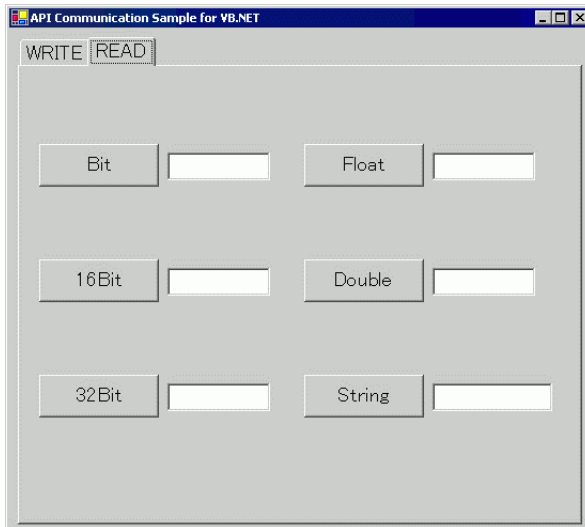
■ Sample Summary

- WRITE



Click the button to write the input data into the text box.

- READ



Click the button to read the data into the text box.

The sample uses the following symbols as examples.


Symbol Name	Address browsed by the symbol
Buf_Bit	The USR 200.00 bit
Buf_16	USR201
Buf_32	USR203
Buf_Float	USR207
Buf_Double	USR209
Buf_Str	USR213

■ How to specify device addresses directly

- When one Device/PLC driver is specified for WinGP
WriteDeviceBit("#WinGP", "M100", nDataAry(0), 1)
- When more than one device/PLC driver is specified for WinGP
WriteDeviceBit("#WinGP.PLC1", "M100", nDataAry(0), 1)
Device/PLC name connected to WinGP
- When using the memory link driver
WriteDeviceBit("#WinGP.#MEMLINK", "10000", nDataAry(0), 1)
- When using WinGP Internal Device
WriteDeviceBit("#WinGP", "USR10000", nDataAry(0), 1)
WriteDeviceBit("#WinGP", "LS10000", nDataAry(0), 1)
Or
WriteDeviceBit("#WinGP.#INTERNAL", "USR10000", nDataAry(0), 1)
WriteDeviceBit("#WinGP.#INTERNAL ", "LS10000", nDataAry(0), 1)

■ VB .NET 2003 Program Example

Sample Program Location: (In GP-Pro EX CD-ROM)\WinGP\SDK\Pro-SDK\DotNet\EasySmpl


Imports ProEasyDotNet  Imports ProEasy object.

Public Class Form1
Inherits System.Windows.Forms.Form

#Region "code generated with Windows form designer"

Public Sub New()
MyBase.New()

' This call is necessary for Windows form designer.
InitializeComponent()

' ProEasy Initialization  After calling InitializeComponent(), runs initialization.

Dim iResult As Integer = ProEasy.EasyInit()' WinGP Initialize SDK once at the beginning


```

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

```

```
End Sub
```

' Form overwrites the dispose to execute post processing on the component list.
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

```

- Snip (Codes designed by Windows form designer are omitted hereafter) -

```
#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
Read 16. Click

```

```
Try
```

```

' Read data.
Dim nDataAry (1) As Short

```

```
'Read
```

```
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))
```

Here the symbol "Buf_16" (USR201) configured in GP-Pro EX is used.
You can also specify the device address directly.
 " ■ How to specify device addresses directly" (page 38-40)

```
    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
```

```
    ' Read data.  
    Dim nDataAry (1) As Integer
```

```
    'Read.  
    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
```

```
    ' Read data.  
    Dim nDataAry (1) As Short
```

```
    'Read  
    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
        ' Read data.
        Dim nDataAry (1) As Integer

        'Read
        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
    ' Read data.
    Dim nDataAry (1) As Single

    'Read
    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
        ' Read data.
        Dim nDataAry (1) As Double

        'Read
        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 = CDb1 (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
```

```
    ' Read data.
```

```
    Dim nDataAry As String
```

```
    'Read
```

```
    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
    ' Write data.
    Dim nDataAry (1) As Short
    nDataAry (0) = CShort (Val(Me.WBuf_Bit.Text))

    'Write
    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
    ' Write data.
    Dim nDataAry (1) As Short
    nDataAry (0) = CShort (Val(Me.WBuf_16.Text))

    'Write
    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
    ' Write data.
    Dim nDataAry (1) As Integer
    nDataAry (0) = CInt (Val(Me.WBuf_32.Text))

    'Write
    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
    ' Write data.
    Dim nDataAry (1) As Short
    nDataAry (0) = CShort (Val("&h" + Me.WBuf_BCD16.Text))

    'Write
    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
```

```
' Write data.
```

```
Dim nDataAry (1) As Integer
```

```
nDataAry (0) = CInt (Val("&h" + Me.WBuf_BCD16.Text))
```

```
'Write
```

```
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
```

```
' Write data.
```

```
Dim nDataAry (1) As Single
```

```
nDataAry (0) = CSng (Val(Me.WBuf_Float.Text))
```

```
'Write
```

```
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  
    ' Write data.  
    Dim nDataAry (1) As Double  
    nDataAry (0) = CDb1 (Val(Me.WBuf_Double.Text))  
  
    'Write  
    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  
    ' Write data.  
    Dim nDataAry As String  
    nDataAry = Me.WBuf_Str.Text  
  
    'Write  
    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

    ' In VB.NET, Variant type is no longer used. Instead Object type is used.
    'Along the change, WriteDeviceVariant() has been
    'changed to WriteDeviceEasyObject()

End Sub

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

    'Only WriteSymbol system found is WriteSymbolVariant().

End Sub

End Class
```

■ VB6 Program Example

Sample Program Location:(In GP-Pro EX CD-ROM)\WinGP\SDK\Pro-SDK\VB\EasySmpl

NOTE • The sample VB6 program does not support Windows Vista®. It will not run in a Vista environment.

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()

' Write data.

Dim nDataAry (1) As Integer

nDataAry (0) = CInt (Val(Me.WBuf_Bit.Text))

'Write

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

Here the symbol "Buf_16" (USR201) configured in GP-Pro EX is used.

You can also specify the device address directly.

 ■ How to specify device addresses directly" (page 38-40)

```
Private Sub Write16_Click()
```

```
    ' Write data.
```

```
    Dim nDataAry (1) As Integer
```

```
    nDataAry (0) = CInt (Val(Me.WBuf_16.Text))
```

```
    'Write
```

```
    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()
```

```
    ' Write data.
```

```
    Dim nDataAry (1) As Long
```

```
    nDataAry (0) = CLng (Val(Me.WBuf_32.Text))
```

```
    'Write
```

```
    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()
```

```
    ' Write data.
```

```
    Dim nDataAry (1) As Single
```

```
    nDataAry (0) = CSng (Val(Me.WBuf_Float.Text))
```

```
    'Write
```

```
    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()
```

```

' Write data.
Dim nDataAry (1) As Double
nDataAry (0) = CDb1 (Val(Me.WBuf_Double.Text))

'Write
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()
```

```

' Write data.
Dim nDataAry As String
nDataAry = Me.WBuf_Str.Text

'Write
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()

    ' Read data.
    Dim nDataAry (1) As Integer

    'Read
    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()

    ' Read data.
    Dim nDataAry (1) As Integer

    'Read
    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()

    ' Read data.
    Dim nDataAry (1) As Long
```

```
'Read
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()

' Read data.
Dim nDataAry (1) As Single

'Read
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()
```

```
    ' Read data.
```

```
    Dim nDataAry (1) As Double
```

```
    'Read
```

```
    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()
```

```
    ' Read data.
```

```
    Dim nDataAry As String * 255
```

```
    'Read
```

```
    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
```

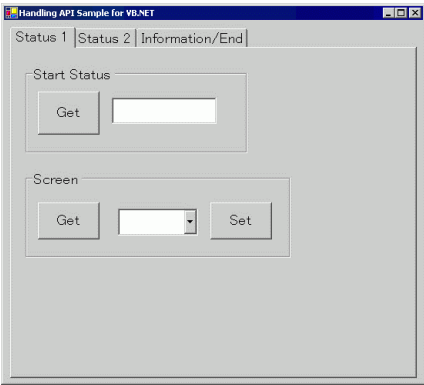
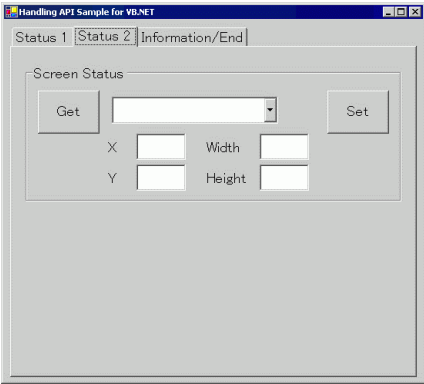
```
    Me.Buf_Str.Text = nDataAry
```

```
End Sub
```

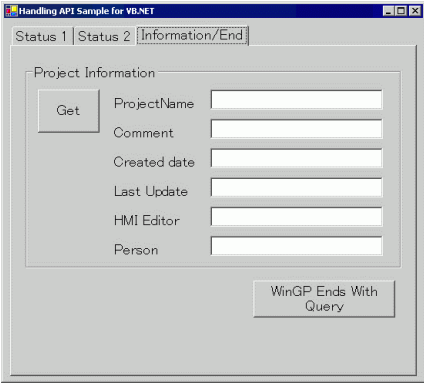

38.5.4 Sample to retrieve the WinGP status and change the settings (Handling API)

■ Sample Summary

Switching the tabs from [Status 1] to [Information/End] allows you to retrieve the WinGP status and change the settings.

<p>[Status 1] tab</p> 	<p>In the [Start Status] area, click the [Get] button. The WinGP startup state is displayed as one of the six shown below.</p> <ul style="list-style-type: none"> • Starting • Offline • Online • Transfer mode • Ending • Not executing <p>In the [Screen] area, click the [Get] button to display the screen number currently displayed in WinGP. Also, the screens available for display in WinGP are listed in the ComboBox. In the list, select the screen you are switching to and click the [Set] button to switch the screen displayed in WinGP.</p>
<p>[Status 2] tab</p> 	<p>In the [Screen Status] area, click the [Get] button. The WinGP display state is displayed in the ComboBox as one of the three shown below.</p> <ul style="list-style-type: none"> • Maximized (Full screen) • Window screen • Minimized <p>Change the display in the ComboBox and click the [Set] button to change the display state. Settings for X and Y coordinates, Width, and Height are available only in the Window mode.</p>

[Information/End] tab



In [Project Information] on the top left, click the [Get] button. This displays the information below displayed in WinGP.

ProjectName	Project file name
Comment	Project title
Created date	Project creation date
Last Update	Project last update date
HMI Editor	GP-Pro EX version
Person	Creator

[WinGP Ends With Query] button displays a confirmation message asking "Do you want to exit?". Click [Yes] to exit WinGP.

■ VB .NET 2003 Program Example

Sample Program Location: (In GP-Pro EX CD-ROM)\WinGP\SDK\Pro-SDK\DotNet\RtCtrlSmpl

Imports

System.Runtime.InteropServices — Imports System.Runtime.InteropServices.

Public Class Form1

Inherits System.Windows.Forms.Form

Dim ghWinGP As Int32 = 0' API handle.

#Region "code generated with Windows form designer"

Public Sub New()
MyBase.New()

' This call is necessary for Windows form designer.

InitializeComponent()

After calling InitializeComponent(), runs initialization.

'Initialize API (API).

Dim nResult As Integer = InitRuntimeAPI()

'Gets the handle at this stage (API).

ghWinGP = GetRuntimeHandle (9800)

If ghWinGP = 0 Then

MsgBox ("(API) Failed to get handle.")

End If

End Sub

' Form overwrites the dispose to execute post processing on the component list.
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

```

- Snip (Codes designed by Windows form designer are omitted hereafter) -

#End Region

' 5 Gets the startup state.

Private Sub Bt_GetStartState_Click (ByVal sender As System.Object, ByVal e As System.EventArgs)

Handles Bt_GetStartState.Click

Me.Cursor = Cursors.WaitCursor 'Changes the cursor to an hourglass.

Try

'Gets the state (API).

Dim Status As Int32

Dim RetVal As Int32 = GetRuntimeStartState(ghWinGP, Status)

'Any error?

If RetVal <> API_ERROR.E_SUCCESS Then

MsgBox ("Err(" + Str(RetVal).Trim() + "):GetRuntimeStartState()")

End If

'Display the state

Select Case Status

Case 0

Me.StartState.Text = "Starting"

Case 1

Me.StartState.Text = "Online"

Case 2

Me.StartState.Text = "Offline"

Case 3

Me.StartState.Text = "Transfer mode"

Case 4

Me.StartState.Text = "Ending"

Case 5

```
        Me.StartState.Text = "Not execute"  
    End Select  
  
    Catch ex As Exception  
        MsgBox(ex.Message)  
  
    End Try  
  
    Me.Cursor = Cursors.Default ' Changes the cursor back to the original.  
  
End Sub  
  
Private Sub GetScreenState_Click (ByVal sender As System.Object, ByVal e As  
System.EventArgs)  
Handles BT_GetScreenState.Click  
  
    Me.Cursor = Cursors.WaitCursor 'Changes the cursor to an hourglass.  
  
    Try  
  
        'Gets the state.  
        Dim Status As Int32  
        Dim RetVal As Int32 = GetScreenState(ghWinGP, Status)  
  
        'Any error?  
        If RetVal <> API_ERROR.E_SUCCESS Then  
            MsgBox ("Err(" + Str(RetVal).Trim() + "):GetScreenState()")  
        End If  
  
        'Display the state  
        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 ' Changes the cursor back to the original.  
  
End Sub
```

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

Handles BT_SetScreenState.Click

Me.Cursor = Cursors.WaitCursor 'Changes the cursor to an hourglass.

Try

'Gets the value

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)

'Screen state settings.

Dim RetVal As Int32 = SetScreenState(ghWinGP, State, PosX, PosY, Width, Height)

'Any error?

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 ' Changes the cursor back to the original.

End Sub

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

Handles GetDispScreen.Click

Me.Cursor = Cursors.WaitCursor 'Changes the cursor to an hourglass.

Dim CurScrNo As Int32 ' Screen number currently displayed

Try

'Gets the state.

Dim RetVal As Int32 = GetDisplayScreenNumber(ghWinGP, CurScrNo)

```

'Any error?
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

    ' Gets the number of screens.
    Dim ScreenCount As Int32 = 0
    Dim RetVal As Int32 = GetEnumScreenNumberCount(ghWinGP, ScreenCount)

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

    ' Gets the screen number.
    If ScreenCount > 0 Then

        'Gets the screen number.
        Dim ScreenNumber (ScreenCount - 1) As Int32
        RetVal = EnumScreenNumber(ghWinGP, ScreenCount, ScreenNumber (0))

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

        ' -----Display the state-----

        'Delete all.
        Me.CB_DispScreen.Items.Clear()

        'Set the screen number you got.
        Dim idx As Int32
        For idx = 0 To ScreenNumber.Length - 1
            Me.CB_DispScreen.Items.Add (ScreenNumber (idx))
        Next
    End If
End Try

```

```
'Display the screen number currently displayed.
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 ' Changes the cursor back to the original.

End Sub

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

    Me.Cursor = Cursors.WaitCursor 'Changes the cursor to an hourglass.

Try

    ' Gets the screen number.
    Dim Screen As Int32
    Screen = Val(Me.CB_DispScreen.Text)

    'Changes the screen number.
    Dim RetVal As Int32 = SetDisplayScreenNumber(ghWinGP, Screen)

    'Any error?
    If RetVal <> API_ERROR.E_SUCCESS Then
        MsgBox ("Err(" + Str(RetVal).Trim() + "):SetDisplayScreenNumber()")
    End If
```

'Gets the screen number again and compare it with the set value to see whether the screen number has been changed successfully.

Dim NowScrNo As Long

RetVal = GetDisplayScreenNumber(ghWinGP, NowScrNo)

If RetVal = API_ERROR.E_SUCCESS Then

 If NowScrNo = Screen Then

 'MsgBox ("Screen change number = No=" + Str(NowScrNo))

 End If

End If

Catch ex As Exception

 MsgBox(ex.Message)

End Try

Me.Cursor = Cursors.Default ' Changes the cursor back to the original.

End Sub

Private Sub GetProjectInfo_Click (ByVal sender As System.Object, ByVal e As System.EventArgs)

Handles GetProjectInfo.Click

Me.Cursor = Cursors.WaitCursor 'Changes the cursor to an hourglass.

Try

 'Parameter range to get.

 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)


```

'Gets the project information.
Dim RetVal As Int32
RetVal = GetProjectInformation(ghWinGP, _
    ProjectFileName, _
    ProjectComment, _
    ProjectFastTime, _
    ProjectLastTime, _
    ProjectIDownload, _
    HMIEditorVersion, _
    ControlEditorVersion, _
    MakingPerson)

'Any error?
If RetVal <> API_ERROR.E_SUCCESS Then
    MsgBox ("Err(" + Str(RetVal).Trim() + "):GetProjectInformation()")
End If

'Display the information you got.
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 'Changes the cursor back to the original.

End Sub

' 13 Exit.
'Exit following a confirmation dialog.
'WinGP does not end if you select "Do not exit" in the dialog box
'You can go back to the return value (NULL) with API_ERROR.E_SUCCESS.

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

    Me.Cursor = Cursors.WaitCursor 'Changes the cursor to an hourglass.

Try

```

```
'Exit (API).
Dim RetVal As Int32 = StopRuntime(ghWinGP, 1)

'Any error?
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 ' Changes the cursor back to the original.

End Sub

End Class
```

■ VB6 Program Example

Sample Program Location: (In GP-Pro EX CD-ROM) \WinGP\SDK\Pro-SDK\VB\RtCtrlSmpl

NOTE

- The sample VB6 program does not support Windows Vista®. It will not run in a Vista environment.
- The sample program executable file operates properly on Japanese and English operating systems only. To run the executable file in other operating system environments, re-create the executable file in that operating system environment.

Option Explicit

Private Sub Form_Load()

'Initialize API (API).

Dim nResult As Long

nResult = InitRuntimeAPI

'Gets the handle at this stage (API).

ghWinGP = GetRuntimeHandle (9800)

If ghWinGP = 0 Then

 MsgBox ("(API) Failed to get handle")

End If

End Sub

Private Sub Bt_GetStartState_Click()

Screen.MousePointer = vbHourglass

'Gets the state (API).

Dim Status As Long

Dim RetVal As Long

RetVal = GetRuntimeStartState(ghWinGP, Status)

'Any error?

If RetVal <> CLng(API_ERROR.E_SUCCESS) Then

 MsgBox ("Err(" + Str(RetVal) + "):GetRuntimeStartState()")

End If

'Display the state

Select Case Status

 Case 0

 Me.StartState.Text = "Starting"

```
Case 1
    Me.StartState.Text = "Online"
Case 2
    Me.StartState.Text = "Offline"
Case 3
    Me.StartState.Text = "Transfer mode"
Case 4
    Me.StartState.Text = "Ending"
Case 5
    Me.StartState.Text = "Not execute"
End Select

Screen.MousePointer = vbDefault

End Sub

Private Sub BT_GetScreenState_Click()

    Screen.MousePointer = vbHourglass

    'Gets the state.
    Dim Status As Long
    Dim RetVal As Long
    RetVal = GetScreenState(ghWinGP, Status)

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

    'Display the state
    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 ' Changes the cursor to an hourglass.

    'Gets the value
    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)
```

```
'Screen state settings.
```

```
Dim RetVal As Long
RetVal = SetScreenState(ghWinGP, State, PosX, PosY, Width, Height)
```

```
'Any error?
```

```
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 ' Changes the cursor to an hourglass.
```

```
Dim CurScrNo As Long 'Screen number currently displayed.
```

```
'Gets the state.
```

```
Dim RetVal As Long
RetVal = GetDisplayScreenNumber(ghWinGP, CurScrNo)
```

```
'Any error?
```

```
If RetVal <> API_ERROR.E_SUCCESS Then
    MsgBox ("Err(" + Str(RetVal) + "):GetDisplayScreenNumber()")
End If
```

```
' Gets the number of screens.
```

```
Dim ScreenCount As Long
RetVal = GetEnumScreenNumberCount(ghWinGP, ScreenCount)
```

```
'Any error?
```

```
If RetVal <> API_ERROR.E_SUCCESS Then
    MsgBox ("Err(" + Str(RetVal) + "):GetEnumScreenNumberCount()")
```

End If

' Gets the screen number.

If ScreenCount > 0 Then

'Gets the screen number.

Dim ScreenNumber() As Long

ReDim ScreenNumber (ScreenCount - 1) As Long

RetVal = EnumScreenNumber(ghWinGP, ScreenCount, ScreenNumber (0))

'Any error?

If RetVal <> API_ERROR.E_SUCCESS Then

MsgBox ("Err(" + Str(RetVal) + "):EnumScreenNumber()")

End If

' -----Display the state-----

'Set the screen number you got.

Me.CB_DispScreen.Clear

Dim idx As Long

For idx = 0 To ScreenCount - 1

Me.CB_DispScreen.AddItem (ScreenNumber (idx))

Next

'Display the screen number currently displayed.

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 'Changes the cursor back to the original.

End Sub

Private Sub SetDispScreen_Click()

Screen.MousePointer = vbHourglass ' Changes the cursor to an hourglass.

' Gets the screen number.

Dim ScrNo As Long

ScrNo = Val(Me.CB_DispScreen.Text)

'Changes the screen number.

Dim RetVal As Long

RetVal = SetDisplayScreenNumber(ghWinGP, ScrNo)

'Any error?

If RetVal <> API_ERROR.E_SUCCESS Then

 MsgBox ("Err(" + Str(RetVal) + "):SetDisplayScreenNumber()")

End If

'Gets the screen number again and compare it with the set value to see whether the screen number has been changed successfully.

Dim NowScrNo As Long

RetVal = GetDisplayScreenNumber(ghWinGP, NowScrNo)

If RetVal = API_ERROR.E_SUCCESS Then

 If NowScrNo = ScrNo Then

 MsgBox ("Screen change number = No=" + Str(NowScrNo))

 End If

End If

Screen.MousePointer = vbDefault 'Changes the cursor back to the original.

End Sub

Private Sub GetProjectInfo_Click()

Screen.MousePointer = vbHourglass ' Changes the cursor to an hourglass.

'Parameter range to get.

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

'Gets the project information.

Dim RetVal As Long

RetVal = GetProjectInformation(ghWinGP, _

 ProjectFileName, _

 ProjectComment, _

 ProjectFastTime, _

 ProjectLastTime, _

 ProjectIDownload, _

```
    HMIEditorVersion, _  
    ControlEditorVersion, _  
    MakingPerson)
```

```
'Any error?
```

```
IfRetVal <> API_ERROR.E_SUCCESS Then
```

```
    MsgBox ("Err(" + Str(RetVal) + "):GetProjectInformation()")
```

```
End If
```

```
'Display the information you got.
```

```
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 'Changes the cursor back to the original.
```

```
End Sub
```

```
' 13 Exit
```

```
'Exits following the confirmation dialog box.
```

```
'WinGP does not end if you select "Do Not Exit" in the dialog box.
```

```
'You can go back to the return value (NULL) with API_ERROR.E_SUCCESS.
```

```
Private Sub StopWinGP_Q_Click()
```

```
    Screen.MousePointer = vbHourglass ' Changes the cursor to an hourglass.
```

```
'Exit (API).
```

```
Dim RetVal As Long
```

```
RetVal = StopRuntime(ghWinGP, 1)
```

```
'Any error?
```

```
IfRetVal <> API_ERROR.E_SUCCESS Then
```

```
    MsgBox ("Err(" + Str(RetVal) + "):StopRuntime()")
```

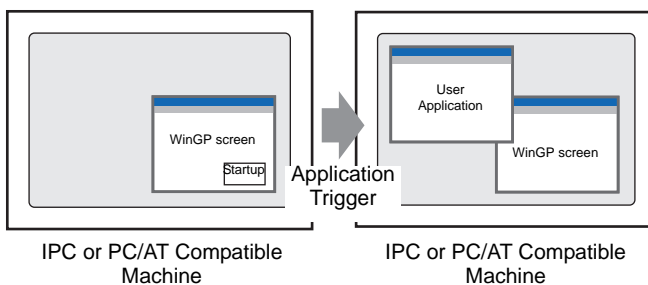
```
End If
```

```
Screen.MousePointer = vbDefault 'Changes the cursor back to the original.
```

```
End Sub
```


38.6 Running the Application from WinGP


38.6.1 Details

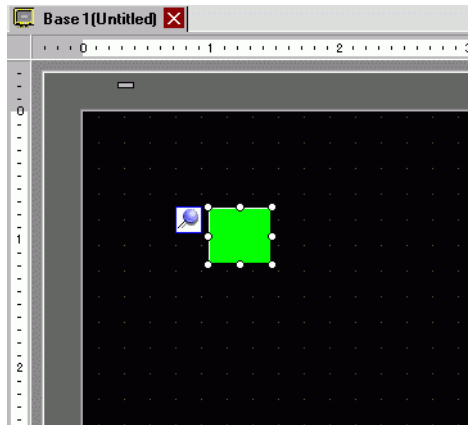


On the WinGP screen, you can execute other applications. There are four ways to execute applications as below.

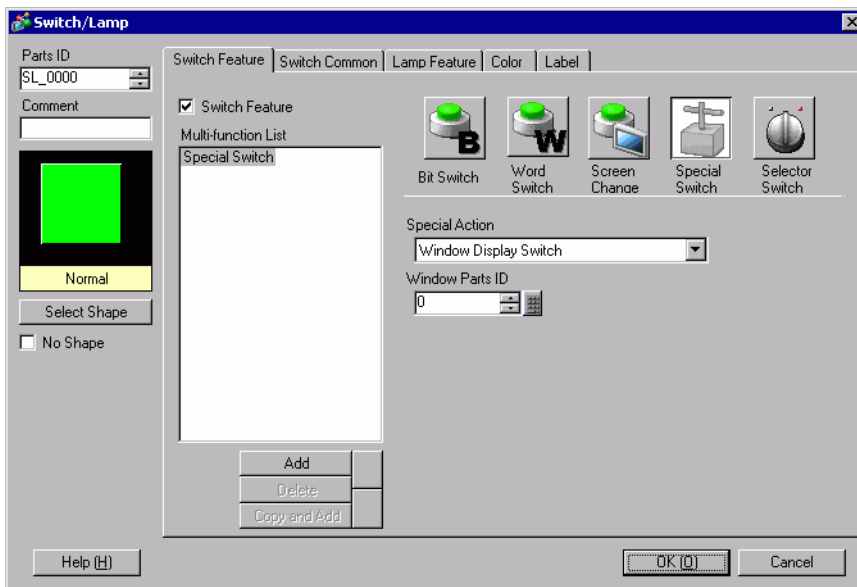
Using a switch for startup.	☞ "38.6.2 Switch Startup Settings" (page 38-74)
Using D-Script for startup.	☞ "38.6.3 D-Script startup settings" (page 38-77)
Startup on WinGP offline screen.	☞ [Maintenance/Troubleshooting]
Start up by trigger action.	

38.6.2 Switch Startup Settings

1 On the [Parts (P)] menu, select [Switch Lamp (C)] and [Special Switch (P)] or click  on the tool bar to place the switch on the screen.



2 Double-clicking the Switch part opens the Settings dialog box.

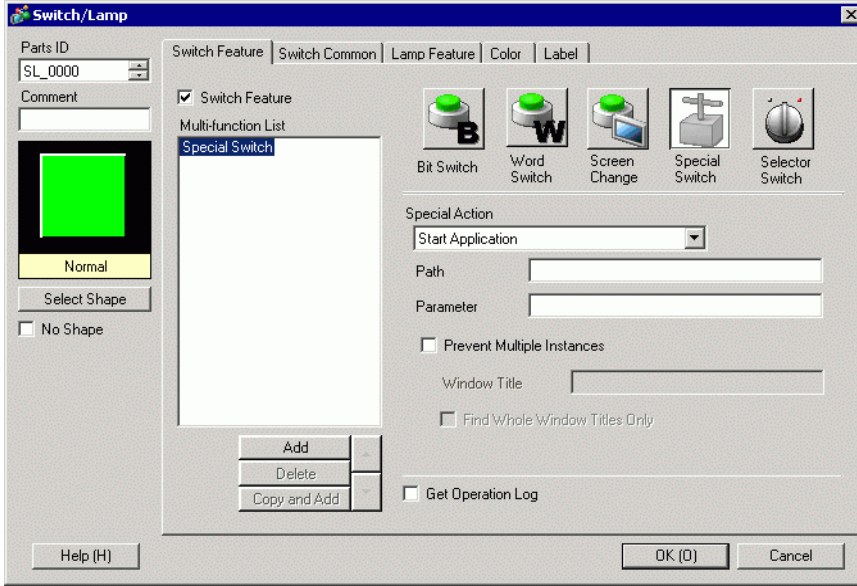


3 In [Select Shape], select the Switch shape.

NOTE

- Some switch shapes do not allow you to change the color.

4 In [Special Action], select [Start Application].



5 Enter [EXE path].

For example, execute sample.exe in C:\Documents and Settings\user\Local Settings\Temp

Specification Method	Example
Specify the full path	For example, C:\Documents and Settings\user\Local Settings\Temp\sample.exe
EXE name only	In the IPC or PC/AT compatible machine, use the [Control Panel] [System] [Detail] [Environment variables] to specify the executable files under System. For example, sample.exe (With an environment variable, specify the Path = C:\Documents and Settings\user\Local Settings\Temp.)
Specify the path with an environment variable	In the IPC or PC/AT compatible machine's [Control Panel], enter [System], [Detail], then [Environment Variables], only when the executable file exists in the folder where [TEMP] is set for the environment variables, you can specify the path with an environment variable.
	For example, %TEMP%\sample.exe (With an environment variable, specify TEMP = C:\Documents and Settings\user\Local Settings\Temp.)

6 Select the option (Argument) to run the executable using the [Parameter]. Up to 255 characters can be used to set the [Parameter].

For example, start a Microsoft Excel file

EXE path	Specify the EXCEL.EXE path. For example, execute sample.exe in C:\Program Files\Microsoft Office\Office\EXCEL.EXE
Parameter	Specify the excel book (*.xls) path in " ". For example, C:\Documents and Settings\user\desktop\ProductionControl.xls"

7 To stop multiple instances, select the [Prevent Multiple Instances] check box and enter [Window Title].

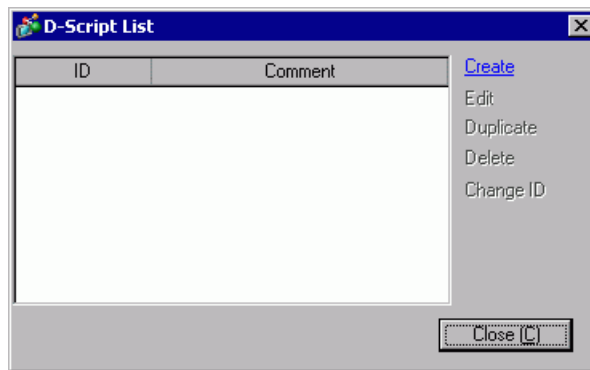
 "10.15.4 Special Switch ◆ Application Trigger" (page 10-84)

38.6.3 D-Script startup settings

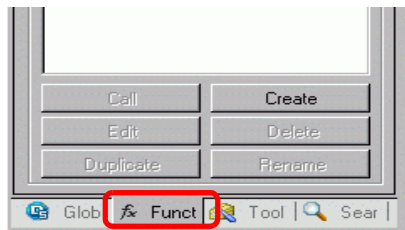
NOTE

- Please refer to the Settings Guide for details.
 ☞ "21.11.7 Others ■ Application Trigger" (page 21-145)
- On the [Common] menu, you can select [Global D-Script] or [Extended Script] to start EXE.

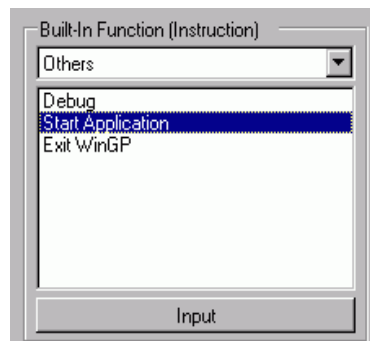
- 1 On the [Parts (P)] menu, select [D-Script (R)] and click [Create] in the [D-Script List] dialog box.



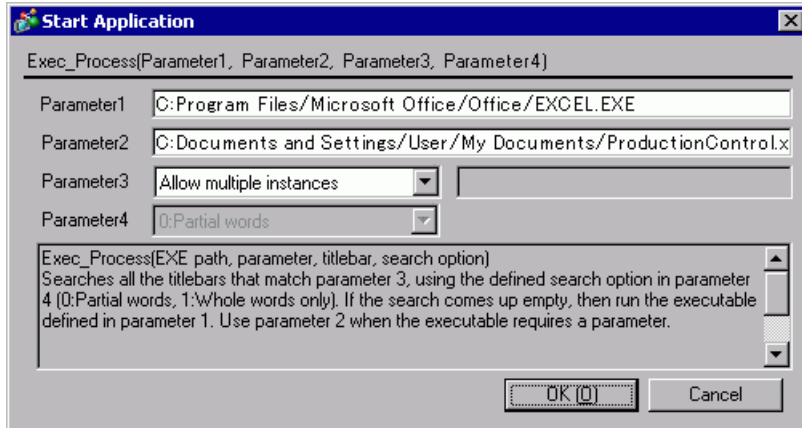
- 2 Click the [Function] tab. The [Built-In Function (Instruction)] allow you to easily place a command to use in the script.



- 3 On the [Built-In Function (Instruction)] pull-down menu, click [Others] and double-click [Start Application].



4 Configure the settings in the dialog box as shown below.



Parameter 1	Specify the EXE file path. ☞ "38.6.2 Switch Startup Settings" (page 38-74)
Parameter 2	Select the option (Argument) to run the executable using the [Parameter]. Up to 255 characters can be used to set the [Parameter]. ☞ "38.6.2 Switch Startup Settings" (page 38-74)
Parameter 3	Select [Allow Multiple Instances] or [Prevent Multiple Instances]. If you select [Prevent Multiple Instances], enter the window title. ☞ "21.11.7 Others ■ Application Trigger" (page 21-145)
Parameter 4	Select [0:Partial words] or [1:Whole words only]. ☞ "21.11.7 Others ■ Application Trigger" (page 21-145)

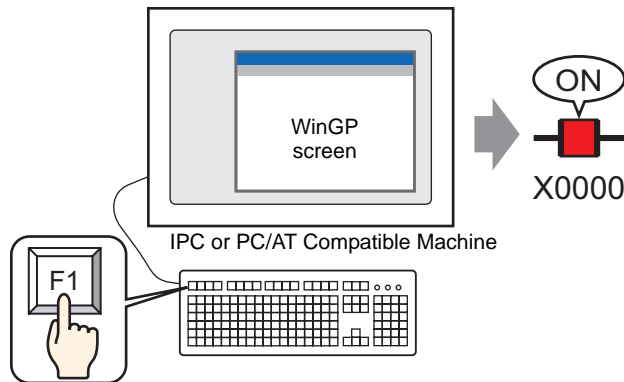
5 Click [OK] to enter the parameter configured in procedure 4 in [Script Expression Area].

For example:

```
Exec_Process("C:\Program Files\Microsoft Office\Office\EXCEL.EXE",
"C:\Documents and Settings\User\My Documents\ProductionControl.xls", "",0)
```

38.7 Allocating the switch feature to the function key

38.7.1 Details



Press the function key on the keypad while WinGP is running to operate the switch feature. In the function settings, allocate a switch feature to a function key on the keypad. The function settings include [Global Function] where switch feature can be set as a function key common to project data or [Local Function] where different switch features can be set for a function key on each base screen.


NOTE

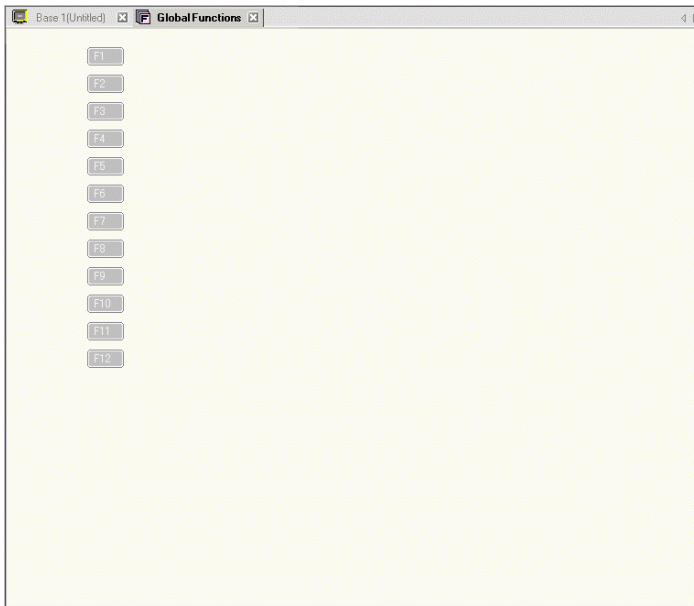
- For more information on the switch features that can be set for the function key, see:
☞ "38.7.3 Switch/Key parts that can be set for a function key" (page 38-82)

38.7.2 Setup Procedure

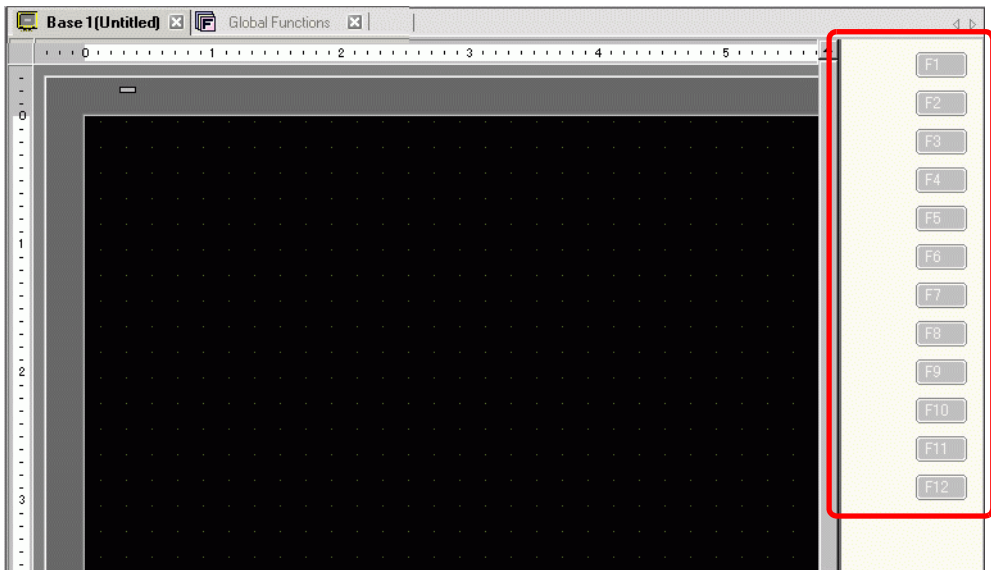
NOTE

- Please refer to the Settings Guide for details.
☞ "39.4 Setting Up Common Function Keys for All Hand-held GP Screen" (page 39-9)

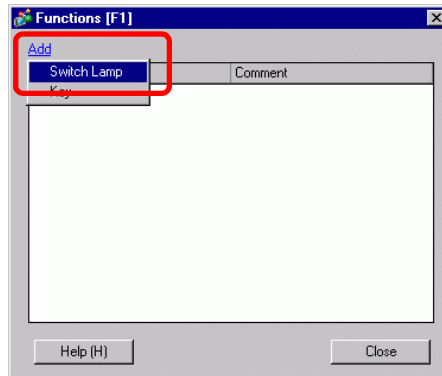
1 In the [Common Settings (R)] menu, select the [Global Function Settings (C)] command or click , and the following screen appears.



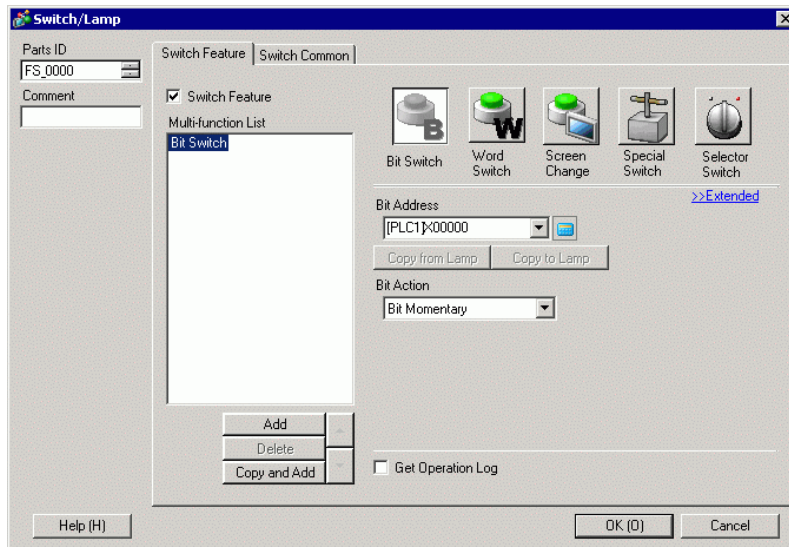
For a local function, you can set from the function area displayed on the drawing screen.



- 2 Double-click the function key to which you want to allocate the feature (for example, F1) to display the [Function Settings] dialog box. [Add] can be used to select the part attribute (for example, Switch Lamp).



- 3 Right-click the added [Part ID] and click [Edit (E)] or double-click the [Part ID] to display the setting dialog box.



- 4 Set the switch feature and click [OK].
- 5 Click [Close] to close the [Functions] dialog box.

NOTE

- The function name of the function in which features is allocated becomes black.



38.7.3 Switch/Key parts that can be set for a function key

◆ Switch Parts

The following are the Switch/Key Parts that can be set for the function key.

Category	Feature that can be set	Action	Remarks
Bit Switch	Bit Set	O	
	Bit Reset	O	
	Bit Momentary	O	
	Bit Invert	O	
	Comparison	O	
Word Switch	Write Data	O	
	Add Data	O	
	Subtract Data	O	
	Digit Addition	O	
	Digit Subtraction	O	
	scripts	O	
Screen Change	Screen Change	O	
	Previous Screen	O	
Special Switch	Window Display Switch	O	*1 It can be set but will not function in WinGP. *2 "Ladder start monitor switch" will not operate in WinGP.
	Alarm History Switch	O	
	Text Alarm Switch	O	
	Historical Trend Graph Switch	O	
	Sampling Data Display Switch	O	
	File Item Switch	O	
	File Manager Display Switch	O	
	Data Transfer Switch	O	
	Switch for CSV Display	O	
	Movie Player Switch	X ^{*1}	
	Start monitor switch	O ^{*2}	
	Application Trigger	O	
	WinGP, exiting	O	
	Remote PC Access window display Switch	X ^{*1}	
	Reset	O	
	Offline	O	
	Security	O	
Switch for Selector List	O		
Transfer Device/PLC Data	O		
Operation Lock	O		
Selector Switch		O	
Key Switch	Keypad Key	O	
	FEP Feature Key	O	

◆ Switch Common Settings

The switch common settings that can be set for the function key are as follows.


Category	Feature that can be set	Action	Remarks
Switches	Groups	O	Only a bit switch can be set.
	hierarchical screen change	O	Only the change screen switch can be set.
Switch Common	Interlock	O	[Interlocked Condition Display] cannot be set.
	Delay Feature: ON Delay	O	[In-Delay Status Display] cannot be set.
	Delay Feature: OFF Delay	O	
	Delay Feature: Double Touch	O	
	Option: Reverse Display	X	
	Option: Buzzer	O	
Option: AUX Output	O		
Lamp Feature		X	
Color		X	
Label		X	
Select Shape		X	
Animation Feature		X	

38.7.4 About Action

◆ Action of the function keys

- You can set multiple switch features for one function key. When you press the function key, it will operate in the order you set.

[F1] key setting


	Set Order	Switch Feature
	1	Word Switch Write Data D0100
	2	Word Switch Write Data D0200
	3	Bit Switch Bit Set X0000

Action

Press the [F1] key, write data to D0100, write data to D0200, and turn ON X0000.

- When you set switch features for both local function key and the global function key, it will operate the function key and then global function key.
- When the untarget switch feature is set, it will be ignored.

[F1] key setting

	Set Order	Switch Feature
	1	Word Switch Write Data D0100
	2	Remote PC Access window display Switch
	3	Bit Switch Bit Set X0000

Action

Press the [F1] key, write data to D0100, and turn ON X0000

(Remote PC Access Window Switch will not be operated because it is not a target.)

- When the change screen switch is in the middle of the set order, the switch action and screen change action are processed separately, and the order that the change screen switch is to be functioned is undefined.
If you want to function the change screen after all switches run, set the change screen switch in the last set order.
- When the function key is set for the called screen, the function key of the called screen also operates. It will operate the function key of the call-to screen, and then function key of the called screen.
- If the active window was changed to an application other than WinGP while pressing the function key, it will be determined that the function key is OFF. If the function key is still pressed after the WinGP returns to the active window again, a process will be executed again.
- When more than two function keys are pressed at the same time, the switch will work in the pressed order.

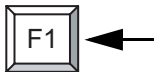
- The action of the shortcut key "activate the menu bar" for [F10], Windows specification will be enabled in WinGP.
The [F10] key will operate as a function key of WinGP similar to the other function keys.
- When you want to activate the menu bar in WinGP, use the [Alt] key.
- If you press the function key on the offline screen while the system menu is displayed, it will not function.
Similarly, if you delete the system menu while pressing the function key, it will not function.

◆ **Action in the operation log**

- If the set switch is a target when you press the function key, the operation log will be output. A log will be output for each switch set for the function key.

For example:

[F1] key setting

	Set Order	Switch Feature
	1	Word Switch Write Data D0100
	2	Word Switch Write Data D0200
	3	Bit Switch Bit Set X0000

Operation Log Data

Number	Date	Time	User ID	LEVEL	Screen	Parts ID	Comment	Action	Address	...
1	07/10/23	09:00		0	B1	SL-0000	Switch1	Bit Set	[PLC1]D0100	
2	07/10/23	09:00		0	B1	SL-0001	Switch1	Word Set	[PLC1]D0200	
3	07/10/23	09:00		0	B1	SL-0002		Bit Set	[PLC1]X0000	

The screen outputs the screen number being displayed when the function key is pressed.

The same is applied to the log of the global function key switch.

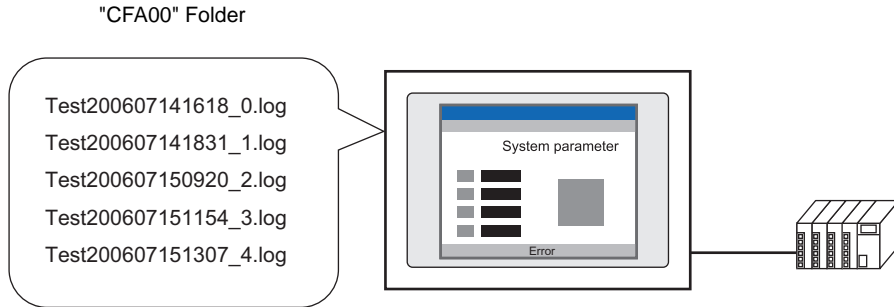
When the function key exists in the called screen, the call-to screen number will be input rather than the called screen number.

38.8 Keep History of Error Messages Displayed in WinGP

38.8.1 Introduction

You can save system and application errors displayed in WinGP as log files. Every time an error occurs, the date and time, type (Error or Warning), and error message is saved to the file.

You can save up to 1000 error messages in the log file.



Format of the error log file

For example, log filename "Test200607141618_0.log" opened as text

Date	Time	Type	Error Message
------	------	------	---------------

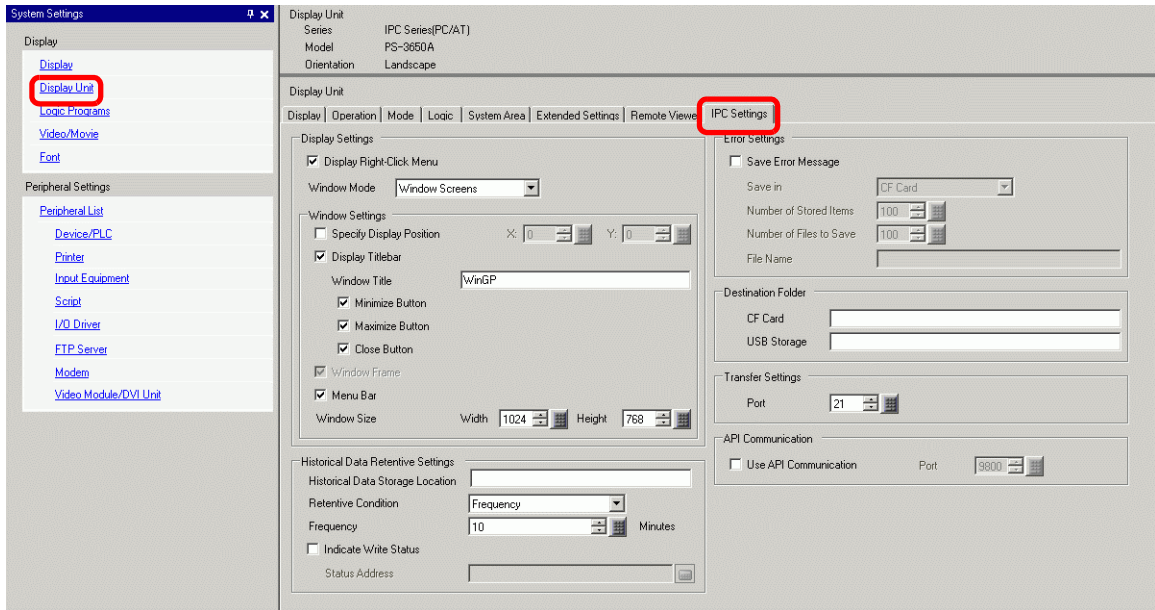
2006/07/			
14,16:18:59.563,		ERROR,	osKRboot1[c:\runtime_Desktop\win\power\src\pw_main.cpp:831]
2006/07/14,17:26:30.062,		WARNING,	RHAA070:PLC1:Cable is not connected (or PLC power is OFF)
...			

NOTE

- Error messages are written one by one to the file. However, if an error occurs within 10 minutes of the previous write, error messages are collected and when 10 minutes have elapsed all the error messages are written to the file in a single step. Collected error messages are also written to the file when exiting WinGP.

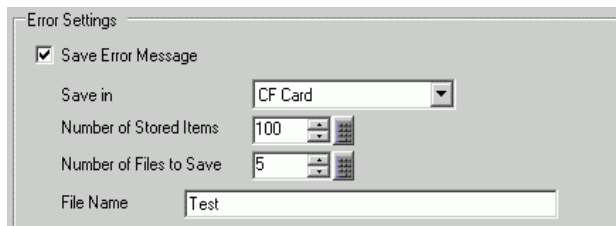
38.8.2 Setup Procedure

1 In GP-Pro EX, in the [System Settings] [Display Unit] select the [IPC Settings] tab.



2 Select the [Save Error Message] check box and in the [Save in] list select where to save error messages. (For example, CF card)

3 In the [Number of Stored Items] field define how many errors to save in one file. In the [Number of Files to Save] field define how many total files to save in the folder. After a file reaches the error limit, the system automatically creates the next file. Once all the log files are created in the folder, the oldest file is deleted and a new log file is created for new error messages.



4 In the [File Name] field, type 0 to 16 single-byte characters for the file name. (For example, "Test".)

The file name is defined using the following.

(Arbitrary file name) [Date-time]_[ID].[Extension]

Date-Time: yyymmddhhmm

ID: This is the File ID, allocated automatically using a value from zero to the [Number of Files to Save].

Extension: "log"

For example, when the date is 2006, January 14 4:18 PM, the file name is:

"Test200607141618_0.log"

38.9 API Function List

There are two types of APIs that you can use with WinGP:

38.9.1 Handling API and Device Access API.

◆ Summary

This API gets the WinGP status, or changes the WinGP settings from the user-created program. By linking the program with the API DLL file, the application created with the handling API can operate with WinGP on IPC and PC/AT compatible machines.

◆ Handling API DLL file

The API is provided in a DLL file. The file name is RtCtrlAPI.dll and installed in WINDOWS folder.

◆ Supported Languages

The following 5 programming languages can be used for handling API.

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

◆ Function List

- Get WinGP handle

Creates the WinGP handle for the communication destination and returns it to the application.

The following functions specify the handles retrieved by this function.

Function Name	INT32 GetRuntimeHandle (UINT32ul_PortNo);
Argument	ul_PortNo: (i) the IPC port number where WinGP is located
Return value (NULL)	WinGP handle

- WinGP handle release

Releases the handle retrieved by the get WinGP handle function.

Function Name	bool ReleaseRuntimeHandle (INT32l_RuntimeHandle);
Argument	l_RuntimeHandle : (i) WinGP Handle
Return value (NULL)	true: Succeed /false: Fail

- API Initialization

Initialize the WinGP operations/state get API.

Function Name	bool InitRuntimeAPI (void);
Argument	None
Return value (NULL)	true: Succeed /false: Fail

- Exit API

Executes post processing when you finish using WinGP Operation/State Get API.

Function Name	bool CleanupRuntimeAPI (void);
Argument	None
Return value (NULL)	true: Succeed /false: Fail

- ' Gets the startup state.

Gets the start up state of WinGP.

Function Name	INT32 GetRuntimeStartState (INT32 l_RuntimeHandle, INT32 *pl_RuntimeCondition);
Argument	l_RuntimeHandle : (i) WinGP handle from which it gets the information *pl_RuntimeCondition: (o) WinGP state 0: STARTING 1: START_ONLINE (Online) 2: START_OFFLINE (Offline) 3: START_TRANSFER (Transfer mode) 4: ENDING (Ending) 5: NOTEXECUTE (Not executed)
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Gets the screen number currently displayed

Gets the screen number currently displayed in WinGP from WinGP.

Function Name	INT32 GetDisplayScreenNumber (INT32 l_RuntimeHandle, INT32 *pl_DisplayScreenNumber);
Argument	l_RuntimeHandle : (i) WinGP handle from which it gets the information pl_DisplayScreenNumber: (o) Screen number If offline, Screen None (0) is returned.
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Gets the screen state

Gets the WinGP display state.

Function Name	INT32 GetScreenState (INT32 l_RuntimeHandle, INT32 *pl_ScreenState);
Argument	l_RuntimeHandle : (i) WinGP handle from which it gets the state pl_ScreenState (o) Screen state 0: FULLSCREEN (Full screen) 1: WINDOWSCREEN (Window screen) 2: MINIMUMSCREEN (Minimized) -1: UNCERTAINTY (Unknown)
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Gets the language settings
Returns the language setting number.

Function Name	INT32 GetLanguage (INT32 l_RuntimeHandle, INT32 l_LanguageKind, INT32 *pl_LanguageNumber);
Argument	l_RuntimeHandle : (i) WinGP handle from which it gets the information l_LanguageKind : (i) Language setting type 0: SYSTEMLANGUAGE (System language settings) 1: USERLANGUAGE (User language settings) pl_LanguageNumber : (o) Language setting number 0: SYSTEMLANGUAGE (System language settings) 0: Japanese 1: English 1: USERLANGUAGE (User language settings)
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Gets the touch buzzer settings
Returns the information on the buzzer sound selected in WinGP.

Function Name	INT32 GetTouchBuzzer (INT32 l_RuntimeHandle, INT32 *pl_BuzzerState);
Argument	l_RuntimeHandle : (i) WinGP handle from which it gets the information pl_BuzzerState : (o) Buzzer state 0: BUZZERON (No Buzzer) 1: BUZZEROFF (Buzzer) -1: UNCERTAINTY (Unknown)
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Gets the project information

Gets the project information in WinGP.

Function Name	INT32 GetProjectInformation(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)
Argument	l_RuntimeHandle : (i) The WinGP handle from which it gets the information ps_ProjectFileName : (o) Project file name ps_ProjectComment : (o) Project title (Comment) pus_ProjectFastTime : (o) Project creation date pus_ProjectLastTime : (o) Project last update date ps_ProjectIDownload : (o) Download date pus_HMIEditorVersion : (o) HMIEditor version pus_ControlEditorVersion: (o) CONTROL editor version pus_MakingPerson : (o) Creator name
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Gets the version information

Returns the WinGP version.

Function Name	INT32 GetRuntimeVersion(INT32 l_RuntimeHandle, UINT16 *pus_VersionInfo);
Argument	l_RuntimeHandle : (i) WinGP handle from which it gets the information pus_VersionInfo : (o) Version information
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Exit Operation

Requests WinGP to end.

Function Name	INT32 StopRuntime(INT32 I_RuntimeHandle, INT32 I_StopMode);
Argument	I_RuntimeHandle : (i) WinGP handle for operation I_StopMode : (i) End mode (Unused) 0: Normal end 1: End confirmation dialog enabled
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Changing the display screen number

Requests screen number change in WinGP

Function Name	INT32 SetDisplayScreenNumber(INT32 I_RuntimeHandle, INT32 I_ScreenNumber);
Argument	I_RuntimeHandle : (i) WinGP handle for operation I_ScreenNumber : (i) Screen number
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Changing the screen state

Changes the screen display state in WinGP.

Function Name	<pre>INT32 SetScreenState(INT32 l_RuntimeHandle INT32 l_ScreenState, INT32 l_PosX, INT32 l_PosY, INT32 l_Width, INT32 l_Height);</pre>
Argument	<pre>l_RuntimeHandle : (i) WinGP handle for operation l_ScreenState : (i) Screen State 0: FULLSCREEN (Full screen) 1: WINDOWSCREEN (Window screen) 2: MINIMUMSCREEN (Minimized) l_PosX : (i) X on the screen coordinate system (*1) l_PosY : (i) Y on the screen coordinate system (*1) l_Width : (i) Window screen width (*1) l_Height : (i) Window screen height (*1)</pre> <p>(*1) Coordinate and size are added only on the Window screen.</p> <p>The argument is available for settings only when [Screen Status] is set to [WINDOWSCREEN] for the 2nd argument.</p>
Return value (NULL)	<pre>Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)</pre>

- Changing the language settings

Changes the language settings in the system language settings/user language settings in WinGP.

The change is reflected after WinGP restarts.

Function Name	INT32 SetLanguage(INT32 l_RuntimeHandle, INT32 l_LanguageKind, INT32 l_LanguageNumber);
Argument	l_RuntimeHandle : (i) WinGP handle for operation l_LanguageKind : (i) Language setting type 0: SYSTEMLANGUAGE (System language settings) 1: USERLANGUAGE (User language settings) l_LanguageNumber : (i) Language setting number
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Changing the touch buzzer settings

Changes the touch buzzer settings in WinGP.

Function Name	INT32 SetTouchBuzzer(INT32 l_RuntimeHandle, INT32 l_BuzzerState);
Argument	l_RuntimeHandle : (i) WinGP handle for operation l_BuzzerState : (i) Buzzer settings 0: BUZZERON (No Buzzer) 1: BUZZEROFF (Buzzer)
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Get the number of screens

Gets the screen numbers that can be set in WinGP.

Function Name	INT32 GetEnumScreenNumberCount(INT32 l_RuntimeHandle, INT32 *l_ScreenNumberCount);
Argument	l_RuntimeHandle : (i) WinGP handle for operation l_ScreenNumberCount: (o) Number of display screens
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Listing the display screen numbers

Gets the screen numbers that can be set in WinGP and returns them to arrays.

Set the number of screens to be retrieved/displayed, which must be smaller than the value returned by the Get number of screens function.

Function Name	INT32 EnumScreenNumber(INT32 l_RuntimeHandle, INT32 l_ScreenNumberCount, INT32 *pl_ScreenNumbers);
Argument	l_RuntimeHandle : (i) WinGP handle for operation l_ScreenNumberCount: (i) Number of display screens l_ScreenNumbers : (o) Display screen (Returns the number in arrays)
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- Get the number of languages

Gets the number of languages that can be set in WinGP.

Function Name	INT32 GetEnumLanguageCount(INT32 l_RuntimeHandle, INT32 l_LanguageKind, INT32 *pl_LanguageCount);
Argument	l_RuntimeHandle : (i) WinGP handle for operation l_LanguageKind : (i) Language setting type 0: SYSTEMLANGUAGE (System language settings) 1: USERLANGUAGE (User language settings) pl_LanguageCount : (o) Number of languages that can be specified
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

- List the language numbers

Gets the language numbers that can be set in WinGP.

Function Name	INT32 EnumLanguage(INT32 l_RuntimeHandle, INT32 l_LanguageKind, INT32 l_LanguageCount, INT32 *pl_Languages);
Argument	l_RuntimeHandle : (i) WinGP handle for operation l_LanguageKind : (i) Language setting type 0: SYSTEMLANGUAGE (System language settings) 1: USERLANGUAGE (User language settings) l_LanguageCount : (i) Number of languages that can be specified pl_LanguageCount : (o) Languages that can be set
Return value (NULL)	Status 0 : Completed -1 : Parameter error -2 : Timeout 1 : State WinGP does not accept (ending, etc.)

38.9.2 Device Access API

◆ Summary

API is to read/write to a device/PLC communication with WinGP or a device in WinGP from the user-created program (application).

◆ DDL file for API communication

The API is provided in a DLL file. The DLL file name is ProEasy.dll and is installed in the WINDOWS folder.

◆ Supported Languages

The following five program languages can be used for the device access API.

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


NOTE

- You cannot use the following API with VB.NET or C#. Even if the API is used, its operation cannot be guaranteed.
 - ReadDevice()
 - WriteDevice()
 - ReadSymbol()
 - WriteSymbol()
 - SizeOfSymbol()
-

◆ Devices WinGP SDK can access

The WinGP SDK has access to PLC device and USR, LS Area and symbols and logic instructions variables registered in GP-Pro Ex.

NOTE

- To use structure variables of logic instructions, you need to use the parameters below.
ReadSymbolD/ReadSymbolVariantD/WriteSymbolD/WriteSymbolVariantD as I/F
For details on using a structure variable in the logic instruction, see  "3) Bit offset symbols when accessing the device with a structure variable in the logic instruction" (page 38-163)
 - You cannot use real variables of logic instructions or R_device.
-

◆ **Function List**

- Direct read API of single handle system

Function Name	Bit Data
INT WINAPI ReadDeviceBit(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* owData,WORD wCount);	
Function Name	16-bit data
INT WINAPI ReadDevice16(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* owData,WORD wCount);	
Function Name	32-bit data
INT WINAPI ReadDevice32(LPCSTR sNodeName,LPCSTR sDeviceName,DWORD* odwData,WORD wCount);	
Function Name	16-bit BCDdata
INT WINAPI ReadDeviceBCD16(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* owData,WORD wCount);	
Function Name	32-bit BCDdata
INT WINAPI ReadDeviceBCD32(LPCSTR sNodeName,LPCSTR sDeviceName,DWORD* odwData,WORD wCount);	
Function Name	Single float number data
INT WINAPI ReadDeviceFloat(LPCSTR sNodeName,LPCSTR sDeviceName,FLOAT* oflData,WORD wCount);	
Function Name	Double float number data
INT WINAPI ReadDeviceDouble(LPCSTR sNodeName,LPCSTR sDeviceName,DOUBLE* odbData,WORD wCount);	
Function Name	Text data
INT WINAPI ReadDeviceStr(LPCSTR sNodeName,LPCSTR sDeviceName,LPSTR psData,WORD wCount);	
Function Name	General data
INT WINAPI ReadDevice(LPCSTR sNodeName,LPCSTR sDeviceName,LPVOID pData,WORD wCount,WORD wAppKind);	
Function Name	General data (Variant Type)
INT WINAPI ReadDeviceVariant(LPCSTR sNodeName,LPCSTR sDeviceName,LPVARIANT pData,WORD wCount,WORD wAppKind);	

- Single handle system API

Function Name	Bit Data
INT WINAPI WriteDeviceBit(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* pwData,WORD wCount);	
Function Name	16-bit data
INT WINAPI WriteDevice16(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* pwData,WORD wCount);	
Function Name	32-bit data
INT WINAPI WriteDevice32(LPCSTR sNodeName,LPCSTR sDeviceName,DWORD* pdwData,WORD wCount);	
Function Name	16-bit BCDdata
INT WINAPI WriteDeviceBCD16(LPCSTR sNodeName,LPCSTR sDeviceName,WORD* pwData,WORD wCount);	
Function Name	32-bit BCDdata
INT WINAPI WriteDeviceBCD32(LPCSTR sNodeName,LPCSTR sDeviceName,DWORD* pdwData,WORD wCount);	
Function Name	Single float number data
INT WINAPI WriteDeviceFloat(LPCSTR sNodeName,LPCSTR sDeviceName,FLOAT* pflData,WORD wCount);	
Function Name	Double float number data
INT WINAPI WriteDeviceDouble(LPCSTR sNodeName,LPCSTR sDeviceName,DOUBLE* pdbData,WORD wCount);	
Function Name	Text data
INT WINAPI WriteDeviceStr(LPCSTR sNodeName,LPCSTR sDeviceName,LPCSTR psData,WORD wCount);	
Function Name	General data
INT WINAPI WriteDevice(LPCSTR sNodeName,LPCSTR sDeviceName,LPVOID pData,WORD wCount,WORD wAppKind);	
Function Name	General data (Variant Type)
INT WINAPI WriteDeviceVariant(LPCSTR sNodeName,LPCSTR sDeviceName,LPVARIANT pData,WORD wCount,WORD wAppKind);	

- Group Symbol Read API for Single Handle

Function Name	Group Symbol
INT WINAPI ReadSymbol(LPCSTR sNodeName,LPCSTR sSymbolName,LPVOID oReadBufferData);	
Function Name	Group Symbol (Variant Type)
INT WINAPI ReadSymbolVariant(LPCSTR sNodeName,LPCSTR sSymbolName,LPVARIANT pData);	

- Group Symbol Write API for Single Handle

Function Name	Group Symbol
INT WINAPI WriteSymbolD(LPCSTR sNodeName,LPCSTR sSymbolName,LPVOID pWriteBufferData);	
Function Name	Group Symbol (Variant Type)
INT WINAPI WriteSymbolVariantD(LPCSTR sNodeName,LPCSTR sSymbolName,LPVARIANT pData);	

- Parameter for read/write Argument

sNodeName: The station name is fixed as #WinGP.

sDeviceName: Directly describes the symbol names and device addresses registered in GP-Pro EX.

For example, use a symbol to specify "SWITCH1"

For example, directly specify the device address "M100"

The following table shows the data types that you can specify when using symbols in each function.

Function	Symbol data type							
	Bit	16 Bit		32 Bit		Float	Double	String
		Signed/Unsigned/Hex	BCD	Signed/Unsigned/Hex	BCD			
XXXDeviceBit	○							
XXXDevice16		○						
XXXDevice32				○				
XXXDeviceBCD16			○					
XXXDeviceBCD32					○			
XXXDeviceFloat						○		
XXXDeviceDouble							○	
XXXDeviceStr								○
XXXDevice	○	○	○	○	○	○	○	○

pxxData: Pointer for read/write data

Defines the destination pointer for values that are read or values that are written. For each function, please define the corresponding data format pointer.

Data types for access	Argument type
Bit Data	WORD * pwData
16-bit data	WORD * pwData
32-bit data	DWORD * pdwData
16-bit BCD data	WORD * pwData
32-bit BCD data	DWORD * pdwData
Single float number data	FLOAT * pflData
Double float number data	DOUBLE * pdbData
Text data	LPTSTR psData
General data	LPVOID pData
General data (for VB)	LPVARIANT pData

wCount: Number of read/write data

For the Read/WriteDeviceStr function, the amount of text data is expressed in single-byte units. If the symbol refers to a 16-bit device, use two characters to specify the number. If it refers to a 32-bit device, use four characters.

The following table shows the maximum amount of read/write data.

Data types for access	Read/Write
Bit Data	255
16-bit data	1020
32-bit data	510
16-bit BCD data	1020
32-bit BCD data	510
Single float number data	510
Double float number data	255
Text data	1020 characters (Single-byte)

wAppKind: Data Type Value

To specify the Data Type Value, you can either directly specify the value or specify using a constant name. For more details, please refer to the following.

 "38.9.2 Device Access API ◆ Data Type" (page 38-128)

NOTE

- The Read/WriteDevice function specifies the data type with parameters. It allows you to change the data type dynamically.

Return value (NULL)

Normal end: 0

Abnormal end: Error Code

Supplementary

When using Read/WriteDeviceBit function

PwData stores the same amount of data as in wCount starting from D0 bit.

For example: wCount is 20

	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

To handle sequential multiple data, it is more efficient to read/write in 16/32 bits using Read/WriteDevice16 and Read/WriteDevice32 than using Read/WriteDeviceBit.

“*” contains undefined values. Mask the value using an application program.

When using Read/WriteDeviceBCD16/32 functions

Use these functions for handling data as BCD internally in the device/PLC. Note that data (PxxData summary) to be sent to/received from the functions is binary data, not BCD.

(BCD conversion is performed in the [WinGP SDK].) Negative numbers cannot be handled.

Function Name	Decimal notation	Hexadecimal notation
Read/WriteDeviceBCD16	0-9999	0000 to 270F
Read/WriteDeviceBCD32	0-99999999	00000000 to 05F5E0FF

When using the text data function

For variables to receive text data, secure sufficient data space to receive the data.

- Data Access API in SRAM

Function Name	Read SRAM backup data																												
<p>Read the following data in SRAM save the data as a file in PC. The saved file format for filing data is saved in a binary format and other files are saved in CSV format.</p> <p>INT WINAPI EasyBackupDataRead(LPCSTR sSaveFileName, LPCSTR sNodeName, INT iBackupDataType, INT iSaveMode);</p>																													
<p>Argument</p> <p>sSaveFileName:(In) File path of a destination file of read data (Text pointer)</p> <p>sNodeName: (In) Participating station name of the source data to be read (Text pointer). The station name is fixed to #WinGP.</p> <p>iSaveMode: (In)How to save 0:New (If a file with the same file name exists, the file is deleted and overwritten.) 1:Add (Add data to the end of a file. If there is no file, a new file is created.) Other than those above:Reserved</p> <p>iBackupDataType:(In)Types of read data</p>	<p>Return value (NULL) Normal End: 0 Fatal Error: Error Code</p>																												
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Value</th> <th>Types of data</th> </tr> </thead> <tbody> <tr> <td>0x0001</td> <td>Filing Data</td> </tr> <tr> <td>0x0002</td> <td>Sampling data of sampling group number 1</td> </tr> <tr> <td>0x0003</td> <td rowspan="2">All sampling group data except for sampling group number 1</td> </tr> <tr> <td>0x0004</td> </tr> <tr> <td>0x0005</td> <td>Alarm Block1</td> </tr> <tr> <td>0x0006</td> <td>Alarm Block2</td> </tr> <tr> <td>0x0007</td> <td>Alarm Block3</td> </tr> <tr> <td>0x0008</td> <td>Alarm Block4</td> </tr> <tr> <td>0x0009</td> <td>Alarm Block5</td> </tr> <tr> <td>0x000A</td> <td>Alarm Block6</td> </tr> <tr> <td>0x000B</td> <td>Alarm Block7</td> </tr> <tr> <td>0x000C</td> <td>Alarm Block8</td> </tr> <tr> <td>Other than those above</td> <td>Reserved</td> </tr> </tbody> </table>			Value	Types of data	0x0001	Filing Data	0x0002	Sampling data of sampling group number 1	0x0003	All sampling group data except for sampling group number 1	0x0004	0x0005	Alarm Block1	0x0006	Alarm Block2	0x0007	Alarm Block3	0x0008	Alarm Block4	0x0009	Alarm Block5	0x000A	Alarm Block6	0x000B	Alarm Block7	0x000C	Alarm Block8	Other than those above	Reserved
Value	Types of data																												
0x0001	Filing Data																												
0x0002	Sampling data of sampling group number 1																												
0x0003	All sampling group data except for sampling group number 1																												
0x0004																													
0x0005	Alarm Block1																												
0x0006	Alarm Block2																												
0x0007	Alarm Block3																												
0x0008	Alarm Block4																												
0x0009	Alarm Block5																												
0x000A	Alarm Block6																												
0x000B	Alarm Block7																												
0x000C	Alarm Block8																												
Other than those above	Reserved																												
<p>If a type of data is alarm block 1 to 80, one alarm block stores three types of data; maximum active data, history data, and log data, based on the GP-Pro EX settings. However, this API confirms whether effective data is available based on the following priorities and if any data exists, the data will be subjected.</p> <p>(1) Alarm History (2) Alarm Log (3) Active Alarm</p> <p>If none of the above is available, the error occurs.</p>																													

Continued

Function Name	Extended Read of SRAM Backup Data																													
<p>Read the following data in SRAM save the data as a file in PC. The saved file format for filing data is saved in a binary format and other files are saved in CSV format. This enables access to data that cannot be retrieved from backup data by comparing with EasyBackupDataRead().</p> <p>INT WINAPI EasyBackupDataReadEx(LPCSTR sSaveFileName, LPCSTR sNodeName, INT iBackupDataType, INT iSaveMode, INT iNumber = 0, INT iStringTable = 0x0000);</p>																														
<p>Argument</p> <p>sSaveFileName:(In) File path of a destination file of read data (Text pointer) sNodeName: (In) Participating station name of the source data to be read (Text pointer). The station name is fixed to #WinGP. iSaveMode: (In)How to save 0:New (If a file with the same file name exists, the file is deleted and overwritten.) 1:Add (Add data to the end of a file. If there is no file, a new file is created.) Other than those above:Reserved iBackupDataType:(In)Types of read data</p>	<p>Return value (NULL) Normal End: 0 Fatal Error: Error Code</p>																													
	<table border="1"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">Type of Data</th> </tr> </thead> <tbody> <tr> <td>0x0001</td> <td>Filing Data</td> </tr> <tr> <td>0x0002</td> <td>Sampling data of sampling group number 1</td> </tr> <tr> <td>0x0003</td> <td>All sampling group data except for sampling group number 1</td> </tr> <tr> <td>0x0004</td> <td>1</td> </tr> <tr> <td>0x0005</td> <td>Alarm Block1 Specify the type of alarm using iNumber.</td> </tr> <tr> <td>0x0006</td> <td>Alarm Block2 Specify the type of alarm using iNumber.</td> </tr> <tr> <td>0x0007</td> <td>Alarm Block3 Specify the type of alarm using iNumber.</td> </tr> <tr> <td>0x0008</td> <td>Alarm Block4 Specify the type of alarm using iNumber.</td> </tr> <tr> <td>0x0009</td> <td>Alarm Block5 Specify the type of alarm using iNumber.</td> </tr> <tr> <td>0x000A</td> <td>Alarm Block6 Specify the type of alarm using iNumber.</td> </tr> <tr> <td>0x000B</td> <td>Alarm Block7 Specify the type of alarm using iNumber.</td> </tr> <tr> <td>0x000C</td> <td>Alarm Block8 Specify the type of alarm using iNumber.</td> </tr> <tr> <td>0x8002</td> <td>Sampling group of a specific group number Specify a group number using iNumber.</td> </tr> </tbody> </table>		Value	Type of Data	0x0001	Filing Data	0x0002	Sampling data of sampling group number 1	0x0003	All sampling group data except for sampling group number 1	0x0004	1	0x0005	Alarm Block1 Specify the type of alarm using iNumber.	0x0006	Alarm Block2 Specify the type of alarm using iNumber.	0x0007	Alarm Block3 Specify the type of alarm using iNumber.	0x0008	Alarm Block4 Specify the type of alarm using iNumber.	0x0009	Alarm Block5 Specify the type of alarm using iNumber.	0x000A	Alarm Block6 Specify the type of alarm using iNumber.	0x000B	Alarm Block7 Specify the type of alarm using iNumber.	0x000C	Alarm Block8 Specify the type of alarm using iNumber.	0x8002	Sampling group of a specific group number Specify a group number using iNumber.
Value	Type of Data																													
0x0001	Filing Data																													
0x0002	Sampling data of sampling group number 1																													
0x0003	All sampling group data except for sampling group number 1																													
0x0004	1																													
0x0005	Alarm Block1 Specify the type of alarm using iNumber.																													
0x0006	Alarm Block2 Specify the type of alarm using iNumber.																													
0x0007	Alarm Block3 Specify the type of alarm using iNumber.																													
0x0008	Alarm Block4 Specify the type of alarm using iNumber.																													
0x0009	Alarm Block5 Specify the type of alarm using iNumber.																													
0x000A	Alarm Block6 Specify the type of alarm using iNumber.																													
0x000B	Alarm Block7 Specify the type of alarm using iNumber.																													
0x000C	Alarm Block8 Specify the type of alarm using iNumber.																													
0x8002	Sampling group of a specific group number Specify a group number using iNumber.																													

Continued

Function Name	Extended Read of SRAM Backup Data											
<p>iNumber: Enter a value based on the value in iBackupDataType.</p>												
A value in iBackupDataType	Description											
0x0005 to 0x000C	<p>There are three types of alarm data; Active, History, and Log. Specify the type.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">A value in iNumber</th> <th style="text-align: center;">Description</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td> <p>Check if the alarm block contains available data based on the following priorities and if data exists, the data becomes a target.</p> <p>(1) Alarm History (2) Alarm Log (3) Active Alarm</p> <p>If none of the above is available, the error occurs.</p> </td> </tr> <tr> <td style="text-align: center;">1</td> <td>Targets Active Alarms.</td> </tr> <tr> <td style="text-align: center;">2</td> <td>Targets Alarm History.</td> </tr> <tr> <td style="text-align: center;">3</td> <td>Targets Alarm Log.</td> </tr> </tbody> </table>		A value in iNumber	Description	0	<p>Check if the alarm block contains available data based on the following priorities and if data exists, the data becomes a target.</p> <p>(1) Alarm History (2) Alarm Log (3) Active Alarm</p> <p>If none of the above is available, the error occurs.</p>	1	Targets Active Alarms.	2	Targets Alarm History.	3	Targets Alarm Log.
	A value in iNumber	Description										
	0	<p>Check if the alarm block contains available data based on the following priorities and if data exists, the data becomes a target.</p> <p>(1) Alarm History (2) Alarm Log (3) Active Alarm</p> <p>If none of the above is available, the error occurs.</p>										
	1	Targets Active Alarms.										
	2	Targets Alarm History.										
3	Targets Alarm Log.											
<p>If the subjected data type is not in the alarm block specified by iBackupDataType, an error occurs.</p>												
0x8002	<p>Group number of a sampling group to be read A value between 1 to 64</p>											
Other than those above	Reserved											
<p>iStringTable: (In)Reserved. Always specify 0.</p>												

Function Name	Write SRAM backup data	
<p>The binary format filing data is written in the SRAM.</p>		
<p>INT WINAPI EasyBackupDataWrite(LPCSTR sSourceFileName,LPCSTR sNodeName,INT iBackupDataType);</p>		
<p>Argument</p> <p>sSourceFileName: (In) File path of a filing data file in binary format to be written (Text pointer)</p> <p>sNodeName: (In) Name of a participating station of the location to which the data is written (Text pointer) The station name is fixed to #WinGP.</p> <p>iBackupDataType:(In)"1" Fixed (indicates filing/recipe data)</p>	<p>Return value (NULL)</p> <p>Normal End: 0</p> <p>Fatal Error: Error Code</p>	

- API for Systems

Function Name	Message handling control	
<p>Most of WinGP SDK API functions handles Windows messages within the functions if the process takes a while. You can specify if you use this Windows message process or control this process.</p> <p>If the control is used, Windows messages are accumulated in the message queue and not processed during the function process.</p> <p>As a result, it can prevent from double calling of functions by clicking the icon during function process.</p> <p>However, in this case, please be careful that all Windows message processes are controlled, not only the message "the icon is clicked" but also important messages such as the timer and redrawing window screen will not be processed.</p> <p>You can specify whether to process or control the process for each handle of WinGP SDK. The default is set to process.</p> <p>INT EasySetWaitType(DWORD dwMode);</p>		
<p>Argument</p> <p>dwMode: (In)Process messages if 1 is specified. Control message process if 2 is specified.</p>	<p>Return value (NULL)</p> <p>Normal End: 0</p> <p>Fatal Error: Error Code</p>	

Function Name	Acquiring the method of message process	
<p>This function retrieves what mode is being processed for the message method when WinGP SDK API is calling.</p> <p>INT EasyGetWaitType();</p>		
<p>Argument</p>	<p>Return value (NULL)</p> <p>1:Process messages.</p> <p>2:Control message processing.</p>	

Function Name	Text Conversion of Error Code	
<p>Converts error codes returned by various APIs in the WinGP SDK into error messages. EasyLoadErrorMessage() returns multi-byte text (ASCII) as a message. EasyLoadErrorMessageW() returns a UNICODE text string</p> <p>BOOL WINAPI EasyLoadErrorMessage(INT iErrorCode,LPSTR osErrorMessage); BOOL WINAPI EasyLoadErrorMessageW(INT iErrorCode,LPWSTR owsErrorMessage);</p>		
<p>Argument</p> <p>iErrorCode: (In) Error code returned by the WinGP SDK function osErrorMessage: (Out) Pointer to the area where the converted string (ASCII) is stored (prepare for 512 bytes or more) owsErrorMessage: (Out) Pointer to the area where the converted string (ASCII) is stored (prepare for 1024 bytes or more)</p>	<p>Return value (NULL) Successful operation: Any value other than zero Failed to convert string (for example, unused error code): 0</p>	
<p>Special Item</p> <ul style="list-style-type: none"> • This API is provided to enable compatibility with Pro-Server with Studio. • EasyLoadErrorMessageEx() converts errors into an error message with more details. 		

Function Name	Error code string conversion (status information attached)	
<p>Converts error codes returned by various APIs in the WinGP SDK into error messages. Returns an error message with status information attached, if possible. EasyLoadErrorMessage() always returns the same error message as the defined error code. EasyLoadErrorMessageEx() returns more detailed information, such as the name of the communication partner, where the error occurred, and status when the error occurred. Even the same error code could return different error messages, depending on the location of the error. EasyLoadErrorMessageEx(), EasyLoadErrorMessageExM() return a multi-byte string message (ASCII) EasyLoadErrorMessageEx(), EasyLoadErrorMessageExM() return a string message (UNICODE)</p> <p>BOOL WINAPI EasyLoadErrorMessageEx(INT iErrorCode,LPSTR osErrorMessage); BOOL WINAPI EasyLoadErrorMessageExW(INT iErrorCode,LPWSTR owsErrorMessage);</p>		
<p>Argument</p> <p>iErrorCode: (In) Error code returned by the WinGP SDK function osErrorMessage: (Out) Pointer to the area where the converted string (ASCII) is stored (prepare for 1024 bytes or more) v:(Out) Pointer to the area where the converted string (UNICODE) is stored (prepare for 2048 bytes or more)</p>	<p>Return value (NULL) Successful operation: Any value other than zero Failed to convert string (for example, unused error code): 0</p>	

Function Name	Error code string conversion (status information attached)
Special Item	
<ul style="list-style-type: none"> • EasyLoadErrorMessage() is used to call a function in the WinGP API, and when the function returns an error code, this message is converted into a message. • The WinGP SDK remembers only one set of error status information for each handle. As a result, after an error occurs in the API, call EasyLoadErrorMessage() right away. Do not call a different API function, or else the API will overwrite the error status information and EasyLoadErrorMessage() will not return the desired error status. 	

- Other APIs

Function Name	Read IPC Time as DWORD	
Function for getting the current time as a numeric value (DWORD format) from the defined station. This function is valid only with the time stored in LS2048 (6 words).		
DWORD WINAPI EasyGetGPTime(LPCSTR sNodeName, DWORD* odwTime);		
Argument		Return value (NULL)
sNodeName:	The station name is fixed as #WinGP.	Normal End: 0
odwTime:	Retrieves time in DWORD format, which actually uses ANSI time_t format	Fatal Error: Error Code
Special Item		

Function Name	Read IPC Time as VARIANT	
Function to acquire the current time as a numeric value (Variant format) from the defined station. This function is valid only with the time stored in LS2048 (6 words).		
DWORD WINAPI EasyGetGPTimeVariant(LPCSTR sNodeName, LPVARIANT ovTime);		
Argument		Return value (NULL)
sNodeName:	The station name is fixed as #WinGP.	Normal End: 0
ovTime:	Retrieves time as VARIANT format, which internally is the Date format	Fatal Error: Error Code
Special Item		

Function Name	Read IPC Time as STRING	
<p>Function to acquire the current time as a string (LPTSTR format) from the defined station. This function is valid only with the time stored in LS2048 (6 words).</p>		
<p>DWORD WINAPI EasyGetGPTTimeString(LPCSTR sNodeName, LPCSTR sFormat, LPSTR osTime);</p>		
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>pFormat: Retrieves string as a time formatted string. Formatting codes following the percent sign (%) are replaced with "Special Item." Other characters are not converted and display as is.</p> <p>osTime: Retrieves time as a string (but when not enough space to receive string length + 1 (Null) or more, an unexpected memory space damage will occur. Make sure you reserve enough memory space to receive string length + 1 (for the NULL character). If you don't reserve enough space, you could experience unexpected data loss and operations may not work properly.</p>	<p>Return value (NULL)</p> <p>Normal End: 0</p> <p>Fatal Error: Error Code</p>	

Continued

Function Name	Read IPC Time as STRING
Special Item	
Formatting codes following the percent sign (%) are replaced as shown in the following table. Other characters are not converted and display as is. For example, if the clock is 2006/1/2 12:34:56 and you define %Y_%M %S, the string becomes:	
Formatting Code	Folder
%a	Day - abbreviated (*2)
%A	Day (*2)
%b	Month - abbreviated (*2)
%B	Month (*2)
%c	Locale-related date and time
%#c	Locale-related date and time (long form)
%d	Day as decimal value (01 to 31) (*1)
%H	24 Hour Clock (00 to 23) (*1)
%l	12 Hour Clock (01 to 12) (*1)
%j	Day of year as decimal value (001 to 366) (*1)
%m	Month as decimal value (01 to 12) (*1)
%M	Minutes as decimal value (00 to 59) (*1)
%p	AM/PM for locale (*2)
%S	Seconds as decimal value (00 to 59) (*1)
%U	Week of year as decimal value. The first Sunday of the year is the first week. (00 to 53) (*1)
%w	Day as decimal value. Sunday is 0 (0 to 6) (*1)
%W	Week of year as decimal value. The first Monday of the year is the first week. (00 to 53) (*1)
%x	Date of current locale
%#x	Date of current locale (long form)
%X	Time of current locale (*2)
%y	2-digit Year as decimal value (00 to 99) (*1)
%Y	4-digit Year as decimal value (*1)
%z, %Z	Time zone or time-zone abbreviation. When time zone is unknown, character is not entered (*2)
%%	Percentage symbol (*2)
<p>*1 Suppress leading zeroes by placing a hash mark (#) in front of d, H, I, j, m, M, S, U, w, W, y, or Y. For example, if the value is 05, and the formatting code is %#d, displays 5.</p> <p>*2 The hash mark is ignored when it is placed (for example %#a) in front of a, A, b, B, p, X, z, or Z.</p>	

Function Name	Read IPC Time as STRING VARIANT	
<p>Function for acquiring the current time as a string (Variant format) from the defined station. This function is valid only with the time stored in LS2048 (6 words).</p>		
<p>DWORD WINAPI EasyGetGPTimeStringVariant(LPCSTR sNodeName, LPCSTR sFormat, LPVARIANT ovTime);</p>		
<p>Argument sNodeName: The station name is fixed as #WinGP. pFormat: Retrieves string as a time formatted string. Formatting codes following the percent sign (%) are replaced as shown in the following table. Other characters are not converted and display as is. For details, refer to the "Special Items" section in "Function for Reading String Type on the IPC." ovTime: Retrieves time string as VARIANT format, which internally is the BSTR format</p>	<p>Return value (NULL) Normal End: 0 Fatal Error: Error Code</p>	

Function Name	Read Reference Station Status	
<p>You can get the status of the connected equipment (IPC). Or, you can vary the response timeout value to confirm the connection.</p>		
<p>INT WINAPI GetNodeProperty(LPCSTR sNodeName,DWORD dwTimeLimit,LPSTR osGPType,LPSTR osSystemVersion,LPSTR osComVersion,LPSTR osECOMVersion);</p>		
<p>Argument sNodeName: The station name is fixed as #WinGP. dwTimeLimit: (In) Response Timeout Value. Zero is the default value, which indicates 3000 milliseconds, not zero milliseconds. The set up range, in millisecond units, is 1 to 2147483647, or zero.</p> <p>The following areas return information about object stations. Please reserve 32 bytes or more of memory. osGPType: (Out) Model code osSystemVersion:(Out) System version osComVersion: (Out) PLC protocol driver version (blank) osECOMVersion: (Out) 2-Way driver version (blank)</p>	<p>Return value (NULL) Normal End: 0 Fatal Error: Error Code</p>	

Function Name	Finds the symbol/group byte size	
Find the total buffer byte size required to access the device and group symbols.		
<pre>INT WINAPI SizeOfSymbol(LPCSTR sNodeName,LPCSTR sSymbolName,INT* oiByteSize);</pre>		
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>sSymbolName:(In) Device symbol name or group symbol name to search for</p> <p>oiByteSize: (Out) Byte size to search for</p>	<p>Return value (NULL)</p> <p>Normal End: 0</p> <p>Fatal Error: Error Code</p>	
<p>Special Item</p> <p>In sSymbolName, you can define one element as a device symbol, non-array group, array group, or all array groups.</p>		

Function Name	Finds the number of members in the group	
Finds the number of members in the defined group symbol or symbol sheet, which is the total symbols and groups.		
<pre>INT WINAPI GetCountOfSymbolMember(LPCSTR sNodeName,LPCSTR sSymbolName,INT* oiCountOfMember);</pre>		
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>sSymbolName:(In) Group symbol name or symbol sheet name to search for</p> <p>oiCountOfMember:(Out) Number of members to find</p>	<p>Return value (NULL)</p> <p>Normal End: 0</p> <p>Fatal Error: Error Code</p>	
<p>Special Item</p> <p>When the defined group symbol contains another group symbol, even if there are multiple device symbols within the internal group symbol, the device symbols are counted as one member.</p>		

Function Name	Searches for definition information about symbol, group, symbol sheet	
<p>Searches for the definition information, such as the data format and data size, of the defined device symbol, group symbol, or symbol sheet.</p>		
<p>INT WINAPI GetSymbolInformation(LPCSTR sNodeName,LPCSTR sSymbolName,INT iMaxCountOfSymbolMember,LPSTR osSymbolSheetName,SymbolInformation* oSymbolInformation,INT* oiGotCountOfSymbolMember);</p>		
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>sSymbolName:(In) Symbol, group name, sheet name</p> <p>iMaxCountOfSymbolMember:(In) Define the maximum value (1 or more) for the search information. Defines the quantity in oSymbolInformation.</p> <p>osSymbolSheetName:(Out) Returns the symbol sheet name belonging to sSymbolName. Please prepare a workspace of 66 bytes or more.</p> <p>oSymbolInformation:(Out) Returns detailed information as an array.Please prepare the quantity defined in iMaxCountOfSymbolMember for the workspace.</p> <p>oiGotCountOfSymbolMember:(Out) Returns the information number actually returned to oSymbolInformation.</p>	<p>Return value (NULL)</p> <p>Normal End: 0</p> <p>Fatal Error: Error Code</p>	

Continued

Function Name	Searches for definition information about symbol, group, symbol sheet
	<p>Special Item</p> <ul style="list-style-type: none"> • SymbolInformation Structure <pre> struct SymbolInformation { WORD m_wAppKind; // Data type. When symbol 1 to 12, when group 0x8000 WORD m_wDataCount; // Data size DWORD m_dwSizeOf; // Buffer byte size char m_sSymbolName[64+1]; // Symbol or group name char m_bDummy1[3]; // Reserved char m_sDeviceAddress[256+1]; // Device address (empty for group) char m_bDummy2[3]; // Reserved }; </pre> <p>Information found in oSymbolInformation is returned as a SymbolInformation array when group, sheet, or symbol is set in the first setting.</p> <p>In the second setting and onward, when sSymbolName is a group, sets the group members. When sSymbolName is sheet, the entire sheet information is set. When sSymbolName is symbol, there are no settings after the first.</p> <p>When the object symbol is bit offset symbol, be careful about the following points.</p> <p>(1) When bit offset symbol is used to directly specify the original symbol information (when sSymbolName is directly specified as bit offset symbol), in oSymbolInformation's first SymbolInformation field m_dwSizeOf, a byte count of 2 is set for accessing the bit symbol. Because the original information is one symbol, there can be only one oSymbolInformation.</p> <p>(2) Define the original information as group symbol, and when the group includes a bit offset symbol, the m_dwSizeOf property of the second oSymbolInformation and later is set to zero, because it defines the access size for group access of members.</p> <ul style="list-style-type: none"> • When the member count is unknown, use GetCountOfSymbolMember(), set up a SymbolInformation workspace of the member count return value + 1, then call this function.

- CF Card APIs

Function Name	Read CF Card status															
Gets the CF card connection status of the IPC.																
INT WINAPI EasyIsCFCard(LPCSTR sNodeName) ;																
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP. The node needs to be registered in a network project.</p>	<p>Return value (NULL)</p> <table border="1" data-bbox="683 417 1256 716"> <thead> <tr> <th data-bbox="683 417 852 479">Function Return value</th> <th data-bbox="852 417 1256 479">Status</th> </tr> </thead> <tbody> <tr> <td data-bbox="683 479 852 517">0x00000000</td> <td data-bbox="852 479 1256 517">Normal</td> </tr> <tr> <td data-bbox="683 517 852 575">0x10000001</td> <td data-bbox="852 517 1256 575">There is no CF card or the cover on the CF card slot is open</td> </tr> <tr> <td data-bbox="683 575 852 614">0x10000002</td> <td data-bbox="852 575 1256 614"></td> </tr> <tr> <td data-bbox="683 614 852 653">0x10000004</td> <td data-bbox="852 614 1256 653">Detect CF card problem</td> </tr> <tr> <td data-bbox="683 653 852 691">0x10000008</td> <td data-bbox="852 653 1256 691"></td> </tr> <tr> <td data-bbox="683 691 852 716">Others</td> <td data-bbox="852 691 1256 716">Error unrelated to card</td> </tr> </tbody> </table>		Function Return value	Status	0x00000000	Normal	0x10000001	There is no CF card or the cover on the CF card slot is open	0x10000002		0x10000004	Detect CF card problem	0x10000008		Others	Error unrelated to card
Function Return value	Status															
0x00000000	Normal															
0x10000001	There is no CF card or the cover on the CF card slot is open															
0x10000002																
0x10000004	Detect CF card problem															
0x10000008																
Others	Error unrelated to card															

Function Name	Read CF card file list (optional folder name)																	
<p>The file list in the IPC CF card is output to the file, sent as a parameter. You can optionally define the folder of the file list you want to get.</p> <p>INT WINAPI EasyGetListInCfCard(LPCSTR sNodeName, LPCSTR sDirectory, INT* oiCount, LPCSTR sSaveFileName) ;</p>																		
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>sDirectory: Gets the folder name in uppercase characters</p> <p>oiCount: Number of files read</p> <p>sSaveFileName:Filename where directory information is stored. In the defined file, data stored in the stEasyDirInfo formatted array is, in the quantity returned in the pioCount, stored as binary data. Saves the filename and extension in uppercase characters.</p> <pre> struct stEasyDirInfo { BYTE bFileName[8+1]; // File name (NULL terminated) BYTE bExt[3+1]; // File extension (NULL terminated) BYTE bDummy[3]; // temporary DWORD dwFileSize; // File size BYTE bFileTimeStamp[8+1]; // File timestamp (NULL terminated) BYTE bDummy2[3]; // temporary2 }; </pre>	<p>Return value (NULL)</p> <p>Successful operation: 0</p> <p>Problem operation: Error code</p>																	
<p>Special Item</p> <p>More information about bFileTimeStamp: 8 bytes are divided into two sections. The top 4 bytes are used to store MS-DOS formatted time, and the bottom 4 bytes are used to store MS-DOS formatted date, both as hexadecimal values.</p> <p>MS-DOS formatted dates and times are set up in the following format.</p> <p>For example, when the DOS date/time is 20C42C22, 2C22 is the date and 20C4 is the time. Translated, the date and time is 2002/1/2 04:06:08.)</p> <table border="1" data-bbox="216 1360 1222 1528"> <thead> <tr> <th>Bit</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0 - 4</td> <td>Day of month (1 to 31).</td> </tr> <tr> <td>5 - 8</td> <td>Month of year (1=January, 2=February, ..., 12=December)</td> </tr> <tr> <td>9 - 15</td> <td>9 to 15 Year, starting with the year 1980. Add 1980 to the value indicated by these bits to come up with the actual year.</td> </tr> </tbody> </table> <p>MS-DOS formatted time. The date uses the following format to pack the date into one 16-bit value.</p> <table border="1" data-bbox="216 1653 1222 1792"> <thead> <tr> <th>Bit</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0 - 4</td> <td>The number of seconds, divided by 2 (0 to 29).</td> </tr> <tr> <td>5 - 10</td> <td>Minutes (0 to 59)</td> </tr> <tr> <td>11 - 15</td> <td>Hours (24 hour clock 0 to 23).</td> </tr> </tbody> </table>			Bit	Description	0 - 4	Day of month (1 to 31).	5 - 8	Month of year (1=January, 2=February, ..., 12=December)	9 - 15	9 to 15 Year, starting with the year 1980. Add 1980 to the value indicated by these bits to come up with the actual year.	Bit	Description	0 - 4	The number of seconds, divided by 2 (0 to 29).	5 - 10	Minutes (0 to 59)	11 - 15	Hours (24 hour clock 0 to 23).
Bit	Description																	
0 - 4	Day of month (1 to 31).																	
5 - 8	Month of year (1=January, 2=February, ..., 12=December)																	
9 - 15	9 to 15 Year, starting with the year 1980. Add 1980 to the value indicated by these bits to come up with the actual year.																	
Bit	Description																	
0 - 4	The number of seconds, divided by 2 (0 to 29).																	
5 - 10	Minutes (0 to 59)																	
11 - 15	Hours (24 hour clock 0 to 23).																	

Function Name	Read CF Card file list (define file type)	
<p>The file list in the IPC CF card is output to the file, sent as a parameter. You can optionally define the directory of the file list you want to read by using "sDirectory".</p> <p>INT WINAPI EasyGetListInCard(LPCSTR sNodeName, LPCSTR sDirectory, INT* oiCount, LPCSTR sSaveFileName);</p>		
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>sDirector: Gets the directory name, all in uppercase characters. Only the following directories are supported. LOG (logged data) TREND (trend data) ALARM (alarm data) CAPTURE (screen capture data) FILE (recipe data)</p> <p>oiCount: Number of files read</p> <p>sSaveFileName:Filename where directory information is stored. In the defined file, data stored in the stEasyDirInfo formatted array is, in the quantity returned in the pioCount, stored as binary data. Saves the filename and extension in uppercase characters.</p> <pre>struct stEasyDirInfo { BYTE bFileName[8+1]; // File name (NULL terminated) BYTE bExt[3+1]; // File extension (NULL terminated) BYTE bDummy[3]; // temporary DWORD dwFileSize; // File size BYTE bFileTimeStamp[8+1]; // File timestamp (NULL terminated) BYTE bDummy2[3]; // temporary2 };</pre>	<p>Return value (NULL)</p> <p>Successful operation: 0</p> <p>Problem operation: Error code</p>	

Function Name	Read CF Card file (optional file name)	
<p>Function to read the file contents of the file to save to CF card. You can optionally define the file to read.</p> <p>INT WINAPI EasyFileReadInCfCard(LPCSTR sNodeName, LPCSTR sFolderName, LPCSTR sFileName, LPCSTR pWriteFileName, DWORD* odwFileSize);</p>		
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>sFolderName: Folder name of file on the CF card file to read (Maximum 32 single-byte characters.)</p> <p>sFileName: File name in the 8.3 string format to read from the CF card.</p> <p>pWriteFileName:File name and path for saving the CF card file</p> <p>odwFileSize: File size of the file read from the CF card</p>	<p>Return value (NULL)</p> <p>Successful operation: 0</p> <p>Problem operation: Error code</p>	

Function Name	Read CF Card file (define file type)																																											
<p>Function to read the file contents of the file to save to CF card. File you can read are limited to the file type defined in pReadFileType.</p> <p>INT WINAPI EasyFileReadCard(LPCSTR sNodeName, LPCSTR pReadFileType, WORD wReadFileNo, LPCSTR sWriteFileName, DWORD* odwFileSize);</p>																																												
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>pReadFileType:File type of the file to read from the CF card (see Special Items)</p> <p>wReadFileNo: File number of the file to read from the CF card</p> <p>sWriteFileName:File name and path for saving the CF card file</p> <p>odwFileSize: File size of the file read from the CF card</p>	<p>Return value (NULL)</p> <p>Successful operation: 0</p> <p>Problem operation: Error code</p>																																											
<p>Special Item</p> <p>The supported file types are as follows. You can only read items stored in the defined CF card folder.</p>																																												
<p>File Types</p>																																												
<table border="1"> <thead> <tr> <th data-bbox="197 826 687 861">Data Class</th> <th data-bbox="687 826 915 861">File Type</th> <th data-bbox="915 826 1140 861">Folder</th> </tr> </thead> <tbody> <tr> <td data-bbox="197 861 687 896">Recipe (Filing Data)</td> <td data-bbox="687 861 915 896">ZF or F</td> <td data-bbox="915 861 1140 896">FILE</td> </tr> <tr> <td data-bbox="197 896 687 931">Recipe (CSV Data)</td> <td data-bbox="687 896 915 931">ZR</td> <td data-bbox="915 896 1140 931">FILE</td> </tr> <tr> <td data-bbox="197 931 687 966">Image Screen</td> <td data-bbox="687 931 915 966">ZI or I</td> <td data-bbox="915 931 1140 966">DATA</td> </tr> <tr> <td data-bbox="197 966 687 1000">Sound Data</td> <td data-bbox="687 966 915 1000">ZO or O</td> <td data-bbox="915 966 1140 1000">DATA</td> </tr> <tr> <td data-bbox="197 1000 687 1089">GP-PRO/PB III for Windows exclusive trend graph data (compatible)</td> <td data-bbox="687 1000 915 1089">ZT</td> <td data-bbox="915 1000 1140 1089">TREND</td> </tr> <tr> <td data-bbox="197 1089 687 1178">GP-PRO/PB III for Windows exclusive sampling data (compatible)</td> <td data-bbox="687 1089 915 1178">ZS</td> <td data-bbox="915 1089 1140 1178">TREND</td> </tr> <tr> <td data-bbox="197 1178 687 1213">Alarm1</td> <td data-bbox="687 1178 915 1213">Z1 or ZA</td> <td data-bbox="915 1178 1140 1213">ALARM</td> </tr> <tr> <td data-bbox="197 1213 687 1248">Alarm2</td> <td data-bbox="687 1213 915 1248">Z2 or ZH</td> <td data-bbox="915 1213 1140 1248">ALARM</td> </tr> <tr> <td data-bbox="197 1248 687 1282">Alarm3</td> <td data-bbox="687 1248 915 1282">Z3 or ZG</td> <td data-bbox="915 1248 1140 1282">ALARM</td> </tr> <tr> <td data-bbox="197 1282 687 1317">Alarm4 to 8</td> <td data-bbox="687 1282 915 1317">Z4 to Z8</td> <td data-bbox="915 1282 1140 1317">ALARM</td> </tr> <tr> <td data-bbox="197 1317 687 1379">GP-PRO/PB III for Windows exclusive logging data (compatible)</td> <td data-bbox="687 1317 915 1379">ZL</td> <td data-bbox="915 1317 1140 1379">LOG</td> </tr> <tr> <td data-bbox="197 1379 687 1414">Capture data</td> <td data-bbox="687 1379 915 1414">CP</td> <td data-bbox="915 1379 1140 1414">CAPTURE</td> </tr> <tr> <td data-bbox="197 1414 687 1476">Sampling Group 1 to 64 data</td> <td data-bbox="687 1414 915 1476">ZS1 to ZS64</td> <td data-bbox="915 1414 1140 1476">SAMP01 to SAMP64</td> </tr> </tbody> </table>			Data Class	File Type	Folder	Recipe (Filing Data)	ZF or F	FILE	Recipe (CSV Data)	ZR	FILE	Image Screen	ZI or I	DATA	Sound Data	ZO or O	DATA	GP-PRO/PB III for Windows exclusive trend graph data (compatible)	ZT	TREND	GP-PRO/PB III for Windows exclusive sampling data (compatible)	ZS	TREND	Alarm1	Z1 or ZA	ALARM	Alarm2	Z2 or ZH	ALARM	Alarm3	Z3 or ZG	ALARM	Alarm4 to 8	Z4 to Z8	ALARM	GP-PRO/PB III for Windows exclusive logging data (compatible)	ZL	LOG	Capture data	CP	CAPTURE	Sampling Group 1 to 64 data	ZS1 to ZS64	SAMP01 to SAMP64
Data Class	File Type	Folder																																										
Recipe (Filing Data)	ZF or F	FILE																																										
Recipe (CSV Data)	ZR	FILE																																										
Image Screen	ZI or I	DATA																																										
Sound Data	ZO or O	DATA																																										
GP-PRO/PB III for Windows exclusive trend graph data (compatible)	ZT	TREND																																										
GP-PRO/PB III for Windows exclusive sampling data (compatible)	ZS	TREND																																										
Alarm1	Z1 or ZA	ALARM																																										
Alarm2	Z2 or ZH	ALARM																																										
Alarm3	Z3 or ZG	ALARM																																										
Alarm4 to 8	Z4 to Z8	ALARM																																										
GP-PRO/PB III for Windows exclusive logging data (compatible)	ZL	LOG																																										
Capture data	CP	CAPTURE																																										
Sampling Group 1 to 64 data	ZS1 to ZS64	SAMP01 to SAMP64																																										

Function Name	Write to CF card file (optionally define file name)	
<p>Function to save the file contents of the file to the CF card. You can optionally define the file name to save.</p> <p>INT WINAPI EasyFileWriteInCfCard(LPCSTR sNodeName, LPCSTR pReadFileName, LPCSTR sFolderName, LPCSTR sFileName);</p>		
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>pReadFileName: The file name (full path) of the source file to save to the CF card</p> <p>sFolderName: Folder name of the file to save to CF card (maximum 32 single-byte characters)</p> <p>sFileName: File name (maximum 8.3 string format) of the file to save to CF card</p>	<p>Return value (NULL)</p> <p>Successful operation: 0</p> <p>Problem operation: Error code</p>	

Function Name	CF card file to save (define type)	
<p>Function to save the file contents of the file to the CF card. Files you can save are limited to the file type defined in pWriteFileType.</p> <p>INT WINAPI EasyFileWriteCard(LPCSTR sNodeName, LPCSTR pReadFileName, LPCSTR sWriteFileType, WORD wWriteFileNo);</p>		
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>pReadFileName: The file name (full path) of the source file to save to the CF card</p> <p>sWriteFileType: File type of file to save to CF card See Function to read CF card file (file type), Special Items section</p> <p>wWriteFileNo: File number of file to save to CF card</p>	<p>Return value (NULL)</p> <p>Successful operation: 0</p> <p>Problem operation: Error code</p>	

Function Name	Delete CF card file (optionally define file name)																																											
<p>Deletes specified files in CF Card. You can optionally define the file to delete.</p> <p>INT WINAPI EasyFileDeleteInCfCard(LPCSTR sNodeName, LPCSTR sFolderName, LPCSTR sFileName) ;</p>																																												
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>sFolderName: Folder name of file on the CF card file to delete (Maximum 32 single-byte characters.)</p> <p>sFileName: File name in the maximum 8.3 string format to delete from the CF card.</p>	<p>Return value (NULL)</p> <p>Successful operation: 0</p> <p>Problem operation: Error code</p>																																											
<p>Special Item</p> <p>Supported File Types</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Data Class</th> <th style="text-align: center;">File Type</th> <th style="text-align: center;">Folder</th> </tr> </thead> <tbody> <tr> <td>Recipe (Filing Data)</td> <td>ZF or F</td> <td>FILE</td> </tr> <tr> <td>Recipe (CSV Data)</td> <td>ZR</td> <td>FILE</td> </tr> <tr> <td>Image Screen</td> <td>ZI or I</td> <td>DATA</td> </tr> <tr> <td>Sound Data</td> <td>ZO or O</td> <td>DATA</td> </tr> <tr> <td>GP-PRO/PB III for Windows exclusive trend graph data (compatible)</td> <td>ZT</td> <td>TREND</td> </tr> <tr> <td>GP-PRO/PB III for Windows exclusive sampling data (compatible)</td> <td>ZS</td> <td>TREND</td> </tr> <tr> <td>Alarm1</td> <td>Z1 or ZA</td> <td>ALARM</td> </tr> <tr> <td>Alarm2</td> <td>Z2 or ZH</td> <td>ALARM</td> </tr> <tr> <td>Alarm3</td> <td>Z3 or ZG</td> <td>ALARM</td> </tr> <tr> <td>Alarm4 to 8</td> <td>Z4 to Z8</td> <td>ALARM</td> </tr> <tr> <td>GP-PRO/PB III for Windows exclusive logging data (compatible)</td> <td>ZL</td> <td>LOG</td> </tr> <tr> <td>Capture data</td> <td>CP</td> <td>CAPTURE</td> </tr> <tr> <td>Sampling Group 1 to 64 data</td> <td>ZS1 to ZS64</td> <td>SAMP01 to SAMP64</td> </tr> </tbody> </table>			Data Class	File Type	Folder	Recipe (Filing Data)	ZF or F	FILE	Recipe (CSV Data)	ZR	FILE	Image Screen	ZI or I	DATA	Sound Data	ZO or O	DATA	GP-PRO/PB III for Windows exclusive trend graph data (compatible)	ZT	TREND	GP-PRO/PB III for Windows exclusive sampling data (compatible)	ZS	TREND	Alarm1	Z1 or ZA	ALARM	Alarm2	Z2 or ZH	ALARM	Alarm3	Z3 or ZG	ALARM	Alarm4 to 8	Z4 to Z8	ALARM	GP-PRO/PB III for Windows exclusive logging data (compatible)	ZL	LOG	Capture data	CP	CAPTURE	Sampling Group 1 to 64 data	ZS1 to ZS64	SAMP01 to SAMP64
Data Class	File Type	Folder																																										
Recipe (Filing Data)	ZF or F	FILE																																										
Recipe (CSV Data)	ZR	FILE																																										
Image Screen	ZI or I	DATA																																										
Sound Data	ZO or O	DATA																																										
GP-PRO/PB III for Windows exclusive trend graph data (compatible)	ZT	TREND																																										
GP-PRO/PB III for Windows exclusive sampling data (compatible)	ZS	TREND																																										
Alarm1	Z1 or ZA	ALARM																																										
Alarm2	Z2 or ZH	ALARM																																										
Alarm3	Z3 or ZG	ALARM																																										
Alarm4 to 8	Z4 to Z8	ALARM																																										
GP-PRO/PB III for Windows exclusive logging data (compatible)	ZL	LOG																																										
Capture data	CP	CAPTURE																																										
Sampling Group 1 to 64 data	ZS1 to ZS64	SAMP01 to SAMP64																																										

Function Name	Change CF card file name	
<p>Function to change the name of the file on the CF card.</p> <p>INT WINAPI EasyFileRenameInCfCard(LPCSTR sNodeName, LPCSTR sFolderName, LPCSTR sFileName, LPCSTR sFileRename) ;</p>		
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>sFolderName: Folder name of a file to be renamed in CF Card (maximum 32 characters)</p> <p>sFileName: File name to be renamed in CF Card (Max. 8.3 format text)</p> <p>sFileRename: Renamed file name (Max. 8.3 format text)</p>	<p>Return value (NULL)</p> <p>Successful operation: 0</p> <p>Problem operation: Error code</p>	

Function Name	Delete CF Card File																																											
<p>Deletes specified files in CF Card. Files to be deleted are limited to the file type specified in the "pDeleteFileType".</p> <p>INT WINAPI EasyFileDeleteCard(LPCSTR sNodeName, LPCSTR pDeleteFileType, WORD wDeleteFileNo);</p>																																												
<p>Argument</p> <p>sNodeName: The station name is fixed as #WinGP.</p> <p>pDeleteFileType: Delete File Types in CF Card (refer to Special Remarks.)</p> <p>wDeleteFileNo: File Numbers of Delete Files in CF Card</p>	<p>Return value (NULL)</p> <p>Successful operation: 0</p> <p>Problem operation: Error code</p>																																											
<p>Special Item</p> <p>When this function is called to the files that do not exist, the operation ends normally without resulting an error.</p> <p>The supported file types are as follows. You can only read items stored in the defined CF Card folder.</p> <p>■Supported File Types</p> <table border="1" data-bbox="200 846 1249 1439"> <thead> <tr> <th>Data Class</th> <th>File Type</th> <th>Folder</th> </tr> </thead> <tbody> <tr> <td>Recipe (Filing Data)</td> <td>ZF or F</td> <td>FILE</td> </tr> <tr> <td>Recipe (CSV Data)</td> <td>ZR</td> <td>FILE</td> </tr> <tr> <td>Image Screen</td> <td>ZI or I</td> <td>DATA</td> </tr> <tr> <td>Sound Data</td> <td>ZO or O</td> <td>DATA</td> </tr> <tr> <td>GP-PRO/PB III for Windows exclusive trend graph data (compatible)</td> <td>ZT</td> <td>TREND</td> </tr> <tr> <td>GP-PRO/PB III for Windows exclusive sampling data (compatible)</td> <td>ZS</td> <td>TREND</td> </tr> <tr> <td>Alarm1</td> <td>Z1 or ZA</td> <td>ALARM</td> </tr> <tr> <td>Alarm2</td> <td>Z2 or ZH</td> <td>ALARM</td> </tr> <tr> <td>Alarm3</td> <td>Z3 or ZG</td> <td>ALARM</td> </tr> <tr> <td>Alarm4 to 8</td> <td>Z4 to Z8</td> <td>ALARM</td> </tr> <tr> <td>GP-PRO/PB III for Windows exclusive logging data (compatible)</td> <td>ZL</td> <td>LOG</td> </tr> <tr> <td>Capture data</td> <td>CP</td> <td>CAPTURE</td> </tr> <tr> <td>Sampling Group 1 to 64 data</td> <td>ZS1 to ZS64</td> <td>SAMP01 to SAMP64</td> </tr> </tbody> </table>			Data Class	File Type	Folder	Recipe (Filing Data)	ZF or F	FILE	Recipe (CSV Data)	ZR	FILE	Image Screen	ZI or I	DATA	Sound Data	ZO or O	DATA	GP-PRO/PB III for Windows exclusive trend graph data (compatible)	ZT	TREND	GP-PRO/PB III for Windows exclusive sampling data (compatible)	ZS	TREND	Alarm1	Z1 or ZA	ALARM	Alarm2	Z2 or ZH	ALARM	Alarm3	Z3 or ZG	ALARM	Alarm4 to 8	Z4 to Z8	ALARM	GP-PRO/PB III for Windows exclusive logging data (compatible)	ZL	LOG	Capture data	CP	CAPTURE	Sampling Group 1 to 64 data	ZS1 to ZS64	SAMP01 to SAMP64
Data Class	File Type	Folder																																										
Recipe (Filing Data)	ZF or F	FILE																																										
Recipe (CSV Data)	ZR	FILE																																										
Image Screen	ZI or I	DATA																																										
Sound Data	ZO or O	DATA																																										
GP-PRO/PB III for Windows exclusive trend graph data (compatible)	ZT	TREND																																										
GP-PRO/PB III for Windows exclusive sampling data (compatible)	ZS	TREND																																										
Alarm1	Z1 or ZA	ALARM																																										
Alarm2	Z2 or ZH	ALARM																																										
Alarm3	Z3 or ZG	ALARM																																										
Alarm4 to 8	Z4 to Z8	ALARM																																										
GP-PRO/PB III for Windows exclusive logging data (compatible)	ZL	LOG																																										
Capture data	CP	CAPTURE																																										
Sampling Group 1 to 64 data	ZS1 to ZS64	SAMP01 to SAMP64																																										

Function Name	Get Free Space in CF Card	
<p>To acquire free space in CF Card connected to an assigned station.</p>		
<p>INT WINAPI EasyGetCfFreeSpace(LPCSTR sNodeName, INT* oiUnallocated);</p>		
<p>Argument sNodeName: The station name is fixed as #WinGP. oiUnallocated: Free Space in CF Card (Acquired in a byte unit)</p>	<p>Return value (NULL) Successful operation: 0 Problem operation: Error code</p>	
<p>Special Item</p>		

Function Name	Function Name	
<p>FTP Passive Mode Settings Communicates via FTP protocol to access CF Card FTP protocol in WinGP SDK supports Normal Mode and Passive Mode. This API sets each mode.</p>		
<p>INT WINAPI EasyFileSetPassiveMode(INT iPassive);</p>		
<p>Argument iPassive: (In) 0: Normal Mode Other than 0: Passive Mode Normal Mode is set at the time of WinGP SDK initialization.</p>	<p>Return value (NULL) Successful operation: 0 Problem operation: Error code</p>	
<p>Special Item</p>		

- Queuing Access Control API

Function Name	Start Queuing Device Read Request.	
<p>Queuing device read request until ExecuteQueuingAccess() is called after this API is called. Queuing is performed in a unit of WinGP SDK handle.</p> <p>INT WINAPI BeginQueuingRead();</p>		
Argument	Return value (NULL) Successful operation: 0 Problem operation: Error code	
<p>Special Item</p> <ul style="list-style-type: none"> • Do not call API to execute device write operations after calling BeginQueuingRead() until ExecuteQueuingAccess() is called. After these function calls, cache read and direct read commands will be queued. However, cache read and direct read commands cannot be mixed. • To cancel a queuing command, call CancelQueuingAccess(). • The maximum number of queuing commands is 1500, the maximum byte number is under 1 MB. 		

Function Name	Start Queuing Device Write Request	
<p>Queuing device read request until ExecuteQueuingAccess() is called . Queuing is performed in a unit of WinGP SDK handle.</p> <p>INT WINAPI BeginQueuingWrite();</p>		
Argument	Return value (NULL) Successful operation: 0 Problem operation: Error code	
<p>Special Item</p> <ul style="list-style-type: none"> • Do not call API to execute device write operations after calling BeginQueuingWrite() until the ExecuteQueuingAccess(). After these calling, cache write and direct write commands will be queued. However, cache write and direct write commands cannot be mixed. • To cancel a queuing command, call CancelQueuingAccess(). • The maximum number of queuing commands is 1500, the maximum byte number is under 1 MB. 		

Function Name	Start Queuing Device Read/Write Request	
<p>Accesses to device data according to queuing device read/write request.</p>		
<p>INT WINAPI ExecuteQueuingAccess();</p>		
Argument	Return value (NULL) Successful operation: 0 Problem operation: Error code	
<p>Special Item</p> <ul style="list-style-type: none"> • When access to all devices succeeded, ExecuteQueuingAccess() returns the successful completion and when access to any device failed, it returns an access error. If you want to know whether each access was successful or not, call IsQueuingAccessSucceeded() to check for details. • No action can be registered to queuing access. 		

Function Name	Cancel Queuing Device Read/Write Request	
<p>Cancels queuing device read/write request.</p>		
<p>INT WINAPI CancelQueuingAccess();</p>		
Argument	Return value (NULL) Successful operation: 0 Problem operation: Error code	
<p>Special Item</p> <p>Until ExecuteQueuingAccess() is called after calling BeginQueuingWrite() or BeginQueuingRead(), the queuing device access request continues.</p> <p>If the request is no longer required, call this API. The API cancels the request and ends the queuing operation.</p>		

Function Name	Cancel Queuing Device Read/Write Request	
<p>Asks to check whether the device access to ExecuteQueuingAccess() succeeded or not after ExecuteQueuingAccess() is called.</p> <p>INT WINAPI IsQueuingAccessSucceeded(INT iIndex);</p>		
<p>Argument iIndex: (In) Checking Request No.</p> <p>When BeginQueuingWrite() or BeginQueuingRead() is called, API for device access is called several times to queue the device access requests until ExecuteQueuingAccess() is called. However, the actual device access results are only available after ExecuteQueuingAccess() is executed.</p> <p>To find out the result of device access, indicates a request number (a number from 0) of the device after ExecuteQueuingAccess() is executed.</p>	<p>Return value (NULL) XX: Error Code 0: Device access to the specified number was successful.</p>	
<p>Special Item For example: BeginQueuingWrite(); WriteDevice16("Node1","LS100",Data,10); WriteDevice16("Node1","LS200",Data,10); WriteDevice16("Node1","LS300",Data,10); ExecuteQueuingAccess() To check whether access to "LS200" in "Node 1" was successful with the above registration, use IsQueuingAccessSucceeded(1). If 0 is returned, the access was successful.</p>		

◆ Data Type

- Basic data type to specify the data type or receive the data as response in API

Definition name	Decimal	Hexadecimal	Description
EASY_AppKind_Bit	1	0x0001	Bit Data
EASY_AppKind_SignedWord	2	0x0002	16 Bit Signed Data
EASY_AppKind_UnsignedWord	3	0x0003	DataWithout16 bit code
EASY_AppKind_HexWord	4	0x0004	16 Bit Hex Data
EASY_AppKind_BCDWord	5	0x0005	16-bit BCD data
EASY_AppKind_SignedDWord	6	0x0006	32 Bit Signed Data
EASY_AppKind_UnsignedDWord	7	0x0007	DataWithout32 bit code
EASY_AppKind_HexDWord	8	0x0008	32 Bit Hex Data
EASY_AppKind_BCDDWord	9	0x0009	32-bit BCD data
EASY_AppKind_Float	10	0x000A	Single float number data
EASY_AppKind_Real	11	0x000B	Double float number data
EASY_AppKind_Str	12	0x000C	Text data

- Data type available in special cases

Definition name	Decimal	Hexadecimal	Description
EASY_AppKind_NULL	0	0x0000	Default (Write the existing contents) Shows that API is using the data type defined by the symbol for API that can use symbols as the device address.
EASY_AppKind_BOOL	513	0x0201	BOOL (Write the existing contents) Handles the Bit data in 1-bit unit and as VARIANT-type BOOL.

◆ Specify the Device/PLC

When specifying a device in GP-Pro EX, selecting a symbol name also selects the connected device/PLC. In the device access API, you need to also define the connected device/PLC name.

For example, `ReadDevice 16("#WinGP","PLC.1 valve", Data,10);`

◆ Device Length**Operation When Accessing a 16-bit Device Using 32-bit Access**

WinGP allocates 32-bit symbols to 16-bit devices. When you use a symbol or directly use the 32-bit data type to access, it allows the 16-bit device to handle the data as a 32-bit device.

In such a case, WinGP sees the two sequential 16-bit devices as one

◆ Symbol Index (16 Bits)

Only device names defined in the device access API can specify the index of symbols. The index specification of symbol is used to specify an address offset using brackets [] after the symbol name as shown below. The device address is incremented by the specified numeric value multiplied by the size of the symbol data type.

(Symbol name)[Numeric Value]

For example, `Valve[2]`

If a symbol "valve" is assigned to D100 and is signed 16-bit, it indicates D102. If it is assigned to D100 and is unsigned 32-bit, it indicates D104.

◆ Windows Message Processing

Many Windows programs are event-driven programs that display dialogs and output sounds corresponding to events such as "clicking an icon," "moving the mouse," and "pressing a key."

When any such event occurs, Windows sends the application a message that indicates the type of event.

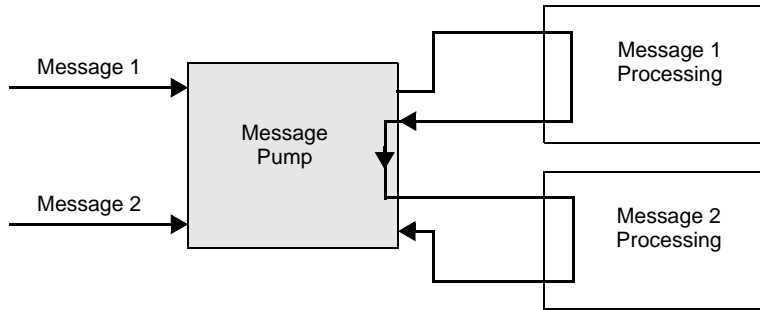
Upon receiving the message, the applications acknowledge that the event has occurred and execute the processing.

In this document, the part that receives messages in order from Windows and branches them into the respective processing (DoEvents, in VB, and the part where GetMessage() and DispatchMessage() are performed in VC) is called the message pump.

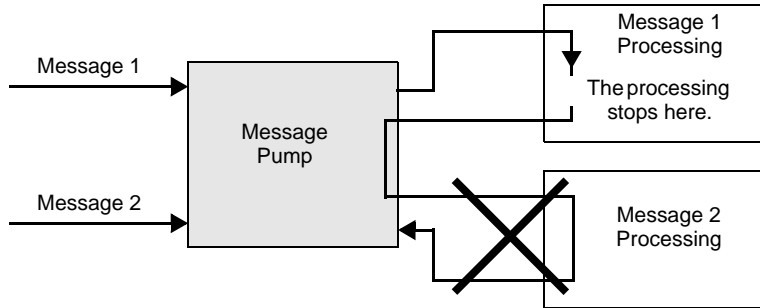
When normally programmed in VC and VB, the message pump hides in the VC and VB framework. If the message pump does not perform properly, Windows applications perform unintended operations.

For example, if a routine takes a long time to process a message and does not return, the application cannot receive an event from Windows during the processing time and cannot process the event.

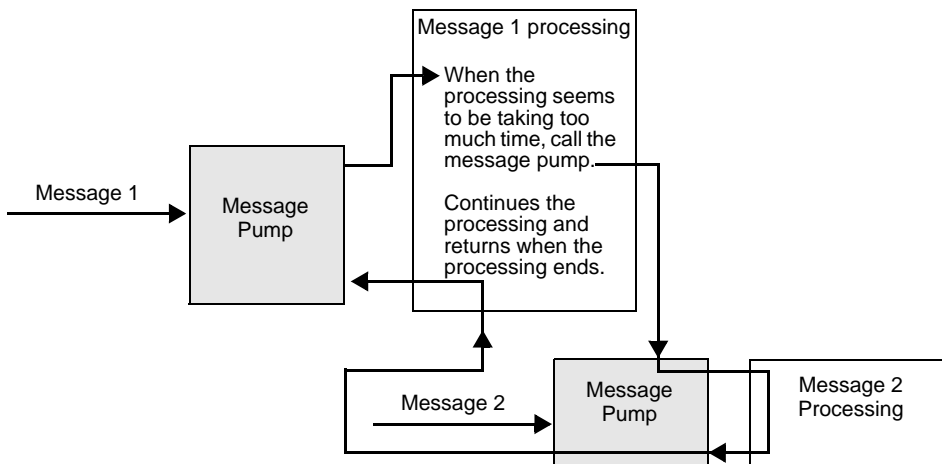
For example, when Windows sends messages in the order of Message 1 and Message 2. The message pump retrieves Message 1 and calls a subroutine for Message 1. Once returning, it retrieves the next message (Message 2) and calls a subroutine for Message 2.



If it takes a long time to process Message 1 at this point, the message pump does not return and the message pump processing 2 cannot be performed.



In such case, force the operation of the message pump. (Referred to DoEvents in VB and GetMessage() and DispatchMessage() in , VC)



Windows applications are designed based on applications to operate the message pump properly. In order to prevent such an event as shown in the example, WinGP SDK operates the message pump in the function when processing takes too long.

◆ Prohibit Double Calling API

- API double calling

WinGP SDK prohibits calling another device access API during a call to one device access API (Double calling). However, the device access API is operating the message pump in the API, so if an event happens, the user program starts.

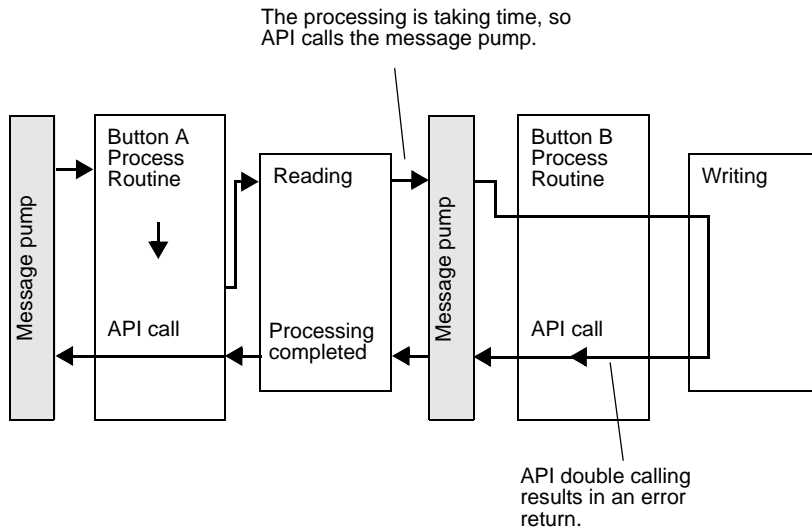
In the course of the message processing routine, double calling might occur when API is called.

The following shows a case resulting in double calling:

(1) Pressing two buttons results in double calling

There are two buttons of A and B. If you press A, it calls the device read API. If you press B, it calls the device write API.

In this case, if you press B button while calling the device read API while pressing A button, the device write API is also called, which leads to API double calling and an error results.



(2) Double calling with a timer

A timer event is often used for cyclic processing in Windows program. Program carefully for programs using the timer event; otherwise, API double calling might result.

- 1) Call, read, and display the device read API cyclically once every second.
- 2) Pressing the button calls the device write API and writes the value into the device.

The following situations will cause errors when using the timer event.

- During reading triggered by a timer event in 1), the 2) button is pressed and the 2) processing starts.
- During the 2) writing, a timer event occurs and the 1) reading is performed.

- Solutions to avoid API double calling

The following shows solutions to avoid API double calling.

- (1) In the user program, improve the algorithm to prevent API double calling.

For example,

- Always cancel the timer at the start of the timer processing routine and the button processing routine.
- During a processing triggered by 1 button pressed, ignore any other button pressed or if the button is pressed again.

- (2) Do not allow message processing in API.

Call `EasySetWaitType()` with the argument 2. In this case, other messages than that causing double calling are not processed either, which may lead to other problems such as the applications performing unintended operations.

◆ Reading Text in VB

There are two ways to read text in VB as shown below.

- (1) Using `ReadDeviceStr` in VB to read text

In this case, you need to specify (fix) the location size to store the already read text.

```
Public Sub Sample1()
```

```
    Dim strData As String * 10' Correct specification method specifying the read size
```

```
    'Dim strData As String      ' Wrong specification method not specifying the text size
```

```
    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) Using ReadDeviceVariant in VB to read text

If not specifying the location size to store the already read text, use Variant type.

```
Public Sub Sample2()
```

```
    Dim IErr As Long
```

```
    Dim vrData As Variant    'For the location to store the read data, specify the Variant  
                             type.
```

```
    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
```

It should be noted that WinGP SDK uses NULL at the end of the text. Thus, text acquired by the above method has the NULL character at the end, which needs to be removed.

The following shows sample functions to shorten the text up to the NULL.

```
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
```

◆ Error Code List

Error Code that can be checked with "return value".

NOTE

- The terms "Pro-Server" and "Pro-Studio" in the Error Messages are required to be replaced as "WinGP SDK".

- "REAA****" Error Info

Error Code	Error Message	Cause and Troubleshooting
0xC0A10010 REAA016 -1063190512 3231776784	Could not use the XX port (No: XX). (XX: Port name/No.)	Could not use the XX port (No: XX). The system port number may have already been used.
0xC0A10011 REAA017 -1063190511 3231776785	Attempted to access a write-protect area (XX) (XX: Device name)	Cannot write to Write Inhibit Area (LS0000-LS0019, LS2032-LS2095, LS9000-LS9999) via D-Script or Network.
0xC0A10012 REAA018 -1063190510 3231776786	Attempted to access a device outside the address range (XX) (XX: Device name)	A device beyond the valid device range was accessed.
0xC0A10015 REAA021 -1063190507 3231776789	An invalid ID (Node, Device, Address) has been specified.	An invalid ID was specified. Attempted to access a nonexistent device.
0xC0A10016 REAA022 -1063190506 3231776790	An invalid ID (Node, Device, Address) has been specified.	
0xC0A1001A REAA026 -1063190502 3231776794	Illegal/Undefined Device Address	An invalid device was specified. Attempted to access a nonexistent device.
0xC0A1001B REAA027 -1063190501 3231776795	Illegal/Undefined Device Address	
0xC0A1001C REAA028 -1063190500 3231776796	Illegal/Undefined Device Address	

- * 1st line: Error code.
2nd line: Integrated error code.
3rd line: Error code with decimal code.
4th line: Error code without decimal code.

- "RYAA****" Error Info

Error Code	Error Message	Cause and Troubleshooting
0xC0AF0001 RYAA001 -1062273023 3232694273	The specified shared memory already exists.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0AF0002 RYAA002 -1062273022 3232694274	The specified shared memory does not exist.	
0xC0AF0003 RYAA003 -1062273021 3232694275	A shared memory already exists, but its memory size is less than specified.	Please close another application or restart the OS.
0xC0AF0004 RYAA004 -1062273020 3232694276	Cannot create a shared memory due to insufficiency of memory or resource.	
0xC0AF0005 RYAA005 -1062273019 3232694277	Could not start TdasEngine because it is already in execution or termination.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0AF0006 RYAA006 -1062273018 3232694278	Could not stop TdasEngine because it is already in suspension or termination.	
0xC0AF0007 RYAA007 -1062273017 3232694279	Could not register the operation in TdasEngine.	
0xC0AF0008 RYAA008 -1062273016 3232694280	Cannot execute State Transition of TdaInfo because a small service is now in transition.	
0xC0AF0009 RYAA009 -1062273015 3232694281	The device name (XX) specified as the destination NODE does not exist. (XX: Device/PLC name)	
0xC0AF000A RYAA010 -1062273014 3232694282	Cannot execute the operation due to the invalid state of the small service.	
0xC0AF000B RYAA011 -1062273013 3232694283	Cannot execute the operation because the small service is not in operation.	

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0AF00C RYAA012 -1062273012 3232694284	Cannot execute the operation because the small service is not in operation.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0AF00D RYAA013 -1062273011 3232694285	The I/F of an unsupported small service was called.	
0xC0AF010 RYAA016 -1062273008 3232694288	Could not register the item because of insufficient memory.	Please close another application or restart the OS.
0xC0AF011 RYAA017 -1062273007 3232694289	Accessed a device in which no item is registered.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0AF012 RYAA018 -1062273006 3232694290	Accessed an out-of-range device.	A device beyond the valid device range was accessed.
0xC0AF013 RYAA019 -1062273005 3232694291	Failed to register the specified cluster because an invalid item is specified in it.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0AF014 RYAA020 -1062273004 3232694292	The specified data type is invalid.	
0xC0AF015 RYAA021 -1062273003 3232694293	The specified access type is illegal.	
0xC0AF016 RYAA022 -1062273002 3232694294	The specified data type is illegal.	
0xC0AF017 RYAA023 -1062273001 3232694295	The no. of data you specified is too many to write (Please reduce it to XX or fewer) (XX: Data number)	

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0AF0018 RYAA024 -1062273000 3232694296	The operation result to write is below the lower limit value.	Attempted to write an out of range value. Please change the setting to write an in-range value.
0xC0AF0019 RYAA025 -1062272999 3232694297	The operation result to write is beyond the upper limit value.	
0xC0AF001A RYAA026 -1062272998 3232694298	Could not send processing request to the network destination due to insufficient memory.	Please close another application or restart the OS.
0xC0AF001B RYAA027 -1062272997 3232694299	The specified group was not found.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0AF001C RYAA028 -1062272996 3232694300	The two compared access tickets differ in their nodes, equipment, or devices.	
0xC0AF001D RYAA029 -1062272995 3232694301	The specified access ticket is not for this node.	
0xC0AF001E RYAA030 -1062272994 3232694302	Could not register the cache because of insufficient memory.	Please close another application or restart the OS.
0xC0AF0020 RYAA032 -1062272992 3232694304	The access ticket you tried to use in block access is not of the block type.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0AF0021 RYAA033 -1062272991 3232694305	The small service to process was not found.	
0xC0AF0022 RYAA034 -1062272990 3232694306	The size of block access to the device exceeded the limit.	The max buffer size for Device Block Write/Read is 10KB. Please set a size less than that.
0xC0AF0023 RYAA035 -1062272989 3232694307	A different network project is being used.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0AF0030 RYAA048 -1062272976 3232694320	A communication error occurred during communication with the destination node. Please confirm that the network connection to the node has been properly established.XX (XX: Destination node name)	Check that the LAN card settings are correct.
0xC0AF0031 RYAA049 -1062272975 3232694321	The destination node did not responded within the specified time. Please confirm that the network connection to the node has been properly established.XX (XX: Destination node name)	
0xC0AF0032 RYAA050 -1062272974 3232694322	The destination node did not responded within the specified time. Please confirm that the network connection to the node has been properly established.XX (XX: Destination node name)	
0xC0AF0033 RYAA051 -1062272973 3232694323	Communication with the destination Node stopped because the destination or local Node closed.	Set 'WinGP' node online.
0xC0AF0040 RYAA064 -1062272960 3232694336	Failed to read the device.	The data may have been read into an illegal or undefined device address. Please specify a proper device address.
0xC0AF0041 RYAA065 -1062272959 3232694337	Failed to write the device.	

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0AF0045 RYAA069 -1062272955 3232694341	The specified request is not supported.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0AF0046 RYAA070 -1062272954 3232694342	The specified request is not supported.	
0xC0AF0050 RYAA080 -1062272944 3232694352	The project ID of the network project file is different. (A different network project is being used.)	
0xC0AF0051 RYAA081 -1062272943 3232694353	The network project file does not have necessary data.	
0xC0AF0052 RYAA082 -1062272942 3232694354	The network project file is damaged.	
0xC0AF0053 RYAA083 -1062272941 3232694355	The network project file does not exist.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0AF0067 RYAA103 -1062272921 3232694375	Operation was interrupted because GP Online was terminated.	The operation was interrupted because 'WinGP' Online was terminated. To finish the operation, put 'WinGP' online and retry.

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

- "SAAA****" Error Info

Error Code	Error Message	Cause and Troubleshooting
0xC0B00001 SAAA001 -1062207487 3232759809	System Error	Please restart your PC. It must be other than 0, correctly created, and not discarded.
0xC0B00002 SAAA002 -1062207486 3232759810	Cannot process due to a shortage of OS resource or memory.	
0xC0B00003 SAAA003 -1062207485 3232759811	Cannot execute any new process until the server returns a processing result.	
0xC0B00004 SAAA004 -1062207484 3232759812	The process was interrupted because Pro-Server EX was terminated.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B00005 SAAA005 -1062207483 3232759813	The process was interrupted because Pro-Server EX was terminated during the process.	
0xC0B00006 SAAA006 -1062207482 3232759814	Cannot process because Pro-Server EX has already been terminated.	
0xC0B00007 SAAA007 -1062207481 3232759815	The specified connector has already been registered. The application is already in execution.	Please restart your PC. It must be other than 0, correctly created, and not discarded.
0xC0B00008 SAAA008 -1062207480 3232759816	An error occurred in an OLE function. Cannot convert the data.	
0xC0B0000A SAAA010 -1062207478 3232759818	Cannot refer to the resource because Pro-Server EX has not been started.	
0xC0B0000B SAAA011 -1062207477 3232759819	Cannot request the system to execute processing because Pro-Server EX has not been started.	
0xC0B0000C SAAA012 -1062207476 3232759820	The system is broken. Cannot process.	

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B00011 SAAA017 -1062207471 3232759825	An error occurred when accessing the XX file. The file is either locked (shared) or broken. (XX: File name)	Please restart your PC. It must be other than 0, correctly created, and not discarded.
0xC0B00012 SAAA018 -1062207470 3232759826	Too many connectors to register.	
0xC0B00029 SAAA041 -1062207447 3232759849	Failed to get device info from the PRW file.	The screen project file may be damaged. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B0002A SAAA042 -1062207446 3232759850	Failed to get symbol info from the PRW file.	
0xC0B0002B SAAA043 -1062207445 3232759851	Failed to get the device address from the PRW file.	
0xC0B0002C SAAA044 -1062207444 3232759852	Failed to get setting info from the PRX file.	
0xC0B0002D SAAA045 -1062207443 3232759853	Failed to create a temporary file.	
0xC0B0002E SAAA046 -1062207442 3232759854	Cannot open the PRX file.	The screen project file may be damaged. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B0002F SAAA047 -1062207441 3232759855	Failed to delete the temporary file.	Please execute it again.

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B00030 SAAA048 -1062207440 3232759856	The specified screen file has an error. XX	The screen project file may be damaged. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B00031 SAAA049 -1062207439 3232759857	The PRW file does not have necessary data.	
0xC0B00032 SAAA050 -1062207438 3232759858	The specified file is not a PRW file.	
0xC0B00062 SAAA098 -1062207390 3232759906	The network project file is broken. Cannot read. Please confirm whether the file you specified is a real network project file.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B00063 SAAA099 -1062207389 3232759907	Cannot write to the network project file.	Please check whether the disk space is sufficient, and the hard disk has any problem.
0xC0B00064 SAAA100 -1062207388 3232759908	The file is not a network project file, or its version is old. Cannot read the data.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B00065 SAAA101 -1062207387 3232759909	The specified device was not found in (XX). It may have been deleted or renamed. Please check it again. (XX: NODE name)	
0xC0B00066 SAAA102 -1062207386 3232759910	The specified NODE (XX) has not been registered. There is a conflict. Please check it again. (XX: NODE name)	
0xC0B00067 SAAA103 -1062207385 3232759911	The specified NODE info is incorrect. No NODE info exists.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B00068 SAAA104 -1062207384 3232759912	The device setting in the system area of the specified NODE (XX) has an error. Please check the device you set. (XX: NODE name)	

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B00069 SAAA105 -1062207383 3232759913	(XX: XX) is invalid as a device/ symbol. Cannot analyze. (XX: Device/Symbol name)	A nonexisting device/symbol name has been specified. Please confirm the symbol, following the direction of the message. If it does not solve the problem, the screen project file may be damaged. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B0006C SAAA108 -1062207380 3232759916	The network setting is broken.	Please review the network settings.
0xC0B00078 SAAA120 -1062207368 3232759928	(Symbol Sheet: XX Symbol: XX Address: XX) is invalid as a device address. (XX: Symbol Sheet name, XX: Symbol name, XX: Address)	The screen project file may be damaged. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B0007C SAAA124 -1062207364 3232759932	(Symbol Sheet: XX Symbol: XX Address: XX) is beyond the valid device range. (XX: Symbol Sheet name, XX: Symbol name, XX: Address)	The screen project file may be damaged. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B00082 SAAA130 -1062207358 3232759938	The specified NODE (XX) has not been registered in the network project. (XX: NODE name)	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.
0xC0B00083 SAAA131 -1062207357 3232759939	The specified NODE (XX) is not a GP2000 Series NODE. (XX: NODE name)	
0xC0B00084 SAAA132 -1062207356 3232759940	The device of the specified NODE (XX) is not supported. (XX: NODE name)	
0xC0B00095 SAAA149 -1062207339 3232759957	(Symbol Sheet: XX Symbol: XX No. of Devices:XX) is beyond the range of the no. of devices (Valid Range:XX-XX)	Decrease the number of symbols registered on the GP-Pro EX Symbol screen.
0xC0B00096 SAAA150 -1062207338 3232759958	(Symbol Sheet: XX Group: XX) has the no. of rows beyond the limit. Please reduce it. (XX rows or less)	

- * 1st line: Error code.
2nd line: Integrated error code.
3rd line: Error code with decimal code.
4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B0009C SAAA156 -1062207332 3232759964	In a symbol sheet, 2 symbol/ group names are the same. (Symbol Sheet: XX Name1: XX Name2: XX)	Change the name of the symbol registered on the 'GP-Pro EX' Symbol screen.
0xC0B0009D SAAA157 -1062207331 3232759965	The device driver of (Node: XX) is not supported. (Necessary driver has not been installed.) (XX: NODE name)	Please install according to Device/PLC differences.
0xC0B000A9 SAAA169 -1062207319 3232759977	(%s:%s)The specified Device/ Symbol is beyond the valid device range. (xx: Device, xx: Number)	A device beyond the valid device range was accessed.
0xC0B000E0 SAAA224 -1062207264 3232760032	Warning: In different symbol sheets, 2 symbol/group names are the same. To use the same name, please specify the sheet name to which this symbol/group belongs. (xx: Existing Symbol Sheet name, xx: Symbol Sheet, Symbol/Group: (Sheet: XX Name: XX)) (XX: Existing Symbol Sheet name, XX: Symbol Sheet name to which Same Symbol name belongs, XX: Same Symbol name)	In the 'GP-Pro EX' Symbol setting screen, please change the name to avoid name duplication.
0xC0B000E1 SAAA225 -1062207263 3232760033	Warning: The symbol/group name is the same as a symbol sheet name. To use the same name, please specify the sheet name to which this symbol/group belongs.(xx: Symbol Sheet, Symbol/Group: (Sheet: XX Name: XX)) (XX: Existing Symbol Sheet name, XX: Symbol Sheet name to which Same Symbol name belongs, XX: Same Symbol name)	

- * 1st line: Error code.
2nd line: Integrated error code.
3rd line: Error code with decimal code.
4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B000E4 SAAA228 -1062207260 3232760036	Warning: The array variable (XX) has too many elements for API Communication for WinGP to access the whole array. In API Communication for WinGP, only XX elements from the head are accessible. (XX: Symbol name, XX: Array Element number)	(1) Consider registering the array by splitting it into multiple parts in 'GP-Pro EX'. (2) If the array cannot be split, when importing the 'GP-Pro EX' project file to the network project in 'Pro-Server EX', there is a function that automatically splits array variables that exceed the number that can be accessed simultaneously, and registers them as multiple symbols. Consider using 'Pro-Server EX' instead of 'WinGP SDK'.

- * 1st line: Error code.
2nd line: Integrated error code.
3rd line: Error code with decimal code.
4th line: Error code without decimal code.

- "SAAF***" Error Info

Error Code	Error Message	Cause and Troubleshooting
0xC0B00201 SAAF001 -1062206975 3232760321	Cannot initialize TCP/IP.	From Windows' s [Control Panel]-[Network Connection] please confirm that connection setting is enabled and that the TCP/IP protocol has been installed, which can be confirmed in the property of the connection setting. 'WinGP SDK' does not work without the TCP/IP.
0xC0B00203 SAAF003 -1062206973 3232760323	This PC does not have a valid IP address allocated. Please check the TCP/IP environment of this PC.	Please confirm that the LAN card works properly. Please check the LAN cable, too.
0xC0B00204 SAAF004 -1062206972 3232760324	Cannot load the PLCInfo.xml file.	Please update the protocol driver. If it does not solve the problem, please install 'WinGP SDK' again.
0xC0B00205 SAAF005 -1062206971 3232760325	Cannot load the Editor Driver.	

- * 1st line: Error code.
2nd line: Integrated error code.
3rd line: Error code with decimal code.
4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B00206 SAAF006 -1062206970 3232760326	An error occurred in Active X I/F.	Please confirm that the OS version is appropriate. If the phenomenon still reoccurs despite the restart, please install 'WinGP SDK' again.
0xC0B00207 SAAF007 -1062206969 3232760327	Cannot execute because of the version inconsistency of DLL and EXE for Pro-Server EX. This program will be shut down. (xx: program name)	Please confirm that there are not multiple different versions of 'Pro-Server EX' or DLLs of 'WinGP SDK' are installed in one PC. Only 1 version of 'Pro-Server EX' or 'WinGP SDK' can be installed in a PC.
0xC0B00209 SAAF009 -1062206967 3232760329	The file Core. ID was not found.	Please restart your PC. If it does not solve the problem, please install 'WinGP SDK' again.
0xC0B0020B SAAF011 -1062206965 3232760331	ProNet.dll has not been installed properly	
0xC0B0020C SAAF012 -1062206964 3232760332	Cannot start Pro-Server EX. Please close all the applications that use Pro-Studio EX or Pro-Server EX, and try again.	Cannot start 'WinGP SDK' because 'WinGP SDK' or an application using 'WinGP SDK' may not have stopped normally. Please close 'WinGP SDK' and all the applications running on it, and then try again.
0xC0B00211 SAAF017 -1062206959 3232760337	This API is not supported.	The API you use is unavailable. Please consider another method.
0xC0B00212 SAAF018 -1062206958 3232760338	The specified string is invalid as a device address.	Please reconfirm the address specification method. Please confirm that no change has been made to devices and nodes. Please confirm that the necessary device driver has been installed.
0xC0B00213 SAAF019 -1062206957 3232760339	The specified device supports bit access only.	Please confirm the device to access and the access method.
0xC0B00214 SAAF020 -1062206956 3232760340	The specified device driver is not supported (The necessary device driver has not been installed).	Please install according to Device/PLC differences.

- * 1st line: Error code.
2nd line: Integrated error code.
3rd line: Error code with decimal code.
4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B00215 SAAF021 -1062206955 3232760341	The parameter value is invalid.	Please review the API parameters.
0xC0B00216 SAAF022 -1062206954 3232760342	The device no. is out of range.	Please check the device number.
0xC0B00217 SAAF023 -1062206953 3232760343	The specified device does not exist.	Please check that the Device/PLC settings or System Area Start Address settings are correct.
0xC0B00218 SAAF024 -1062206952 3232760344	The specified group symbol does not exist.	Please check the group symbol specification.
0xC0B0021A SAAF026 -1062206950 3232760346	In Queuing Access, read-access and write-access, or cache access and direct access, cannot be mixed.	Please confirm that no different access method exists between the start of queuing and the actual access. If there is the necessity of using a different access method, please use another queuing access.
0xC0B0021D SAAF029 -1062206947 3232760349	The specified node has not been registered in the network project.	Please check the node specification.
0xC0B0021F SAAF031 -1062206945 3232760351	The API was redundantly called. The specified access handle for Pro-Server EX is already running.	Consider using EasySetWaitType() to avoid calling the API simultaneously.
0xC0B00220 SAAF032 -1062206944 3232760352	In data-type conversion, the data type of the conversion source/destination is unsupported.	Please check the contents of the Variant type.
0xC0B00221 SAAF033 -1062206943 3232760353	Backup data type specified is not supported.	Please check the data type specification.
0xC0B00222 SAAF034 -1062206942 3232760354	Failed to open the SRAM backup data file or to create its copy in the PC.	Please check the specifications of the destination file/folder in the PC, disk space, and the access rights to the file, etc.

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B00223 SAAF035 -1062206941 3232760355	In Read/Write Backup Data, failed to access the file.	In reading or writing SRAM Backup Data, an error occurred in accessing the specified file. Please check the free space of the PC and the file access right, and then execute it again.
0xC0B00224 SAAF036 -1062206940 3232760356	In Write SRAM Backup Data, the specified file size is too large. It must be 96KB or less.	Please confirm that the file specified in Write SRAM Backup Data is correct. Also, please specify a file of the size of 96Kbytes or less.
0xC0B00225 SAAF037 -1062206939 3232760357	Numeric value error. Please set a correct value.	Please confirm that the string is valid as a numeric value.
0xC0B00226 SAAF038 -1062206938 3232760358	The specified data count is 0 or out of range.	Please check the data count.
0xC0B00227 SAAF039 -1062206937 3232760359	The max number of access destinations is too high (It must be 1500 or less).	Please consider dividing it for successful access.
0xC0B00228 SAAF040 -1062206936 3232760360	The total buffer size of the data to access is too high. (It must be 1MB or less.)	
0xC0B00230 SAAF048 -1062206928 3232760368	Cannot start Pro-Server EX.	Please restart your PC. If it does not solve the problem, please install 'WinGP SDK' again.
0xC0B00238 SAAF056 -1062206920 3232760376	Reading out logging data from a GP3000 Series / WinGP NODE is not allowed .	Please change the setting to not execute Read Logging Data when the target is a 'WinGP' Node.
0xC0B00239 SAAF057 -1062206919 3232760377	Reading out trend data from a GP3000 Series / WinGP NODE is not allowed .	Please change the setting to not execute Read Trend Data when the target is a 'WinGP' Node.
0xC0B00240 SAAF064 -1062206912 3232760384	The specified access handle for Pro-Server EX is invalid.	Please restart your PC. It must be other than 0, correctly created, and not discarded.
0xC0B00241 SAAF065 -1062206911 3232760385	Cannot continue because this command is unsupported.	Please restart your PC. If it does not solve the problem, please install 'WinGP SDK' again.

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B00242 SAAF066 -1062206910 3232760386	Cannot process because Pro-Server EX stopped.	Please exit all applications before you close 'WinGP SDK'.
0xC0B00243 SAAF067 -1062206909 3232760387	While waiting for a processing result from the server, the API received the application quitting message.	If you do not want to receive WM_QUIT, please use a multihandle system API in EasySetWaitTypeM(2).
0xC0B00244 SAAF068 -1062206908 3232760388	The file name consists of more than 256 characters. Supposed to be within 256 characters.	Please check the file name specification.
0xC0B00245 SAAF069 -1062206907 3232760389	Queuing access registration has not started.	Please check the sequence of the program.
0xC0B00246 SAAF070 -1062206906 3232760390	Actual queuing access has not been made.	
0xC0B00247 SAAF071 -1062206905 3232760391	The device access to the specified no. failed.	Please check the cable or device operating environment.
0xC0B00248 SAAF072 -1062206904 3232760392	The device access with the specified no. has not been registered. Please check the preregistered access count and no.	Please check the sequence of the program.
0xC0B0024C SAAF076 -1062206900 3232760396	The specified group no. is not within the range of sampling data group no.	Please review the API parameters.
0xC0B0024D SAAF077 -1062206899 3232760397	In Queuing Access, Read and Write cannot be mixed.	Please check the sequence of the program.
0xC0B00250 SAAF080 -1062206896 3232760400	No word exists.	Please review the API parameters.
0xC0B00251 SAAF081 -1062206895 3232760401	Invalid name/word. Illegal characters are included.	

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B00252 SAAF082 -1062206894 3232760402	The specified node has not been registered in the network project.	Please review the API parameters.
0xC0B00253 SAAF083 -1062206893 3232760403	The specified device has not been registered.	
0xC0B00254 SAAF084 -1062206892 3232760404	Array Index Specification Error	Please check the array specification method.
0xC0B00255 SAAF085 -1062206891 3232760405	The specified device is an undefined symbol or an invalid address.	Please check the device address specification method.
0xC0B00256 SAAF086 -1062206890 3232760406	The symbol name is invalid, or the group specification is too deeply nested.	
0xC0B00257 SAAF087 -1062206889 3232760407	Index specification is unavailable for a string-type symbol.	
0xC0B00258 SAAF088 -1062206888 3232760408	The specified index value is too high.	
0xC0B00259 SAAF089 -1062206887 3232760409	Group symbol specification is unavailable for this device specification.	
0xC0B0025A SAAF090 -1062206886 3232760410	Please specify a group symbol to specify a device.	
0xC0B0025B SAAF091 -1062206885 3232760411	The symbol sheet name is invalid, or it is unavailable for the specified device.	
0xC0B0025C SAAF092 -1062206884 3232760412	Device names are redundantly specified.	A fatal error occurred. Restart 'WinGP' and 'WinGP SDK' after executing a forced transfer in 'GP-Pro EX'.

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B0025D SAAF093 -1062206883 3232760413	Cannot use the specified symbol because its data type is different from the one required here.	The symbol data type and the specified data type are different and cannot be used. Check the Symbol name or data type.
0xC0B0025E SAAF094 -1062206882 3232760414	Failed to analyze the option-specifying string.	Please review the API parameters.
0xC0B00262 SAAF098 -1062206878 3232760418	Failed to read the file.	Please confirm that the specified file exists in the CF-card folder. If exists, please confirm the right of access to the file.
0xC0B00263 SAAF099 -1062206877 3232760419	Failed to writing to the file.	Please check the access rights to the write destination. If there is no problem with the access right, please check whether the CF-card has enough free space.
0xC0B00264 SAAF100 -1062206876 3232760420	The specified file was not found.	Please confirm that the specified file exists.
0xC0B00265 SAAF101 -1062206875 3232760421	Failed to delete the file.	Please confirm that the specified file exists in the CF-card folder. If exists, please confirm the right of access to the file.
0xC0B00266 SAAF102 -1062206874 3232760422	Failed to rename the file.	Please confirm that the specified file exists in the CF-card folder. If it does, please check the access right to the file and whether the new file name does not contain any forbidden characters.
0xC0B00267 SAAF103 -1062206873 3232760423	Cannot open the file list retention file.	Please check the access rights to the destination folder. If there is no problem with the access right, please check whether the drive has enough free space.
0xC0B00269 SAAF105 -1062206871 3232760425	No file name has been inputted.	Please input a file name.
0xC0B0026A SAAF106 -1062206870 3232760426	Too long file path.	Please shorten the file path.

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code	Error Message	Cause and Troubleshooting
0xC0B0026C SAAF108 -1062206868 3232760428	Connection to GP3000 Series NODE was reset.	After confirming the GP3000 Series NODE/'WinGP' Node is still on and the cable is properly connected, please execute it again
0xC0B0026D SAAF109 -1062206867 3232760429	The destination NODE does not respond.	
0xC0B0026E SAAF110 -1062206866 3232760430	Could not complete the operation because connection was broken during the process.	
0xC0B0026F SAAF111 -1062206865 3232760431	Cannot connect to the specified node because it does not exist.	Please use the #WinGP node name.
0xC0B00272 SAAF114 -1062206862 3232760434	The parameter value is invalid.	Please review the inputted parameter, and set a correct value.
0xC0B00273 SAAF115 -1062206861 3232760435	Failed to acquire CF Card's File List	Please confirm that the specified file type is correct. Also, please check the access right to the destination folder. If there is no problem with the access right, please check whether the drive has enough free space.
0xC0B00274 SAAF116 -1062206860 3232760448	Could not connect to GP3000 Series NODE / WinGP NODE.	'WinGP' NODE may be busy. Please execute it again after a brief interval. Or, if the connection with 'WinGP' NODE is established using the transfer tool, please exit the tool and then execute it again.
0xC0B002A6 SAAF166 -1062206810 3232760486	Read SRAM Backup Data is now being used.	Please execute Read SRAM Backup Data again.
0xC0B002A7 SAAF167 -1062206809 3232760487	Parameter Error in Read SRAM Backup Data	Please execute Read SRAM Backup Data using a correct parameter.
0xC0B002A8 SAAF168 -1062206808 3232760488	Failed to write to a saved file.	If the hard-disk capacity of the PC is insufficient, please increase it and execute the operation again. Or please restart the PC and execute again.

- * 1st line: Error code.
 2nd line: Integrated error code.
 3rd line: Error code with decimal code.
 4th line: Error code without decimal code.

Error Code		Message
Decimal	Hexa	
9300	2454h	Cannot find network project file.
9301 : 9329	2455h : 2471h	Reserved
9330	2472h	Cannot execute the command because resources are insufficient. Terminated program.
9331	2473h	The system resource was dead-locked. Terminated program.
9332	2474h	System Error
9333	2475h	Cannot execute the command because program versions do not match. Terminated program.
9334 : 9339	2476h : 247Bh	Reserved
9340	247Ch	An error occurred when accessing the <%s> file.
9341	247Dh	Pro-Server is being used by too many applications.
9342	247Eh	OS resources are insufficient (insufficient memory).
9343	247Fh	The set connector is used by another application.
9344	2480h	Pro-Server has not been started. Could not reference data.
9345	2481h	Pro-Server has been terminated. Could not reference data.
9346	2482h	Pro-Server has been terminated. Cannot continue.
9347	2483h	Pro-Server has not been started. Cannot continue.
9348	2484h	Could not start Pro-Server.
9349	2485h	Could not start Pro-Studio.
9350	2486h	Unsupported command. Cannot continue.
9351	2487h	Failed in loading the network project file.
9352	2488h	The entered node name has already been registered.
9353	2489h	The node name entered has not been registered.
9354	248Ah	Backup data type specified is not supported.
9355	248Bh	Failed to writing to the file.
9356	248Ch	Could not create a file to store the SRAM backup data.
9357	248Dh	The node name entered has not been registered.
9358	248Eh	Pro-Server is already operating. Cannot start two copies.
9359	248Fh	Reserved
9360	2490h	'%s' has not been entered.
9361	2491h	0 cannot be entered in '%s'.
9362	2492h	'%s' should be: "xxx.xxx.xxx.xxx" format; where xxx is a value between 0 and 255.

Continued

Error Code		Message
Decimal	Hexa	
9363	2493h	An invalid value has been entered in '%s'.
9364	2494h	A character unavailable for '%s' is involved.
9365	2495h	'%s' has not been entered yet.
9366	2496h	Cannot start a new process until the process result is returned from the server.
9367	2497h	Cannot terminate the application while waiting for the process result.
9368	2498h	Read permission required to execute this command. Log on to the network again.
9369	2499h	Write permission required to execute this command. Log on to the network again.
9370	249Ah	Administrator permission required to execute this command. Log on to the network again.
9371	249Bh	The specified number is not registered.
9372	249Ch	Reserved
:	:	
9375	249Fh	
9376	24A0h	Cannot read the file (Core.ID)
9377	24A1h	Reserved
:	:	
9389	24ADh	
9390	24AEh	Mode of Appointed handle is EASY_TB_STATUS_NOW or EASY_TB_STATUS_LAST_READ. Please execute after changing its mode to EASY_TB_STATUS_PAST or EASY_TB_STATUS_INDEX.
9391	24AFh	Unable to open the designated LS Area
9392	24B0h	Designated LS Area is not open
9393	24B1h	Failed to acquire CF Card's File List
9394	24B2h	Failed to read CF Card's file(s)
9395	24B3h	Failed to write CF Card's file(s)
9396	24B4h	CF Card is not inserted
9397	24B5h	CF Card is not initialized
9398	24B6h	CF Card is damaged
9399	24B7h	Unable to access the designated file
9400	24B8h	The Pro-Easy.DLL function was called twice. The function of PfnApiEasy.DLL is already running.
9401	24B9h	The specified access handle for Pro-Server EX is not effective.
9402	24BAh	Pro-Server has stopped and can not perform processing.
9403	24BBh	The error occurred in the function of OLE. Data cannot be converted.

Continued

Error Code		Message
Decimal	Hexa	
9404	24BCh	The effective data for the specified data-type variant does not exist in the original data, or is not enough.
9405	24BDh	Original data and destination data types cannot be converted by data-type variant.
9406	24BEh	The specified argument is not enabled.
9407	24BFh	Can not create the time bar.
9408	24C0h	The symbol name is not registered.
9409	24C1h	Cannot open the distribution sheet.
9410	24C2h	The specified time bar has already been locked.
9411	24C3h	The specified time bar has already been linked.
9412	24C4h	The specified handle is not linked.
9413	24C5h	The specified handle is not linked to the database.
9414	24C6h	Specified handle is locked or played, Please execute after clearing to its status. Please execute after clearing to its status.
9415	24C7h	The argument is wrong.
9416	24C8h	Please set the type to either "Date", or compatible with the "Date" type.
9417	24C9h	The specified time is out of the valid range.
9418	24CAh	The invalid argument has been set.
9419	24CBh	Database of appointed handle is closed.
9420	24CCh	Database access error.
9421	24CDh	INI file ('%s') in the action contents cannot be opened.
9422	24CEh	'%s' of INI file ('%s') in the action contents cannot be analyzed.
9423	24CFh	Action '%s' uses action contents not yet installed in the network project.
9424	24D0h	There are too many actions to register.
9425	24D1h	The specified action has already been registered.
9426	24D2h	The action contents which action '%s' uses cannot be started. The designated action is not registered.
9427	24D3h	An error occurred on the Active-X IF.
9428	24D4h	The designated action has been registered in the registry.
9429	24D5h	Reserved
:	:	
9449	24E9h	
9450	24EAh	The node name or symbol name is not specified.
9451	24EBh	The node name is not specified.
9452	24ECh	The data type setting is not valid.
9453	24EDh	The node name and symbol is not delimited with '!'. Continued

Error Code		Message
Decimal	Hexa	
9454	24EEh	The symbol name has not been registered or it is not a valid device address.
9455	24EFh	Cannot continue the process - no valid device is specified.
9456	24F0h	Cannot make word-access to 32-bit devices.
9457	24F1h	The address is out of the valid range.
9458	24F2h	The number of points setting is invalid.
9459	24F3h	The number of points setting is 0 or exceeds the setting range.
9460	24F4h	Cannot convert the set symbol into a device address.
9461	24F5h	A value input error occurred. Enter a correct value.
9462	24F6h	The specified lifetime is invalid.
9463	24F7h	The designated bit location is incorrect.
9464	24F8h	Reserved
:	:	
9469	24FDh	
9470	24FEh	Unable to connect to designated Node
9471	24FFh	Node is a Windows PC. Unable to perform processing.
9472	2500h	Failed to save captured screen data as JPEG file
9473	2501h	Screen Capture is not supported.
9474	2502h	Capture Approval Flag is not ON.
9475	2503h	Failed to acquire CF Card free space data
9476	2504h	Data Transfer is not supported
9477	2505h	ProNet.dll has not been installed properly
9478	2506h	Unable to perform due to the 2-Way Driver's version not being 4.50 or higher
9479	2507h	Reserved
9480	2508h	Failed to delete CF card file.
9481	2509h	Failed to change CF Card's internal file
9482	250Ah	The file name consists of more than 256 characters. Supposed to be within 256 characters.
9483	250Bh	Reserved
:	:	
9499	251Bh	
9500	251Ch	Pro-Server schedule management thread initialization error
9501	251Dh	Pro-Server LAN management thread initialization error
9502	251Eh	Pro-Server timer management thread initialization error
9503	251Fh	Pro-Server DDE control thread initialization error
9504	2520h	Pro-Server API control thread initialization error
9505	2521h	Pro-Server API parameter error

Continued

Error Code		Message
Decimal	Hexa	
9506	2522h	Response time out
9507	2523h	Pro-Server failed in initializing the LAN.
9508	2524h	No data
9509	2525h	Invalid device
9510	2526h	Invalid address
9511	2527h	The address is out of the valid range.
9512	2528h	Data type error
9513	2529h	Transmission message error
9514	252Ah	Cannot initialize Pro-Server cache function.
9515	252Bh	Cannot load the network project because the database is used.
9516	252Ch	Reserved
:	:	
9559	2557h	
9560	2558h	System Error (DLL load error)
9561	2559h	System Error (DLL version may be old.)
9562	255Ah	System Error
9563	255Bh	The designated property ID is not defined. (Version may be old.)
9564	255Ch	Value conversion error. Incorrect characters as numbers are designated.
9565	255Dh	Too many characters.
9566	255Eh	The number is too large.
9567	255Fh	System Error (Cannot start COMM.)
9568	2560h	System Error (Cannot start GP-Viewer runtime.)
9569	2561h	Cannot open the %s file.
9570	2562h	File read error.
9571	2563h	File write error.
9572	2564h	No tags exist. (No parameter class declarations exist.)
9573	2565h	No end tags exist. (No parameter class declarations exist.)
9574	2566h	Found the unexpected end tag (No parameter class declarations exist.)
9575	2567h	Signatures do not match.
9576	2568h	Unsupported parameter.
9577	2569h	Reached the file end.
9578	256Ah	The incorrect structure.
9579	256Bh	Cannot continue the process due to a memory lack.
9580	256Ch	Cannot analyze the device name.
9581	256Dh	DB name is not designated.
9582	256Eh	Cannot access to DB.

Continued

Error Code		Message
Decimal	Hexa	
9583	256Fh	Cannot edit DB because it is locked (edited) by another program (for example, Data View).
9584	2570h	Either the node name or the device name is not designated.
9585	2571h	Cannot use DB because it has been closed. (DB in use is automatically closed once when NPJ is saved/loaded.)
9586	2572h	The database may be broken.
9587	2573h	Data not saved.
9588	2574h	Cannot find data at the designated time.
9589	2575h	No polling setups exist.
9590	2576h	The database has not been opened. (Or it has already been closed.)
9591	2577h	Already polling start.
9592	2578h	Old data will be overwritten, instead of newest data.
9593	2579h	Defined record is deleted.
9594	257Ah	Exceeds designated file size.
9595	257Bh	Designated file number does not exist
9596 : 9599	257Ch : 257Fh	Reserved
9600	2580h	Cannot continue the process due to a resource lack in GP.
9601 : 9619	2581h : 2593h	Reserved
9620	2594h	The network project item has been registered redundantly. (The network project file has been broken.)
9621 : 9639	2595h : 25A7h	Reserved
9640	25A8h	The Provider information data that is not registered in the network project file were sent from other node. (Network projects differ between the Provider and the Receiver nodes.)
9641	25A9h	Either that the device write failed at the Receiver node or that no partner nodes exist while providing data.
9642 : 9659	25AAh : 25BBh	Reserved
9660	25BCh	Data Read failed.
9661	25BDh	Invalid access range of the read device.
9662 : 9669	25BEh : 25C5h	Reserved

Continued

Error Code		Message
Decimal	Hexa	
9670	25C6h	It is an access range wrong point by the write of device.
9671 : 9699	25C7h : 25E3h	Reserved
9700	25E4h	Received the first trigger establish command for non-existing provider information.
9701 : 9709	25E5h : 25EDh	Reserved
9710	25EEh	Received the second trigger establish command for non-existing provider information.
9711 : 9729	25EFh : 2601h	Reserved
9730	2602h	GP is busy. It is busy sending screen data or saving SRAM backup data to another PC.
9731	2603h	SRAM backup data read error. (The item ID differs from the previous ID.)
9732	2604h	SRAM backup data read error. (The data type differs from the previous type.)
9733	2605h	SRAM backup data read error. (The block number differs from the previous number.)
9734	2606h	SRAM backup data read error. (The requested data amount is 0 or differs from the previous amount.)
9735 : 9739	2607h : 260Bh	Reserved
9740	260Ch	GP is busy. It is busy sending screen data or saving SRAM backup data to another PC.
9741	260Dh	SRAM backup data read error. (The item ID differs from the previous ID.)
9742	260Eh	SRAM backup data read error. (The data type differs from the previous type.)
9743	260Fh	SRAM backup data read error. (The block number differs from the previous number.)
9744	2610h	SRAM backup data read error. (The requested data amount is 0 or differs from the previous amount.)
9745 : 9749	2611h : 2615h	Reserved
9750	2616h	CF command error.

Continued

Error Code		Message
Decimal	Hexa	
9751	2617h	CF Access error.
9752	2618h	No CF card unit.
9753 : 9779	2619h : 2633h	Reserved
9780	2634h	Transmission error occurred with PLC during data write. (Code:%02x:%04x)
9781	2635h	The designated SRAM backup data is not in the GP.
9782	2636h	The GP's SRAM backup data is incorrect. (Code:%04x)
9783	2637h	New alarm block is not supported.
9784 : 9789	2638h : 263Dh	Reserved
9790	263Eh	No remote access rights. (not connected remotely)
9800	2648h	Parameter error.
9801	2649h	Data count is over.
9802	264Ah	File create error.
9803	264Bh	Error on creating EXCEL sheet.
9804	264Ch	Write file error.
9805	264Dh	File open error.
9806	264Eh	Read only file.
9807	264Fh	Print out error.
9808	2650h	Save folder access error.
9809	2651h	Reserved
9810	2652h	Unable to find message table file.
9811	2653h	Unable to open message table file.
9812	2654h	Unable to find designated sheet in message table file.
9813	2655h	Message table is incorrect.
9814	2656h	No equivalent enabled code.
9815	2657h	Error occurred during POP confirmation. Refer to Log Viewer for the details
9816	2658h	Unable to send mail. Refer to Log Viewer for the details
9817	2659h	Unable to send portion of mail. Refer to Log Viewer for the details
9818 9819	265Ah 265Bh	Reserved
9820	265Ch	Unable to find designated database
9821	265Dh	Unable to find designated Table. Or, there are no records in the designated Table

Continued

Error Code		Message
Decimal	Hexa	
9822	265Eh	Unable to find the designated field name
9823	265Fh	Unable to find the designated data
9824	2660h	Field data is incorrect
9825	2661h	Validation failed
9826	2662h	Error occurred while accessing the database
9827	2663h	Unable to create the Pro-Server handle
9828	2664h	There are no character data
9829	2665h	Reserved
:	:	
9839	266Fh	
9840	2670h	Unable to open Action Report Sheet Template, or unable to append sheet
9841	2671h	Failed to start EXCEL
9842	2672h	Unable to open Template Book
9843	2673h	Action System Error
9844	2674h	Unable to save Output Book
9845	2675h	Designated Template Sheet (%s) does not exist in Template Book
9846	2676h	Failed to append sheet
9847	2677h	Unable to interpret command (%s) and cannot execute
9848	2678h	Failed to print
9849	2679h	Designated data type is not supported
9850	267Ah	Pro-Server version is old and cannot be started
9851	267Bh	Action Report sheet is corrupted
9852	267Ch	Designated group does not exist
9853	267Dh	Unable to paste image
9854	267Eh	File header is corrupted - unable to read
9855	267Fh	Unable to open designated CSV file (%s)
9856	2680h	Action Area Size is too small
9857	2681h	Unable to create or read temporary file
9858	2682h	No usable files exist in GP/GLC
9859	2683h	Designated data type is not supported
9860	2684h	A file name is too long, and Output Book cannot be make
9861	2685h	An error occurred while macro run. Refer to Log Viewer for the details
9862	2686h	Unable to save GP Screen Capture data.
9863	2687h	Check if the Permission Flag has turned ON.
9864	2688h	The file name is error.
9865	2689h	The specified file does not exist in the CF card.

Continued

Error Code		Message
Decimal	Hexa	
9866	268Ah	Not the browser application's designated folder. Browser cannot be displayed.
9870	268Eh	Error downloading Binary file
9871	268Fh	Binary file Read failed
9872	2690h	Binary file Open error
9873	2691h	Binary file Analysis failed
9874	2692h	Error writing to Excel file
9875	2693h	Error writing to CSV file
9876	2694h	Error creating Binary file
9877	2695h	Designated file does not exist
9878	2696h	Conversion from Excel file to Binary file failed
9879	2697h	Conversion from CSV file to Binary file failed
9880	2698h	Provided data is outside range
9881	2699h	Failed in GP log data upload.
9882	269Ah	There is no data to support.
9883	269Bh	This data requires more than one sheet.
9884	269Ch	Microsoft Excel is not installed on this machine.
9885	269Dh	Wrong parameter is designated.
9886	269Eh	Failed to write data.
9887	269Fh	Failed to read CSV file.
9888	26A0h	An error occurred in deleting an unnecessary file.
9889	26A1h	Action Failed.
9891	26A3h	No corresponding data in ACCESS file.
9892	26A4h	Command error.
9893	26A5h	Failed in automatic upload of ACCESS data.
9894	26A6h	Cannot open the specified table.

38.9.3 Bit data access

WinGP SDK provides three ways to handle bit data when accessing the bit device.

- 1) 16-bit unit: Handles the data as a bit array in a 16-bit unit on the bit device.
The specified number of bit data is stored/used right-aligned from the D0 bit.
The data buffer requires sufficient space for 16 bit even if the specified number is 1. Also, the number needs to be specified in 16-bit units.

For example, data buffer storing order when a 20 bit device is specified:

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

Applicable API

ReadDeviceBit/WriteDeviceBit()

When specifying (EASY_AppKind_Bit) for the data type in ReadDevice/WriteDevice(), ReadDeviceVariant/WriteDeviceVariant()

When specifying a bit symbol and a group that includes any bit symbol in ReadSymbol/WriteSymbol()

- 2) Variant BOOL unit: Handles 1 bit as the Variant BOOL data.
The data buffer is a BOOL type where 1 bit is 1 Variant. It handles the specified number of data as a BOOL-type array.

Applicable API

When specifying 0x201 (EASY_AppKind_BOOL) for the data type in ReadDeviceVariant/WriteDeviceVariant()

When specifying the bit symbol and the group that includes any bit symbols in ReadSymbolVariant/WriteSymbolVariant()

- 3) Bit offset symbols when accessing the device with a structure variable in the logic instruction
When you directly specify the bit offset symbol to access the device, the data buffer handles the data either in "16-bit unit" or "Variant BOOL unit" as described above.
Note that the group symbol itself has bit offset symbols and no data is secured for the bit offset symbols in the data buffer when accessing the device with the structure variable in the logic instruction.
The bit offset symbols never exist by themselves and always have parent word symbols. A data area is secured for the parent. Use part of the respective secured area for the bit offset symbols.

38.10 Settings Guide

38.10.1 System Settings [Display Unit Settings] [IPC Settings] Settings Guide

The screenshot shows the 'Display Unit' settings window with the following sections and options:

- Display Settings:**
 - Display Right-Click Menu
 - Window Mode: **Window Screens** (dropdown)
- Window Settings:**
 - Specify Display Position (X: 0, Y: 0)
 - Display Titlebar
 - Window Title: **WinGP**
 - Minimize Button
 - Maximize Button
 - Close Button
 - Window Frame
 - Menu Bar
 - Window Size: Width **1024**, Height **768**
- Historical Data Retentive Settings:**
 - Historical Data Storage Location: [Empty text box]
 - Retentive Condition: **Frequency** (dropdown)
 - Frequency: **10** Minutes
 - Indicate Write Status
 - Status Address: [Empty text box]
- Error Settings:**
 - Save Error Message
 - Save in: **CF Card** (dropdown)
 - Number of Stored Items: **100**
 - Number of Files to Save: **100**
 - File Name: [Empty text box]
- Destination Folder:**
 - CF Card: [Empty text box]
 - USB Storage: [Empty text box]
- Transfer Settings:**
 - Port: **21**
- API Communication:**
 - Use API Communication
 - Port: **9800**

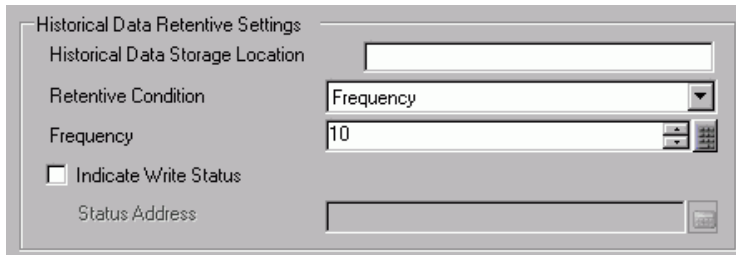
■ Display Settings

This close-up view of the 'Display Settings' section shows the following configuration:

- Display Right-Click Menu
- Window Mode: **Window Screens** (dropdown)
- Window Settings:**
 - Specify Display Position (X: 0, Y: 0)
 - Display Titlebar
 - Window Title: **WinGP**
 - Minimize Button
 - Maximize Button
 - Close Button
 - Window Frame
 - Menu Bar
 - Window Size: Width **1024**, Height **768**

Setting	Description
Display right-click menu	Specifies whether to display the menu by right-click on the window in WinGP. "38.10.2 Window Frame Settings Guide ■ Right-click menu" (page 38-172)
Window Mode	Select the size of the window screen, either [Full Screen] or [Window], when starting [WinGP]. When [Window] is selected, the window opens to the defined window size. When [Full Screen] is selected, the window is displayed at full screen regardless of the screen size.
Window Settings	Defines the window display position when WinGP starts. Use X and Y coordinates to set the display position.
Specify Display Position	<ul style="list-style-type: none"> • X: 0 to maximum horizontal resolution of the selected model minus 1 • Y: 0 to maximum vertical resolution of the selected model minus 1
Display Titlebar	Specifies whether to display the title bar in the window frame. ☞ "38.10.2 Window Frame Settings Guide" (page 38-171)
Window Title	Specifies the window title name to display on the title bar in no more than 63 single-byte characters.
Minimize Button	Specifies whether to display the Minimize window button.
Maximize Button	Specifies whether to display the Maximize window button.
Close	Specifies whether to display the Close window button. Window Settings
Window Frame	Specifies whether to display the window border. NOTE • If [Display Titlebar] is selected, [Window Frame] is always displayed and the check box is selected.
Menu Bar	Specifies whether to display the menu bar inside the window frame.
Window Size	Specifies the window size with [Width] and [Height]. Use 0 - maximum resolution of the selected model for [Width] and [Height] settings. NOTE • The value can be specified between 0 to 1024 when using PS-2000B.

■ Historical Data Retentive Settings



Setting	Description												
Historical Data Storage Location	A feature that mimics SRAM functionality, specifies the full path to the location to save backup data with up to 255 single-byte characters, including the drive and folder names. If no settings are defined, the location defaults to "NAND\PRJ001\USER\SCREEN" in the WinGP installation folder.												
Retentive Condition	Selects a condition to execute backup from [Frequency], [Bit ON], or [Bit Change]. <ul style="list-style-type: none"> • Frequency Backs up the data as specified in [Loop Update Time]. • Bit ON Backs up the data only when the bit specified in [Control Bit Address] turns ON. The data is saved only after 1 minute has elapsed since the last save. • Bit Change Backs up the data only when the bit specified in [Control Bit Address] turns ON. The data is saved only after 1 minute has elapsed since the last save. 												
Frequency	Specifies the loop update time to repeat backup when [Frequency] is selected in [Backup Trigger] using 1 to 60 minutes.												
Control Bit Address	Specifies the address to control backup when [Bit ON] or [Bit Change] is selected in [Backup Trigger].												
Indicate Write Status	Specifies whether to use the bit address to show the backup data write status.												
Status Address	The backup data write status is shown by ON and OFF of the bit address specified here. <ul style="list-style-type: none"> • ON Writing data • OFF Writing no data <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Bit</th> <th>Name</th> <th>Bit ON condition</th> <th>Bit OFF condition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Writing</td> <td>When file write starts</td> <td>File write ends</td> </tr> <tr> <td>1</td> <td>Write Error</td> <td>When write fails</td> <td>When write starts</td> </tr> </tbody> </table>	Bit	Name	Bit ON condition	Bit OFF condition	0	Writing	When file write starts	File write ends	1	Write Error	When write fails	When write starts
Bit	Name	Bit ON condition	Bit OFF condition										
0	Writing	When file write starts	File write ends										
1	Write Error	When write fails	When write starts										

■ Error Settings

The screenshot shows a dialog box titled "Error Settings". It has the following controls:

- A checkbox labeled "Save Error Message" which is currently unchecked.
- A "Save in" dropdown menu with "CF Card" selected.
- A "Number of Stored Items" spinner control set to "100".
- A "Number of Files to Save" spinner control set to "100".
- A "File Name" text input field.

Setting	Description
Save Error Message	<p>Specifies whether to save system errors and application errors displayed on the [WinGP] window.</p> <p>NOTE</p> <ul style="list-style-type: none"> • If no more than 10 minutes have past since the last save, the error log file is not saved until 10 minutes pass to avoid frequent write access. If so, all summaries recorded in the 10 minutes are saved in the error log file. • Even the error occurred consecutively, all errors are saved in the error log. • If the clock time of the IPC or PC/AT compatible machine is changed while the error log function is operating, the error log will not be saved in the order of elapsed time
Save in	<p>Define Save in as either [CF Card] or [USB storage].</p> <p>NOTE</p> <ul style="list-style-type: none"> • When you select [CF Card] or [USB storage], the [LOG] folder is created in the Save in folder and the error log file is created in the folder.
Number of Stored Items	<p>Specifies the number of error messages to save per error log file using 1 to 1000.</p>
Number of Files Saved	<p>Specifies the number of error files to save the error log files using 0 to 1024.</p> <p>NOTE</p> <ul style="list-style-type: none"> • If [Number of Files to Save] is set to 0, the files are saved until the [CF Card] or [USB Storage] capacity is reached. • Until the number of error log files reach the number set in the [Number of Stored Items], records are added to the latest error log file. However, if you change the date or time, an error log file may be created with the wrong date or time. In this case, with the new date, records are not added even if the system has not reached the [Number of Stored Items]. • When the number of error messages exceeds [Number of Files to Save] in [Error Settings], the oldest file is deleted to add a new file.

Continued

Setting	Description
File Name	<p>Specifies the file name prefix of the error log file using 0 to 16 single-byte characters.</p> <p>The file name is specified in the following format. [Prefix][Date/Time]_[ID].[Extension]</p> <p>For example:</p> <p>[Prefix] : Test [Saved Date/Time]: 2006/7/14 16:18 [ID] : Same 0 (0 - Serial Number) When multiple files are created at the same time, this number specifies the order of files created. [Extension]: log (Fixed characters)</p> <p>File Name: Test200607141618_0.log</p> <p>NOTE</p> <ul style="list-style-type: none">• If no file name is specified, the file is named simply as [Save Date Time] and [ID].

■ Set Destination Folder

Specify the folder to save the data from the [CF Card Destination Folder] or [USB storage Destination Folder] specified in [Information] - [Destination Folder (C)] on the [Project (F)] menu.

On models other than IPC Series (PC/AT), the screen transfer operation stores data to the CF card or USB storage. For IPC Series (PC/AT), the folder defined here replaces the functionality of the CF card or USB storage.

The image shows a dialog box titled "Destination Folder". It contains two input fields: "CF Card" and "USB Storage". Both fields are currently empty.

Setting	Description
CF Card	Specify the folder with a full path to replace CF Card. The path must be less than 239 characters using single or double-byte characters. When this folder is left empty, data is saved to "CFA00" in the WinGP installation folder.
USB storage	Specify the folder with a full path to replace USB Storage (USB memory). The path must be less than 239 characters using single or double-byte characters. When this folder is left empty, data is saved to "USBHD" in the WinGP installation folder.

NOTE

- You can set folders to replace CF Card or USB Storage on the network. However, file names may not be displayed correctly depending on the connected environment (OS or language settings).

IMPORTANT

- When the operating system for the IPC or PC/AT compatible machine is Windows XP Embedded, you can set the Write Filter (Write Protection) in the system drive (C drive) using the IPC tool. When the designated reference folder is C:\, and the Write Filter is enabled, then files cannot be written. Please select a drive with the Write Filter disabled.
- Define a Destination Folder that is different from the [CF Card Folder] or [USB Storage Destination Folder]. Otherwise, an error will occur.

■ Transfer Settings

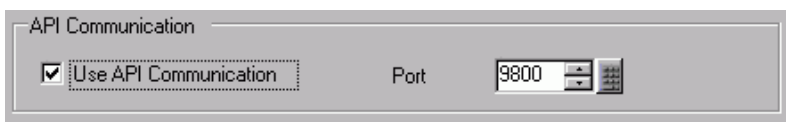


Setting	Description																								
Port	<p>Specifies the port number to use for transfer from 0 to 65535. When you change port numbers, make sure it matches the port number defined in the Project's LAN transfer settings.</p> <p>The screenshot shows a dialog box titled "Select Display Unit". It contains a table with columns: IP Address, Port, PASV, Display Unit, Node, and Automatic/... The 'Port' column is highlighted with a red box. The table contains the following data:</p> <table border="1"> <thead> <tr> <th>IP Address</th> <th>Port</th> <th>PASV</th> <th>Display Unit</th> <th>Node</th> <th>Automatic/...</th> </tr> </thead> <tbody> <tr> <td><input type="checkbox"/> 192.168.0.1</td> <td>21</td> <td>Do Not Use</td> <td>AGP-3450T</td> <td></td> <td>Automatic</td> </tr> <tr> <td><input type="checkbox"/> 192.168.0.2</td> <td>21</td> <td>Do Not Use</td> <td>AGP-3500T</td> <td></td> <td>Automatic</td> </tr> <tr> <td><input type="checkbox"/> 192.168.0.3</td> <td>21</td> <td>Do Not Use</td> <td>AGP-3550T</td> <td></td> <td>Automatic</td> </tr> </tbody> </table>	IP Address	Port	PASV	Display Unit	Node	Automatic/...	<input type="checkbox"/> 192.168.0.1	21	Do Not Use	AGP-3450T		Automatic	<input type="checkbox"/> 192.168.0.2	21	Do Not Use	AGP-3500T		Automatic	<input type="checkbox"/> 192.168.0.3	21	Do Not Use	AGP-3550T		Automatic
IP Address	Port	PASV	Display Unit	Node	Automatic/...																				
<input type="checkbox"/> 192.168.0.1	21	Do Not Use	AGP-3450T		Automatic																				
<input type="checkbox"/> 192.168.0.2	21	Do Not Use	AGP-3500T		Automatic																				
<input type="checkbox"/> 192.168.0.3	21	Do Not Use	AGP-3550T		Automatic																				

NOTE

- If you forgot the port number for the Transfer Tool, in the Offline Mode check [WinGP Settings] - [Transfer].

■ API Communication




Setting	Description
Use API communication	Specifies whether to use API communication (handling API or device access API).
Port	<p>Specifies the port number to use for API transfer from 0 to 65535. Define a number that is outside the range 8000 to 8019, and is different than the [Transfer Settings] [Port].</p> <p>NOTE</p> <ul style="list-style-type: none"> • In [Peripheral Settings], check which port is used by the other device/ PLC to avoid using the same port.

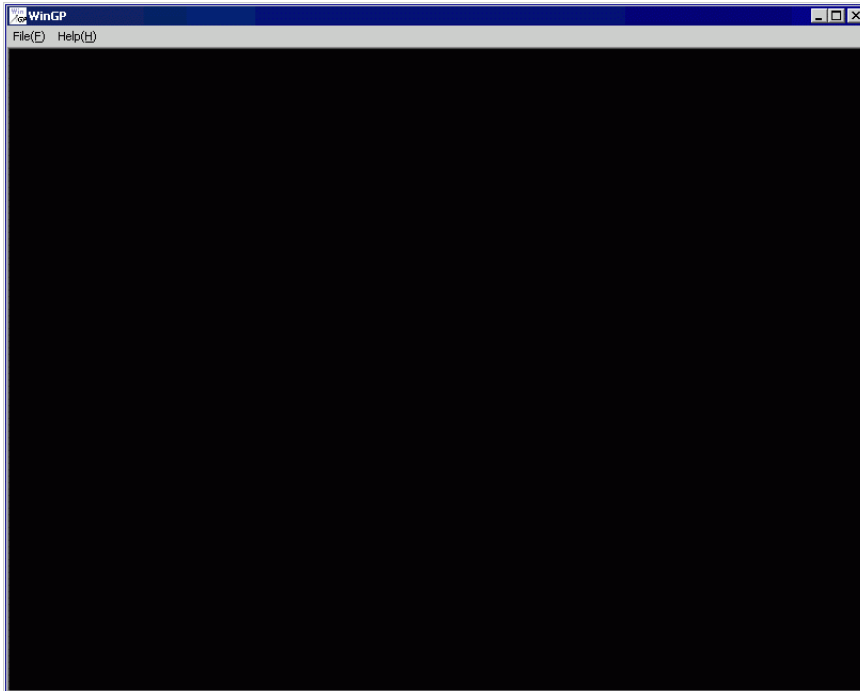
38.10.2 Window Frame Settings Guide

■ Window Frame

This section describes the WinGP window frame you can use to emulate a display unit.


NOTE

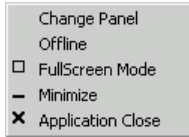
- See the following for information about display settings.
 "38.10.1 System Settings [Display Unit Settings] [IPC Settings] Settings Guide ■ Display Settings" (page 38-164)

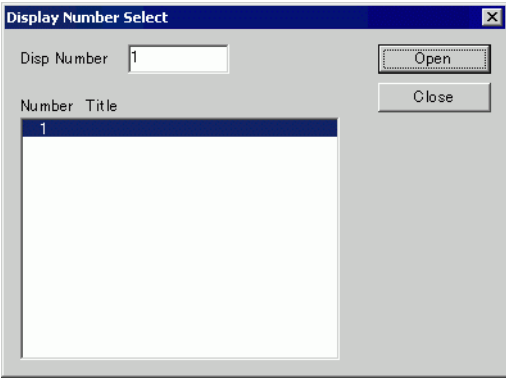


Setting	Description
Title Bar	Displays Window Title, minimize or maximize window, and close button. The window title set in the [System Settings]-[IPC Settings] appears. If no title is set, blank space is displayed on the title.
Minimize Button	Hides the window and displays the icon on the task bar.
Maximize Button	Changes the window to full screen.
Close Button	Exit WinGP.
Menu Bar	<ul style="list-style-type: none"> • Help Displays [Version Information]. • File Displays [Exit] to exit WinGP.
Window Frame	Changes the window size by dragging and dropping the cursor on the window frame. If the size is changed to smaller than the original size, the scroll bar is displayed.

■ Right-click menu

Menu displays when right-clicking the WinGP window frame or clicking on the keypad . This menu is available when in the [System Settings] window, [Display Unit] area, IPC Settings tab, the [Display Right-Click Menu] check box is selected.



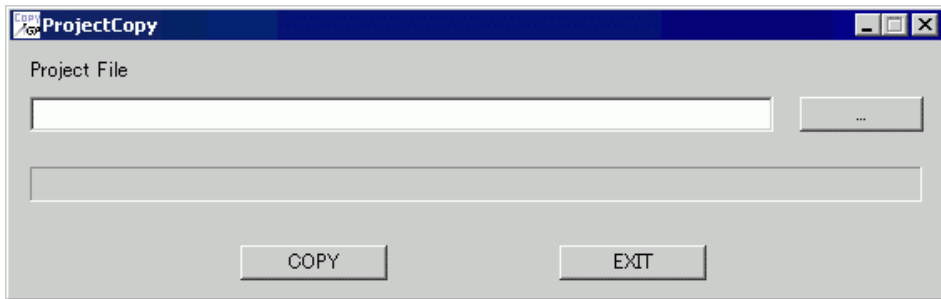
Setting	Description
Screen Change	<p>[When you select [Screen Change], the [Select Display Screen] dialog box appears and allows you to switch the display screen.</p>  <p>NOTE</p> <ul style="list-style-type: none"> • If offline, this item is not displayed on the menu. • You cannot change screens while the password input screen is open.
Screen Number	<p>Specifies the screen number to switch from 1 to 9999.</p> <p>NOTE</p> <ul style="list-style-type: none"> • Only screens in the project can be opened in the Simulation.
Number	Displays the screen number.
Title	Displays the screen title.
Opening Keypads	Opens the screen selected in [Screen Number] or the screen number list.
Close	Displays the [Select Display Screen] dialog box.
Offline (Online)	Switches to offline mode. If displaying in offline mode, switches to the online screen.

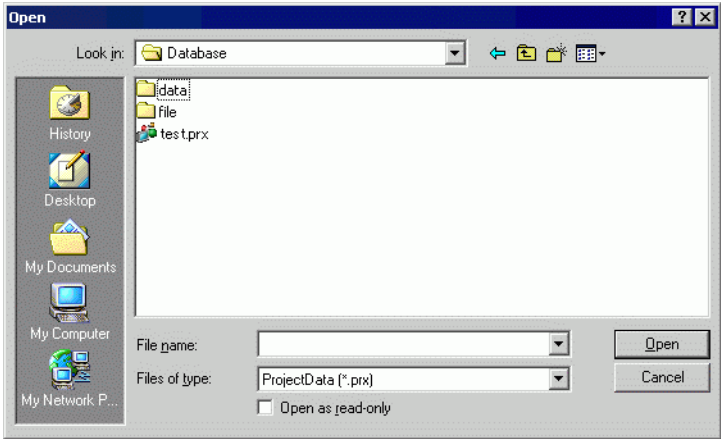
Continued

Setting	Description
Full Screen Mode	Displays the full screen. NOTE <ul style="list-style-type: none">• [If the screen is displayed in [Full Screen Mode], the [Window] is displayed and changes the window to the original size.• Also, you can touch the upper-right and lower-left of the IPC screen and display [System Menu]-[Reset] to reset the screen size of [Full Screen Mode].
Minimized	Hides the window and displays the icon on the task bar.
Close	Exit WinGP.

38.10.3 ProjectCopy (Copy Tool) Settings Guide

From the [Start] menu, point to [Programs], [Pro-face], [WinGP], and then select [Project Copy]. The following dialog box appears. You can copy only the screen data of a project file.



Setting	Description
Project File	Enters or displays the project file path to be copied.
Browse	Specifies the project file location to be copied. 
Copy	Starts copying a project file.
Exit	Exits ProjectCopy.

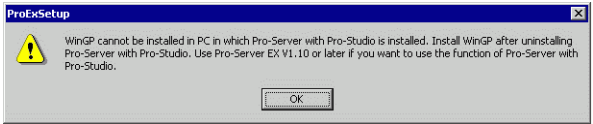
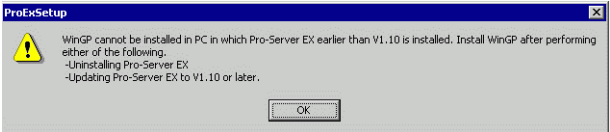
38.11 Restrictions

- You cannot start more than one WinGP.
- If the number of parts on one screen exceeds 1280 in IPC, a warning message appears. Reduce the number of parts placed on the screen. You can place and transfer parts even when this message is displayed.
- If the number of addresses on one screen exceeds 3000 in IPC, a warning message appears. Reduce the number of addresses placed on the screen. You can place and transfer the addresses even when this message is displayed.
- If many parts are placed in IPC, a warning appears when you save. This is because the parts and addresses limits change when converted into another series in [Change Display Unit].
- If the specified number of alarm history and word monitoring in the new model exceeds the limit of the post-conversion model, an error appears when you change the display unit, however, you can change the model.
- You can set the data size up to 8 blocks.
- For blocks 1 to 8, you can register bit/word monitoring up to a total of 10000.
- If the total capacity of SRAM used in the GP-Pro EX settings exceeds 5 MB, a warning appears at the time of error checking and sampling and alarm features do not operate properly. You can use up to 5 MB data to save and transfer the project files.
- If you turn OFF the IPC without shutting down the OS, the backup file for exiting WinGP cannot be saved and the record will be from the last save.
In an IPC with battery backup features, a standby mode (resume) signal is sent when the power is OFF. Upon receiving the signal, WinGP saves the backup file.
- The touch buzzer sound setting is a feature used to specify a unique buzzer used by PC runtime. The setting differs from that of the IPC touch panel unit. If you enable both the buzzer of the IPC touch panel unit and that of PC runtime, a buzzer will sound twice when you touch the PC runtime screen. If you enable the IPC touch panel buzzer, disable the PC runtime buzzer.
- If you disabled [Script Settings]-[Comm.]-[Flow Control] in the system settings, the status [EXIT_SIO_STAT] cannot detect sending errors in [SIO Port Operation].
- If you select [Prevent Multiple Instances] in [Start Application] for special switch, trigger actions, and scripts, multiple instances occur when [Window Title] is not input.
- For the [Window Title] you wish to prevent multiple instances, enter the exact window title in the [Start Application] for the special actions of special switch, trigger actions, and scripts.
- You can transfer project files to the GP if it has [Start Application] and [Exit WinGP] settings for the special actions of special switch, trigger actions, and scripts on models other than IPC, but the file will not run on GP.


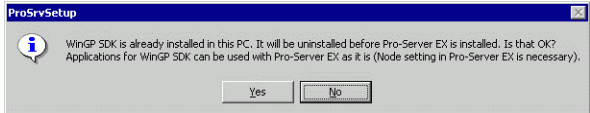
- Do not connect the USB license key before WinGP installation is complete. If you connect the USB license key to the PC before installing WinGP, the operating system automatically starts the wizard which you need to cancel. If you continue with the wizard, then exit the wizard without finding a device driver, it will be registered to the device manager as an unauthorized device. In this scenario, the license cannot be authenticated. Please delete unauthorized devices from the device manager, and restart. Then, install WinGP.
- When a communication error occurs with a device/PLC, and in the [System Settings] workspace, in the [Device/PLC] page, either the Port is set to [Ethernet (UDP)] or [Ethernet (UDP)] is not set to [Automatic], it may take approximately 4 minutes to recover.
- Compared to using the GP3000 Series, it may take longer to communicate with the device/PLC. Therefore, the timing for data updates may be delayed using the device monitor function and the data delivery function with Pro-Server EX. You can alleviate this by increasing the communication [Speed] for the [Device/PLC].

38.11.1 Restrictions On Installation

- If the path has more than 200 single-byte characters in the folder where WinGP is installed, an error "Cannot start because the installation folder will exceed 200 characters" appears when simulation starts and it will not operate properly. Use a path less than 200 single-byte characters and reinstall WinGP.
- If you install WinGP on an OS that does not support it, an error message appears and the installation cannot be completed.
- To install, log on with an account with Windows Administrator authority.
- You cannot install WinGP more than once, even to another folder. To uninstall, insert the installation CD in the IPC on which WinGP is installed.
- WinGP does not allow for recovery installation. To recover, uninstall and then reinstall WinGP.
- If downloading WinGP to a PC/AT compatible machine, please connect the USB key after installation is complete. If the USB key is connected before installation, the operating system starts the USB driver wizard automatically. If the wizard starts, make sure you cancel and exit the wizard.
- When installing WinGP on an IPC or PC/AT compatible machine with either Pro-Server with Pro-Studio for Windows or Pro-Server EX installed, you may not be able to install WinGP, depending on the installation conditions, outlined below. The following shows each installation state.

Installation State	WinGP Installation
Pro-Server with Pro-Studio for Windows has already been installed.	<p>The following message appears and WinGP cannot be installed. Please uninstall Pro-Server with Pro-Studio before installing WinGP.</p>  <p>The screenshot shows a dialog box titled "ProExSetup" with a yellow warning icon. The text inside reads: "WinGP cannot be installed in PC in which Pro-Server with Pro-Studio is installed. Install WinGP after uninstalling Pro-Server with Pro-Studio. Use Pro-Server EX V1.10 or later if you want to use the function of Pro-Server with Pro-Studio." There is an "OK" button at the bottom.</p>
Pro-Server EX Ver1.10 or lower has already been installed.	<p>The following message appears and WinGP cannot be installed. You can either uninstall Pro-Server EX or upgrade to a version higher than V1.10 or higher, then install WinGP.</p>  <p>The screenshot shows a dialog box titled "ProExSetup" with a yellow warning icon. The text inside reads: "WinGP cannot be installed in PC in which Pro-Server EX earlier than V1.10 is installed. Install WinGP after performing either of the following." Below this, there are two bullet points: "-Uninstalling Pro-Server EX" and "-Updating Pro-Server EX to V1.10 or later." There is an "OK" button at the bottom.</p>
Pro-Server EX Ver1.10 or later has already been installed.	WinGP can be installed. (WinGP SDK is not installed)
Neither Pro-Server with Pro-Studio for Windows nor Pro-Server EX have been installed.	WinGP can be installed. (WinGP SDK is automatically installed.)

- When you install Pro-Server with Pro-Studio for Windows or Pro-Server EX on an IPC or PC/AT compatible machine with WinGP installed, WinGP may not operate properly. The following shows each action.

S/W to be installed	Action
Pro-Server with Pro-Studio for Windows	Neither Pro-Server with Pro-Studio for Windows nor WinGP will operate. In this case, uninstall both applications. Do not install Pro-Server with Pro-Studio for Windows on an IPC with WinGP installed.
Pro-Server EX before Ver1.10	<p>After the installer for a version of Pro-Server EX before Ver1.10 starts, the following error message appears and the installation will not be completed. Even if Pro-Server EX is not installed, the error message appears as follows.</p> 
Pro-Server EX Ver1.10 or later	<p>After the installer for Pro-Server EX Ver1.10 or later starts, the following error message appears. If you select [Yes], WinGP SDK is uninstalled and Pro-Server EX Ver1.10 installation begins.</p>  <p>If you stop installing Pro-Server EX Ver1.10 midway, reinstall WinGP.</p> <p>NOTE</p> <ul style="list-style-type: none"> If you install WinGP, WinGP SDK is also installed in a folder called SDK where GP-Pro EX is installed. Although the path differs from the path you specified when creating the user application in Pro-Server EX, you can still use the application created in Pro-Server EX without changing the path. If you install Pro-Server EX after installing WinGP and uninstall Pro-Server EX, WinGP SDK becomes unavailable.

- After installation, please restart before using WinGP and log in with an account with Administrator rights. The WinGP will not operate properly without restarting the IPC.

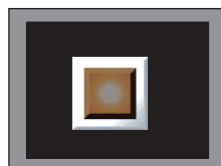
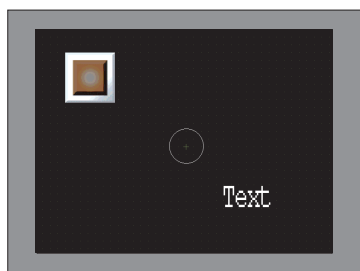
38.11.2 Restrictions on Window Frames

- You can transfer data to an IPC that has different screen resolutions (screen size) but the data will not be displayed properly if the IPC has a lower resolution.

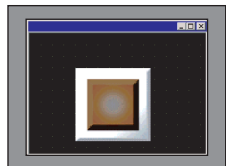
Example 1:

IPC: Create a 800X600 screen and send to a 320X240 IPC

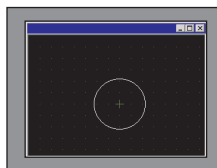
Created data



[Window Mode]: [Full Screen]
Only parts that can be displayed in 320X240 resolution are displayed starting from the top left end.



[Window Mode]: [Window Screen]
[Specify Display Position] is specified, [X Coordinate] is 0, and [Y Coordinate] is 0
Only parts that can be displayed in 320X240 resolution are displayed starting from the top left end.

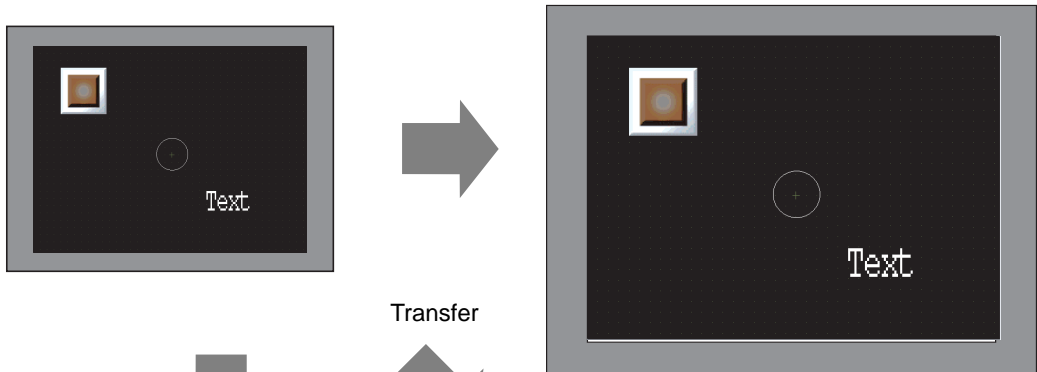


[Window Mode]: [Window Screen]
[Specify Display Position] none
Only parts that can be displayed in 320X240 resolution are displayed starting from the top left end.

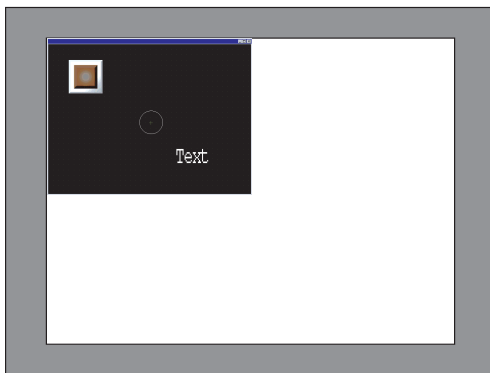
Example 2:

IPC: Create a 800X600 screen and send to a 1600X1200 IPC

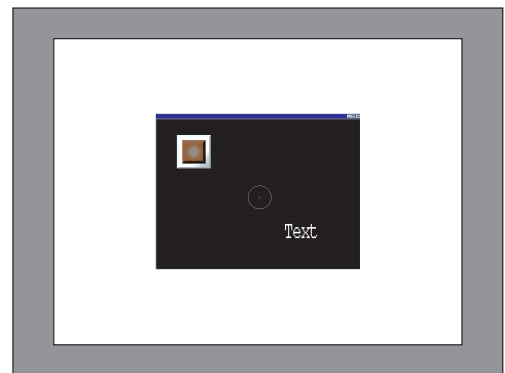
Created data



[Window Mode]: [Full Screen]
With 800X600 resolution, the screen is enlarged to 1600X1200 and displayed.



[Window Mode]: [Window Screen]
[Specify Display Position] is specified, [X Coordinate] is 0, and [Y Coordinate] is 0
In 800X600 resolution, displayed starting from the top left end.



[Window Mode]: [Window Screen]
[Specify Display Position] none
Displayed centered in 800X600 resolution.

- When viewing screen data on a large sized screen with a high resolution, a portion of the window frame will appear outside the screen.
To make sure the window is not outside the screen, do not display the window titlebar, window frame, and menu bar, or display in full-screen mode. Be aware that when you do not display the title bar or display in full-screen mode, the title bar's Exit button does not display.
- If you change IPC models, System Settings keeps the settings before the model change in [IPC Settings] [Display]. Note that X Coordinate and Y Coordinate in [Specify Display Position] return to the initial value of 0 and "Window Size" is initialized to XGA (1024x768) dimensions regardless of the IPC you are converting to.

38.11.3 Restrictions on using Windows XP Embedded

- Windows XP Embedded has the Write Filter feature in system drive. During the Write Filter operation, files cannot be updated in the system drive. The destination folder to update files has to be set in a drive which does not have Write Filter. Thus, you can change the folder to update files by settings.

38.11.4 Restriction on AP Communication

- When using the API with Windows XP SP2, make sure you use Windows Update to install the following patch: Update for Windows XP Service Pack 2 (KB884020).

■ Handling API Restrictions

- All the text information on the handling API are in Unicode. In API, the version information and the project information are read in Unicode. Convert the code if you wish to use the information in another text code (ASCII, etc.).
- You cannot use the handling API in IPC unless it has TCP/IP settings. Be sure to check that the network settings have TCP/IP protocol installed.

■ Handling API Restrictions

- To use the device access API, start WinGP first. An error results if you use the device access API without starting WinGP. A timeout error results if you start the device API after exiting WinGP.
- Do not set IPC standby while API is communicating using the user application. A user application should control such that IPC goes to standby mode only after the operation of device access API is completed.
- To add a protocol to update the Pro-Server EX version, you need to install the protocol module updated in GP-Pro EX to IPC with WinGP SDK installed.
- In ReadSymbolD(), ReadSymbolVariantD(), WriteSymbolD(), WriteSymbolVariantD() API, you cannot use any array variable that exceeds the following array size.

Array variable type	Maximum size accessible with WinGP API communication
Bit Variable	255
Integer Variable	510
Float Variable	510

- If you install Pro-Server EX V1.10, you have to control Pro-Server EX separately.
- You cannot use the device API in IPC unless it has TCP/IP settings. Be sure to check that the network settings have TCP/IP protocol installed.
- If you exit WinGP while accessing the device access API, all the returns from API result in an error.

- If you compile the header created in Visual C++ Ver.6, C:\Program files\Pro-face\WinGP\SDK\VC\Public\ProEasy.h or Pro-Studio [Programming Support]-[VC: Statement] via clip board, LPVARIANT might result in undefined error. LPVARIANT is defined in afxdisp.h. Include this by defining #include <afxdisp.h> in stdafx.h to avoid an error.

38.11.5 Transfer Restrictions

- You cannot transfer using modem or COM port.
- During the initialization process after start up , WinGP displays a screen asking for a retransfer request if any error (damage or loss) is found in the necessary file.
- If you transfer the project file to a different type of IPC, an error dialog box is displayed indicating that the model differs and the transfer cannot be completed. To transfer the file to a different model, convert the model using the editor before transfer.
- You need to exit WinGP because [ProjectCopy](Copy Tool) updates the files used in WinGP. If you try to use the copy operation while WinGP is operating, an error message is displayed and the copy operation is not executed.
- When OS is Windows XP Embedded, you can set the Write Filter in a driver (C drive) of the system using IPC tool. WinGP is installed in C drive and the Write Filter is enabled, WinGP system files or screen data cannot be updated. Disable the Write Filter before starting transfer.
- WinGP allows for changing the port number with the transfer tool. You cannot LAN transfer from the transfer tool if you forget the new port number.

■ Restrictions when using [ProjectCopy] (Copy Tool)

- Only screen data transfer is available using Project Copy [Copy Tool]. Receiving screen data or full transfer of project is not available. In the following cases, please use the Transfer Tool.
 - The first time you transfer the project after installing WinGP
 - Change or add a Device/PLC
 - Change or add a font
 - After upgrading GP-Pro EX, the run-time system or protocol driver is updated and you update the project.
- You cannot send the WinGP system program using the Copy Tool. Please use the Transfer Tool when you upgrade WinGP.

38.11.6 Restrictions on error logs

- If an error log is opened when the error log feature starts writing, writing to the file cannot be completed.
- When the number of error messages exceeds [Number of Files to Save] in [Error Settings], the oldest file is deleted to add a new file.
- If no more than 10 minutes have past since the last save, the error log is not saved until 10 minutes pass to avoid frequent write access. If so, all summaries recorded in the 10 minutes are saved in the error log file.

38.11.7 Function Key Restrictions

- The number of switches that can be allocated to one function key is unlimited.
- The number of switches that can be allocated to all local function keys is the maximum number that can be placed on one screen. The maximum number of parts that can be placed on one screen is determined by the total number of parts placed on the screen and on the function keys.

$(\text{Number of parts placed on B1}) + (\text{Number of parts placed on B1 local function key}) \leq (\text{maximum number of parts on one screen})$

- The number of switches that can be allocated to all function keys is limited even in the maximum device number that can be placed on one screen. Count lead devices in the base screen, and then the local function key. If it exceeds the maximum number of devices, the switches after that will not work.
- The maximum number of parts placed on a global function key, and devices are not dependent on the maximum number of screens. The number limit exists in the global function key itself. Count the number in the set order, and if it exceeds the following restricted number, any part after that will not work.
 - Number of Parts: up to 384
 - Number of lead devices: up to 1152 devices
- The function key cannot be registered in a Package.
- The function key is also not accepted while the operation is prohibited in the operation lock feature.

