Panasonic®

PROGRAMMABLE CONTROLLER

Control FPWIN Pro7 Introduction Guidance

WUME-FPWINPRO7-03

2024.4 | industry.panasonic.com/

Introduction

Thank you very much for purchasing a Panasonic product. Please carefully read through User's Manual before use, and use the product properly with adequate understanding.

Type of Manual

- This manual explains the basic operations of "Control FPWIN Pro7," programming software for the FP7 series and the FP series. Along with this manual, also refer to the User's Manual for the PLC model to be used.
- The manuals can be downloaded on our website https://industry.panasonic.com/global/en/downloads/?tab=manual .

Name of unit / purpose of use	Name of manual	Manual code
Programming Software FPWIN Pro7	Introductory Guide to FPWIN Pro7	WUME-FPWINPRO7

Description in this document is based on Control FPWIN Pro7 V7.2.4. (as of October 2018).

For detailed functions that are not described herein, refer to "Help" in Software FPWIN Pro7.

Table of Contents

1 Installation

1.1 System Requirements

The system requirements for using FPWIN Pro7 are as follows.

- At least 1 GB of memory (recommended)
- CD-ROM drive
- At least 600 MB of hard disk space
- Monitor of WXGA (1280×800) or higher, or equivalent (recommended)
- Mouse
- Serial interface for connecting PLC

To use FPWIN Pro7, Windows® 10, and Windows® 11 need to be pre-installed.

◆ **NOTES**

- **You must log in as administrator.**

1.2 Installation

◆ **PROCEDURE**

1. **Insert the CD of FPWIN Pro7 into the CD-ROM drive.**

   The content of CD is displayed on the browser, and is started automatically.

2. **Select a language.**

3. **Select "Software to be installed."**

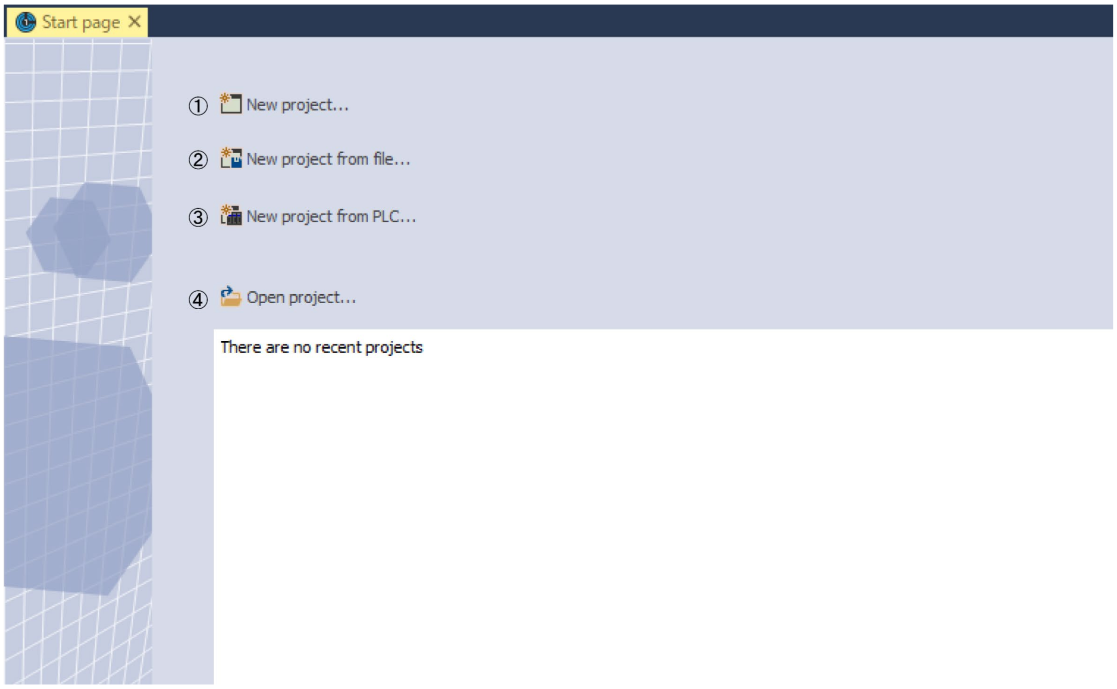
   Carefully read the content displayed on the screen, and proceed with installation in accordance with the instructions.

4. **In the start menu, select "All programs" -> " Panasonic Industry Control FPWIN Pro7," and confirm that it starts up correctly.**

2
Startup and Names of Parts

2.1 Start Page

When FPWIN Pro7 is started up, the Start Page is displayed.


Start page
① New project...
② New project from file...
③ New project from PLC...
④ Open project...
There are no recent projects

In the Start Page, select one of the following.

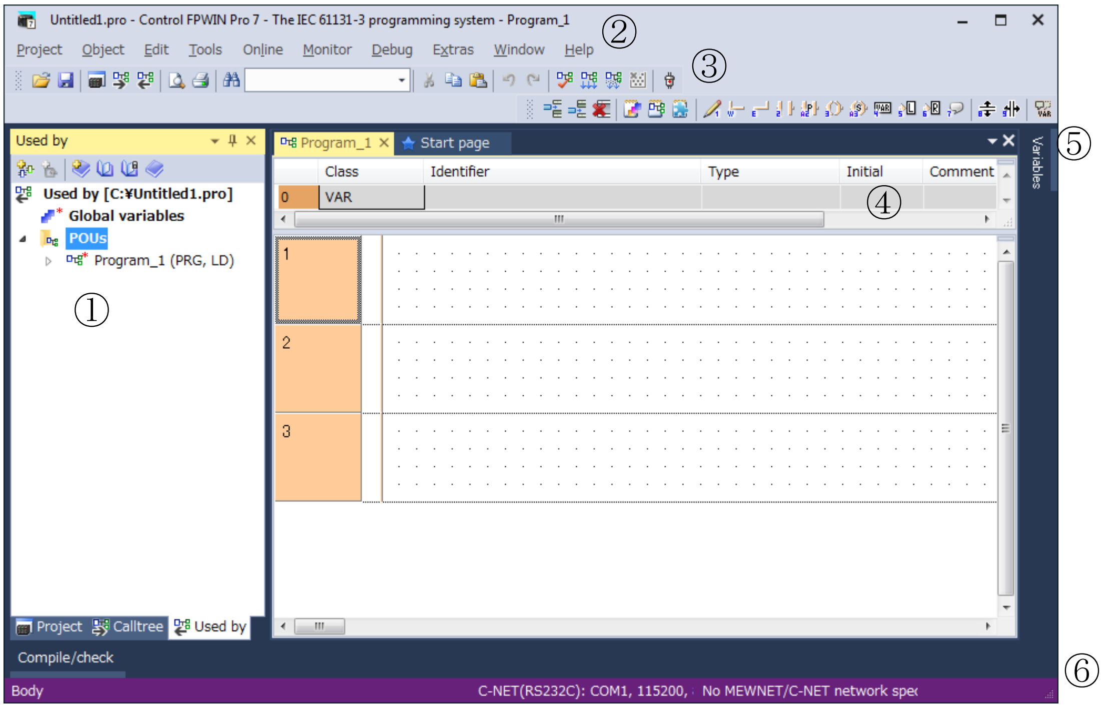
①	New project	Using Wizard, create a new project. It is also possible to create an empty project that contains no POU.
②	New project from file	Open a new project from a project file that is saved in the PC (or external media).
③	New project from PLC	Open a project that is saved in the comment memory of the connected PLC.
④	Open project	A dialog box is opened to browse existing projects saved in the computer. The projects that have been used recently are displayed in a list.

◆ **NOTES**

The number of projects that have been used recently to be displayed in the list can be specified by proceeding from the Menu Bar to "Extras" -> "Options" -> "Program options" -> "General" -> "Number of recent projects on project menu"It can be specified by proceeding in order of the number.

2.2 Main Window

The GUI of the Main Window of FPWIN Pro7 has the following components.


Untitled1.pro - Control FPWIN Pro 7 - The IEC 61131-3 programming system - Program_1
Project Object Edit Tools Online Monitor Debug Extras Window Help
Used by
Used by [C:¥Untitled1.pro]
Global variables
POUs
Program_1 (PRG, LD)
Project Calltree Used by
Compile/check
Program_1
Start page
Class Identifier Type Initial Comment
0 VAR
1
2
3
Variables
Body
C-NET(RS232C): COM1, 115200, No MEWNET/C-NET network spec
①
②
③
④
⑤
⑥

Components of user interface

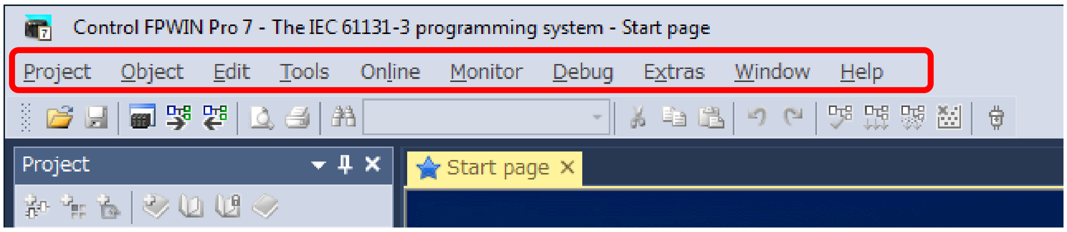
①	Navigator Pane	Refer to "6. Navigator Pane"
②	Menu Bar	Refer to "3. Menu Bar Project" Refer to "4. Menu Bar: Online " Refer to 5. Menu Bar: Extended Functions"
③	Tool Bar	Refer to "2.2.2 Tool Bar"
④	Programming Window	Refer to "7. Programming Editor"
⑤	Selection Pane	Refer to "8. Selection Pane"
⑥	Status Bar	Refer to "2.2.3 Status Bar"

The components can be moved to any place on the screen. To move components, drag the relevant Title Bar.

Most commands displayed in the Tool Bar and the Menu Bar can also be selected in the Popup Menu that is displayed by right-clicking. The Popup Menu is displayed from the Tool Bar, Menu Bar, etc.

2.2.1 Menu Bar

The menu that is displayed may vary depending on the status of the program (e.g. whether or not a project is opened). Read the subsequent sections assuming that a project is opened, if there is no particular reference to the program status.


Control FPWIN Pro 7 - The IEC 61131-3 programming system - Start page
Project Object Edit Tools Online Monitor Debug Extras Window Help
Project
Start page

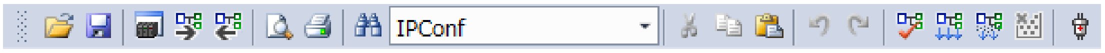
Menu	Function	Reference
Project (P)	The Project Menu contains all menu items for processing the project itself in FPWIN Pro7.	3.Menu Bar Project
Object (O)	The Object Menu contains menu items for performing all processes concerning an Object (POU), such as Open, Save As, Change Name, and Comment.	
Edit (E)	This section contains menu items for editing programs, as well as menu items for creating and changing variables.	
Tool (T)	Menu commands that can only be used in the respective editors can be confirmed.	
Online (L)	Set parameters required in the online mode.	4.Menu Bar: Online
Monitor (M)	Multiple menus are prepared for monitoring a program, where the program has been compiled and downloading to PLC has been completed. During monitoring, it is possible to change variable values, or to forcibly set values.	
Debug (D)	The program can be debugged in each step, before proceeding to the next breakpoint. There are two options. - Debugging in the PLC Simulation Mode for all PLC types that support the simulation - Debugging for PLC models FP2 and FP2SH (Not recommended if there are risk to users or devices)	
Extras (X)	In the Extended Function Menu, it is possible to back up or restore projects, declare external variables, and set options for more efficient operations.	5.Menu Bar: Extended Functions
Window (W)	In the menu items of the Window Menu, the layout of windows on the screen can be adjusted.	
Help (H)	The following items of the Help Menu are displayed. - Table of Contents for Help - Search for Topics - Details of Version Upgrading - First Step Guide - Version Information	

For details of the menus without reference information, refer to "Help" in FPWIN Pro7.

2.2.2 Tool Bar

Icons are displayed in the Tool Bar. These icons represent menu items that are frequently used.

The Standard Tool Bar is supposed to be displayed at any time.

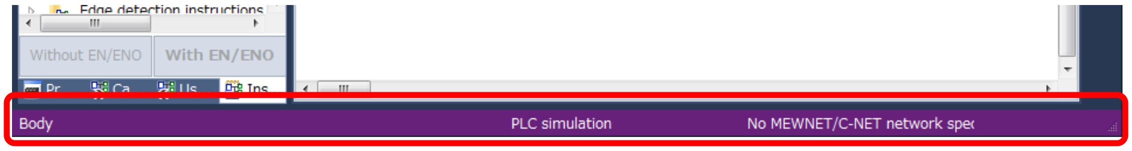


◆ **NOTES**

- **If the Standard Tool Bar is not displayed, reboot FPWIN Pro7.**
- **The display of other Tool Bars and icons vary by the programming mode and editor.**
- **Only icons that are active can be used.**

2.2.3 Status Bar

The Status Bar is at the bottom of the FPWIN Pro7 window. It is possible to display the PLC model, communication setting, time, and other information in the Status Bar.

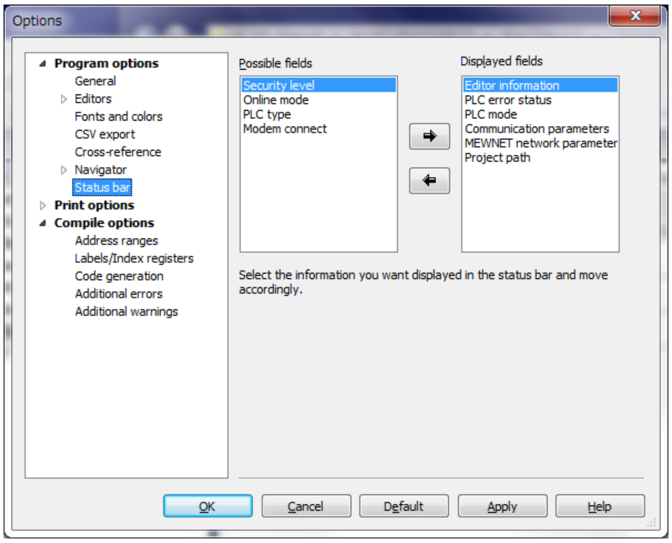


Items to be displayed in the Status Bar can be specified under "Status Bar" in the Options dialog box.
The Options dialog box can be selected by proceeding from the Menu Bar to "Extras" -> "Options" -> "Status Bar." Open "Status Bar Setting" by right-clicking on the Status Bar.

◆ **PROCEDURE**

1. **Double-click on a field that is not active on the Status Bar.**

   The "Options" dialog box as indicated below is displayed.



Select the information you want displayed in the status bar and move accordingly.

2. **In the "Possible fields," select items to be added to the display.**
3. **Press the ➔ button.**
4. **In the list of "Displayed fields," select items to be deleted.**
5. **Press the ➔ button.**
6. **Press the [OK] button.**

   After the setting is completed, it is possible to check, on the Tooltip, the behavior when an active field is double-clicked on the Status Bar.

◆ **NOTES**

When the [Default] button is pressed in the Options window, the setting in the Status Bar is returned to the default value.

3
Menu Bar Project

3.1 Create a New Project

To create a new project, there are three methods as follows.

- Use the "New project wizard" wizard
- Open a new project from the file
- Open a new project from PLC

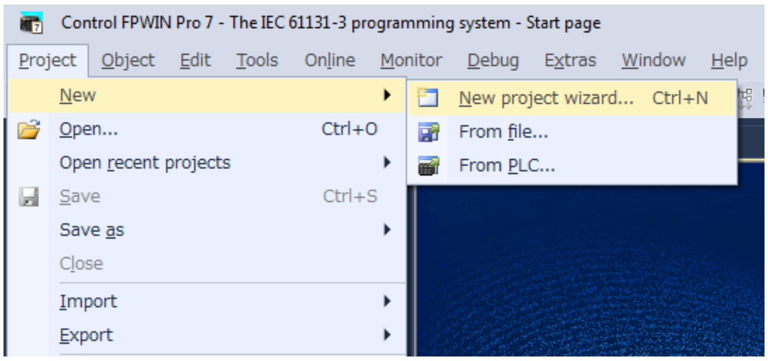
3.1.1 Use the "New project wizard"

By using the "Create a new project" wizard, a new project can be created with FPWIN Pro7.

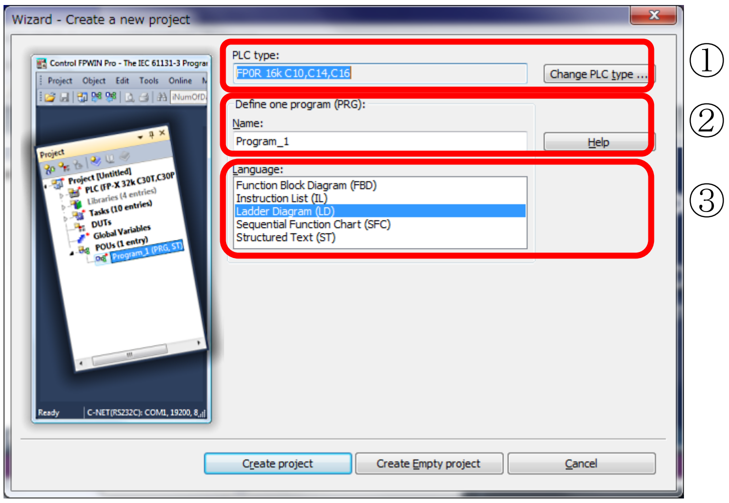
The "New project wizard" is displayed by proceeding from the Menu Bar to "Project" -> "New" -> "New project wizard." (The selection can also be selected in the FPWIN Pro7 startup screen.)

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Project" -> "New" -> "New project wizard." Alternatively, select "New project" in the Start Page.**



2. Change the PLC model if necessary.



PLC type:
FP0R 16k C10,C14,C16

Define one program (PRG):
Name:
Program_1

Language:
Function Block Diagram (FBD)
Instruction List (IL)
Ladder Diagram (LD)
Sequential Function Chart (SFC)
Structured Text (ST)

① **PLC type:** Press the [Change PLC model] button to display a list of PLC models and open the dialog box. The PLC models with parentheses indicated cannot be selected. In the case of the FPWIN Pro7 full version, all PLC models can be selected.

3. Define one Program (PRG).

In FPWIN Pro7, programming is performed by dividing the entire program task into smaller programs.

② **Name:** Enter the name of the first program of the project. When the project is created, this program is assigned to Task "Programs." This task is executed in every scan of PLC.

③ **Language:** Specify the type of editor to create a program.

4. Select [Create project], [Create empty project] or [Cancel] button.

To create a new project, press the [Create project] button.
To use an import object (*.asc, *.st, *.type), select [Create empty project].

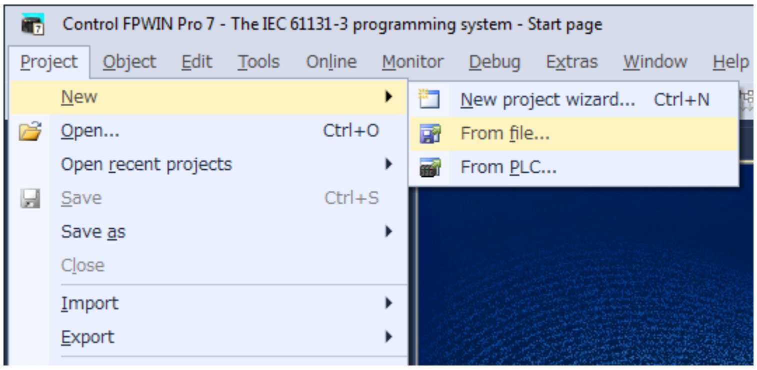
3.1.2 Open a new project from the file

To open a project, the following files can be selected.

- FPWIN Pro7 export files (*.asc)
- Packed backup files (*.pcd, *.pce)
- FPWIN GR files (*.fp, *.spg)

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Project" -> "New" -> "From File."**



2. **Select a path.**
3. **Select a file name.**
4. **Press the [Open] button.**

   From the selected file, a new project is created. If the [Cancel] button is clicked, the process is canceled.

For more details about packed backup files (*.pcd, *.pce) and FPWIN GR files (*.fp, *.spg), refer to the online Help.

3.1.3 Open a new project from PLC

Proceed from the Menu Bar to "Project" -> "New" -> "From PLC."Alternatively, select "Read a project saved in PLC" in the Start Page, to upload the project stored in PLC to FPWIN Pro7. (Example: Proceed from the Menu Bar to "Project" -> "New" -> "PLC.")

When this command is executed, an empty project is created, and an attempt is made to establish communication with PLC. If this process is normally completed, and if program code and project information are present in PLC, they are uploaded. Project information contains the content of all editors, PLC configuration, compiler options, the content of user library, etc. If user library is contained, it is displayed in the project folder.

◆ **NOTES**

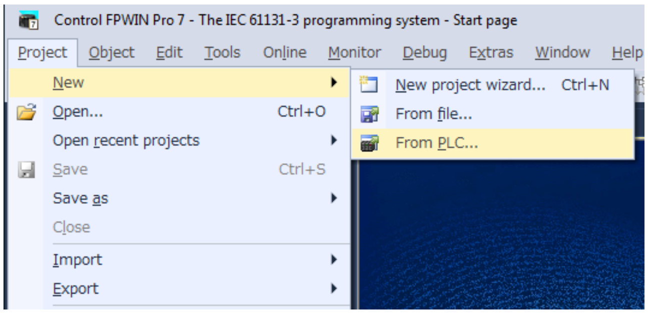
The project can be opened only if PLC is equipped with comment memory. In the case of a PLC model that has no comment memory, only the program code of PLC is uploaded. In this case, the program code is converted into ladder diagrams (for more details, refer to "Help" in FPWIN Pro7).

The following PLC models are equipped with comment memory.

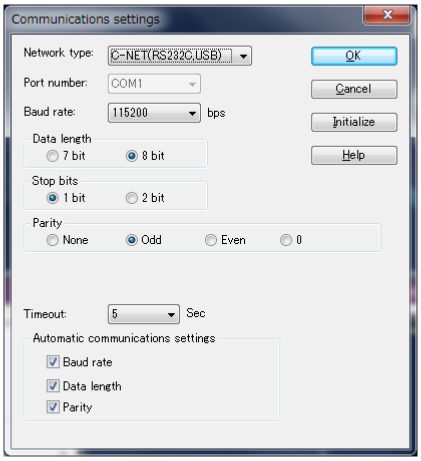
PLC model	Necessity of options	Type of option memory
FP7	Unnecessary	
FP0H	Unnecessary	
FP0R	Unnecessary	
FP-X、FP-XH	Unnecessary	
FP Σ	Unnecessary	
FP2	Necessary	Additional memory unit (FP2-EM1, FP2-EM2, FP2-EM3)
FP2SH	Unnecessary	

◆ **PROCEDURE**

1. Proceed from the Menu Bar to "Project" -> "New" -> "From PLC."



2. Set communication parameters to enable communication with PLC.


Communications settings

Network type: C-NET(RS232C,USB)

Port number: COM1

Baud rate: 115200 bps

Data length: 7 bit [✓ 8 bit]

Stop bits: [✓ 1 bit] 2 bit

Parity: None [✓ Odd] Even 0

Timeout: 5 Sec

Automatic communications settings

- ✓ Baud rate
- ✓ Data length
- ✓ Parity

When a message "The program code and the code in PLC do not match" is displayed, press the [OK] button. This error message is displayed when the project of PLC is incremental compiled.

3.2 Operations of the Project

The following operations can be performed for the created project.

- Open
- Close
- Save, Save As

The processes of Open, Close, Overwrite Save, Save As, etc. are the same as those in other applications of Windows.

3.2.1 Open

In the Menu Bar, click "Project" -> "Open," to open an FPWIN Pro7 project that is saved in PC. This operation can be performed while another project is open.

To open a project created using the former version (FPWIN Pro), refer to the online Help.

◆ **NOTES**

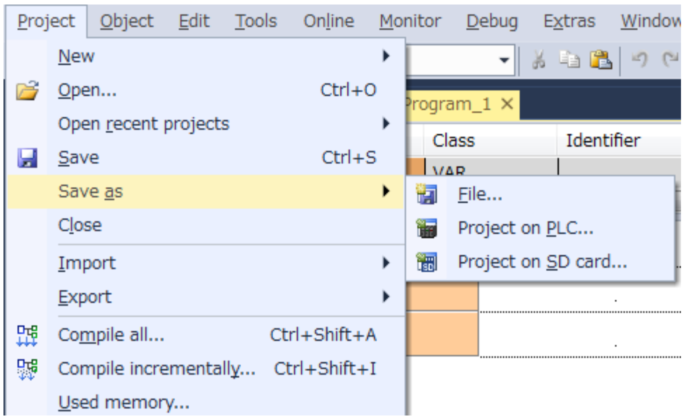
If a project to be opened has not been closed normally, a dialog box is opened. If [Backup] is selected, the previous editing is lost. If [Restore] is selected, the previous editing is restored as far as possible.

3.2.2 Open a project that has been used recently

In the list of projects that have been used recently, the edited project is displayed. Select a project. The number of projects that have been used recently to be displayed in the list can be changed by proceeding from the Menu Bar to "Extras" -> "Options" -> "Program options" -> "General."

3.2.3 Save a project

When creating a program, save the data frequently to avoid the loss of data, in preparation for system interruption, power failure, or other emergency cases. A newly created project can be saved by proceeding from the Menu Bar to "Project" -> "Save As" -> "File."



If a change is made to the project that has been saved, Overwrite Save the current project by proceeding from the Menu Bar to "Project" -> "Save," or by clicking 💾.

◆ **NOTES**

If you proceed from the Menu Bar to "Project" -> "Save," only a copied file is saved without saving the original project, unlike in the case of "Project" -> "Save As." While the copy is open, the original project is closed.

3.3 Import

3.3.1 Import of objects

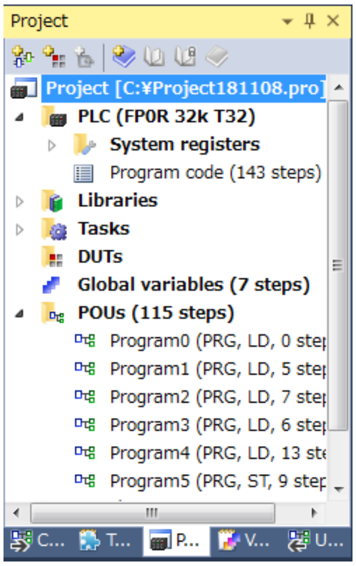
Objects such as POU, list of global variables, and tasks, can be imported. It is possible to save a part of the program in a hard disk, etc., and reuse it in another project or elsewhere. Proceed from the Menu Bar to "Project" -> "Import" -> "Object," to extract and import individual objects from the exported project.

Files that can be imported are as follows.

- FPWIN Pro7 files (*.asc)
- Reusability level: POU files (*.st)
- Reusability level: Type file (*.typ)

◆ **PROCEDURE**

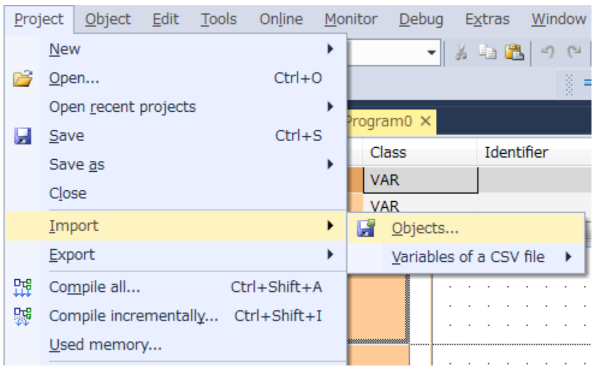
1. **In the project pane, select objects such as POU and tasks.**



Project
Project [C:¥Project181108.pro]
PLC (FP0R 32k T32)
System registers
Program code (143 steps)
Libraries
Tasks
DUTs
Global variables (7 steps)
POUs (115 steps)
Program0 (PRG, LD, 0 step
Program1 (PRG, LD, 5 step
Program2 (PRG, LD, 7 step
Program3 (PRG, LD, 6 step
Program4 (PRG, LD, 13 ste
Program5 (PRG, ST, 9 step

Only file types that correspond to the objects that are selected in the "Project" pane are imported. To select multiple objects, use <Shift> or <Ctrl>.

2. **Proceed from the Menu Bar to "Project" -> "Import" -> "Objects"**



3. **Select a folder.**
4. **Select a file name.**

   To select multiple objects, use <Shift> or <Ctrl>.
5. **Press the [OK] button.**

   The content of the selected file is imported into the current FPWIN Pro7 project, and is displayed under the objects of the import destination in the project pane.

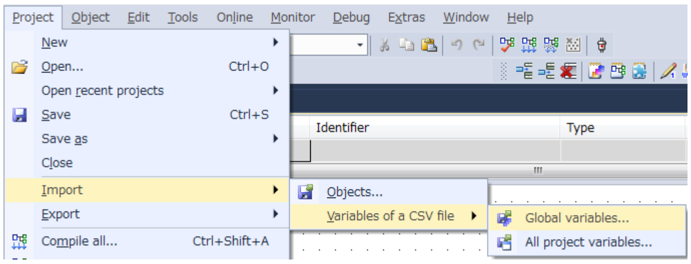
When the import process is completed normally, a message is displayed.

3.3.2 Import global variables (CSV file)

Proceed from the Menu Bar to "Project" -> "Import" -> "Variables of a CSV file" -> "Global variables," to import variables from the text file into the list of global variables. Text data must be in the CSV (Comma Separated Values) format. In the CSV format, individual values are separated by a comma (",") or a semicolon (";"). (The separator can be specified by proceeding from the Menu Bar to "Extras" -> "Options" -> "Program options" -> "Export of CSV.")

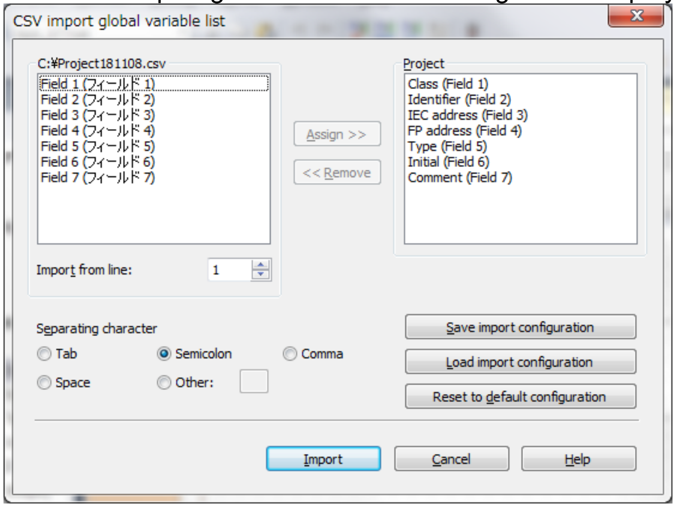
◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Project" -> "Import" -> "Variables of a CSV file" -> "Global variables."**



2. **Select a path.**
3. **Select a file name.**
4. **Press the [Open] button.**

   Any CSV file can be imported.
   The "CSV import global variable list" dialog box is displayed.


CSV import global variable list

C:¥Project181108.csv
Field 1 (フィールド 1)
Field 2 (フィールド 2)
Field 3 (フィールド 3)
Field 4 (フィールド 4)
Field 5 (フィールド 5)
Field 6 (フィールド 6)
Field 7 (フィールド 7)

Project
Class (Field 1)
Identifier (Field 2)
IEC address (Field 3)
FP address (Field 4)
Type (Field 5)
Initial (Field 6)
Comment (Field 7)

Import from line: 1

Separating character: Tab, [✓ Semicolon], Comma, Space, Other:

Assign >>
<< Remove
Save import configuration
Load import configuration
Reset to default configuration
Import
Cancel
Help

In this dialog box, the following options can be selected.

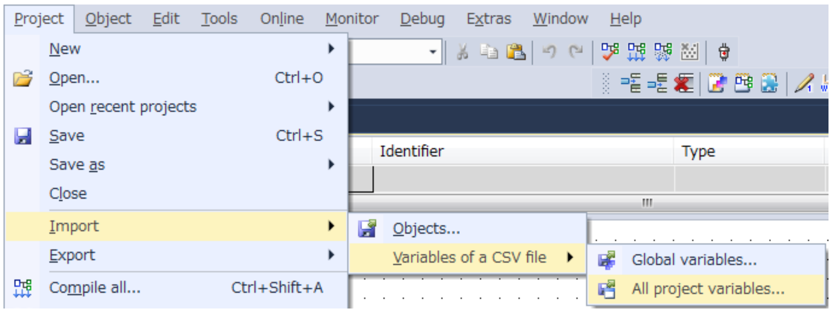
Option	Description
Import from line	Import is started from the specified line. The content of the selected line is displayed to the left of the list box.
[Assign]	The item selected in the list on the left is assigned to the item selected in the list on the right. The assignment is displayed along with the field number. In the list on the right, multiple items can be selected.
[Remove]	The item selected in the list on the right is deleted. "Unspecified" is indicated for the item.
Separating character	Specify the separator for the CSV file to be imported.
[Save import configuration]	The import setting is saved in a file.
[Load import configuration]	The import setting is read from a file.
[Reset to default configuration]	Standard values are applied to all settings.
[Import]	Import is started from the specified "Import from line."
[Cancel]	The import process is canceled, and the dialog box is closed.

3.3.3 Import all project variables (CSV file)

Proceed from the Menu Bar to "Project" -> "Import" -> "Variables of a CSV file" -> "All project variables," to import variables from the text file into the list of global variables. Text data must be in the CSV (Comma Separated Values) format. In the CSV format, individual values are separated by a comma (",") or a semicolon (";"). (The separator can be specified by proceeding from the Menu Bar to "Extras" -> "Options" -> "Program options" -> "Export of CSV.")

◆ PROCEDURE

1. **Proceed from the Menu Bar to "Project" -> "Import" -> "Variables of a CSV file" -> "All project variables."**



2. **Select a path.**
3. **Select a file name.**
4. **Press the [OK] button.**

◆ NOTES

A CSV file needs to have been created by proceeding from executing the Menu Bar to "Project" -> "Export" -> "Variables of a CSV file" -> "All project variables."

3.4 Export

3.4.1 Export of objects

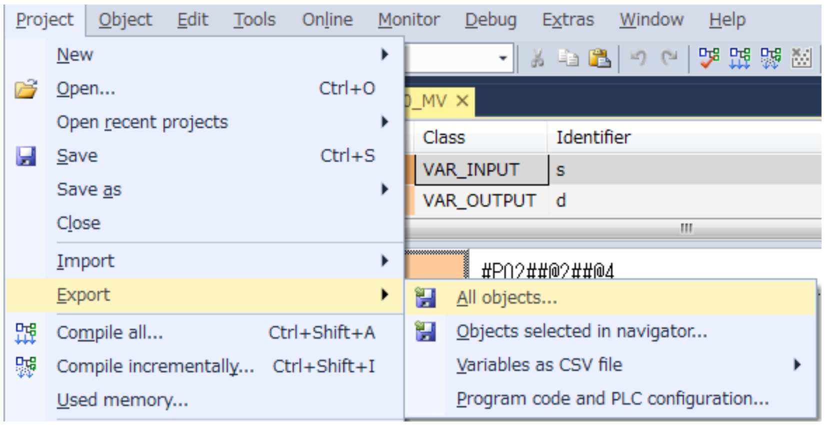
Objects such as POU, list of global variables, and tasks, can be exported. It is possible to save a part of the program in a hard disk, etc., and reuse it in another project or elsewhere.

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Project" -> "Export" -> "All objects."**

   It is possible to export the entire project as a text (ASCII) file, for the purpose of backup, etc.

   Alternatively, proceed from the Menu Bar to "Project" -> "Export" -> "Objects selected in Navigator."
   Only selected objects (Example: List of global variables) are exported. To select multiple objects, use <Shift> or <Ctrl>. Instead of the "Project" menu, the selection can also be made in the context menu that is displayed by right-clicking.



2. **Select a path.**

3. **Select a file.**

   File types that can be selected:
   FPWIN Pro7 file (*.asc); reusability level: POU file (*.st); or type file (*.typ)
   The following two formats can be used.

Unicode	UTF-16 format (default) This is a format for correctly displaying all Unicode characters. This format can be exported in FPWIN Pro of Version 6.0 or later, or in FPWIN Pro7.
Multibyte	(Multi-Byte Character Set, MBCS) format This format can be exported in FPWIN Pro of all establishing versions, or in FPWIN Pro7.

4. **Enter the file name.**
5. **Press the [Save] button.**

◆ **NOTES**

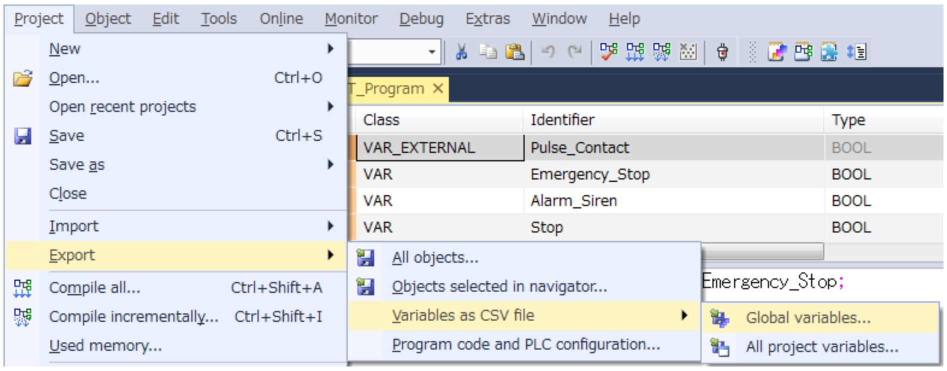
- **The file type "reusability level:POU files (*.st)" can be used only for POU created by ST Editor.**
- **The file type "reusability level: Type files (*.typ)" can be used only for structures created under "Structure (DUT) of Navigator."**

3.4.2 Export of global variables (CSV file)

Proceed from the Menu Bar to "Project" -> "Export" -> “Variables of a CSV file” -> "Global variables," to export a file in the CSV (Character Separated Values) format. In the CSV format, individual values are separated by a comma (",") or a semicolon (";"). (The separator can be specified by proceeding from the Menu Bar to “Extras” -> "Options" -> "Program options" -> "Export of CSV.") It is also possible to import the information of variables in an external program using these text data.

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Project" -> "Export" -> “Variables as CSV file” -> "Global variables."**



Class	Identifier	Type
VAR_EXTERNAL	Pulse_Contact	BOOL
VAR	Emergency_Stop	BOOL
VAR	Alarm_Siren	BOOL
VAR	Stop	BOOL

2. **The "CSV export global variable list" dialog box is displayed.**
3. **Select a path.**
4. **Enter the file name.**
5. **Select a file.**

   The following two formats can be used.

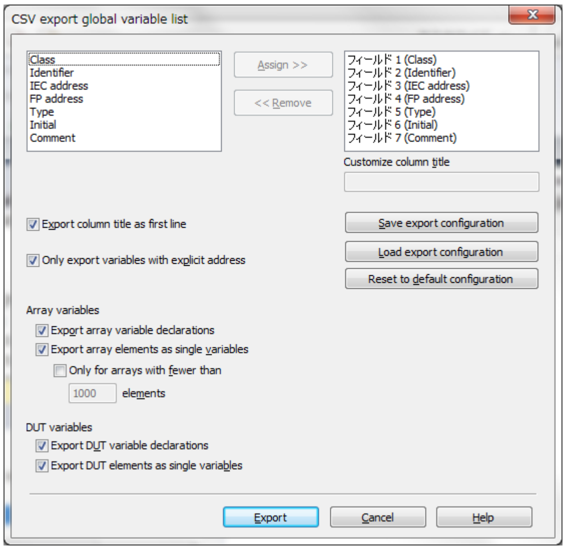
Unicode	UTF-16 format (default) This is a format for correctly displaying all Unicode characters. This format can be exported in FPWIN Pro of Version 6.0 or later, or in FPWIN Pro7.
Multibyte	(Multi-Byte Character Set, MBCS) format This format can be exported in FPWIN Pro of all establishing versions, or in FPWIN Pro7.

6. **Press the [OK] button.**

All variables in FPWIN Pro7 are exported into a text file, along with the variable names and addresses. In addition, the information of the entire project is also exported. (Examples: Project name, Compile time, Version of FPWIN Pro7)

The following registered information is saved in the text data.
Class, Identifier, IEC address, FP address, Type, Initial, Comment

The “CSV export global variable list” dialog box is displayed.


Assign >>
<< Remove
Save export configuration
Load export configuration
Reset to default configuration
Export
Cancel
Help

CSV export global variable list

Class
Identifier
IEC address
FP address
Type
Initial
Comment

フィールド 1 (Class)
フィールド 2 (Identifier)
フィールド 3 (IEC address)
フィールド 4 (FP address)
フィールド 5 (Type)
フィールド 6 (Initial)
フィールド 7 (Comment)

Customize column title

[✓] Export column title as first line

[✓] Only export variables with explicit address

Array variables

[✓] Export array variable declarations

[✓] Export array elements as single variables

[ ] Only for arrays with fewer than

1000 elements

DUT variables

[✓] Export DUT variable declarations

[✓] Export DUT elements as single variables

In this dialog box, the following options can be selected.

Option	Description
[Assign]	The item selected in the list on the left is assigned to the item selected in the list on the right. The assignment is displayed along with the field number. In the list on the right, multiple items can be selected. The selected items are displayed on the right. It is also possible to rename them in the "Customize column title" text box.
[Remove]	The item selected in the list on the right is deleted. "Unspecified" is indicated for the item.
[Save export configuration]	The export setting is saved in a file.
[Load export configuration]	The export setting is read from a file.
[Reset to default configuration]	Standard values are applied to all settings.
Export column title as first line	Select this when a header is written into the CSV file. In the header, names and other information can be registered in Field 1, Field 2, and Field 3.
Only export variables with explicit address	If this check box is turned on, it is possible to export only variables that have explicit addresses entered into the list of global variables.
Export array variables declarations	If this check box is turned on, it is possible to export Array Variable Declaration.
Export array elements as single variables	If this check box is turned on, it is possible to export all elements of the array. It is possible to set an upper limit to the number of elements of array to be exported.
Only for arrays with fewer than	To avoid the export of a large-scale array, define the limitation here. Only arrays that contain elements fewer than the limitation will be exported.
Export DUT variable declarations	If this check box is turned on, it is possible to export DUT Variable Declaration.
Export DUT elements as single variables	If this check box is turned on, it is possible to export all elements of DUT in the export file as a single element.

Option	Description
[Export]	Export is started.
[Cancel]	The export process is canceled, and the dialog box is closed.

◆ **NOTES**

To access DUTs and arrays that are not directly supported, individual elements are exported into the CSV file as variables.

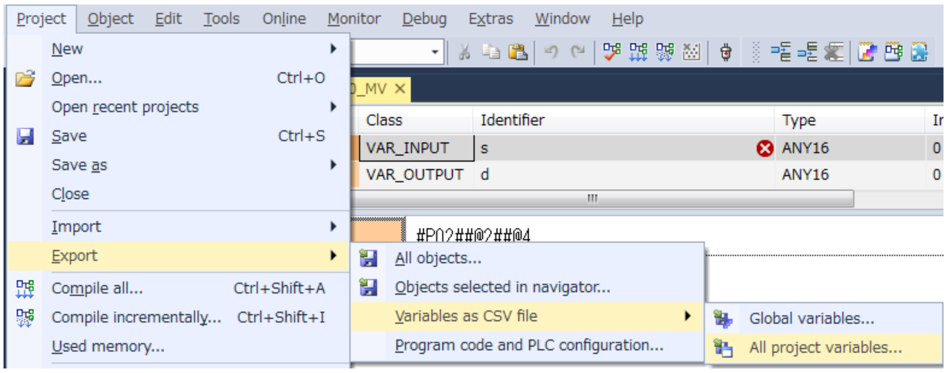
For the exchange of global variables with other softwares, refer to the Online Help.

3.4.3 Export of project variables (CSV file)

Proceed from the Menu Bar to "Project" -> "Export" -> “Variables of a CSV file” -> "All project variables," to export a file in the CSV (Character Separated Values) format. In the CSV format, individual values are separated by a comma (",") or a semicolon (";"). (The separator can be specified by proceeding from the Menu Bar to “Extras” -> "Options" -> "Program options" -> "Export of CSV.") It is also possible to import the information of variables in an external program using these text data.

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Project" -> "Export" -> "Variables as CSV file" -> "All project variables."**



2. **Select a path.**
3. **Select a file.**

   The following two formats can be used.

Unicode	UTF-16 format (default) This is a format for correctly displaying all Unicode characters. This format can be exported in FPWIN Pro of Version 6.0 or later, or in FPWIN Pro7.
Multibyte	(Multi-Byte Character Set, MBCS) format This format can be exported in FPWIN Pro of all establishing versions, or in FPWIN Pro7.

4. **Press the [Save] button.**

3.4.4 Export of program code and PLC configuration

A program that has been compiled can be saved in a hard disk, etc. in an FP HEX file, Motorola HEX file, or Intel HEX file format. By using the EPROM writing program, a file in the Motorola HEX or Intel HEX format can be written into the EP-ROM attached to PLC as a program memory.

This function can be used in models to which EP-ROM can be attached as an option memory.

◆ **NOTES**

When attaching EP-ROM to PLC, make sure that the write protection switch of PLC is in the "Writing enabled" position, and that the EP-ROM slot is not switched to "Read protection" during the reading process.

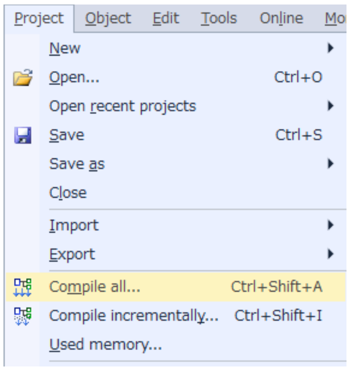
File format	Description
*.fp *.spg	FP HEX format This format can be used when you want to have a program memorized in an IC card, and transfer it to PLC later, or when you want to read a program using the programming software "FPWIN GR, NPST-GR."
*.itl	Intel HEX format
*.mtl	Motorola HEX format

3.5 Full Compile

A program needs to be compiled before downloading. Proceed from the Menu Bar to "Project" -> " Compile all," and compile the entire project. Individual POUs are re-checked for syntax errors during the compile process, and are converted into program code.

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Project" -> "Compile all," click , or press <Ctrl>+<Shift>+<A>.**


Project Object Edit Tools Online Mo
New
Open... Ctrl+O
Open recent projects
Save Ctrl+S
Save as
Close
Import
Export
Compile all... Ctrl+Shift+A
Compile incrementally... Ctrl+Shift+I
Used memory...

If no error occurs in the compile process, proceed from the Menu Bar to "Online " -> "Download program code and PLC configuration" to download the compiled project.

◆ **NOTES**

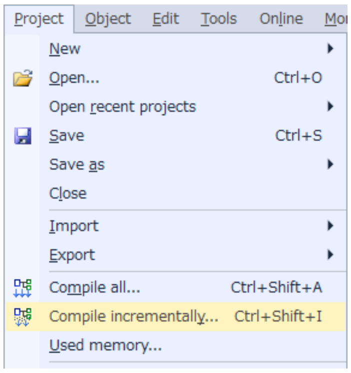
Only program (PRG)-type POUs registered in "Task" are compiled.

3.6 Incremental Compile

In the Navigator, objects that contain changes that have not been compiled are indicated with an asterisk (*). Proceed from the Menu Bar to "Project" -> "Compile incrementally," and compile these objects only.

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Project" -> "Compile incrementally," click , or press <Ctrl>+<Shift>+<I>.**



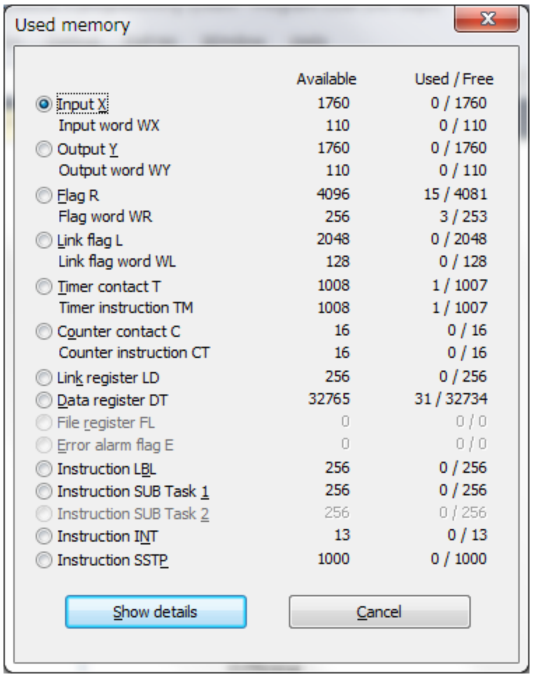
If no error occurs, proceed from the Menu Bar to "Online " -> "Download program code and PLC configuration" to perform downloading.

◆ **NOTES**

- **If "Compile incrementally" is executed multiple times in a series, the memory is used fragmentally. This may cause memory shortage in the case of PLC that has small memory. Therefore, it is recommended to perform "Compile all" on a periodic basis. "Compile all " restores fragmental memory assignment.**
- **POU is a unit that comprises a program. If a part of POU is changed, the entire POU needs to be re-compiled. To minimize the time for compile, its recommended to divide the entire program into as many POUs as possible in the programming process. (Structured programming)**
- **Only program (PRG)-type POUs registered in "Task" are compiled.**

3.7 Used memory

Proceed from the Menu Bar to "Project" -> "Used memory," to display the list of memory areas that are used / not used.



Used memory

	Available	Used / Free
● Input X	1760	0 / 1760
Input word WX	110	0 / 110
○ Output Y	1760	0 / 1760
Output word WY	110	0 / 110
○ Flag R	4096	15 / 4081
Flag word WR	256	3 / 253
○ Link flag L	2048	0 / 2048
Link flag word WL	128	0 / 128
○ Timer contact T	1008	1 / 1007
Timer instruction TM	1008	1 / 1007
○ Counter contact C	16	0 / 16
Counter instruction CT	16	0 / 16
○ Link register LD	256	0 / 256
○ Data register DT	32765	31 / 32734
○ File register FL	0	0 / 0
○ Error alarm flag E	0	0 / 0
○ Instruction LBL	256	0 / 256
○ Instruction SUB Task 1	256	0 / 256
○ Instruction SUB Task 2	256	0 / 256
○ Instruction INT	13	0 / 13
○ Instruction SSTP	1000	0 / 1000

◆ **NOTES**

- **The numbers of relays and registers that can be used vary by the PLC model and PLC configuration that are used.**
- **SUB instruction refers to the memory for user-defined Function Block (FB) and Function (FUN).**
- **SSTP instruction is used only in the Sequential Function Chart (SFC). The list of SSTP instructions used / not used is displayed.**

When [Show details] button is pressed, detailed information concerning the item selected in the "Used memory" dialog box is displayed.

[Display details] External input / External output, Internal relay, Link relay, Timer, Counter, Register instructions

Enter the starting address of the word address to be displayed into the "Display address" box in the top left. In the bottom left of the dialog box, symbols that indicate user areas used / not used, and system areas used / not used.

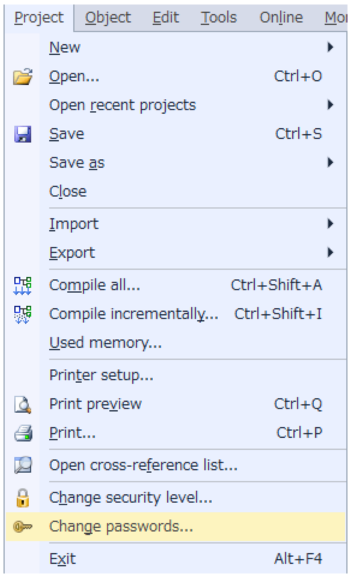
The information that is displayed when the [Show details] button is pressed varies by the selected item.

3.8 Change password

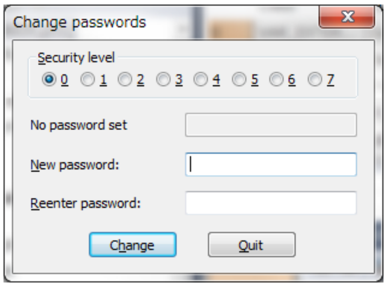
By setting a security level for each object, it is possible to individually control access to each object in the project.First, set a password for each security level.The editing of an object can only be performed by users who knows the password for the security level of the object.

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Project" -> "Change passwords"**



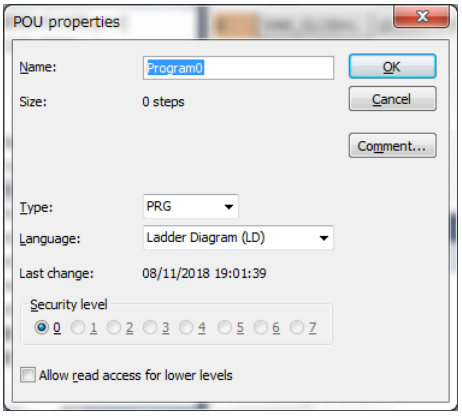
2. **Set a password for each security level.**



When setting the password for the first time, enter the password into the "New password" field and the "Reenter password" field, and press the [Change] button, for each security level.A message that confirms that the password has been registered is displayed.

3. **In the Project Navigator, select an object for access control.**

4. **Proceed from the Menu Bar to "Object" -> "Properties."**



Name: Program0
Size: 0 steps
Type: PRG
Language: Ladder Diagram (LD)
Last change: 08/11/2018 19:01:39
Security level: [✓ 0] 1 2 3 4 5 6 7
Allow read access for lower levels

Setting is possible only for a security level for which a password has been set, and which is lower than the current security level.

5. **Select a security level. Turn on the check box for "Allow read access for lower levels" when necessary.**

3.9 Change Security Level

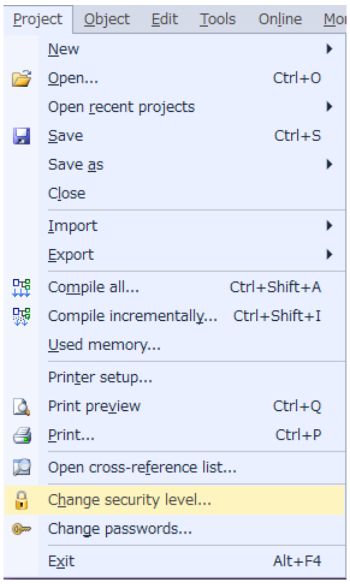
Proceed from the Menu Bar to "Project" -> "Change security level," to change the access level to an object that is protected with a password.

◆ **NOTES**

- **It is not possible to change the security level for an object of a security level higher than that of the current project.**
- **If access to an object of a higher security level is controlled, proceed from the Menu Bar to "Object" -> "Properties," and check "Allow read access for lower levels" to enable reading.**

◆ **PROCEDURE**

1. **In the Project Navigator, select an object.**
2. **Proceed from the Menu Bar to "Project" -> "Change security level."**



3. **Select a security level, and enter a password.**
4. **Press the [OK] button.**

   This operation enables access to the selected object. It can be changed when necessary. Access control to the object remains valid after the change. It is not necessary to re-set the access control.

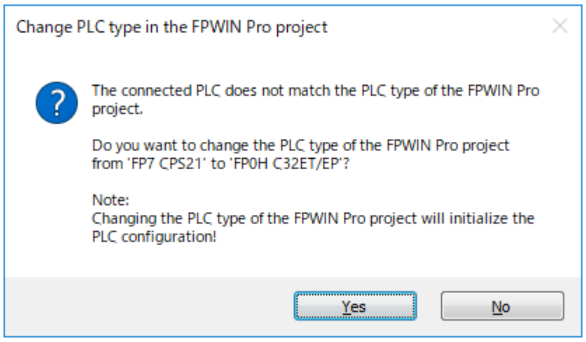
4

Menu Bar: Online

4.1 Online Mode

Proceed from the Menu Bar to "Online " -> "Online mode." Alternatively, press the <Shift> + <Esc> keys. Alternatively, click to switch between Online mode and Offline mode. In the Status Bar, the present mode ("Online" or "Offline") is displayed (if it is included in items to be displayed in the Status Bar). When Online, the icon is highlighted.

When switching from Offline mode to Online mode, a check is performed that the PLC to be connected matches the PLC model selected in the project. If it does not match, a confirmation message box is displayed whether or not to change the PLC model setting in the project. If the [No] button is pressed, FPWIN Pro7 is switched to Offline mode.


Change PLC type in the FPWIN Pro project

The connected PLC does not match the PLC type of the FPWIN Pro project.

Do you want to change the PLC type of the FPWIN Pro project from 'FP7 CPS21' to 'FP0H C32ET/EP'?

Note:
Changing the PLC type of the FPWIN Pro project will initialize the PLC configuration!

To change the PLC model, perform the process in the "PLC type" menu (4.4 Convert PLC models). Unless this process is performed, FPWIN Pro7 is not switched to Online mode. A message box is displayed to change the PLC model.

4.2 Online Edit Mode

Proceed from the Menu Bar to "Online " -> "Online edit mode." Alternatively, use to make a change in the body of the PLC program. Confirm that the project in the "Project" pane has been downloaded into PLC (Proceed from the Menu Bar to "Online " -> "Download program code and PLC configuration"). After the change has been made, proceed from the Menu Bar to "Online" -> "Download program code changes." Alternatively, select .

◆ NOTES

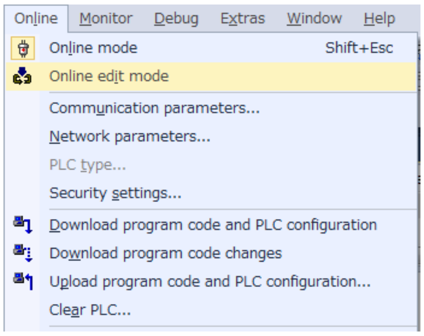
- **Online edit mode can be used both in the PROG. Mode and in the RUN Mode.**
- **It is not possible to declare a new variable in the online edit mode. It is convenient to declare several dummy variables during Offline mode. Alternatively, directly enter the address into the body.**
- **In the case of FP2, FP2SH, FP0, or FPX0:**
  The download of the changed portion of the program may not exceed the number of steps that can be downloaded in a single scan (128 steps). Therefore, minimize changes, or divide them into small amounts.
- **In the case of all other PLCs:**
  The number of steps that can be downloaded in a single scan is 512. If the number of steps exceed 512, the entire program code is downloaded. Therefore, multiple scans may become necessary. After the download of the entire program is completed, PLC is switched to the new program.

◆ PROCEDURE

1. **Open the body of the program to be changed.**

   Confirm that the project has been downloaded into PLC.

2. **Proceed from the Menu Bar to "Online " -> "Online edit mode," or click .**


Online
Monitor
Debug
Extras
Window
Help
Online mode
Shift+Esc
Online edit mode
Communication parameters...
Network parameters...
PLC type...
Security settings...
Download program code and PLC configuration
Download program code changes
Upload program code and PLC configuration...
Clear PLC...

3. **Change the content of the body.**

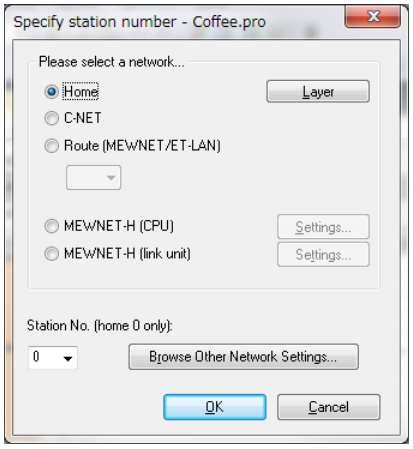
4. **Proceed from the Menu Bar to "Online " -> "Download program code changes," or click .**

   The changed portion is automatically compiled, and downloaded into PLC.

4.3 Specify Communication Station

The parameters for connecting to PLC network, such as MEWNET-H, MEWNET-P, MEWNET-W, and C-NET, should be set in the "Specify station number" dialog box. It is necessary to specify, on the computer, the network that is used by the PLC to be accessed. It is also possible to access PLC via layers.

Proceed from the Menu Bar to "Online " -> “Specify station number,” to open the “Specify station number” dialog box. In this dialog box, it is possible to set the type of network and parameters to access PLC.


Specify station number - Coffee.pro

Please select a network...

[✓ Home]

C-NET

Route (MEWNET/ET-LAN)

MEWNET-H (CPU)

MEWNET-H (link unit)

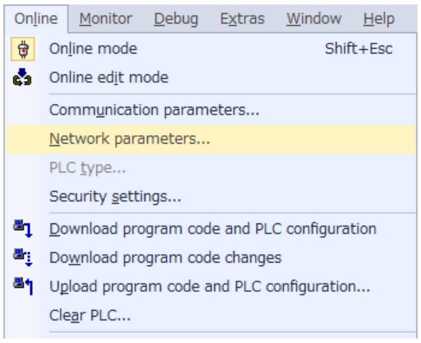
Station No. (home 0 only): 0

◆ **NOTES**

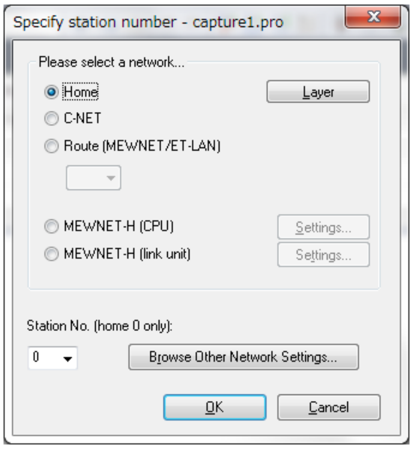
In the case of network such as MEWNET-H, MEWNET-P, MEWNET-W, and C-NET, it is necessary to perform settings, on the computer, for PLC on the network to be accessed. It is also possible to access PLC in a different layer. (Regarding settings for MEWNET-H, refer to the Online Help.)

◆ **PROCEDURE**

1. Proceed from the Menu Bar to "Online " -> "Network parameters"


Online
Monitor
Debug
Extras
Window
Help
Online mode
Shift+Esc
Online edit mode
Communication parameters...
Network parameters...
PLC type...
Security settings...
Download program code and PLC configuration
Download program code changes
Upload program code and PLC configuration...
Clear PLC...

2. Select a network type.


Specify station number - capture1.pro
Please select a network...
Home
Layer
C-NET
Route (MEWNET/ET-LAN)
MEWNET-H (CPU)
Settings...
MEWNET-H (link unit)
Settings...
Station No. (home 0 only):
0
Browse Other Network Settings...
OK
Cancel

Item	Description
Home	Select this for directly connecting PLC to the computer, without using a network.
C-NET	Select this for accessing PLC via C-NET. The station numbers should be specified in the range from 1 to 99. Note: The same communication parameters should be set for all PLCs on C-NET, except for the station numbers.

Item	Description
Route (MEWNET/ET-LAN)	To access PLC that is directly connected to the computer, and to PLC that is linked via the MEWNET-W, MEWNET-P or ET-LAN network, select a Route No. The station numbers should be specified in the range from 1 to 63. The route numbers should be specified as follows. Starting from the unit that is the nearest to the CPU unit, count the following units. The Route No. for the first unit is 1. - MEWNET-W link unit - MEWNET-P link unit - ET-LAN unit - Computer Communication Unit (CCU) Route No. is a number assigned to a link unit. Example: If the CPU unit, ET-LAN unit, I/O unit, and MEWNET-W link unit are attached in this order, ET-LAN is assigned as the first route, and MEWNET-W is assigned as the second route. In this case, to access PLC on MEWNET-W, select "Route 2."
MEWNET-H (CPU)	Select this if the computer and the CPU tool port of PLC are directly connected, and PLC on MEWNET-H is to be accessed via the CPU of PLC. To set a layer or station number, press the [Setting] button. For more details concerning the MEWNET-H setting, refer to the Online Help.
MEWNET-H (link unit)	Select this if the computer is directly connected to the RS232C port on the MEWNET-H unit, and PLC on that MEWNET-H unit is to be accessed. To set a layer or station number, press the [Setting] button. For more details concerning the MEWNET-H setting, refer to the Online Help. Note: For the setting of RS232C port on the MEWNET-H unit, refer to the MEWNET-H Link Unit Manual.

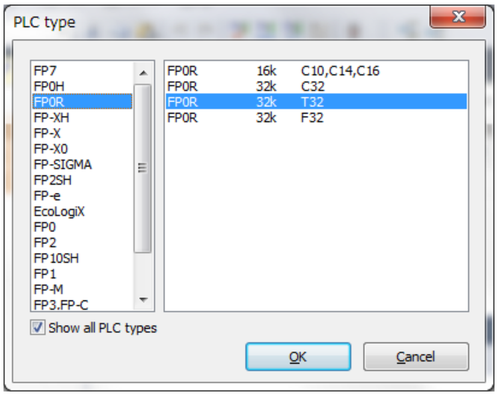
3. **When all settings have been completed, press the [OK] button.**
4. **Proceed from the Menu Bar to "Online " -> "Online mode." Alternatively, click .**

4.4 Convert PLC models

Proceed from the Menu Bar to "Online " -> "PLC type," to change the PLC model. The PLC model can be changed only in the Offline mode.

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Online " -> " PLC type."**



PLC type

FP7
FP0H
FP0R
FP-XH
FP-X
FP-X0
FP-SIGMA
FP2SH
FP-e
EcoLogiX
FP0
FP2
FP10SH
FP1
FP-M
FP3.FP-C

FP0R	16k	C10,C14,C16
FP0R	32k	C32
FP0R	32k	T32
FP0R	32k	F32

[✓] Show all PLC types

2. **Select a PLC model on the left side.**
3. **Select a capacity type on the right side.**
4. **Press the [OK] button.**

The "Convert the PLC setting and compile options" dialog box is displayed.

In this dialog box, selection can be made from the two options (Use default value / Hold the current setting) for conversion into a new PLC model.

When [Hold the current setting] is selected, a list is displayed indicating items that are set in default values due to the absence of the items in the new model.

The name of the new PLC model is displayed in the "PLC" portion of the project pane. Depending on the setting, the information is also displayed in the Status Bar.

◆ **NOTES**

- **When the [Use default value] button is pressed, the compile option setting is reset to the default value. In this process, the storage destination of global variables may be changed from a hold area to a non-hold area. Therefore, after changing PLC models, adjust the setting of hold/non-hold area under "Compile options." Proceed from the Menu Bar to "Extras" -> "Options" -> "Compile options" -> "Code generation," to hold the variables or change the compiler setting.**
- **When the PLC model is changed, all system registers are initialized, and returned to the default values.**

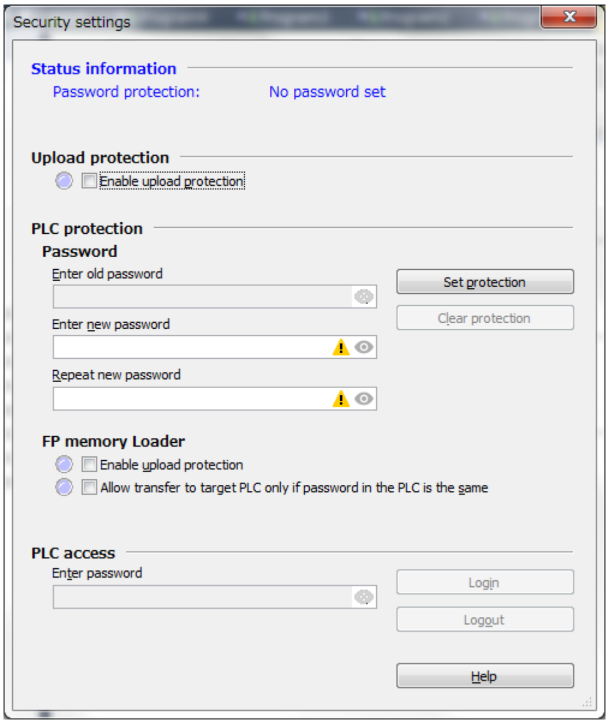
4.5 Security Setting

Proceed from the Menu Bar to "Online " -> "Security settings" to display the current security information, and open a dialog box for performing protection setting of PLC. If PLC is protected with a password, it is necessary to enter the password in this dialog box.

Security setting that can be performed varies by PLC model.

- Password protection
- Prohibition of upload of PLC program
- Security setting for FP memory loader

4.5.1 Security Setting



Security settings

Status information

Password protection: No password set

Upload protection

☐ Enable upload protection

PLC protection

Password

Enter old password

Enter new password

Repeat new password

Set protection

Clear protection

FP memory Loader

☐ Enable upload protection

☐ Allow transfer to target PLC only if password in the PLC is the same

PLC access

Enter password

Login

Logout

Help

The LED of the dialog box indicates the current protection status of PLC. To display Tooltip, align the mouse cursor over the LED.

■ Status information

Security information indicates whether or not a password is set. If password protection is applied, it is necessary to enter the password into the entry field of "PLC access" and log into PLC.

"Acceptable retries of password" refers to how many times passwords may be tried until the correct password is entered. If invalid passwords are entered three times, and then PLC is powered back on, then three more trials can be made.

■ Upload protection

If upload is disabled in setting, the following operations cannot be made.

- Upload of a project or program code to PC
- Upload of system register to PC
- Transfer of program to master memory cassette (only FP-X, FP-XH)

The setting of this function can be canceled using FPWIN Pro7. Note that all programs, system registers, and password information will be deleted.

If upload is prohibited in setting, it is possible to edit the file in the Online edit mode of FPWIN Pro7. Note that the program in FPWIN Pro7 will not match the program in PLC.

■ PLC protection

In this field, a new password up to 32 digits can be set, or an existing password can be changed. (The number of acceptable digits varies by PLC model.)

◆ **NOTES**

- **Be sure to remember the password. Without the password, a program in PLC that is password-protected cannot be read.**
- **Our support team cannot reset the password in the case of forgotten password.**
- **If the [Clear protection conditions] button is pressed without logging in, not only the password, but also the parameters stored in the program and comment memory will be deleted. This process may take an extremely long time.**

Setting of protection conditions

1. Enter a password into the entry field of "Enter new password."
2. Enter the password into the entry field of " Repeat new password."
3. Press the [Set protection] button to set the password.
4. To activate the password that has been set, press the [Logout] button.

Change protection conditions

To change the current password, you need to log into PLC, or the password needs to be displayed in the entry field of "Current password."

1. Enter a password into the entry field of "Enter new password."
2. Enter the password into the entry field of "Repeat new password."
3. Press the [Set protection] button to change the password.
4. To activate the password that has been set, press the [Logout] button.

Clear protection conditions

1. Press the [Clear protection] button.

◆ **NOTES**

To access PLC for which a password has been set, it is necessary to log in every time the PLC is powered on.

■ Memory setting for FP memory loader

Enable upload protection	When this setting is turned on, it is impossible to upload a PLC program to the FP memory loader (Ver. 2 or later).
Allow transfer to target PLV only if password in the PLC is the same	A program can be transferred from PLC to another PLC, via the FP memory loader (Ver. 2 or later), only if the two PLCs share the same password.

■ PLC access

Log into and log out of PLC that is password-protected.

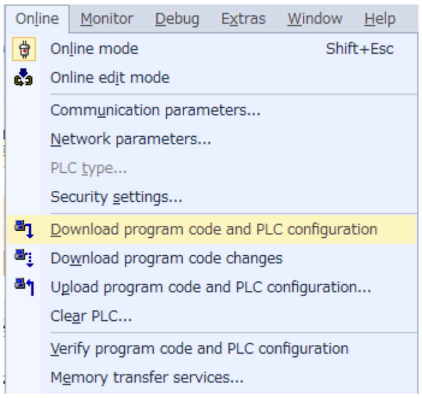
Login	To log into PLC, enter the password into the "Enter password" field, and press the [Login] button.
Logout	After you log out of PLC, it becomes impossible to perform upload or download.

4.6 Download program code and PLC configuration

To download a compiled program, it is necessary to switch PLC into the PROG. mode.

◆ **PROCEDURE**

1. **Switch PLC to the PROG.Mode.**
2. **Proceed from the Menu Bar to "Online " -> "Download program code and PLC configuration," or select .**


Online
Monitor
Debug
Extras
Window
Help
Online mode
Shift+Esc
Online edit mode
Communication parameters...
Network parameters...
PLC type...
Security settings...
Download program code and PLC configuration
Download program code changes
Upload program code and PLC configuration...
Clear PLC...
Verify program code and PLC configuration
Memory transfer services...

3. **Proceed from the Menu Bar to "Online " -> "Change PLC Mode." Alternatively, select , and switch to the RUN Mode.**

◆ **NOTES**

If PLC is equipped with comment memory, it is possible to download a project, instead of program code, by proceeding from "Project" -> "Save as" -> "Save project in PLC." Project information contains the content of all editors, PLC configuration, compiler options, the content of user library, etc.

The following PLC models are equipped with comment memory.

PLC	Necessity of option	Type of option memory
FP7	Unnecessary	
FP0H	Unnecessary	
FP0R	Unnecessary	
FP-X, FP-XH	Unnecessary	
FPΣ	Unnecessary	
FP2	Necessary	Additional memory unit: FP2-EM1, FP2-EM2, FP2-EM3
FP2SH	Unnecessary	

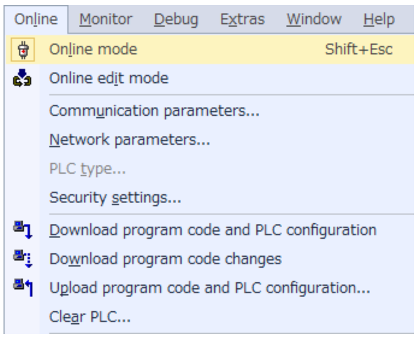
4.7 Download the changed portion of program code

It is possible to download a small changed portion of the program during Online mode, without switching to PROG. Mode. Note that the following restrictions are applied.

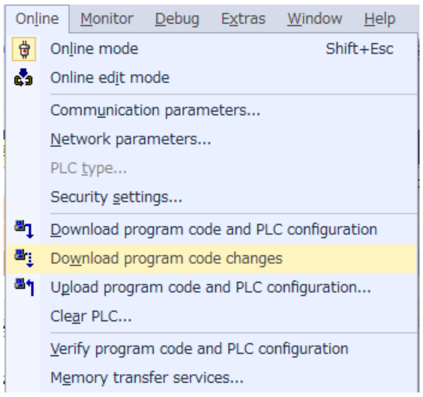
- It is not possible to change or delete global variables, DUT, or POU header.
- It is possible to change only programs and functions. It is not possible to change function blocks.
- It is possible to add a program, but not possible to delete a program.
- The process requires that the PLC configuration has not been changed.

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Online " -> "Online mode." Alternatively, click . Switch to Offline mode.**
2. **Change the program.**
3. **Proceed from the Menu Bar to "Online " -> "Online mode." Alternatively, click . Switch to Online mode.**


Online
Monitor
Debug
Extras
Window
Help
Online mode Shift+Esc
Online edit mode
Communication parameters...
Network parameters...
PLC type...
Security settings...
Download program code and PLC configuration
Download program code changes
Upload program code and PLC configuration...
Clear PLC...

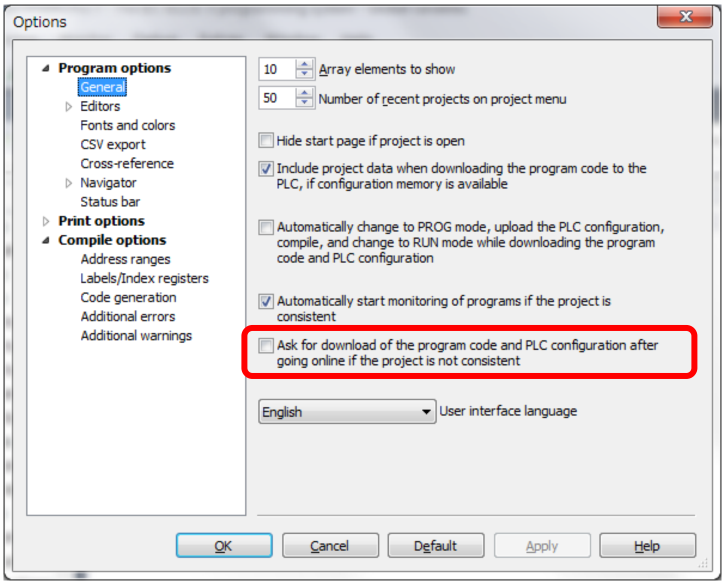
4. Proceed "Online " -> "Download program code changes," or click .


Online
Monitor
Debug
Extras
Window
Help
Online mode Shift+Esc
Online edit mode
Communication parameters...
Network parameters...
PLC type...
Security settings...
Download program code and PLC configuration
Download program code changes
Upload program code and PLC configuration...
Clear PLC...
Verify program code and PLC configuration
Memory transfer services...

The program is compiled, and downloaded into PLC. A confirmation message is displayed to indicate that the change has been completed normally.

◆ **NOTES**

- **The process above requires that "Ask for download of the program code and PLC configuration after going online if the project is not consistent" is unchecked (proceed “Extras” -> "Options" -> Program options" -> "General").**


Options

- Program options
  - General
  - Editors
  - Fonts and colors
  - CSV export
  - Cross-reference
  - Navigator
  - Status bar
- Print options
- Compile options
  - Address ranges
  - Labels/Index registers
  - Code generation
  - Additional errors
  - Additional warnings

10 Array elements to show

50 Number of recent projects on project menu

☐ Hide start page if project is open

☑ Include project data when downloading the program code to the PLC, if configuration memory is available

☐ Automatically change to PROG mode, upload the PLC configuration, compile, and change to RUN mode while downloading the program code and PLC configuration

☑ Automatically start monitoring of programs if the project is consistent

☐ Ask for download of the program code and PLC configuration after going online if the project is not consistent

English User interface language

4.8 Upload program code and PLC configuration

In FPWIN Pro7, the following options for upload can be selected.

- Upload project information
- Upload program code and PLC configuration

■ Upload program information

Proceed from the Menu Bar to "Project" -> "New" -> "PLC," to upload the entire project from PLC to FPWIN Pro7.

■ Upload program code and PLC configuration

Proceed from the Menu Bar to "Online " -> "Upload program code and PLC configuration," to upload program code and PLC configuration from PLC.

◆ **NOTES**

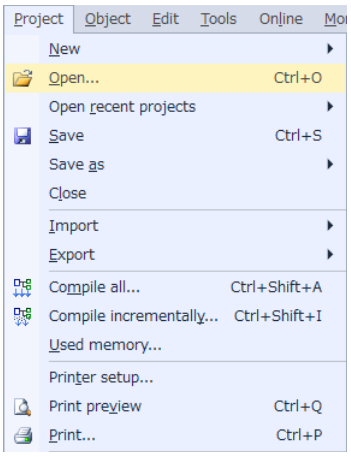
Use the correction of program code only when the source program cannot be used for special reasons. In this case, the program will be displayed on the screen in the mnemonic format, not as the information of IL, LD and FBD of FPWIN Pro7. Once it is uploaded from PLC, all documentation information of the program (such as variable names) will be lost.

If you are familiar with the basic set of instructions of program code, it is possible to correct an uploaded program. Naturally, the process is more difficult for a complex program than for a small program.

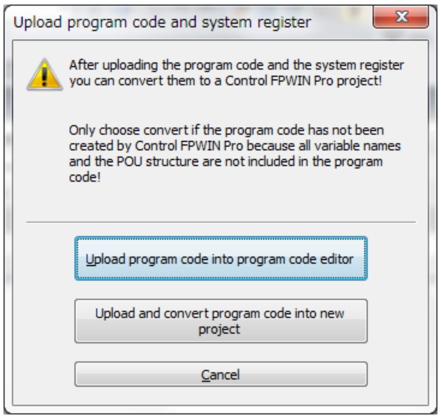
Before changing the program, it is recommended to back up the program by proceeding from the Menu Bar to "Object" -> "Export program code."s

◆ **PROCEDURE**

1. Edit a project, or open a project.



2. **Proceed from the Menu Bar to "Online " -> "Online mode." Alternatively, click .**
3. **Proceed from the Menu Bar to "Online " -> "Upload program code and PLC configuration." Alternatively, click .**
4. **In the "Upload program code and system register" window, select one of the following.**
   - Upload program code into program code editor
   - Upload and convert program code into new project


Upload program code and system register

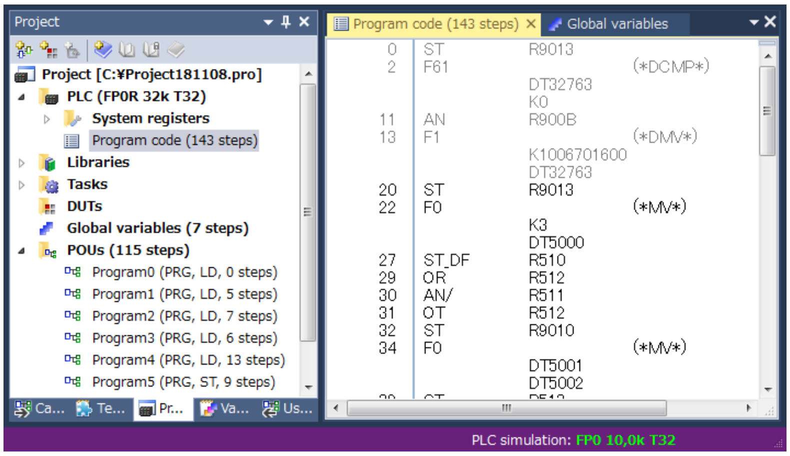
After uploading the program code and the system register you can convert them to a Control FPWIN Pro project!

Only choose convert if the program code has not been created by Control FPWIN Pro because all variable names and the POU structure are not included in the program code!

Upload program code into program code editor
Upload and convert program code into new project
Cancel

5. **Under "PLC" in the "Project" pane, open "Program code."**

   Confirm the content.



Project [C:¥Project181108.pro]
- PLC (FP0R 32k T32)
  - System registers
  - Program code (143 steps)
- Libraries
- Tasks
- DUTs
- Global variables (7 steps)
- POUs (115 steps)
  - Program0 (PRG, LD, 0 steps)
  - Program1 (PRG, LD, 5 steps)
  - Program2 (PRG, LD, 7 steps)
  - Program3 (PRG, LD, 6 steps)
  - Program4 (PRG, LD, 13 steps)
  - Program5 (PRG, ST, 9 steps)

```
 0   ST      R9013
 2   F61                     (*DCMP*)
             DT32763
             K0
11   AN      R900B
13   F1                      (*DMV*)
             K1006701600
             DT32763
20   ST      R9013
22   F0                      (*MV*)
             K3
             DT5000
27   ST_DF   R510
29   OR      R512
30   AN/     R511
31   OT      R512
32   ST      R9010
34   F0                      (*MV*)
             DT5001
             DT5002
```

PLC simulation: FP0 10,0k T32

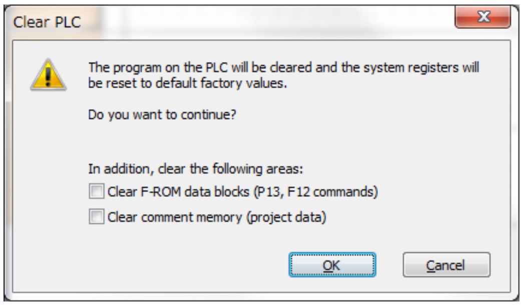
After checking and compiling the changed program, proceed from the Menu Bar to "Object" -> "Download program code and PLC configuration," and download it to PLC.

◆ **NOTES**

- **The process above requires that "When switched to online, urge download if program code or PLC configuration does not match" is unchecked (proceed "Extras" -> "Options" -> Program options" -> "General").**

4.9 Clearance of PLC

Proceed from the Menu Bar to "Online " -> "Clear PLC," to delete the current program of PLC, and return the system register of PLC to the factory setting.


Clear PLC

The program on the PLC will be cleared and the system registers will be reset to default factory values.

Do you want to continue?

In addition, clear the following areas:

☐ Clear F-ROM data blocks (P13, F12 commands)

☐ Clear comment memory (project data)

In the applicable PLC models, two options can be selected.

Option	Description
Clear F-ROM data blocks (P13, F12 commands)	F-ROM extended data area is cleared.
Clear comment memory (project data)	The project that is stored in the comment memory of PLC is cleared.

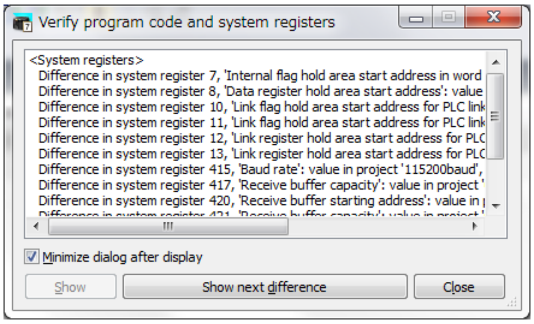
◆ **NOTES**

- **It is not required to delete the old program before writing a new program into PLC.**
- **The values that are set in PLC as factory setting differ from the standard setting of system register on FPWIN Pro7.**

The program and system register setting values of PLC are overwritten by the program and system register setting values of FPWIN Pro7.

4.10 Matching of program code and PLC configuration

Proceed from the Menu Bar to "Online " -> "Matching of program code and PLC configuration," to match "program code and PLC configuration" between PLC and the computer. The compiled data is uploaded from PLC to the computer for matching. The results of matching are displayed on the screen.



```
<System registers>
  Difference in system register 7, 'Internal flag hold area start address in word
  Difference in system register 8, 'Data register hold area start address': value
  Difference in system register 10, 'Link flag hold area start address for PLC link
  Difference in system register 11, 'Link flag hold area start address for PLC link
  Difference in system register 12, 'Link register hold area start address for PLC
  Difference in system register 13, 'Link register hold area start address for PLC
  Difference in system register 415, 'Baud rate': value in project '115200baud',
  Difference in system register 417, 'Receive buffer capacity': value in project '
  Difference in system register 420, 'Receive buffer starting address': value in
```

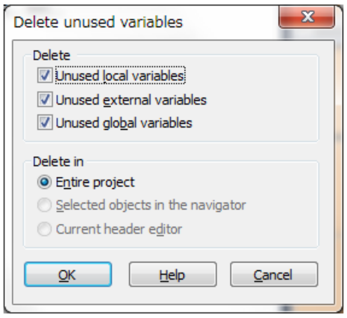
Verify program code and system registers
Minimize dialog after display
Show
Show next difference
Close

5

Menu Bar: Extended Functions

5.1 Delete Variables Not Used

Proceed from the Menu Bar to “Extras” -> "Delete unused variables," to delete, from the POU header or the list of global variables, variables that are not used in the POU body. Note that only variables that are not used in the POU body can be deleted from the POU header or the list of global variables.


Delete unused variables

Delete
- [✓] Unused local variables
- [✓] Unused external variables
- [✓] Unused global variables

Delete in
- [✓ Entire project]
- Selected objects in the navigator
- Current header editor

In this operation, the following options can be selected.

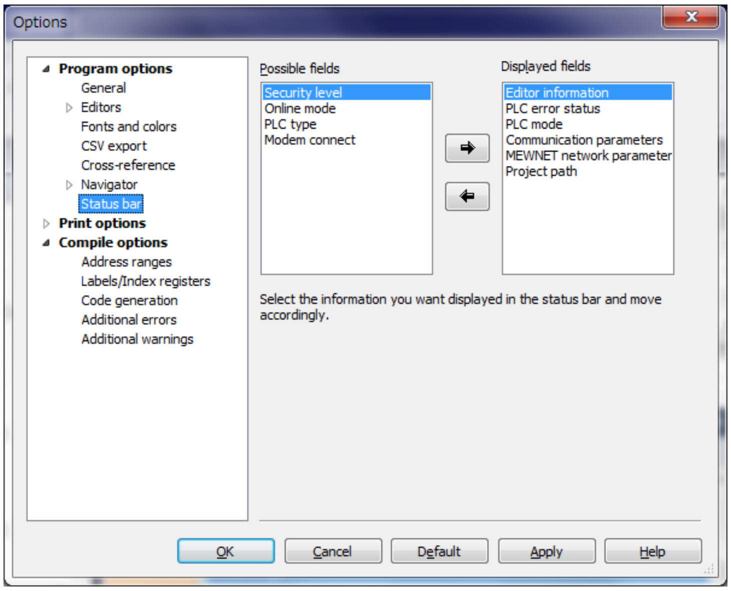
Variables to be deleted	Description
Unused local variables	All local variables that are not used in the POU body are deleted from the corresponding POU header.
Unused external variables	All external variables "VAR_EXTERNAL" that are not used in the POU body are deleted from the corresponding POU header.
Unused global variables	All global variables that are not used in any POU body are deleted from the list of global variables.

◆ **NOTES**

POUs in the user library are excluded from this operation.

5.2 Option

By proceeding from the Menu Bar to "Extras" -> "Option," to perform option setting.



Possible fields

Security level
Online mode
PLC type
Modem connect

Displayed fields

Editor information
PLC error status
PLC mode
Communication parameters
MEWNET network parameter
Project path

Select the information you want displayed in the status bar and move accordingly.

5.2.1 Program options

The following settings can be performed.
For more details concerning each setting item, refer to the Online Help.

Item	Description
General	Select options for general program behavior. The display language for Tools can be changed under "Select language." The change takes effect upon reboot.
Editors	Settings can be performed for Declaration editor, Programming Editor, and other editors.
Fonts and colors	Colors, fonts, and zooming can be adjusted in the font and color menu.
CSV export	The separator for the CSV format can be selected.
Cross-reference	Settings can be performed for declaration/reference file (*.sct). These files are required for generating a cross reference list cross reference list in the checking or compiling process. The timing for generating a cross reference list can also be set. Proceed from the Menu Bar to "Project" -> "Open cross reference," to display the cross reference list, and confirm the parameters and interdependence of variables, and the positions of declaration/use.
Navigator	Select the display of detailed information, and objects to be displayed on the project tab and the task configuration tab.
Status Bar	The Status Bar is at the bottom of the FPWIN Pro7 window. It is possible to display the PLC model, communication setting, time, and other information in the Status Bar. Items to be displayed in the Status Bar can be specified under "Status Bar" in the Options dialog box.

5.2.2 Printing options

In the "Current printing style" list box, selection can be made between "Common printing style" or "Project-specific printing style."

Common printing style

When a new project is opened in FPWIN Pro7, "Current printing style" is set as "Common printing style." By default, the selection is set to "Common printing style." The setting of "Common printing style" can be changed, and is applied when the next new project is created. The setting of "Common printing style" that is changed in a project is applied as "Common printing style" that is used in all other projects.

Project-specific printing style

When "Project-specific printing style" is selected, project-specific setting can be performed. This setting is not saved as setting for new projects. To perform printing setting specific to the current project, use "Project-specific printing style." When "Project-specific printing style" is changed, the changed setting is not applied to other projects.

◆ **NOTES**

When the [Default value] button is pressed, the printing setting is returned to the default value.

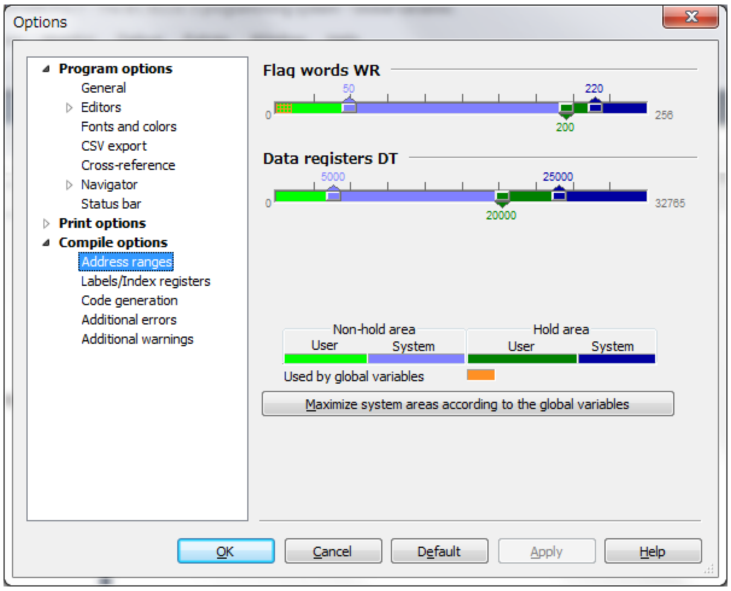
For more details, refer to the online Help.

5.2.3 Compile options

■ Address range

In the address range setting, memory areas are assigned to individual variables to which no physical address is directly assigned.

Proceed from the Menu Bar to "Extras" -> "Options" -> "Compile options" -> "Address range," to assign hold areas and non-hold areas, as well as the area for system (compiler) and the area for user. The range of memory area can be adjusted by moving the sliding button or by double-clicking the sliding button and entering values.



Memory areas that can be set are as follows.

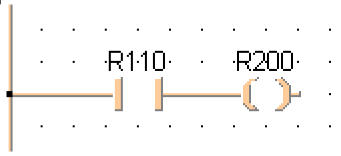
- Internal relay (WR)
- Data register (DT)
- File register (FL) (may not be displayed depending on the PLC model that is used)

By using [Maximize system areas according to the global variables], the maximum address area can be set automatically for the system (compiler). The user area (area that is assigned to variables that are entered by the user) is limited to the area that is assigned by global variables that have explicit addresses.

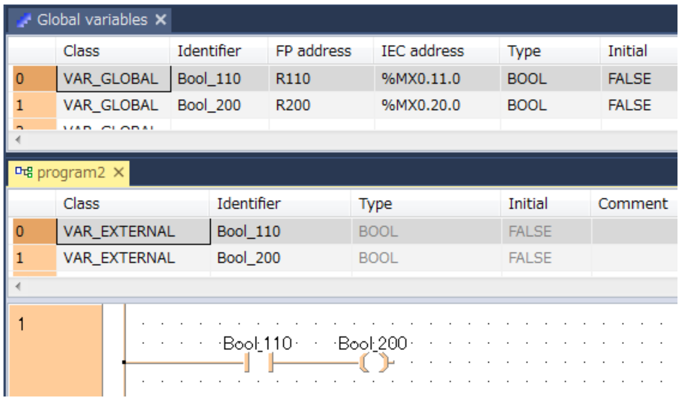
◆ **NOTES**

When [Maximize system areas according to the global variables] is selected, do not use an explicit address in the body of the editor. It will not be recognized.

For example, if R110 and R200 are directly used as variable names in the body, they are reflected in WR11 and WR20, but a warning message is displayed in the compile process.



By assigning explicit addresses to global variables as follows, WR11 and WR20 can be recognized.



	Class	Identifier	FP address	IEC address	Type	Initial
0	VAR_GLOBAL	Bool_110	R110	%MX0.11.0	BOOL	FALSE
1	VAR_GLOBAL	Bool_200	R200	%MX0.20.0	BOOL	FALSE

	Class	Identifier	Type	Initial	Comment
0	VAR_EXTERNAL	Bool_110	BOOL	FALSE	
1	VAR_EXTERNAL	Bool_200	BOOL	FALSE	

If power supply is interrupted, or when the mode is switched from RUN to PROG. mode, the values in the non-hold area are deleted, but the values in the hold area are retained. The values in the hold area are not initialized until the program is downloaded to PLC.

Proceed from the Menu Bar to "Extras" -> "Options" -> "Compile options" -> "Code generation" and turn on the check box for "Do not initialize hold-type variables in the user area," to prevent the initialization of variables to which the user has directly assigned addresses.

◆ **NOTES**

- **The addresses for the hold area can be set by proceeding from "PLC" in the "Project" pane -> "System register" -> "hold/non-hold."**
- **When compile options are changed, the entire project needs to be re-compiled.**
- **For a variable for which no address is entered in the list of global variables, a memory area is assigned automatically by the compiler.**

- **Certain memory areas cannot be used depending on the PLC model (such areas are displayed in gray).**
- **Do not directly assign an address unless necessary. Use this when it is necessary to access the input/output (X, Y) of PLC, or a specific memory area.**
- **The compiler assigns an address automatically. This prevents an error caused by double assignment, and enables the automatic refreshment of addresses in the case of change to the PLC model.**

The following table indicates memory areas that are assigned by the compiler to variables, based on the class and data type.

Class	Data type	Memory area
VAR, VAR_GLOBAL	BOOL	Relay word WR non-hold area
VAR_RETAIN, VAR_GLOBAL_RETAIN	BOOL	Relay word WR hold area
VAR, VAR_GLOBAL	INT、DINT、WORDDWORD、TIME、REAL、STRING	Data register DT, non-hold type File register FL, non-hold type
VAR_RETAIN, VAR_GLOBAL_RETAIN	INT、DINT、WORDDWORD、TIME、REAL、STRING	Data register DT, hold type File register FL, hold type

The compiler also automatically generates labels required for loop.

The number of labels to be reserved for the system (compiler) can be specified by proceeding from the Menu Bar to "Extras" -> "Options" -> "Compile options" -> "Labels / index registers."

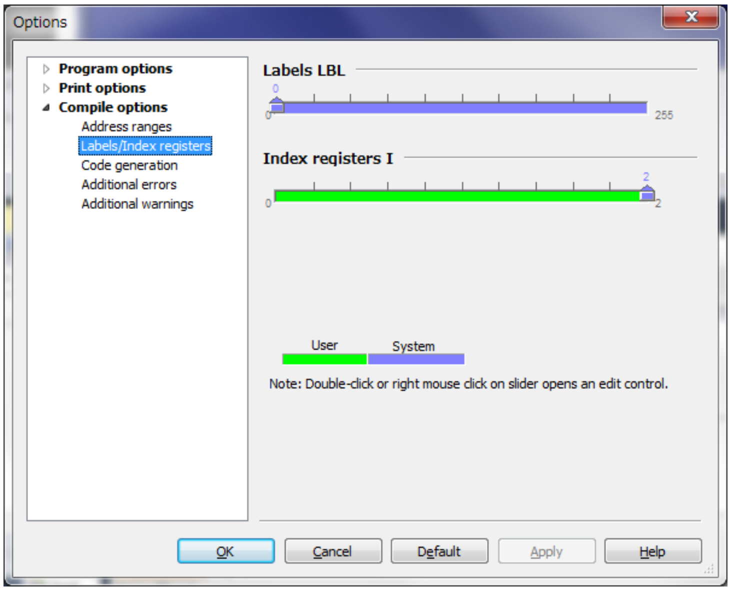
◆ **NOTES**

When the [Default value] button is pressed, the compiler setting can be returned to the default value.

■ Labels / index registers

Labels and index registers are divided by the compiler into user area and system area.

The number of labels (LBL) or index registers (I) can be adjusted by moving the sliding button. Values can also be directly entered by double-clicking the sliding button, or by right-clicking.



Labels LBL: 0 (0 – 255)

Index registers I: 2 (0 – 2)

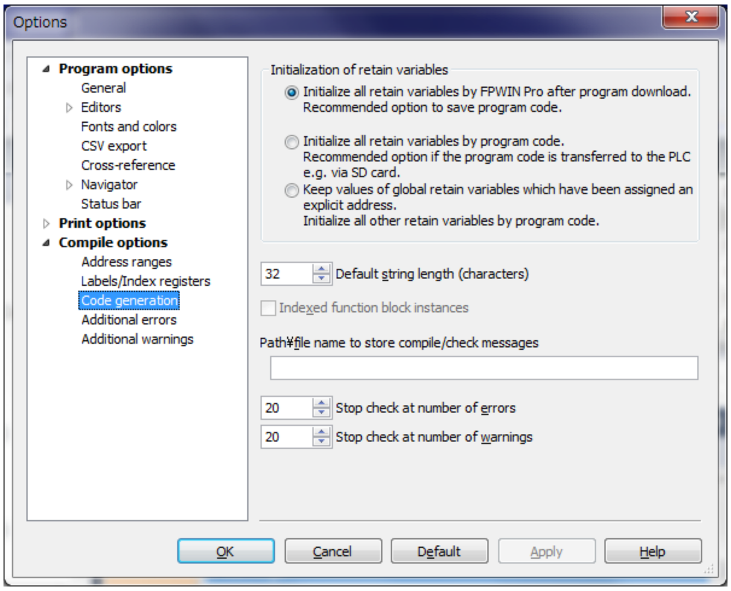
🟩 User 🟦 System

Note: Double-click or right mouse click on slider opens an edit control.

◆ **NOTES**

When the [Default value] button is pressed, the compiler setting can be returned to the default value.

■ Code generation



Options

- Program options
  - General
  - Editors
  - Fonts and colors
  - CSV export
  - Cross-reference
  - Navigator
  - Status bar
- Print options
- Compile options
  - Address ranges
  - Labels/Index registers
  - Code generation
  - Additional errors
  - Additional warnings

Initialization of retain variables

- [✓] Initialize all retain variables by FPWIN Pro after program download. Recommended option to save program code.
- Initialize all retain variables by program code. Recommended option if the program code is transferred to the PLC e.g. via SD card.
- Keep values of global retain variables which have been assigned an explicit address. Initialize all other retain variables by program code.

32 Default string length (characters)

☐ Indexed function block instances

Path¥file name to store compile/check messages

20 Stop check at number of errors

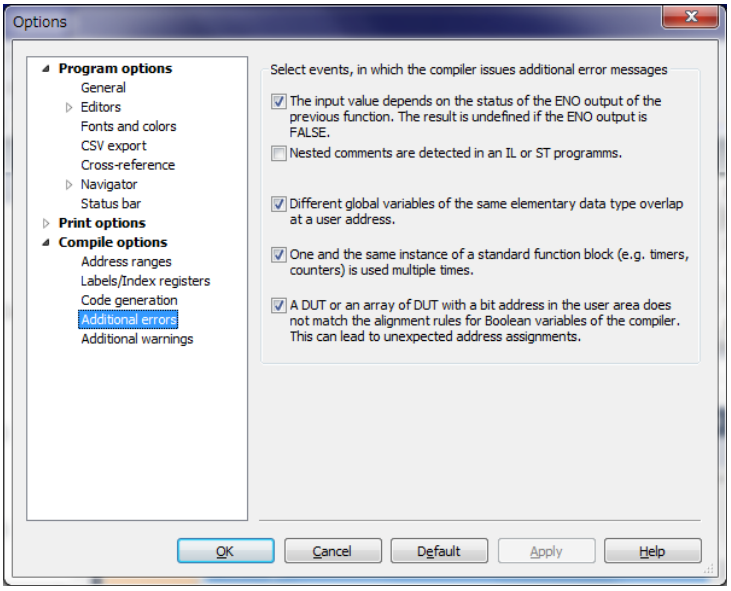
20 Stop check at number of warnings

Code generation	Description
Initialization of retain variables	When this option is disabled, the hold area of the user area is not initialized. For all global variables in "VAR_GLOBAL_RETAIN," to which the user has assigned FP address / IEC addresses, the values are retained after a new program is downloaded. When this option is enabled, hold-type variables in the user area are initialized every time a changed program is downloaded.
Default string length (characters)	Enter the default number of characters for STRING (max. 255 characters).
Indexed function block instances	If this option is enabled, it becomes possible to specify how the compiler generates program code for a function block created by the user. This option can be specified only in models FP2/FP2SH.
Path\file name to store compile/check messages	Enter the folder name//file name in which messages to be displayed in the "Check/compile result" dialog box. (Example: C:\folder name\file name.txt) Proceed from the Menu Bar to "Object" -> "Check," or click . Alternatively, proceed from the Menu Bar to "Project" -> "Compile all/Compile incrementally." Every time the process is executed, the content of this file is automatically overwritten.
Stop check at number of errors/warnings	Enter the number of errors or the number of warnings at which the compile process should be canceled.

◆ **NOTES**

When the [Default value] button is pressed, the compiler setting can be returned to the default value.

■ Additional errors



Select events, in which the compiler issues additional error messages

- [✓] The input value depends on the status of the ENO output of the previous function. The result is undefined if the ENO output is FALSE.
- [ ] Nested comments are detected in an IL or ST programms.
- [✓] Different global variables of the same elementary data type overlap at a user address.
- [✓] One and the same instance of a standard function block (e.g. timers, counters) is used multiple times.
- [✓] A DUT or an array of DUT with a bit address in the user area does not match the alignment rules for Boolean variables of the compiler. This can lead to unexpected address assignments.

Select events, in which the compiler issues additional error messages

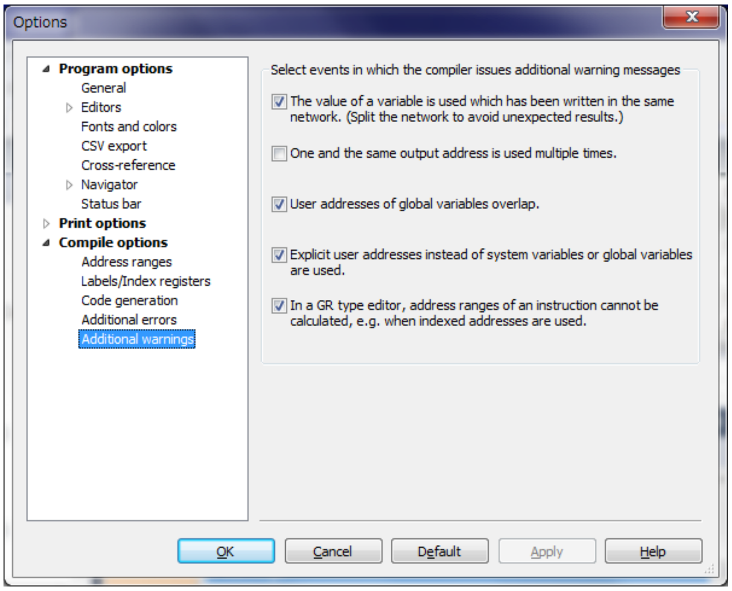
Additional error	Description
The input value depends on the status of the ENO output of the previous function. The result is undefined if the ENO output is FALSE.	For information concerning this check box, refer to Section "9.2.2 EN/ENO connection of FUN or FB."
Nested comments are detected in an IL or ST programs.	If this check box is on, no comment that has a nesting structure is permitted. Just as intended in IEC-1131, the comment is considered to end at the position of the first comment end symbol "*)."
Different global variables of the same elementary data type overlap at a user address.	If this check box is on, an error message is displayed when user assigns the same address to more than one global variables of the same data type. Example: If R0 is assigned to two global variables of the BOOL type, an error message is displayed. If DT0 is assigned to one global variable of the INT type and to one global variable of the WORD type, no error message is displayed.
One and the same instance of a standard function block (e.g. timers, counters) is used multiple times.	If the same instance of the standard function bock (timer, counter, etc.) is used multiple times, an error or warning message is outputted.

Additional error	Description
A DUT or an array of DUT with a bit address in the user area does not match the alignment rules for Boolean variables of the compiler. This can lead to unexpected address assignments.	If DUT is used as DUT array, but the number of the BOOL-type variables of DUT does not equal a multiple of 1, 2, 4, 8 or 16, an error message is displayed. If DUT is used as DUT array, but the number of the BOOL-type variables of DUT does not equal a multiple of 1, 2, 4, 8, or 16, disable this option. If the number of the BOOL-type variables of DUT does not equal a multiple of 1, 2, 4, 8, or 16, the BOOL memory area of each array element is filled by the compiler through to the next assignment (i.e. a multiple of 4, 8, or 16). It is recommended to enable this option, because the BOOL memory area may be overwritten when it has become full. The BOOL-type memory that has become full should not be used, because it may be overwritten by the compiler. (It is recommended to enable the warning option "More than one global variables have the same user address")

◆ **NOTES**

When the [Default] button is pressed, the compiler setting can be returned to the default value.

■ Additional warnings



Select events in which the compiler issues additional warning messages

- [✓] The value of a variable is used which has been written in the same network. (Split the network to avoid unexpected results.)
- [ ] One and the same output address is used multiple times.
- [✓] User addresses of global variables overlap.
- [✓] Explicit user addresses instead of system variables or global variables are used.
- [✓] In a GR type editor, address ranges of an instruction cannot be calculated, e.g. when indexed addresses are used.

Select events in which the compiler issues additional warning messages

Additional warning	Description
The value of a variable is used which has been written in the same network. (Split the network to avoid unexpected results.)	This wiring is displayed if writing and reading is performed for a single variable in the same network Such network is handled by the compiler. The value of the variable before the network is processed is applied to all positions in the network preceding the reading of the variable.
One and the same output address is used multiple times.	A warning is issued if output in the program is doubly defined.
Use addresses of global variables overlap.	This error is displayed if the user designated the same address to more than one global variables.
Explicit use addresses instead of system variables or global variables are used.	If this option is enabled, a warning is outputted when an explicit user address is used. It is recommended to use a global variable or system variable.
In a GR type editor, address ranges of an instruction cannot be calculated, e.g. when indexed addresses are used.	If this option is enabled, a warning is outputted into the GR program to be imported when the address area used in the GR program cannot be processed.

◆ **NOTES**

When the [Default value] button is pressed, the compiler setting can be returned to the default value.

6

Navigator Pane

6.1 Navigation of Project

In FPWIN Pro7, control tasks are stored as a project. A project consists of multiple objects, which are displayed in a list in Navigator.

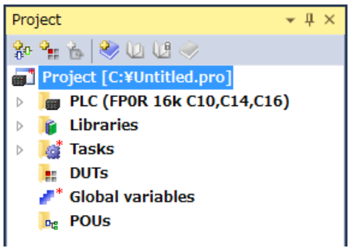
A project consists of the following.

- Program Organization Unit (POU)
- Global variable
- Task

Optional objects of project are indicated below.

- Data Unit Type (DUT)
- Library

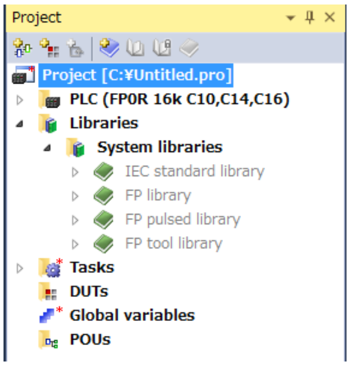
After setting a project, the following objects are displayed in the "Project" pane.



Objects displayed in Navigator

In Project Navigator, pools (library, task, POU pool, etc.) and objects (individual POUs, etc.) are displayed.

For a layered object (library pool, etc.), the ◢ symbol is displayed.
The ▷ symbol indicates that the lower layers are already displayed.


Project
Project [C:¥Untitled.pro]
▷ PLC (FP0R 16k C10,C14,C16)
◢ Libraries
◢ System libraries
▷ IEC standard library
▷ FP library
▷ FP pulsed library
▷ FP tool library
▷ Tasks
DUTs
Global variables
POUs

Open Library Pool

Components consist of user-defined objects. These include the following.

- Program code
- Machine program
- Global variables (list of global variables)
- Header and body of Program Organization Unit (POU)

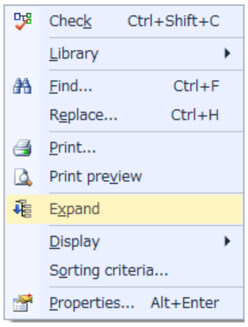
Within the program window, the windows of program code, global variables, POU header/body, etc. can be opened. It becomes possible to edit the program and other content. To expand a pool (POU pool, etc.), double-click the pool name in Project Navigator, or select "Expand" in the context menu that is displayed by right-clicking.

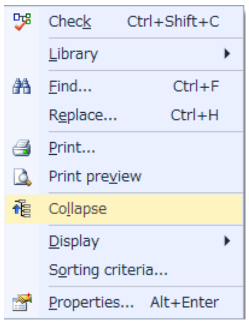
◆ **PROCEDURE**

1. **Double-click on an object that is closed. Alternatively, click ◢ or ▷. Alternatively, click an object that is closed in Project Navigator.**

   The selected object is highlighted.

2. **In the context menu that is displayed by right-clicking, select "Expand / Collapse."**





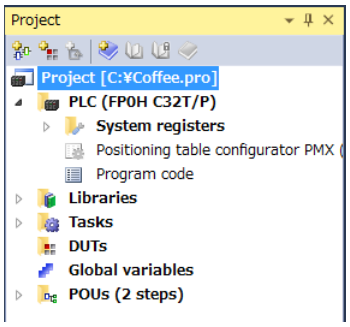
For POU, it is also possible to select "Open" in the context menu that is displayed by right-clicking.

■ Navigator Pane

The Navigator of FPWIN Pro7 displays a complex project in an easy-to-understand way. A project can be opened by double-clicking.

The Navigator has three panes.

- Project
- Task configuration
- POU reference

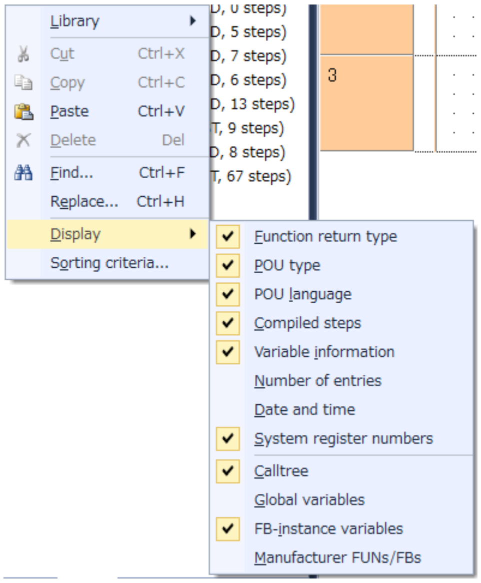


Navigator Pane

The "Project" pane is a standard pane for creating, editing, and the display of a project.

Proceed from the Menu Bar to "Window" -> "Pane" -> "Project," or click , to activate the "Project" pane.

To display detailed information in the "Project" pane, select an item under "Display" in the context menu that is displayed by right-clicking, or select "Display detailed information" in the Navigator option.



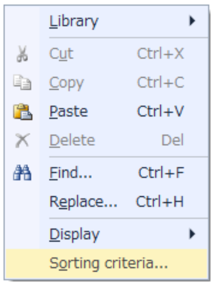
■ Display declaration of variables or reference to objects

◆ **PROCEDURE**

1. **In Navigator or in Programming Editor, select a variable or POU.**
2. **In the context menu that is displayed by right-clicking, select "Jump to variable declaration" or "Jump to reference declaration."**

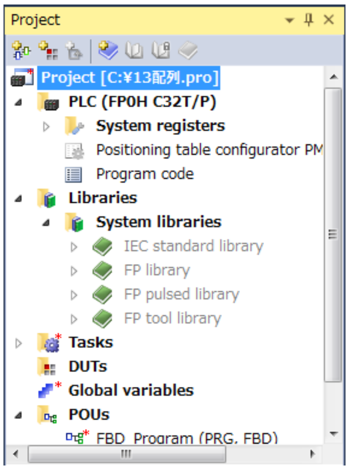
   The declaration of object and the reference declaration are highlighted.
3. **To jump to the next reference, press the <F3> key.**

To change the order of objects displayed in Navigator, select "Sorting criteria" in the context menu. The sorting of POUs is performed from top to bottom.



6.2 "Project" Pane

Proceed from the Menu Bar to "Window" -> "Pane" -> "Project," or select , to activate the "Project" pane. The "Project" pane is a pane for creating, editing, and the display of a project.


Project

Project [C:¥13配列.pro]
PLC (FP0H C32T/P)
System registers
Positioning table configurator PM
Program code
Libraries
System libraries
IEC standard library
FP library
FP pulsed library
FP tool library
Tasks
DUTs
Global variables
POUs
FBD_Program (PRG, FBD)

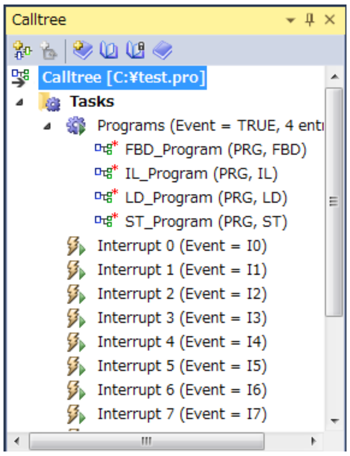
Project

◆ **NOTES**

- **To display task configuration in the "Project" pane, select "Display" -> "Task configuration" "Calltree" in the context menu that is displayed by right-clicking, or select "Task configuration tab" in the Navigator option.**
- **To display detailed information in the"Project" pane, select "Display" in the context menu that is displayed by right-clicking, or select "Navigator" under "5.2.1Program options."**

6.3 "Calltree" Pane

Proceed from the Menu Bar to "Window" -> "Pane" -> "Calltree." Alternatively, select to activate the "Calltree." pane. The calltree pane is displayed by POU that has been assigned to Task, or by POU that is not assigned. POU and global variables that are used for it are displayed in a tree structure. The task configuration indicates the program in which functions and function blocks are used, and how they are interrelated.



Calltree

Calltree [C:¥test.pro]
- Tasks
  - Programs (Event = TRUE, 4 entı
    - FBD_Program (PRG, FBD)
    - IL_Program (PRG, IL)
    - LD_Program (PRG, LD)
    - ST_Program (PRG, ST)
  - Interrupt 0 (Event = I0)
  - Interrupt 1 (Event = I1)
  - Interrupt 2 (Event = I2)
  - Interrupt 3 (Event = I3)
  - Interrupt 4 (Event = I4)
  - Interrupt 5 (Event = I5)
  - Interrupt 6 (Event = I6)
  - Interrupt 7 (Event = I7)

Task configuration

NOTES

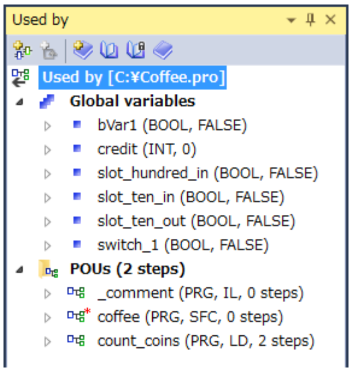
- **To display task configuration in the "Project" pane, select "Display" -> "Calltree" in the context menu that is displayed by right-clicking, or select "Calltree tab" in the Navigator option.**
- **To display detailed information in the"Project" pane, select an item under "Display" in the context menu that is displayed by right-clicking, or select "Navigator" under "5.2.1Program options."**

6.4 "Used by" Pane

Proceed from the Menu Bar to "Window" -> "Pane" -> "Used by." Alternatively, select to activate the " Used by " pane. The Used by tab has two routes.

• POU pool that has individual POUs in lower layers

• Global variable pool that has individual global variables in lower layers


Used by

Used by [C:¥Coffee.pro]

Global variables
- bVar1 (BOOL, FALSE)
- credit (INT, 0)
- slot_hundred_in (BOOL, FALSE)
- slot_ten_in (BOOL, FALSE)
- slot_ten_out (BOOL, FALSE)
- switch_1 (BOOL, FALSE)

POUs (2 steps)
- _comment (PRG, IL, 0 steps)
- coffee (PRG, SFC, 0 steps)
- count_coins (PRG, LD, 2 steps)

POU reference tab

POUs that use global variables, POUs the call POUs (such as function blocks), etc. are displayed in a tree structure.

6.5 PLC

Under the "PLC" item in Project Navigator, settings can be performed for system register, I/O assignment, data logging, etc.

6.5.1 System register setting

System register is a memory area for defining timer, counter, hold area / non-hold area of data register, etc.

In system register, it is also possible to define behavior in the case of error, and parameters for PLC interface.

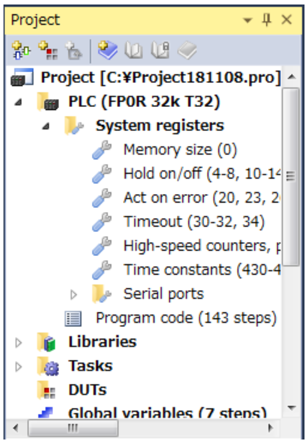
◆ **NOTES**

- **The memory size varies by PLC model. The sum of memory areas used for system register and user program may not exceed the total memory size of PLC.**
- **Two words from the maximum address of data register (four words in the case of PLC that has the second program area) may not be used, because they are reserved for compiler by FPWIN Pro7.**

◆ **PROCEDURE**

1. **Double-click "PLC."**
2. **Double-click "System register."**

   All system registers are displayed in a list. The number in the parentheses that are displayed indicate the number of the system register.



```
Project
Project [C:¥Project181108.pro]
  PLC (FP0R 32k T32)
    System registers
      Memory size (0)
      Hold on/off (4-8, 10-14
      Act on error (20, 23, 2
      Timeout (30-32, 34)
      High-speed counters, p
      Time constants (430-4
    Serial ports
    Program code (143 steps)
  Libraries
  Tasks
  DUTs
  Global variables (7 steps)
```

3. **Double-click a system register.**
4. **Enter setting.**

6.5.2 Program code

When a program code is opened in the project pane, the program is displayed in the mnemonic format. The program code is registered when you "Download program code and PLC configuration" or "Upload program code and PLC configuration."

◆ **NOTES**

Use the correction of program code only when the source program cannot be used for special reasons. In this case, the program will be displayed on the screen in the mnemonic format, not as the information of IL, LD and FBD of FPWIN Pro7. Once it is uploaded from PLC, all documentation information of the program (such as variable names) will be lost.

If you are familiar with the basic set of instructions of program code, it is possible to correct an uploaded program. Naturally, the process is more difficult for a complex program than for a small program.

Before changing the program, it is recommended to back up the program by proceeding from the Menu Bar to "Object" → “Export program code”

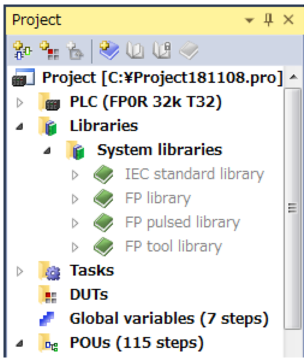
6.6 Library

Double-click "Library" in Project Navigator to display all libraries that can be used in a tree structure. By using functions and function blocks that are stored in the library, certain steps for programming can be saved.

For more details concerning the use of libraries, creation of a user library, etc., refer to the Online Help.

6.6.1 Types of System Library

In FPWIN Pro7, four types of system library are provided as a standard.



Item	Description
IEC standard library	This consists of an aggregation of standard functions and function blocks defined by IEC 61131-3. This includes, for example, data-type conversion function, arithmetic function, bit shift function, comparison function, bistable function block, timer, counter, etc. Also refer to "Advantages of IEC instructions." Examples: ADD, CONCAT, MOVE
FP library	This contains all FP instruction sets, which include FP instructions and F instructions. Although FP instructions were developed for FP7, they are also supported by PLC of other FP series. Most F instructions provide very similar functions, but they operate in different data types. Overload FP instruction integrates these instructions, so that the user can easily select appropriate instructions. Also refer to "Advantages of FP instructions." Examples: FP instructions: FP_MOVE_BLOCK, FP_BAND F Instructions : F0_MV, F159_MTRN, F171_PulseOutput_Home
FP Pulsed Library	This contains all sets of P instructions. While P instructions have the same functions as F Instructions, they are executed only at the leading of a signal. P instructions can be used only for PLC types FP2/2SH. When P instructions are required, select [P instructions] in the "Instruction" pane. Examples: P13_EPWT
FP tool library	This contains address acquisition functions, information functions, and copy functions, which facilitate programming. Tool instructions can be used in all PLC models. For the advantages of Tool instructions over F instructions, refer to "Overview of high-speed counter and pulse output instructions." Examples: Adr_Of_Var I/O, Unit_AnalogInOut_FP0_A21, HscControl_CountingDisable, Pulse_TargetValueMatch_Reset

◆ **NOTES**

- **Functions and function blocks that can be used vary by PLC model.**
- **It is not possible to display or correct POU body for IEC standard functions and function blocks.**

6.6.2 User library

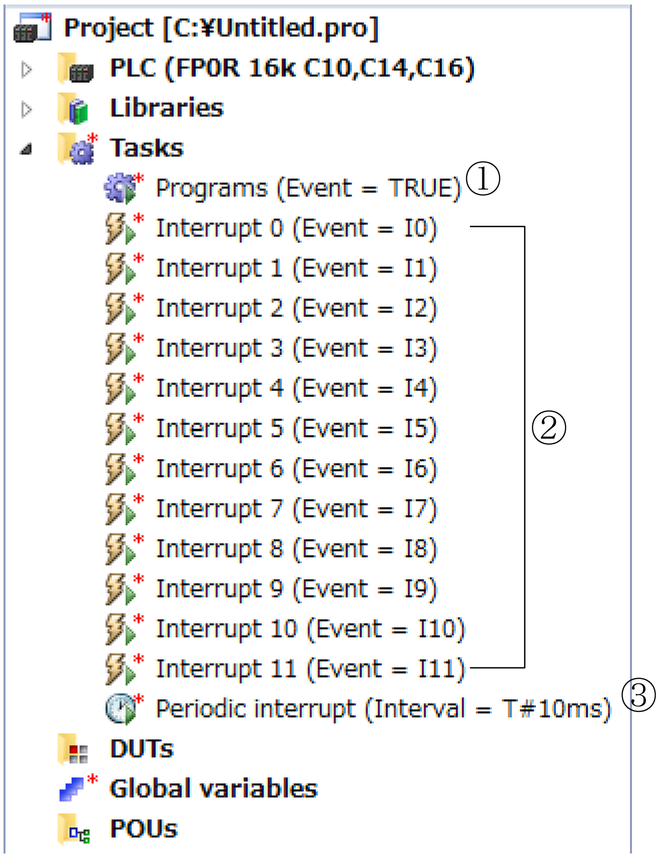
In FPWIN Pro7, up to 50 user libraries can be created. Existing objects and new objects can be registered in each user library. By using user libraries, objects can be more easily retrieved and organized.

User libraries are saved in a hard disk, etc. It is also possible that other users access the same user library via database, etc. In this case, the users can refer to the same library without creating a new library. Therefore, when an object in the library is changed, other users can utilize it immediately.

User library has three statuses: "Open," "Change," and "Install." The current status is continuously displayed in Project Navigator. The usability of the library varies by status.

6.7 Task

Task is positioned in the first layer of the project. Task controls POUs of all program types. Task is registered in Task Pool of Navigator.


Project [C:¥Untitled.pro]
PLC (FP0R 16k C10,C14,C16)
Libraries
Tasks
Programs (Event = TRUE) ①
Interrupt 0 (Event = I0)
Interrupt 1 (Event = I1)
Interrupt 2 (Event = I2)
Interrupt 3 (Event = I3)
Interrupt 4 (Event = I4)
Interrupt 5 (Event = I5)
Interrupt 6 (Event = I6)
Interrupt 7 (Event = I7)
Interrupt 8 (Event = I8)
Interrupt 9 (Event = I9)
Interrupt 10 (Event = I10)
Interrupt 11 (Event = I11)
②
Periodic interrupt (Interval = T#10ms) ③
DUTs
Global variables
POUs

In FPWIN Pro7, three execution modes can be specified.

①	Cyclic (program)	The program is repeated after the entire process is completed. Example: Liquid is continuously pumped out of the container, and its temperature is measured after that.
②	Event driven (interrupt)	If a pre-defined event occurs, the execution of the program is started. Example: If the temperature in the container declines below the minimum value, the heating program is called.
③	Time driven (scheduled interrupt)	The program is executed periodically. Example: The synchronization program is executed every 10 minutes.

◆ **NOTES**

A program-type POU needs to be compiled and assigned to Task. Otherwise, the program will not be executed.

One or more POU can be registered in a task. When more than one POU is registered, they are executed in the order of registration. The registered program is compiled, and subsequently converted into one program, and downloaded into PLC.

To include a program in Task into the scope of compile, select the program, and proceed from the Menu Bar to "Edit" -> "Include into / exclude from scope of compile," or use the context menu.

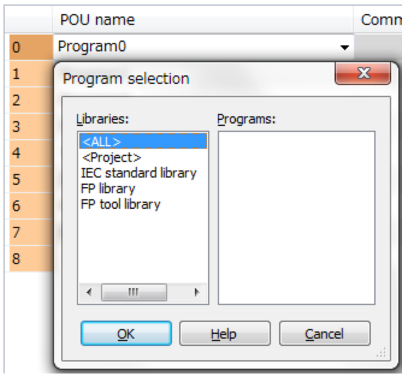
6.7.1 Assign program to Task

When a project is created, the program is automatically registered in Task. Alternatively, the program can be registered later via Task list.

In this procedure, for the cyclic operation of a program, it is registered in Program Task.

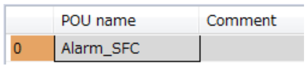
◆ **PROCEDURE**

1. **Open "Task" in the "Project" pane.**
2. **Open "Program."**
3. **Select ▾ under "POU name," and open the "Program selection" dialog box.**



Select a program under "Program."

4. **Press the [OK] button in the "Program selection" dialog box.**



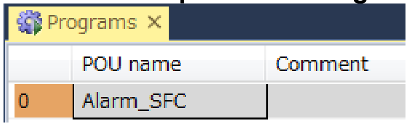
	POU name	Comment
0	Alarm_SFC	

One or more POU can be registered in a task. When more than one POU is registered, they are executed in the order of registration. The registered program is compiled, and subsequently converted into one program, and downloaded into PLC.

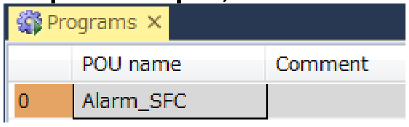
5. **Enter text into the "Comment" field when necessary.**

◆ **NOTES**

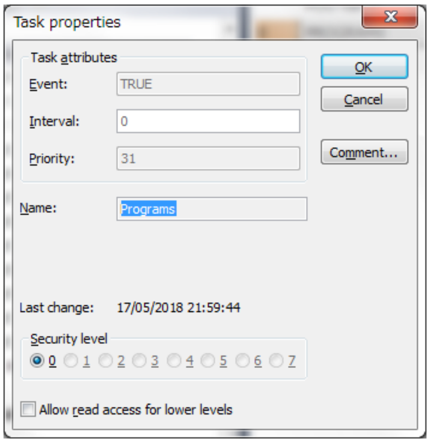
- **SFC program (SFC POU) must always be registered consecutively in Task Pool. It is not possible to register POUs of another format in between.**



- **To include a program in Task into the scope of compile, select the program, and proceed from the Menu Bar to "Edit" -> "Include into / exclude from scope of compile," or use the context menu.**



To confirm the properties of Task, proceed from the Menu Bar to "Object" -> "Properties," or use <Alt>+<Return>.


Task properties

Task attributes
Event: TRUE
Interval: 0
Priority: 31

Name: Programs

Last change: 17/05/2018 21:59:44

Security level
[✓ 0] 1 2 3 4 5 6 7

Allow read access for lower levels

The dialog box consists of the following elements.

Item	Description
Event	In this field, events that are registered in Task are displayed (TRUE if cyclic processing). The event number is linked to the number of the interrupt program.
Interval	This field displays interval time for the time-driven task (scheduled-interrupt task) to call POU. By entering the value of time, the interval can be changed. (Note)
Priority	In FPWIN Pro7, the priority of all tasks is fixed to 31, and may not be changed. Interrupt tasks are processed in the order of display in Task Pool. Therefore, Interrupt0 has higher priority over Interrupt1.

(Note): The interval time needs to be entered in the IEC format. Example: T#10s

6.8 Data Unit Type (DUT)

DUT stands for "Data Unit Type," which consists of variables that have multiple different data types. Define DUT at first, and subsequently use it in the list of global variables or in the POU header, just as standard data types (BOOL, INT, etc.).

◆ **NOTES**

- **DUT may not be used as an element of another DUT or DUT array.**
- **DUT is a variable that consists of different data types (BOOL, INT, WORD, etc.). These groups are used as operation tables.**

6.9 Global Variables

Global variables can be used throughout the project. They can be copied from the list of global variables to the POU header. Global variables can be declared extremely easily in the POU body (by using LD/FBD, ST/IL, or SFC). Alternatively, create the list of global variables using Declaration editor.

Do not directly assign an address unless necessary. Use this when it is necessary to access the input/output (X, Y) of PLC, or a specific memory area.

The compiler assigns an address automatically. This prevents an error caused by double assignment, and enables the automatic refreshment of addresses in the case of change to the PLC model.

The following variables need to be declared on the list of global variables.

- Variables that are assigned to the input/output of PLC (X0, Y0, etc.)
- Variables that need to be assigned to specific addresses (such as DT0), when data is to be exchanged with a programmable display, or for other purposes
- Variables that are referred to as external variables (VAR_EXTERNAL) by POU

Addresses can be registered by using the following format.

FP address	When the entry of FP address is completed, the corresponding IEC address is automatically displayed. Example: X0
IEC address	The respective symbols indicate I = Input, X = Bit, and 0.0 = First input contact of first unit. When the entry of IEC address is completed, the corresponding FP address is automatically displayed. Example: %IX0.0

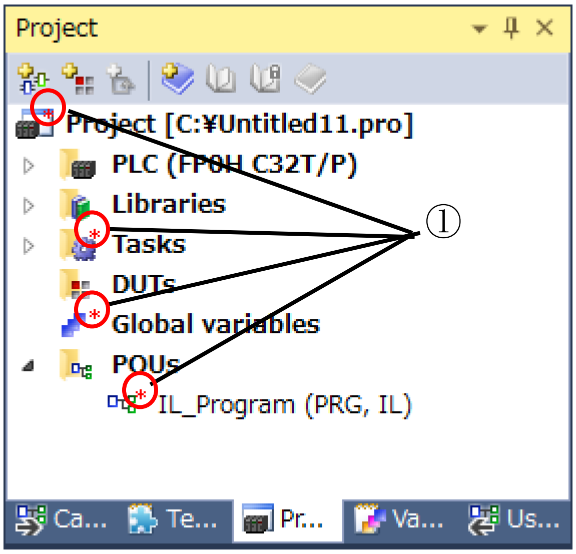
6.10 POU

A program is automatically stored in POU Pool of Project Navigator. More than one program can be registered. Each program consists of Header (variables) and Body (program code), which are written in a programming language.

6.10.1 Use Navigator

The elements of Navigator are called "objects." The context menu in Navigator contains all commands for processing objects.

In the Navigator, objects that contain changes that have yet to be compiled are indicated with a red asterisk (*).



Open POU Pool

①	Object that has yet to be compiled

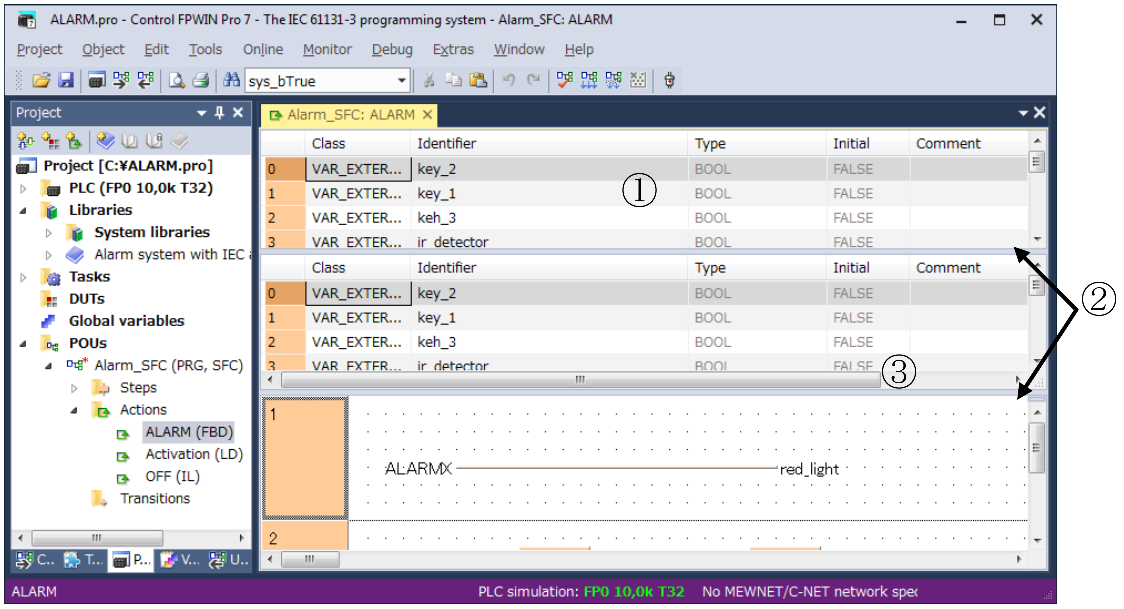
7
Programming Editor

7.1 Programming Window

The programming window consists of POU header and POU body.

■ Customization of work area

- To make changes in color, font, zoom ratio, comment indication, etc., or to restore the default setting, proceed from the Menu Bar to "Extras" -> "Options" -> "Program options" -> "Editor." Options to be used vary by the programming editor.
- To minimize scroll during the editing of different sections in POU, divide POU header or POU body using Window Splitter 2) (mouse pointer:). Double-click the splitter to return to the normal position.
- To lay out tabs horizontally or vertically, use the command in the "Window" menu.
- To switch between POU header and POU body, press <F6>.
- To adjust the height of the window, drag Bar 3) between POU header and POU body using the mouse (mouse pointer:).
- The small triangle in the POU header field indicates that there are multiple lines of comment, etc. To display a comment that has multiple lines, move the mouse pointer over the field, or click inside the field.



	Class	Identifier	Type	Initial	Comment
0	VAR_EXTER...	key_2	BOOL	FALSE	
1	VAR_EXTER...	key_1	BOOL	FALSE	
2	VAR_EXTER...	keh_3	BOOL	FALSE	
3	VAR_EXTER...	ir_detector	BOOL	FALSE	

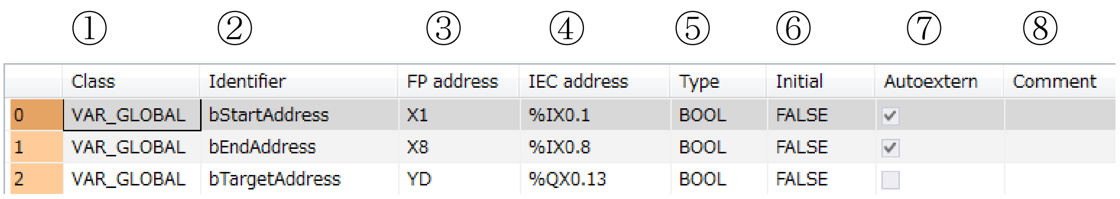
	Class	Identifier	Type	Initial	Comment
0	VAR_EXTER...	key_2	BOOL	FALSE	
1	VAR_EXTER...	key_1	BOOL	FALSE	
2	VAR_EXTER...	keh_3	BOOL	FALSE	
3	VAR_EXTER...	ir_detector	BOOL	FALSE	

Programming window

①	Multi-line field
②	Window splitter for making a copy of the view of POU header or POU body.
③	Window splitter for adjusting the height of the window

7.1.1 Description of fields

A project has the list of global variables. The list of global variables (GVL) is applied to all POUs in the project. Each field in the list of variables respectively indicates the following.



	Class	Identifier	FP address	IEC address	Type	Initial	Autoextern	Comment
0	VAR_GLOBAL	bStartAddress	X1	%IX0.1	BOOL	FALSE	✓	
1	VAR_GLOBAL	bEndAddress	X8	%IX0.8	BOOL	FALSE	✓	
2	VAR_GLOBAL	bTargetAddress	YD	%QX0.13	BOOL	FALSE	☐	

No.	Item name	Description
①	Class	Declares the class of the variable. Examples: VAR_GLOBAL,VAR_GLOBAL_CONSTANT
②	Identifier	The symbol name used in the program. - The name of a variable cannot be started with a number. - For the name of a variable that uses a special syntax, the data type can be determined in accordance with the prefix table.
③ ④	FP address IEC address	This is a physical address assigned to a variable. Do not directly assign an address unless necessary. Use this when it is necessary to access the input/output (X, Y) of PLC, or a specific memory area. The compiler assigns an address automatically. This prevents an error caused by double assignment, and enables the automatic refreshment of addresses in the case of change to the PLC model. Addresses can be registered by using the following format. - FP address (example: X0) When the entry of FP address is completed, the corresponding IEC address is automatically displayed. - IEC address (example: %IX0.0) The respective symbols indicate I = Input, X = Bit, and 0.0 = First input contact of first unit. When the entry of IEC address is completed, the corresponding FP address is automatically displayed.
⑤	Type	After registering an address, the data type is automatically selected beforehand. (Example: BOOL, INT, STRING) It is also possible to select another type from the list.
⑥	Initial	This specifies the default value to be assigned to a variable when PLC is started up. It can be changed when necessary.
⑦	Autoextern	This is automatically inserted into the header of all POUs that are displayed in the "Project" pane. (This function can be activated by proceeding "Extras" -> "Options" -> "Program options" -> "Editor" -> "Declaration editor."
⑧	Comment	This is explanation about the variable.

◆ **NOTES**

- **The name of a variable cannot be started with a number.**
- **FP address (X0, Y1 , etc.) is a reserved word, and may not be used as a variable name.**
- **To insert a new line, proceed "Edit" -> "New declaration of variable" -> "Top / Before / After / Bottom," or select either or .**

8

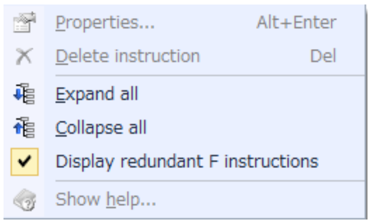
Selection Pane

8.1 Instruction Pane

In the "Instruction" pane, it is possible to select an operator, function, or function block, and insert it into the current POU body.

Proceed from the Menu Bar to "Window" -> "Pane," or select , to open the "Instruction" pane.

It is possible to specify major functions in the context menu.



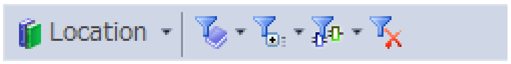
- Properties
- Delete instruction
- Expand all
- Collapse all
- Display redundant F instructions
  This option cannot be used in the PLC model FP7. If this option is activated, all instructions are displayed. In many cases, old instructions have been replaced with FP7 instructions.
- Display Help

Meaning of icons

▯	Function
▯	Function block
▯	Operator

8.1.1 Sorting and filter setting for instructions

① ②



①	Sort condition	Sort can be performed in four methods.
②	Filter	Three filters and text search can be used.

■ Sort condition

1. Location (by library)
2. Categorization (by category of instruction)
3. Name
4. Type (by function, function block, or operator)

■ Filter

One or more filters can be selected. All filters that are currently active are displayed in the Filter Setting Status Bar. To remove all filters, select .

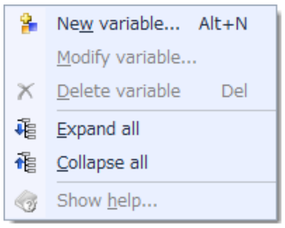
Icon	Name	Description
	Location filter	A filter is set to the library.
	Category filter	A filter is set to the instruction category.
	Instruction type filter	A filter is set to the function, function block, or operator.
	Reset	All filters are disabled, and the text search field is cleared.

8.2 Variable Pane

In the "Variable" pane, it is possible to assign a variable to the current POU body, or to create or change a variable.

Proceed from the Menu Bar to "Window" -> "Pane," or select , to open the "Variable" pane.

It is possible to specify major functions in the context menu.



- New variable
- Modify variable
- Delete variable
- Expand all
- Collapse all
- Show help

The procedure for inserting a variable into POU body slightly varies by Programming Editor.

Global variable

The global variable inserted into the current POU body is automatically copied into the current POU header.

- To save certain steps for programming, use sorting and filter setting.
- By aligning the mouse over a variable to display Tooltip concerning the properties of the variable.

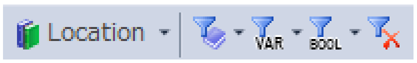
Meaning of icons

■	VAR
→■	VAR_INPUT
■→	VAR_OUTPUT
■→	VAR_OUTPUT_RETAIN
→■→	VAR_IN OUT
■	VAR_GLOBAL
■	VAR_GLOBAL_CONSTANT
■	VAR_GLOBAL_RETAIN
■	VAR_CONSTANT
↗■	VAR_EXTERNAL
↗■	VAR_EXTERNAL_CONSTANT
↗■	VAR_EXTERNAL_RETAIN

🟨	VAR_RETAIN
🟪	System variables
🟪	System constant (in upper case; Example: SYS_MEMORY_AREA_DT)

8.2.1 Sorting and filter setting for variables

① ②



①	Sort condition	Sort can be performed in four methods.
②	Filter	Three filters and text search can be used.

■ Sort condition

1. Location; examples: POU header, the list of global variables, etc.
2. Class
3. Name
4. Data type

■ Filter

One or more filters can be selected. All filters that are currently active are displayed in the Filter Setting Status Bar. To remove all filters, select ✗.

Icon	Name	Description
▽	Location filter	A filter is set to the declaration location. - POU header - List of global variables - System variables - User library (if there is any)
VAR	Class filter	A filter is set to the class.
BOOL	Data type filter	A filter is set to the data type.
✗	Reset	All filters are disabled, and the text search field is cleared.

To display only data types that are valid for the selected variable, proceed from the Menu Bar to "Extras" -> "Options" -> "Program options" -> "Editor" -> "Declaration editor," and select "Use automatic filter in the variable dialog."

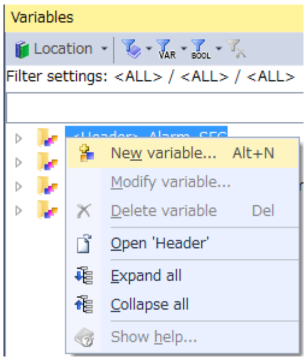
8.2.2 Create a new variable

To create a new variable, Declaration editor or the "Variable" pane can be used. It is also possible to directly declare a new variable in the POU body. This procedure varies by Programming Editor to be used (LD, FBD, ST, IL or SFC).

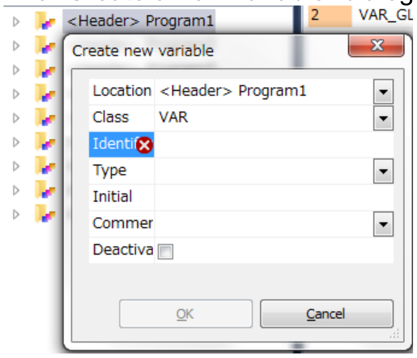
To take the following procedure, it is necessary to have the "Variable" pane opened.

◆ **PROCEDURE**

1. **In the context menu, select "New variable," or proceed from the Menu Bar to "Tools" -> "New variable."**



The "Create a new variable" dialog box is opened.



⊗The [ ] warning symbol indicates that the field entry is invalid. By aligning the mouse over it to display Tooltip concerning required data.

2. **Under "Location," select the location of the variable.**

   A variable can be declared in the current POU header (<Header>), the list of global variables (<Global variables>), or user-defined library (if there is any).

3. **Under "Class," select a variable class.**

   Classes that can be used vary by the location of declaration.

4. **Enter a name in the "Identifier."**

   - Special characters cannot be used. Examples: !, ", $, %, parentheses, etc.
   - A number cannot be used for the starting character.
   - An underscore cannot be used for the ending character. More than one

underscores cannot be used consecutively.
It is not possible to use an address only for the variable name. Example: R0', 'DT0', 'LD0'
It is not possible to use a key word for a variable name. Example: ADD', 'ARRAY', 'INT'

5. **In the case of a global variable, enter an address.**
6. **Select "Type."**
7. **Enter a default value. (Optional)**
8. **Enter a comment. (Optional)**
9. **Press the [OK] button.**

   The new variable is created in the designated location.
   If a new global variable is created in this dialog box, it is declared in the list of global variables, and is automatically inserted into the POU body as VAR_EXTERNAL.

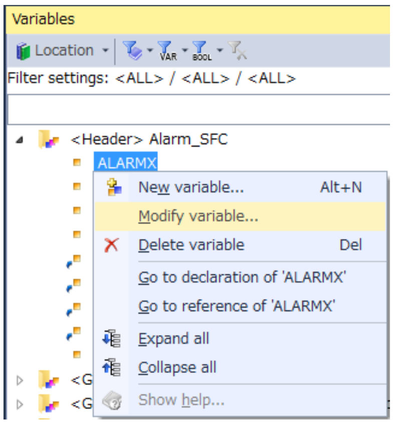
8.2.3 Modify a variable

By using the "Modify variable" command, change the properties of an existing variable. When the variable name is changed, a new variable is created.

To take the following procedure, it is necessary to have the "Variable" pane opened.

◆ **PROCEDURE**

1. **To save certain steps for programming, use sorting and filter setting.**
2. **Select a variable to be changed.**
3. **In the context menu, select "Modify variable"**



4. **In the dialog box, change the current setting.**

   For explanation concerning fields, refer to "8.2.2 Create a new variable."

5. **Press the [OK] button to refresh.**

■ Automatic refreshment of All POUs

Proceed from the Menu Bar to "Extras" -> "Options" -> "Program options" -> "Editor" -> "Declaration editor," and select "Automatically refresh declarations that have been changed in the save process," to reflect all changes that have been made to variables in the list of global variables and POU header. The changes are reflected in all POU headers and POU bodies in which the variables of the current project are used. If changes are saved in the list of global variables or POU header, all those changes are immediately applied.

◆ **NOTES**

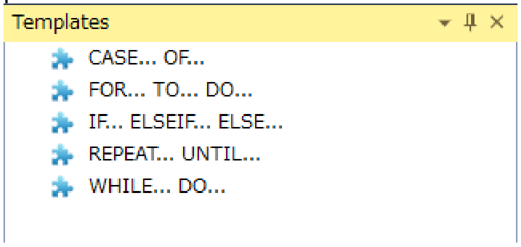
Global variables should only be changed in the list of global variables. Otherwise, the effect of automatic update will be lost.

8.3 Template pane

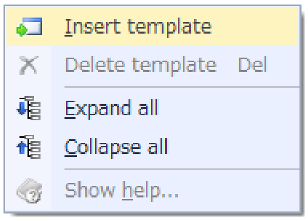
In the "Template" pane, templates are prepared to save certain steps in programming in LD Editor or ST Editor.

In IL Editor, a template is automatically generated when an instruction is inserted from the "Instruction" pane, or when a function block instance is inserted in the "Variable" pane.

Proceed from the Menu Bar to "Window" -> "Pane," or select , to open the "Template" pane.


Templates
CASE... OF...
FOR... TO... DO...
IF... ELSEIF... ELSE...
REPEAT... UNTIL...
WHILE... DO...

It is possible to specify major functions in the context menu.



- Insert a template
- Delete a template
- Expand all
- Collapse all
- Show Help

The procedure for inserting a template into POU body slightly varies by Programming Editor. To take the following procedure, it is necessary to have the "Template" pane opened.

◆ **PROCEDURE**

1. **Align the cursor in the programming window.**
2. **Double-click on a template.**

   The template is inserted into the cursor position.
3. **Enter the template.**

   In accordance with the program requirements, enter the values and variables into the template.

All elements that have been used recently are added to the list. This list can be displayed by pressing <Ctrl>+<Shift>+<v> in the program window. The content of the list may vary depending on the details of programming.

For more details, refer to the Online Help.

9 Programming

9.1 Programming Editor

To select a Programming Editor, the following criteria may be used, for example.

- Minimize required program steps
- Reduce the space of program window, and make the program easy to see
- Save time required for the re-configuration of the program

Generally, LD Editor is optimal for programs that use simple calculation or BOOL operation, or for programs that use the call of functions and function blocks. ST Editor is recommended for functions, function blocks, and complex algorithms.

◆ **NOTES**

Conditional Compile can be used only in ST Editor.

For more details, refer to the Online Help.

9.2 Ladder Diagram (LD) / Function Block Diagram (FBD)

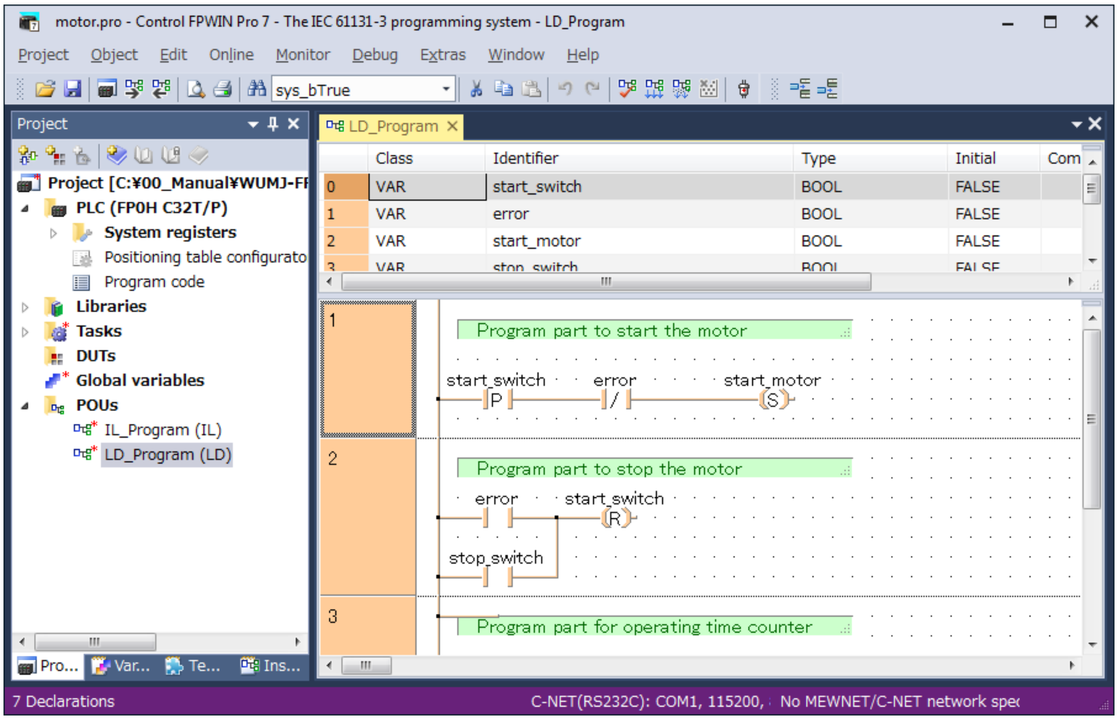
Ladder diagram and function block diagram are programming languages to write programs using standard graphical symbols and programming elements.

Programming elements for LD include bus line, contact, coil, input/output variables, function, function block, jump and return, and horizontal and vertical lines.

FBD uses similar programming elements, but has no bus line, contact, or coil. Examples of use and explanation in this section assume LD, unless otherwise specified.

Example of use of LD

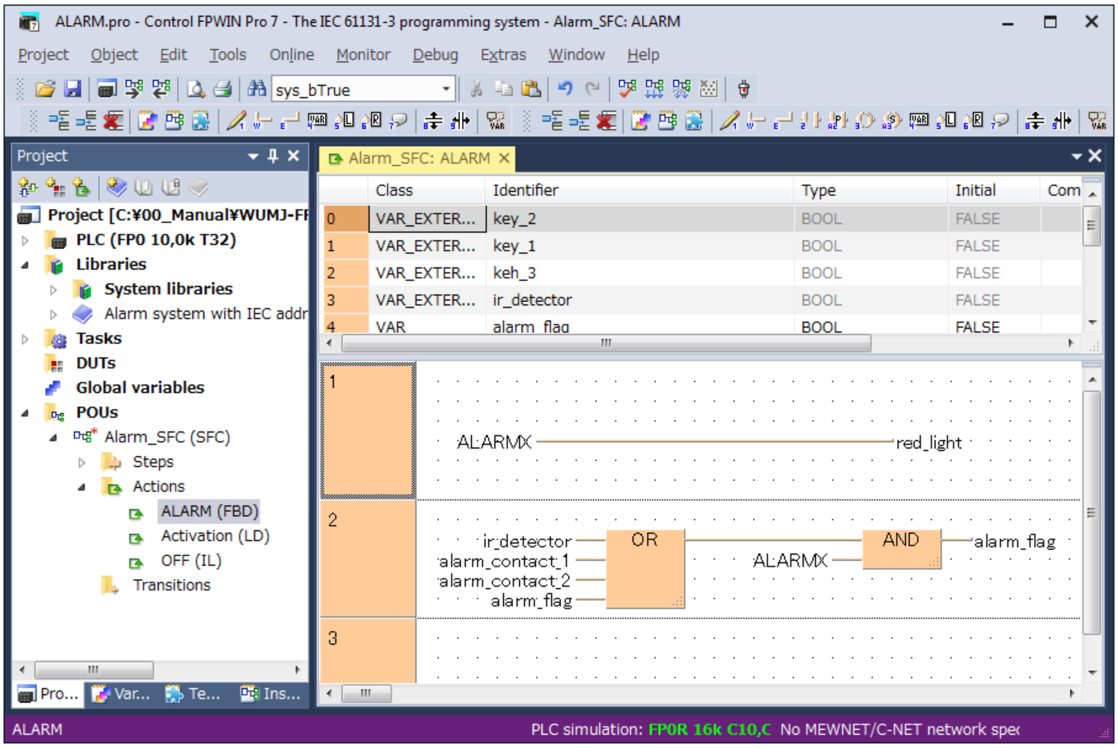
Three networks that are programmed using ladder diagrams


motor.pro - Control FPWIN Pro 7 - The IEC 61131-3 programming system - LD_Program
Project Object Edit Online Monitor Debug Extras Window Help
sys_bTrue
Project
Project [C:¥00_Manual¥WUMJ-FF
PLC (FP0H C32T/P)
System registers
Positioning table configurato
Program code
Libraries
Tasks
DUTs
Global variables
POUs
IL_Program (IL)
LD_Program (LD)
Pro... Var... Te... Ins...
LD_Program
1
Program part to start the motor
start_switch
P
error
/
start_motor
S
2
Program part to stop the motor
error
start_switch
R
stop_switch
3
Program part for operating time counter
7 Declarations
C-NET(RS232C): COM1, 115200, No MEWNET/C-NET network spe

	Class	Identifier	Type	Initial	Com
0	VAR	start_switch	BOOL	FALSE	
1	VAR	error	BOOL	FALSE	
2	VAR	start_motor	BOOL	FALSE	
3	VAR	stop_switch	BOOL	FALSE	

Example of use of FBD

Two networks that are programmed using function block diagrams



	Class	Identifier	Type	Initial	Com
0	VAR_EXTER...	key_2	BOOL	FALSE	
1	VAR_EXTER...	key_1	BOOL	FALSE	
2	VAR_EXTER...	keh_3	BOOL	FALSE	
3	VAR_EXTER...	ir_detector	BOOL	FALSE	
4	VAR	alarm_flag	BOOL	FALSE	

ALARM PLC simulation: FP0R 16k C10,C No MEWNET/C-NET network spec

The program is displayed in the programming window, along with POU header and POU body. The POU body is divided into networks. Each network is equipped with a network information area that contains the network number and information (labels, breakpoints, etc.).

The operation link result is lost between one network and another network. Therefore, to use the result of a network in the subsequent operation, it is necessary to store it (as a variable) before the operation is executed in the subsequent network.

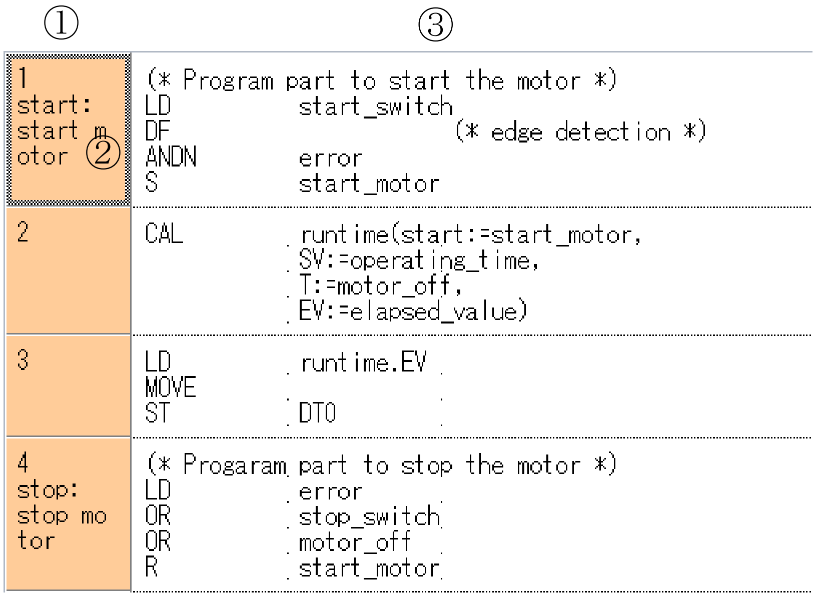
To open Online Help for the function, function block, or operator, select an object in the POU body, and press the <F1> key.

◆ **NOTES**

Within one network, up to 160 elements can be written. When you draw a line, each line is counted as one element. Therefore, draw the entire line as one element.

9.2.1 Overview of network

"Network" refers to a program unit that contains subtasks that are respectively completed. The procedure for creating a network is the same for LD, FBD and IL Editors. A network consists of two columns. The left column contains network header, network number, label, and title. The right column contains network body. The label is required for jump. The title can be used as a comment.



```
1          (* Program part to start the motor *)
start:     LD          start_switch
start m    DF                       (* edge detection *)
otor       ANDN        error
           S           start_motor
2          CAL         runtime(start:=start_motor,
                       SV:=operating_time,
                       T:=motor_off,
                       EV:=elapsed_value)
3          LD          runtime.EV
           MOVE
           ST          DT0
4          (* Progaram part to stop the motor *)
stop:      LD          error
stop mo    OR          stop_switch
tor        OR          motor_off
           R           start_motor
```

Label of IL network

①	Network information areas that have serial network numbers
②	Label (example: start) and title (example: start motor)
③	Programming window

All networks gather to comprise one POU. Within POU, it is possible to insert or delete a network, or to include/exclude it into the scope of compile. It is also possible to add a label or comment.

To optimize network height, double-click the line between the network, or proceed from the Menu Bar to "Tools" and use "Optimize height of network."

◆ **NOTES**

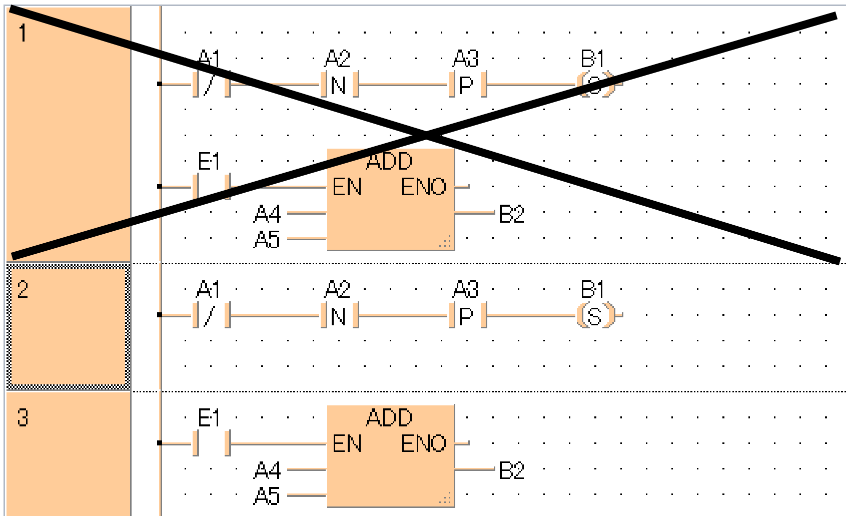
To save program code, do not assign a label if no jump or loop is used.

■ Processing of network

It is recommended to lay out each program chain or subtask in respective networks. The advantages of doing so are as follows.

- The structure becomes clear
- It is possible to perform debugging with the network disabled
- There is no "delayed assignment" of variables

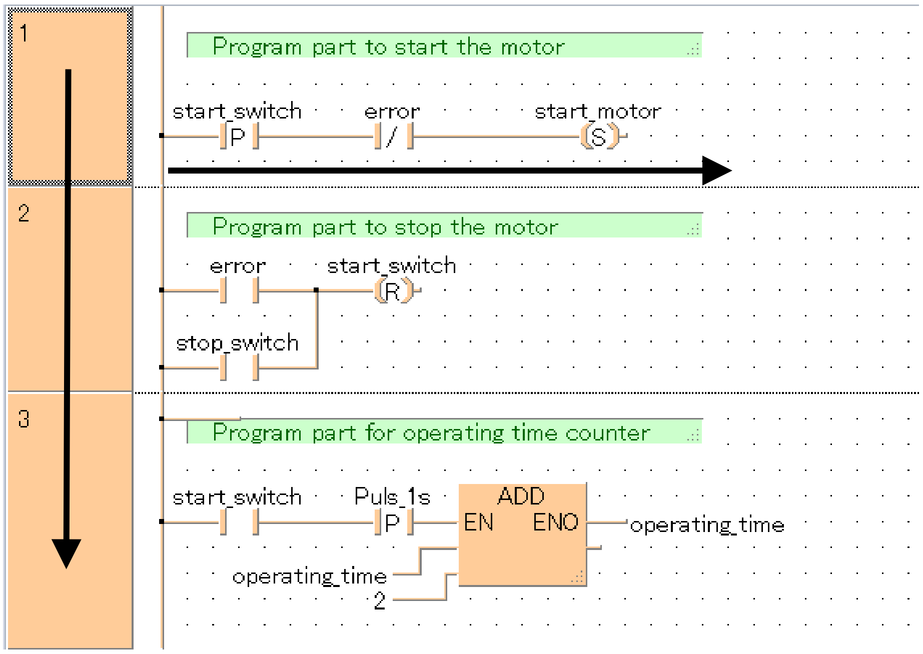
Network 1 contains two program chains. In this case, an error occurs while compiling or checking a program. However, there is no difference in the processing of Network 1, compared to the processing of Network 2 and 3.



For more details, refer to the Online Help.

■ Priority of the execution of networks in POU

Multiple networks written in POU (ladder diagram, function block diagram, instruction list) are executed one by one from top to bottom, as indicated below.



If an input or output value is changed in a position where program is already being executed, the change to the value is detected in the next execution scan of PLC.

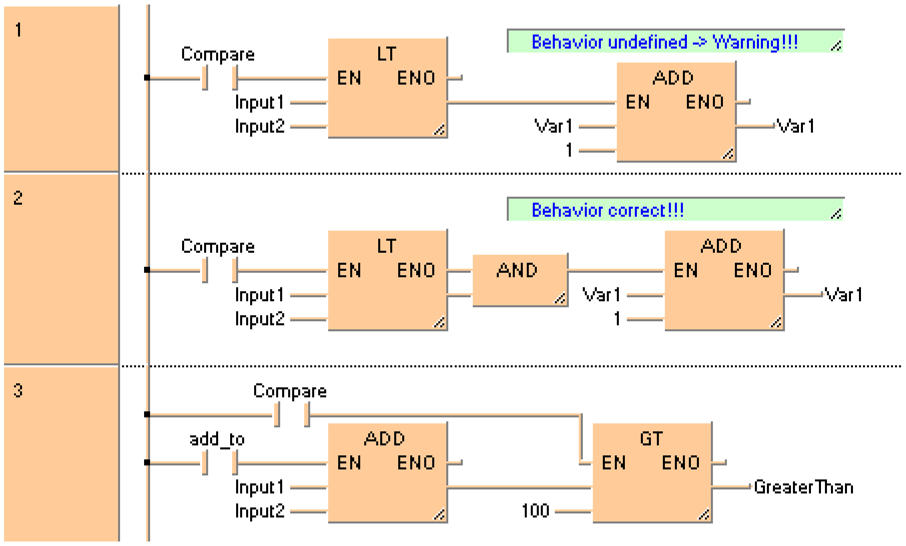
Instructions in one network are executed from left to right at first, and subsequently from top to bottom.

9.2.2 EN/ENO connection of FUN or FB

Whether or not FUN/FB is executed, and whether or not writing is performed into output variables, are determined based on the status (TRUE or FALSE) of Enable Input EN of the head Function (FUN) or Function Block (FB). (Refer to EN/ENO of LD or FBD.)

If the subsequent FUN/FB uses one of the output variables above as an input variable, the compiler generates a temporary variable. Because other temporary variables may use the same address, the value of the input variable is not finalized while nothing is written into the output variable (for example, while ENO is FALSE).

To avoid this, it is necessary to lay out all FUNs and FBs in the network so that they will be executed only when the FUN/FB connected in front of them are executed. The compiler confirms that the EN input of the subsequent FUN or FB is unconnected, and that the AND function is connected.



In the case of Network 1, the Enable Input EN of ADD becomes undefined if the execution condition 'Compare' is "FALSE," and a warning message is displayed.

In the case of Network 2, programming is performed correctly, and no warning is issued.

In the case of Network 3, if 'add_to' is FALSE, and if 'Compare' is TRUE, the subsequent FUN(GT) is executed, but FUN(ADD) in the preceding part is not executed. Even though the behavior becomes undefined, the compiler does not issue a warning message.

9.2.3 Processing order of networks in LD or FBD Editor

Networks in LD or FBD Editor are processed in the following order.

PROCEDURE

1. **The processing of compiler is started at the top left of bus line.**

   If there is an element that is directly connected to the bus line, processing starts in that part. If there is no element that is connected to the bus line, compile is started with all inputs. FPWIN Pro7 processes inputs as the preparatory stage for compile.

2. **Subsequently, the compiler searches for the elements in the highest part, and analyzes and evaluates them.**

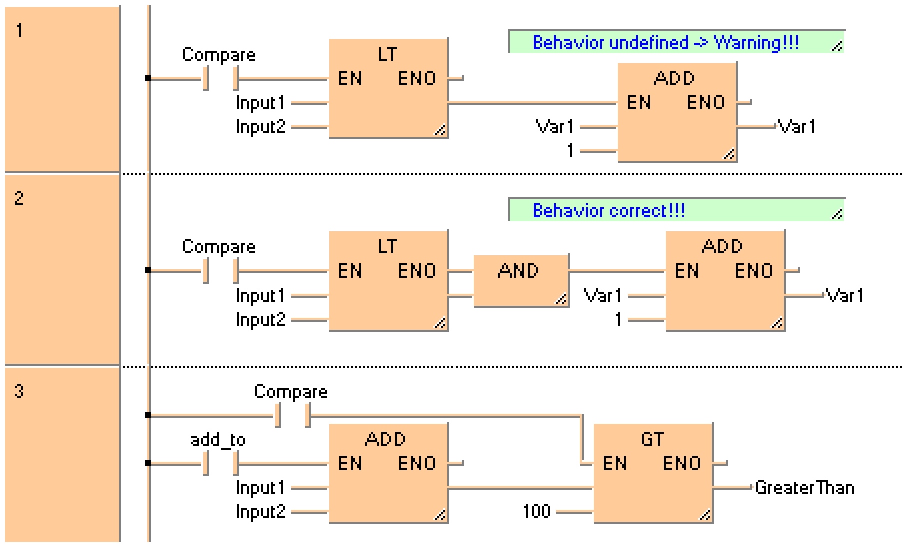
   If two elements are in the same part (line), compile is started with the element on the extreme left. If the element is an output, or a cue in function / function block, the compiler generates the relevant program code. To temporarily store the signal, or to put the signal in the stack, a temporary variable is inserted.

3. **The flows to all elements that are directly connected to the bus line are induced for an element (intermediate data) created in Step 2 above, and the next element is analyzed and evaluated.**

   The compiler returns to Step 2 above, and this behavior is repeated until all elements have been analyzed and evaluated in the end.

4. **The procedure above is canceled or completed.**

An example of execution order is indicated below.



9.2.4 Handling of Jump (JP) in compiler

The jump command is executed at the end of network, regardless of the position where jump is written in the network.

9.2.5 Edit mode

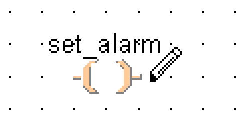
There are two edit modes: Layout mode and Blend mode.

Open "Edit" in the Menu Bar to select "Draw lines" or "Layout mode." It is also possible to select "Draw lines" in the context menu. In this case, it is possible to switch back from "Blend mode" to "Layout mode" by right-clicking.

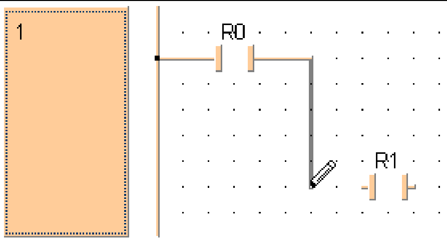
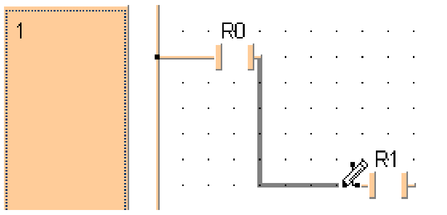
It is also possible to switch between these two modes by double-clicking in an area of programming window without any object.

■ Blend mode

The cursor becomes a pen-like shape.



There are two methods as follows to draw a line from the starting contact to the goal contact.

Edit mode	Description	
Linear	Drag the pen from the start point to the goal point (press the left mouse button). A straight horizontal or vertical line is drawn between the elements.	
Automatic calculation of line	Click once at the start point, and click once more time at the goal point. The line is calculated automatically, and drawn between the elements.	

■ Layout mode

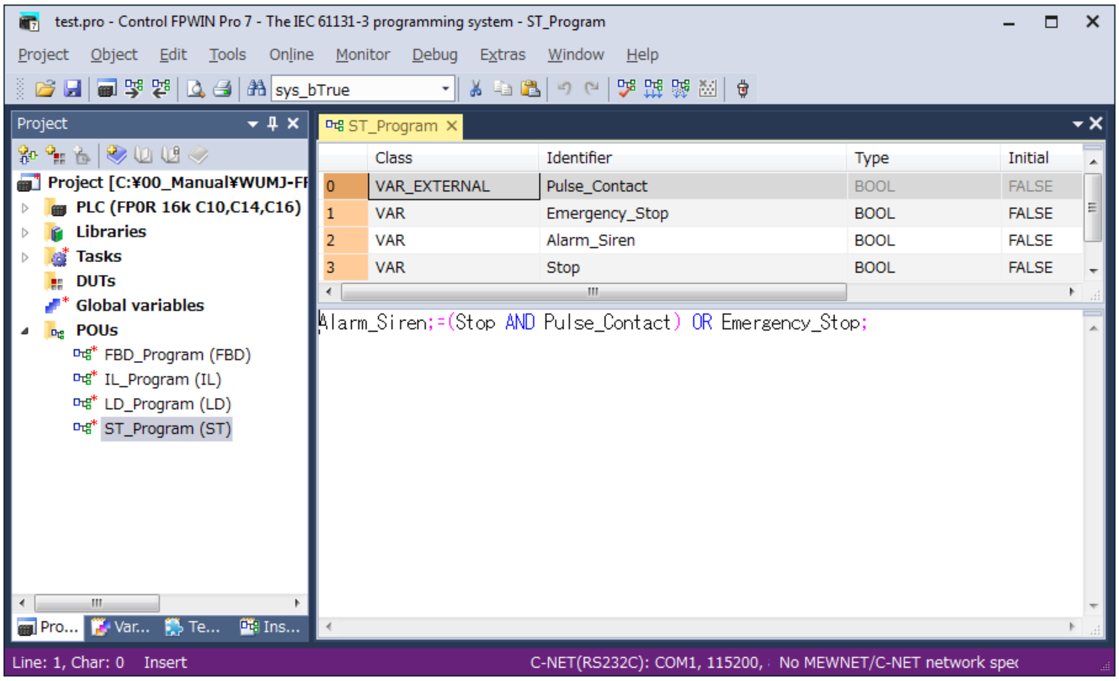
The cursor becomes an arrow shape.

When an instruction is inserted, a line is automatically added to the program elements in the program window. The first element is automatically connected to the bus line. The subsequent elements are automatically connected if aligned in a row. Proceed from the Menu Bar to "Extras" -> "Options" -> "Program options" -> "Editor" -> "LD/FBD editor," to disable this behavior.

Using the layout mode, select elements (contact, coil, jump, return, input/output variable, etc.) in the Tool Bar, in order to lay out elements in the programming window.

9.3 Structured Text (ST) Editor

Structured Text is a text programming language that is very much alike other high-level languages. ST is capable of writing complex programs and programs with a control structure, using an optimized programming language. It can be used in all PLCs, and does not require many resources, such as steps, labels, and calls, compared to other editors.


test.pro - Control FPWIN Pro 7 - The IEC 61131-3 programming system - ST_Program
Project Object Edit Tools Online Monitor Debug Extras Window Help
sys_bTrue
Project
Project [C:¥00_Manual¥WUMJ-FF
PLC (FP0R 16k C10,C14,C16)
Libraries
Tasks
DUTs
Global variables
POUs
FBD_Program (FBD)
IL_Program (IL)
LD_Program (LD)
ST_Program (ST)
Pro... Var... Te... Ins...
ST_Program

	Class	Identifier	Type	Initial
0	VAR_EXTERNAL	Pulse_Contact	BOOL	FALSE
1	VAR	Emergency_Stop	BOOL	FALSE
2	VAR	Alarm_Siren	BOOL	FALSE
3	VAR	Stop	BOOL	FALSE

```
Alarm_Siren:=(Stop AND Pulse_Contact) OR Emergency_Stop;
```

Line: 1, Char: 0   Insert   C-NET(RS232C): COM1, 115200, No MEWNET/C-NET network spec

9.3.1 Expressions

Expressions are structured with operands that are connected by operators, in accordance with the operation priority.

Example	Description
A+B	The sum of operands A and B is called an expression.

If they are of the same operation priority, the operation process proceeds from left to right.

If the values of variables are A:=1.0; B:=2.0; C:=3.0; D:=4.0

```
X:=A+B-C*SQRT(D);                 The operation result will be "-3."
X=1.0+2.0-3.0×√4.0
```

If parentheses are inserted, the operation priority is changed.

```
X:=A+(B-C)*SQRT(D);               The operation result will be "-1."
X=1.0+(2.0-3.0)×√4.0
```

In the notation of BOOL operation, all operations are processed in all cases.

```
IF a<100 AND UserFun1(a) THEN
    a:=a+1;
END_IF;
```

In this example, "UserFun1" is executed even if a>=100.

To avoid the processing of "UserFun1" (for example, if it takes too much time, or if an operation error may be caused at a>=100 and the memory area may be overwritten), the expression may be written as follows.

```
IF a<100 THEN
   IF UserFun1(a) THEN
       a:=a+1;
   END_IF;
END_IF;
```

It is also possible to write array elements in the expression.

```
X:=Array1[i+2];
```

■ Operand

In ST Editor, the following operands can be selected.

Name	Data type	Example
Literal	Value Character string Time	49、3.14159 'This is a text' T#8d_3h_23m
Variable	Single-element variables Element of Array Element of DUT Element of Array or DUT	Var1 Array1[5] Dut1.Var1 Dut1.Array1[i+5]
Function	Function call	Fun1(a,b,c)

Operands are connected with operators. The combination of operators and operands are called "expression."

■ Operator

In ST Editor, the following operators can be selected.

Operator	Function	Priority
()	Round parentheses, function call	Highest
- NOT	Inversion (negation) of the sign Complement	
** * / MOD	Modulo operation Multiplication Division Modulo operation (remainder)	
+ -	Addition Subtraction	
>,<,>=,<=	Comparison	
= <>	Equal Not equal	
&, AND	Logical AND	Lowest
XOR	Exclusive logical OR	
OR	Logical OR	

Operators connect operands.

9.3.2 Statements

Instructions in ST Editor include the following, for example.

- Substitution
- Function block call
- Function call
- Return
- Selection statement
- Repeat statement
- Conditional compile statement

■ Substitution

Key word	Example	Description
:=	a:=87;b:=b+1;c:=SIN(x);	The operation result on the right is substituted into the identifier on the left.

■ Function block call

Example	Description

TON1(IN:= Start1,
PT:=T#300ms ,
Q=> End1 ,
ET=> EV_1);
	Argument of function block that has a formal parameter
	Note:
	In the case of argument that has a formal parameter, the order of parameters is not important.

TON1(IN:=Start1,
PT:=T#300ms);
End1:= TON 1.Q;
Ev1:= TON 1.EV;
	Argument of function block that has no formal parameter
	Note:
	In the case of argument that has no formal parameter, the order of parameters is important.

TON1.IN:=Start1;
TON 1.PT:=T#300ms;
TON 1();
IF TON 1.Q THEN
....
END_IF;
	Unlimited use of formal parameters in program call

(Note): The function block does not return a value.

■ Function call

Example	Description
Y:=SIN(x);	Argument of function that has no formal parameter
Y:=LIMIT(MN:=0,IN:=X,MX:=100);	Argument of function that has a formal parameter Note: In the case of argument that has a formal parameter, the order of parameters is not important. For a user-defined function, EN input or EN output may be omitted. The omitted EN input is processed as "TRUE."

(Note): The function returns a value.

■ Return

Key word	Example	Description
RETURN	RETURN;	Returns to the POU that has made the call

■ Selection statement

Key word	Example	Description

IF	IF a>=0 AND a<=10 THEN
	    b:=0;
	    ELSIF a>=100 THEN
	        b:=1;
	    ELSE
	        b:=2;
	END_IF;
		Branches in accordance with the BOOL-type value of the expression

CASE	CASE a OF
	    0: b:=0;
	    1,2: b:=1;
	    3,4,10...20: b:=2;
	    100..110: b:=3;
	    ELSE b:=4;
	END_CASE;
		Multiple selection based on the value of the variable (multi-direction branches)

■ Repeat statement

Repeat statements FOR, WHILE, REPEAT, and loop exit statement EXIT

Key word	Example	Description

FOR	FOR i:=0 TO 100 DO
	   SUM:=SUM + a[i]
	END_FOR;
	FOR i:=0 TO 100 BY 10 DO
	   IF a[i]>=100 THEN
	      EXIT;
	   END_IF;
	END_FOR;
		Loop that has increment 1 or an increment specified by the user
		Note:
		Do not use the value of the control variable following the completion of the loop ('i' in this case). The value of the control variable is not guaranteed.

WHILE	i:=0;
	WHILE i<=100 AND a[i]<100 DO
	   i:=i+10;
	END_WHILE;
		In this loop, the conditions are checked before executing the program in the loop, and repeats the process until the conditions are matched

REPEAT	i:=0;
	REPEAT
	   i:=i+10;
	   UNTIL i>100 OR a[i]>=100
	END_REPEAT;
		In this loop, the conditions are checked after executing the program in the loop, and repeats the process until the conditions are matched

EXIT	EXIT;	Unconditional exit from the loop

■ Conditional compile statement

Check and compile are executed only if the program part of the expression of a BOOL-type constant is TRUE.

Examples:

```
#IF ConstantBooleanExpression1 #THEN
(* Program part 1 executed if ConstantBooleanExpression1 = TRUE *)

#ELSIF ConstantBooleanExpression2 #THEN
(* Program part 2 executed if ConstantBooleanExpression2 = TRUE *)

#ELSE
(* Program part 3 executed if ConstantBooleanExpression1 and
ConstantBooleanExpression2 = FALSE *)

#END_IF;
```

Purposes:

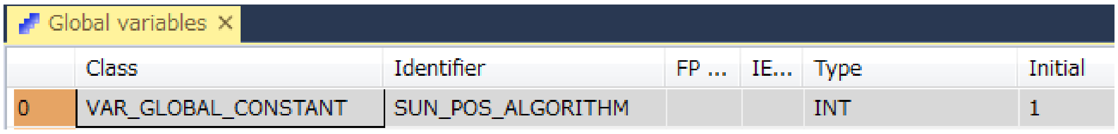
- Version with a different code; examples) recipe, initialization routine
- Used to create program codes for different PLC types from a single program
- Optimization of code

A constant expression of Conditional Compile can be structured by combining the following types.

- Constant-type external variable
- Local variable (constant)
- Literal value
- PLC model constant
- PLC information instruction

Examples:

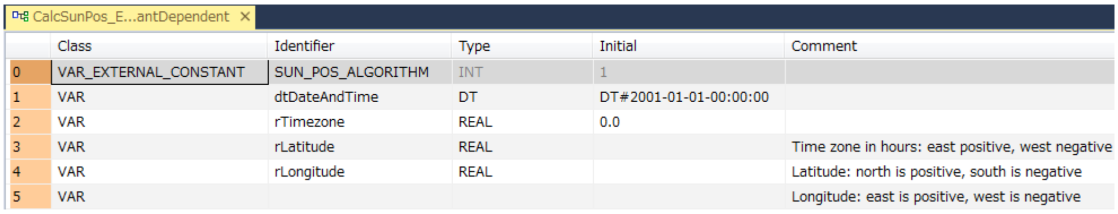
List of global variables



Global variables

	Class	Identifier	FP ...	IE...	Type	Initial
0	VAR_GLOBAL_CONSTANT	SUN_POS_ALGORITHM			INT	1

POU header



CalcSunPos_E...antDependent

	Class	Identifier	Type	Initial	Comment
0	VAR_EXTERNAL_CONSTANT	SUN_POS_ALGORITHM	INT	1	
1	VAR	dtDateAndTime	DT	DT#2001-01-01-00:00:00	
2	VAR	rTimezone	REAL	0.0	
3	VAR	rLatitude	REAL		Time zone in hours: east positive, west negative
4	VAR	rLongitude	REAL		Latitude: north is positive, south is negative
5	VAR				Longitude: east is positive, west is negative

ST Body

```
#if(SUN_POS_ALGORITHM=1) #then

(* PSA Algorithm: 0.06° accuracy, 1950 steps, 3.6 ms cycle time on a FP-X、FP-XH *)
SunPosition_PSA(dtDateAndTime := dtDateAndTime,
            rTimezone := rTimezone,
            rLatitude := rLatitude,
            rLongitude := rLongitude,
            bError => bError,
            rZenith => rZenith,
            rAzimuth => rAzimuth);
#elsif (SUN_POS_ALGORITHM=2) #then
      (* SolPos Algorithm: 0.00257° accuracy, 3700 steps, 6.3 ms cycle time on a FP-X、FP-
XH *)    SunPosition_SolPos(dtDateAndTime := dtDateAndTime,
              rTimezone := rTimezone,
              rLatitude := rLatitude,
              rLongitude := rLongitude,
              rPressure := 1013.0,
              rTemperature := 20.0,
              bError => bError,
              rZenith => rZenith,
              rAzimuth => rAzimuth);
#else
      (* SolPos Algorithm All: 0.00257° accuracy, 7300 steps, 10.3 ms cycle time on a FP-X、
FP-XH *)
       SunPosition_SolPos_All(dtDateAndTime := dtDateAndTime,
                rTimezone := rTimezone,
                rLatitude := rLatitude,
                rLongitude := rLongitude,
                rPressure := 1013.0,
                rTemperature := 20.0,
                dutAdditionalInputs := dutAdditionalInputs,
                bError => bError,
                rZenith => rZenith,
                rAzimuth => rAzimuth,
                dutAdditionalOutputs => dutAdditionaOutputs);
#end_if;
```

PLC model constant

In a PLC model constant (name that starts with upper case), the program to be compiled is determined by the connected PLC. It is also possible to select the following items.

- Current PLC model
- The group of PLC models that are supported by FPWIN Pro7
- Specified PLC models that are supported by FPWIN Pro7

Example:

```
#if((SYS_CURRENT_PLC AND (SYS_FP0 OR SYS_FP_e))<>0) #then (* FP0, FPe *)
   SunPosition_PSA(dtDateAndTime := dtDateAndTime,
               rTimezone := rTimezone,
               rLatitude := rLatitude,
               rLongitude := rLongitude,
               bError => bError,
               rZenith => rZenith,
               rAzimuth => rAzimuth);
#elsif ((SYS_CURRENT_PLC AND (SYS_FP2 OR SYS_FP2SH)<>0) #then
(* FP2,FP2SH *)
   SunPosition_SolPos_All(dtDateAndTime := dtDateAndTime,
                    rTimezone := rTimezone,
                    rLatitude := rLatitude,
                    rLongitude := rLongitude,
                    rPressure := 1013.0,
                    rTemperature := 20.0,
                    dutAdditionalInputs := dutAdditionalInputs,
                    bError => bError,
                    rZenith => rZenith,
                    rAzimuth => rAzimuth,
                    dutAdditionalOutputs => dutAdditionaOutputs);
#else    (* FP Sigma, FP-X, FP-XH, FP0R... *)
   SunPosition_SolPos(dtDateAndTime := dtDateAndTime,
                 rTimezone := rTimezone,
                 rLatitude := rLatitude,
                 rLongitude := rLongitude,
                 rPressure := 1013.0,
                 rTemperature := 20.0,
                 bError => bError,
                  rZenith => rZenith,
                  rAzimuth => rAzimuth);
#end_if;
```

PLC information instruction

PLC information instruction returns a BOOL-type constant.

Example:

```
dutDTBCD := GET_RTC_DTBCD();
#if(IsInstructionSupported('F230_DTBCD_TO_SEC'))#then
   (* FP0R, FPΣ, FP-X, FP-XH FP2, FP2SH *)
   F230_DTBCD_TO_SEC(dut_DTBCD, diSeconds);
 #else
   (* FP0, FPe *)
   iSecond    := WORD_BCD_TO_INT(dutDTBCD.MinSec and16#00FF);
   iMinute    := WORD_BCD_TO_INT(SHR(dutDTBCD.MinSec, 8));
   iHour    := WORD_BCD_TO_INT(dutDTBCD.DayHour and16#00FF);
   iDay    := WORD_BCD_TO_INT(SHR(dutDTBCD.DayHour, 8));
   iMonth    := WORD_BCD_TO_INT(dutDTBCD.YearMonth and16#00FF);
   iYear    := WORD_BCD_TO_INT(SHR(dutDTBCD.YearMonth, 8));
    GET_RTC_DT := CONCAT_DT_INT(iYear+2000, iMonth, iDay, iHour, iMinute, iSecond, 0,
ERROR);
 #end_if;
```

9.3.3 Characteristics of ST Editor

When ST Editor is used, note the following points.

- ST Editor has no functions or function blocks that have EN/ENO. To control the execution of functions and function blocks, use IF statements or CASE statements instead.
  Example
  ```
  IF start then
     F10_BKMV( s1_Start:= source_Array[1],
        s2_End:= source_Array[3],
        d_Start=> target_Array[0]);
  END_IF;
  </SPAN>
  ```
- For a BOOL-type constant, "0" and "1" can also be used, in addition to TRUE and FALSE.
- P instructions of FP Pulsed Library cannot be used in ST Editor.
- The address acquisition function of "FP Tool Library" is common to input and output.

ST Function	Original functions of FP Tool Library	
Adr_Of_Var	Adr_Of_Var_I	Adr_Of_Var_O
AdrLast_Of_Var	AdrLast_Of_Var_I	AdrLast_Of_Var_O
Adr_Of_VarOffs	Adr_Of_VarOffs_I	Adr_Of_VarOffs_O
AdrDT_Of_Offs	AdrDT_Of_Offs_I	AdrDT_Of_Offs_O
AdrFL_Of_Offs	AdrFL_Of_Offs_I	AdrFL_Of_Offs_O

- When a program of control structure that has IF, CASE or other conditional statements is debugged, the program code in that structure is executed even if the execution condition is not "TRUE." However, individual instructions are not executed by themselves.
- Do not use the value of the control variable following the completion of the loop. The value of the control variable is not guaranteed.
- If a program that contains many steps is in a loop, the scan time of PLC may become long. Change the setting of the watchdog timer (system register No.30), or divide the program into several cycles.
- A BOOL-type or value-type constant is defined by providing the name of the basic data type, and a "type prefix" of the constant that consists of '#.' For example, it will be specified as 'INT#2' or 'REAL-3.2.'
  To identify a literal, programming systems from several manufacturers require type-specified constants. Therefore, FPWIN Pro7 supports type-specified literal. Because the compiler of FPWIN Pro7 internally assigns a correct type to literal, it is not required to assign a type to literal as described above.
- The following items are not case sensitive.
  - Variable name (example: abcd, ABCD and aBCd are considered to be the same)
  - Key word (example: "FOR" and "for" are considered to be the same)

9.3.4 Comment

In ST Editor, a comment can be written in any position. A comment should be sandwiched by round parentheses attached with asterisks: '(*' and '*)'. A comment can also be written in multiple lines.

Example:

```
(* this is a comment on one line *)
  (* this is a
  comment on two lines *)
```

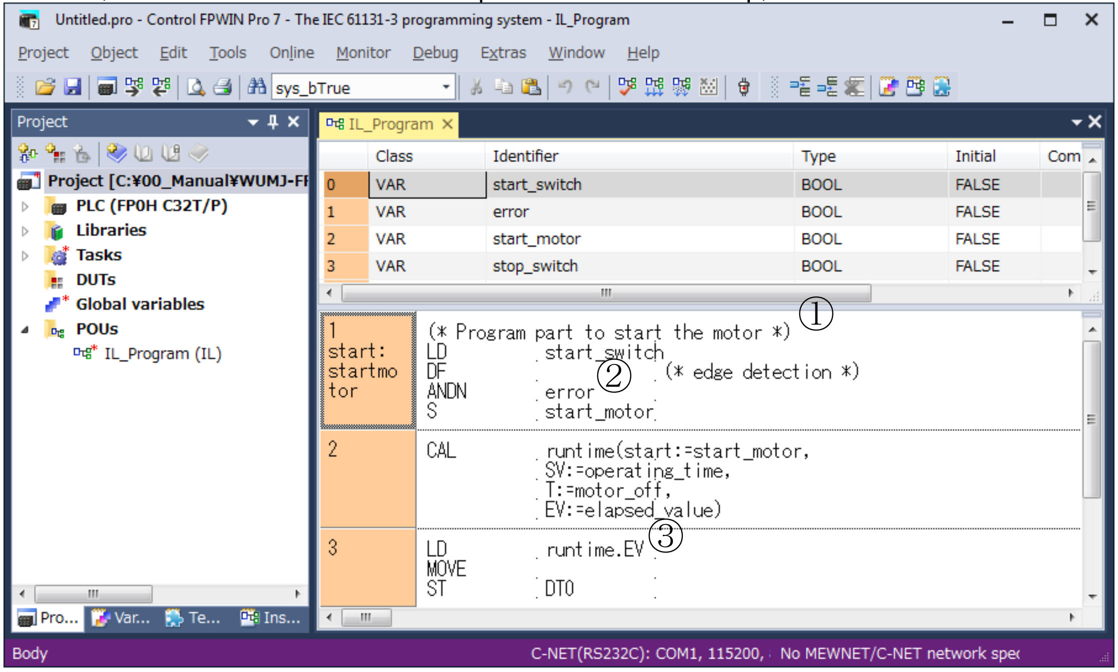
A comment is invalid unless it has *) at the end.

```
    (* this is an invalid comment
```

It is permitted to write a comment that has a nesting structure, such as (* Level 1 (* Level 2 ....*) *), for example. However, such a comment is not common in IEC 61131-3. It may become an error depending on the setting of the compiler. Proceed from the Menu Bar to "Extras" -> "Options" -> "Compile options," and refer to "Additional errors."

9.4 Instruction List Editor (IL)

Instruction List Editor is a text-based editor that has standard operators based on IEC61131-3. In Instruction List, the basic set of instructions based on the standard specification of IEC 61131-3, as indicated in the "Standard operators" in Online Help, can be used.


Untitled.pro - Control FPWIN Pro 7 - The IEC 61131-3 programming system - IL_Program
Project Object Edit Tools Online Monitor Debug Extras Window Help
sys_bTrue
Project
Project [C:¥00_Manual¥WUMJ-FF
PLC (FP0H C32T/P)
Libraries
Tasks
DUTs
Global variables
POUs
IL_Program (IL)
Pro... Var... Te... Ins...
IL_Program
①
②
③
Body
C-NET(RS232C): COM1, 115200, No MEWNET/C-NET network spec

	Class	Identifier	Type	Initial	Com
0	VAR	start_switch	BOOL	FALSE	
1	VAR	error	BOOL	FALSE	
2	VAR	start_motor	BOOL	FALSE	
3	VAR	stop_switch	BOOL	FALSE	

```
1           (* Program part to start the motor *)
start:      LD      start_switch
startmo     DF                      (* edge detection *)
tor         ANDN    error
            S       start_motor

2           CAL     runtime(start:=start_motor,
                    SV:=operating_time,
                    T:=motor_off,
                    EV:=elapsed_value)

3           LD      runtime.EV
            MOVE
            ST      DT0
```

IL Program

①	Comment
②	Operand
③	Operator

The POU body is divided into networks. Each network is equipped with a network information area that contains the network number and information (labels, breakpoints, etc.).

A program consists of the following three columns: operators, operands, and comments. A comment can be put in any position in the programming window, sandwiched by round parentheses attached with asterisks. It is also possible to write a comment of multiple lines or of vacant line.

To change the width of network, drag the vertical line with the mouse.

To display the vertical line, proceed from the Menu Bar to "Extras" -> "Options" -> "Program options" -> "Editor" -> "IL Editor," and select "Display grid."

Usually, IL network is started with Load Instruction (LD). The link result is stored in the accumulator. Note that its content is lost between one network and another network.

◆ **NOTES**

- **To use the result of a network in the subsequent operation, it is necessary to store it (as a variable) before the operation is executed in the subsequent network.**
- **Avoid creating an excessively large network. It may take excessive time to search for errors, jumps, or labels. Effectively utilize the functions of structured programming.**

9.5 Sequential Function Chart (SFC)

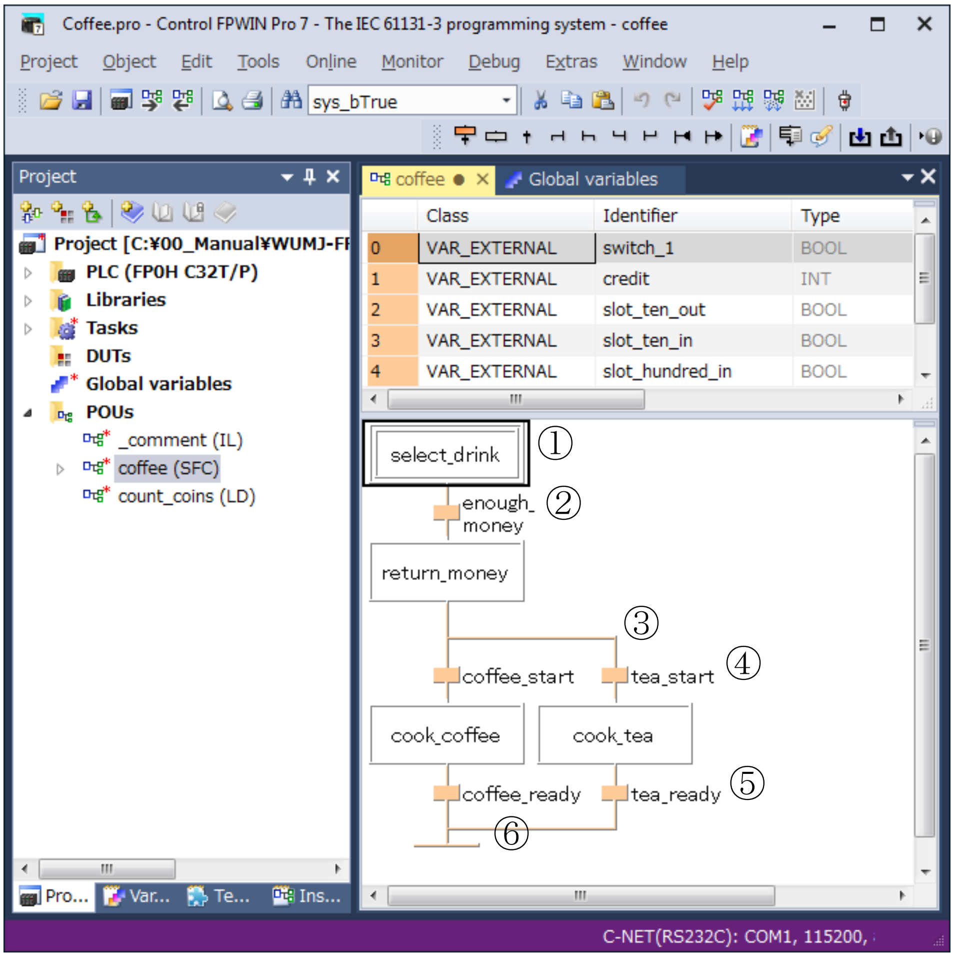
SFC enables a complex program to be handled as a simple structure. A task is divided into subtasks, and subtasks are further divided into steps. A task is executed step by step. Transition functions as a conditional jump between two steps, whether or not to activate the next step, depending on the status of the step (TRUE or FALSE). Each step consists of one or more actions.

◆ **NOTES**

Transition is divided by steps.

When programming is performed in SFC Editor, not only POUs, but also all actions and transitions are registered under "Actions" of Navigator.

Example:



	Class	Identifier	Type
0	VAR_EXTERNAL	switch_1	BOOL
1	VAR_EXTERNAL	credit	INT
2	VAR_EXTERNAL	slot_ten_out	BOOL
3	VAR_EXTERNAL	slot_ten_in	BOOL
4	VAR_EXTERNAL	slot_hundred_in	BOOL

SFC Program

①	Initial step
②	Transition
③	step
④	Divergence
⑤	Data unite
⑥	Final step

9.6 Check of Program

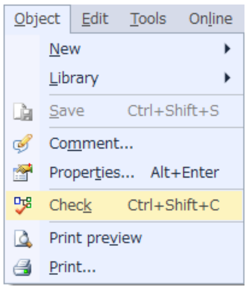
Proceed from the Menu Bar to "Object" -> "Check," or click , to check the program. Check is performed on the entire program defined through to the present point, for syntax errors or undeclared variables. Errors that are detected are displayed in an error list. Select an error in the list, and select [Display] or double-click the error. The program that contains the error is displayed, and the error part is highlighted. Proceed from the Menu Bar to "Extras" -> "Options" -> "Program options" -> "Editor," to set the highlight color.

◆ **NOTES**

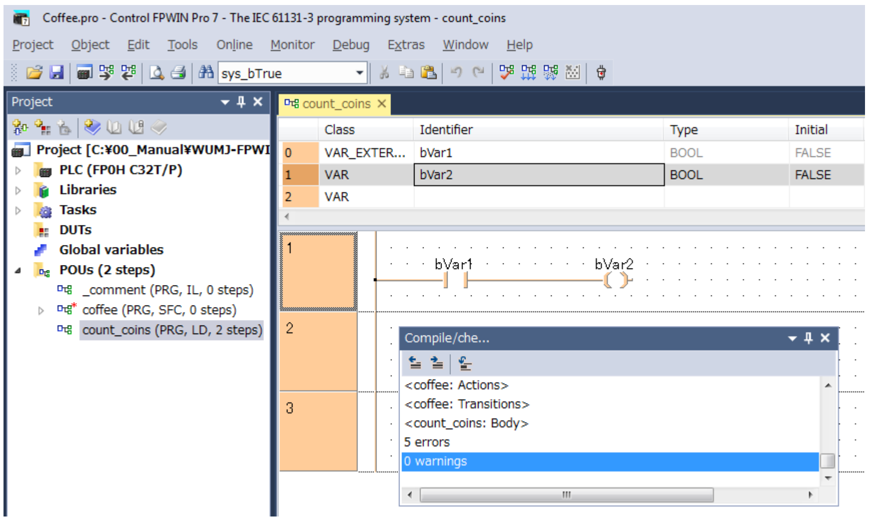
If multiple errors have occurred, correct the first error, and proceed from the Menu Bar to "Objects" -> "Check," or repeat . Other errors in the list may have resulted from the first error.

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to" Object" -> "Check," or click . Alternatively, in the context menu that is displayed by right-clicking, select "Check," and check it.**



2. The compile check window is opened.


Coffee.pro - Control FPWIN Pro 7 - The IEC 61131-3 programming system - count_coins
Project Object Edit Tools Online Monitor Debug Extras Window Help
sys_bTrue
Project
Project [C:¥00_Manual¥WUMJ-FPWI
PLC (FP0H C32T/P)
Libraries
Tasks
DUTs
Global variables
POUs (2 steps)
_comment (PRG, IL, 0 steps)
coffee (PRG, SFC, 0 steps)
count_coins (PRG, LD, 2 steps)
count_coins
bVar1
bVar2
1
2
3
Compile/che...

	Class	Identifier	Type	Initial
0	VAR_EXTER...	bVar1	BOOL	FALSE
1	VAR	bVar2	BOOL	FALSE
2	VAR			

```
<coffee: Actions>
<coffee: Transitions>
<count_coins: Body>
5 errors
0 warnings
```

A message is displayed whether or not the program is correct. If there is an error, such as double lines, the error that needs to be corrected is displayed. If more than one error has occurred, start with the first error to correct, because related errors may have also occurred.

10
Program Organization Unit (POU)

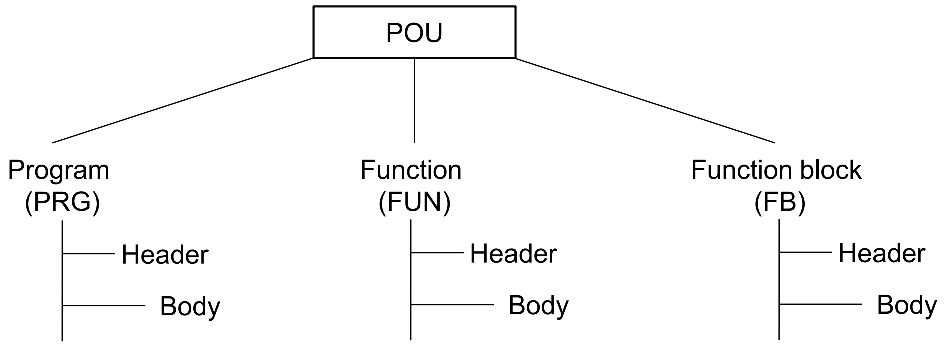
10.1 Overview

Program Organization Unit (POU) is an element that comprises program in FPWIN Pro7. POU contains program to control PLC. A program in FPWIN Pro7 consists of multiple sub-programs. Each sub-program is a self-completed program, and executes its own task. The type of POU to be specified varies by the task to be executed.

The name of POU is displayed in POU Pool of Navigator. In FPWIN Pro7, three types of POU (Program Class) are prepared.

- Program (PRG) (refer to "10.3 Program")
- Function (FUN) (refer to "10.4 Function ")
- Function block (FB) (refer to "10.5 Function Block")

Each POU consists of POU header and POU body, and is displayed in the programming window.



The two parts (header and body) declare variables in the list, based on the concept of IEC Standards to use declaration (symbol name) in place of physical address.

The advantage of this method is that the compiler manages addresses, and user only needs to change the address in POU header (as long as the variable is assigned to the address). It is not necessary to change the program itself.

Local variables are declared in POU header. Local variables can be used only in the POU body to which they belong.

●: Usable

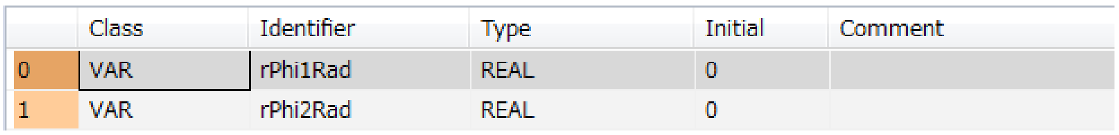
Editor	Program	Function block	Function
Instruction List (IL)	●	●	●
Ladder Diagram Program (LD)	●	●	●
Function Block Diagram (FBD)	●	●	●
Sequential Function Chart (SFC)	●		
Structured Text (ST)	●	●	●

10.2 List of local variables

Each POU has a POU header, which is a list of local variables. VAR_EXTERNAL refers to global variables in the list of local variables.

Each field in the list of variables respectively indicates the following.

① ② ③ ④ ⑤

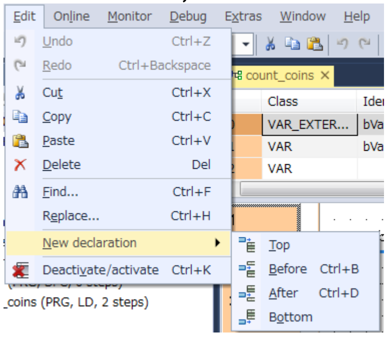


	Class	Identifier	Type	Initial	Comment
0	VAR	rPhi1Rad	REAL	0	
1	VAR	rPhi2Rad	REAL	0	

	Item name	Description
①	Class	Declares the class of the variable. Example: VAR,VAR_GLOBAL
②	Identifier	The symbol name used in the program. - The name of a variable cannot be started with a number. - For the name of a variable that uses a special syntax, the data type can be determined in accordance with the prefix table.
③	T New declaration ype	After registering an address, the data type is automatically selected beforehand. (Example: BOOL, INT, STRING) It is also possible to select another type from the list.
④	Initial	This specifies the default value to be assigned to a variable when PLC is started up. It can be changed when necessary.
⑤	Comment	This is explanation about the variable.

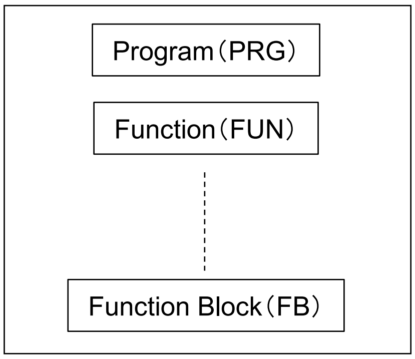
NOTES

- **The name of a variable cannot be started with a number.**
- **FP address (X0, Y1 , etc.) is a reserved word, and may not be used as a variable name.**
- **To insert a vacant line into the list, press <Shift>+<Return>.**
- **To insert a new line, proceed "Edit" -> "New declaration" -> "Top / Before / After / Bottom," or select either ⊒ or ⊒.**



10.3 Program

Program is at the highest layer in the hierarchy of POU. Functions and function blocks can be called from the program.



◆ **NOTES**

Program is called only from Task. Although program can call Function (FUN) and Function Block (FB), it is not possible to call program from Function (FUN) or Function Block (FB).

10.3.1 Processing of interrupt program

Interrupt program performs interrupt process on "Programs," the main program in the Task list.

◆ **NOTES**

- **If multiple interrupts occur at a time, the interrupt that has the smallest interrupt program number is prioritized.**
- **If a new interrupt occurs while an interrupt program is being executed, the interrupt program that has the highest priority is executed after the ongoing interrupt process is completed. In other words, an interrupt process cannot interrupt another interrupt process.**

10.4 Function

Function (FUN) is a Program Organization Unit (POU) that returns the value of an output variable of Class VAR_OUTPUT and VAR_IN_OUT-type, as the result of operation executed. If the output data type of Function is specified as VOID, the function does not return the result.

Function may also access global variables, via VAR_EXTERNAL, VAR_EXTERNAL_RETAIN, or VAR_EXTERNAL_CONSTANT.

Function does not have internal status information. Therefore, if Function is executed with the same value substituted into a variable of
VAR_INPUT, VAR_IN_OUT, VAR_EXTERNAL-type, the same value is always obtained for output variables of Class VAR_OUTPUT, VAR_IN_OUT, or VAR_EXTERNAL-type.

FPWIN Pro7 has two types of Functions.

1. System Instruction Functions in System Library
   - IEC Standard Library
   - FP Library
   - FP Tool Library
2. FP Pulsed Library: user-defined Functions

   The user may define original Functions and add them to the user library. The new Functions may be used in all projects into which the user library is installed.

   Functions can be written in the following programming languages.
   - Ladder Diagram (LD)
   - Function Block Diagram (FBD)
   - Structured Text (ST)
   - Instruction List (IL)

◆ **NOTES**

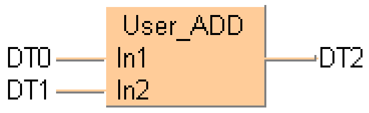
- **Function can be called from Program, Function, or Function Block, but cannot be registered in Task.**
- **Function cannot call the same function recursively.**
- **Function may be structured with the nesting of up to five layers.**
  Example: Fun1 (Fun2 (Fun3 (Fun4 (Fun5 (x)))))
- **Except ST or IL Function that has no formal parameter, Function result and Function output may remain unconnected.**
- **When a Function that has a formal parameter is called in ST Editor, the following conditions are applied.**
  - **In the case of argument that has a formal parameter, the order of parameters is not important.**
  - **For a user-defined function, EN input or EN output may be omitted. The omitted EN input is processed as "TRUE."**

10.4.1 User-defined Function

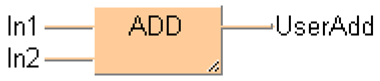
When compiled, a user-defined Function in the project or library is converted into a sub-routine. Whenever these are read out, the input parameters of Class VAR_INPUT and VAR_IN_OUT are transferred. Subsequently, the process jumps to the corresponding sub-routine program. In the end, the output parameters of Class VAR_OUTPUT and VAR_IN_OUT are read out. The number of sub-routines and the number of Functions that can be used vary by PLC model.

A user-defined Function is generated as one sub-routine. A user-defined Function can be called from any position in the program.

Example: when calling a user-defined Function "UserAdd" in Ladder Diagram



Content of user-defined Function



The following program codes are generated.

```
ST      R9010                           Transfer of input parameter
F0                  (*MV*)              R9010 Always = TRUE
        DT0                             DT0 -> UserAdd.In1

        DT550
F0                  (*MV*)
        DT1                             DT1 -> UserAdd.In2
        DT551
CALL    0                               Read of UserAdd subroutine
ST      R9010                           Re-read of output parameter
F0                  (*MV*)
        DT552
        DT2                             UserAdd -> DT2
...
ED                                      End of main program
SUB     0                               UserAdd subroutine
ST      R9010
F22                 (*PLUS_S*)

        DT550
        DT551
        DT552                           if (TRUE)
                                        UserAdd=UserAdd.In1 + UserAdd.In2
RET                                     Return to main program
```

10.4.2 Enable Input and Enable Output

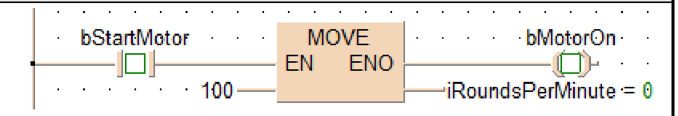
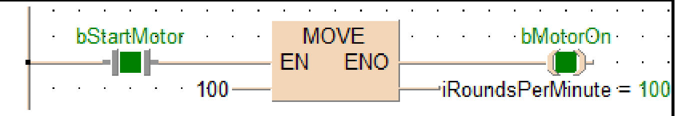
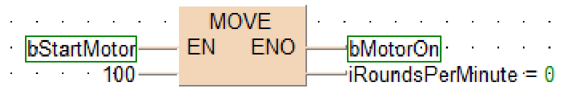
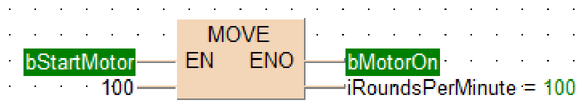
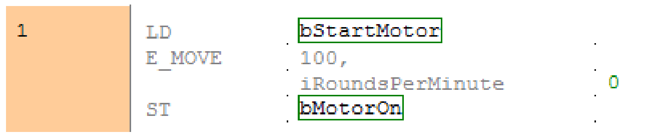
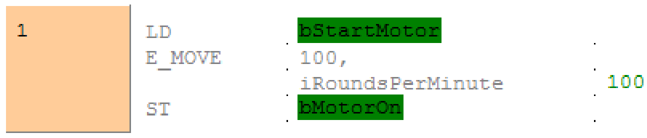
In FBD, LD or IL, a program that has execution conditions can be written, using Functions and Function Blocks that have EN/ENO. EN represents Enable Input, and ENO represents ENABLE Output.

ST Editor has no functions or function blocks that have EN/ENO. To control the execution of functions and function blocks, use IF statements or CASE statements instead.

Both EN and ENO can be used for all Functions and Function Blocks of the IEC specification.

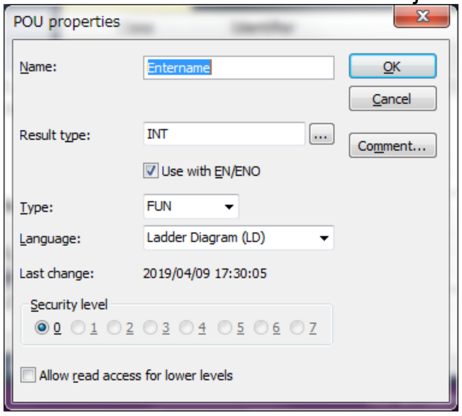
BOOL-type input variables or the BOOL results of a logical expression can be connected to EN Input.

- If EN Input is TRUE
  The Function or Function Block is processed. After normal execution, the corresponding ENO Output is set to TRUE. The ENO Input of the next Function or Function Block that is connected to this EN Output is processed if this is TRUE.
- If EN Input is FALSE
  The Function or Function Block is not processed. The Function result is not changed, and the value of the connected variable is retained.

Programming language		
LD	EN = FALSE T ENO = FALSE	 bStartMotor MOVE EN ENO bMotorOn 100 iRoundsPerMinute = 0
	EN = TRUE T ENO = TRUE	 bStartMotor MOVE EN ENO bMotorOn 100 iRoundsPerMinute = 100
FBD	EN = FALSE T ENO = FALSE	 MOVE bStartMotor EN ENO bMotorOn 100 iRoundsPerMinute = 0
	EN = TRUE T ENO = TRUE	 MOVE bStartMotor EN ENO bMotorOn 100 iRoundsPerMinute = 100
IL	EN = FALSE T ENO = FALSE	1 LD bStartMotor E_MOVE 100, iRoundsPerMinute 0 ST bMotorOn
	EN = TRUE T ENO = TRUE	1 LD bStartMotor E_MOVE 100, iRoundsPerMinute 100 ST bMotorOn

User-defined Functions and Function Blocks can be created with or without EN/ENO. These can be selected in the following methods.

- When a new POU is created, select "Use with EN/ENO." (Proceed from the Menu Bar to "Object" -> "New" -> "POU," or click .)
- Proceed from the Menu Bar to "Object" -> "Properties," and select "Use with EN/ENO."


POU properties

Name: Entername

Result type: INT

☑ Use with EN/ENO

Type: FUN

Language: Ladder Diagram (LD)

Last change: 2019/04/09 17:30:05

Security level: [✓ 0] 1 2 3 4 5 6 7

☐ Allow read access for lower levels

- When inserting an instruction, press the [With EN/ENO] button.

■ Value of ENO Output

- If ENO Output is not explicitly set in the body of user-defined Function or Function Block, its value becomes the same as that of EN Input.
- If ENO Output is set to FALSE in the body of user-defined Function or Function Block, the value of the output variable is not transferred to Output.

◆ **NOTES**

Set EN to TRUE before the processing of POU. If the processing of POU is normally executed, the corresponding ENO Output is set to "TRUE." If ENO is not set, an error may have occurred.

10.5 Function Block

A Function Block (FB) is a small program. Unlike Function (FUN), Function Block (FB) has a memory area to store a value. If an operation is executed, such as addition or subtraction, different results can be obtained for the same input value, depending on the stored data. Function Block (FB) can be used as many times as needed in a single program. Every time Function Block (FB) is called during programming, its copy is created. By naming this copy ("instance"), it can be avoided that the value is overwritten or operated by Function Block (FB) of the same type.

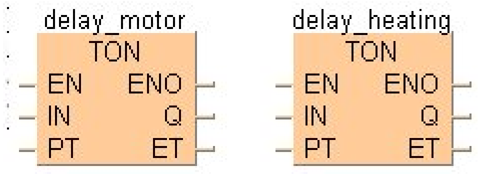
FPWIN Pro7 has two types of Function Blocks.

1. System function block
   - IEC Standard Library: TON, TOF, CTD, etc.
   - FP Library: CT_FB, PID_FB, IsReceptionDoneByTimeOut, etc.
2. User-defined function block

   Function block (FB) can be written in four programming languages.
   - Ladder Diagram (LD)
   - Function Block Diagram (FBD)
   - Structured Text (ST)
   - Instruction List (IL)

Example:

For example, if Function Block (FB) TON (Timer On-Delay) in the IEC Standard Library is used, it is possible to rename and use this Function Block (FB) "delay_motor," etc. Similarly, if this Function Block is called on a different occasion, it can be registered with another name, such as "delay_heating."



◆ **NOTES**

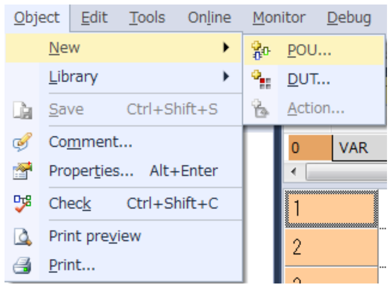
- **Function Block can be called only from Program or Function Block, and cannot be assigned to Task.**
- **Function Block cannot be called from the same Function Block recursively.**
- **Function Block may be structured with the nesting of up to five layers.
Example: FB1 (FB2 (FB3 (FB4 (FB5 (x)))))**
- **For one Function Block, up to 40 input/output variables can be defined.**

10.6 Create a New POU

The procedures for creating a new POU of PRG, FUN, or FB are mostly the same.

◆ **PROCEDURE**

1. **Proceed from the Menu Bar to "Object" -> "New" -> "POU," or click .**



2. **Enter the name.**
3. **Select a POU type.**

   Select Program (PRG), Function (FUN), or Function Block (FB).

Data type	Description	
Program (PRG)	Program Task	Select a Task to which POUs will be assigned.
Function (FUN)	EN/ENO	Specify whether or not to attach an EN/ENO pin to Function.
	Output data type	Specify the output data type that returns the function value.
Function Block (FB)	EN/ENO	Select whether or not to attach an EN/ENO pin to the function block.

4. **Select a programming language.**
5. **Press the [OK] button.**

   The POU name and the program are automatically registered in the Project Navigator. POU header and POU body are laid out in a common programming window. By dragging the splitter, the respective sizes of panes can be adjusted.

   The header and body of POU are displayed. POU header and POU body are laid out in a common programming window. By dragging the splitter, the respective sizes of panes can be adjusted.

   To display detailed information in the"Project" pane, select an item under "Display" in the context menu that is displayed by right-clicking, or select "Display detailed information" in the Navigator option.

11
Variables and Data Types

11.1 Types of Variables

"Variables" refers to the symbol names for input/output contacts and memory areas in PLC. These symbol names can be used in the program in place of explicit user addresses.

◆ **NOTES**

Do not use an explicit user address. It is recommended to use a global variable or system variable.

Global variables and local variables can be used in separation.

- Global variables are effective throughout the project. Global variables refers to, for example, contacts between process inputs and outputs, and are used for operation terminals and displays. Global variables are declared in the list of global variables.
- Local variables are used as places for saving intermediate results in any program. Local variables are declared in the relevant POU header.

Once declared, variables can be changed, directly assigned to inputs/outputs in the program, or assigned to variable groups such as ARRAY or Data Unit Type (DUT). It is also possible to create a recipe in the case of complex control procedure.

By using a cross reference list, it is possible to identify all variables in the program and their statuses of use.

11.1.1 Global variable

Global variables are effective throughout the project. Global variables refers to, for example, contacts between process inputs and outputs, and are used for operation terminals and displays. Global variables are declared in the list of global variables. The list of global variables can be accessed from the "Project" pane or from the "Variable" pane.

Global variables can be declared in various classes. Global variables can be referred to by POU via VAR_EXTERNAL (external variable).

The following variables need to be declared on the list of global variables.

- Variables that are assigned to the input/output of PLC (X0, Y0, etc.)
- For example, variables that need to be assigned to specific addresses (such as DT0), when data is to be exchanged with a programmable display, or for other purposes
- Variables that are referred to as external variables (VAR_EXTERNAL) by POU

◆ **NOTES**

- **Do not directly assign an address unless necessary. Use this when it is necessary to access the input/output (X, Y) of PLC, or a specific memory area.**
- **The compiler assigns an address automatically. This prevents an error caused by double assignment, and enables the automatic refreshment of addresses in the case of change to the PLC model.**
- **Proceed from the Menu Bar to "Extras" -> "Options" -> "Compile options" -> "Code generation," and turn on the check box of "Retain the value of global variable (hold-type) to which an explicit address is assigned," to prevent the initialization of variables to which the user has directly assigned addresses.**
- **To structure a program, it is recommended to declare all global variables in the list of global variables before programming.**

For more details, refer to the online Help.

11.1.2 Local variable

Local variables are used as places for saving intermediate results in any program. Local variables are declared in the relevant POU header.

Local variables can be used only in the body of POU in which they are declared. It is not possible to accessed them from other POUs. In POU header, variables that are copied from the list of global variables are separated from variables that are declared specifically for the POU.

Local variables can be declared in various classes. External variables (VAR_EXTERNAL) refer to global variables, and global variables are also declared in the list of local variables.

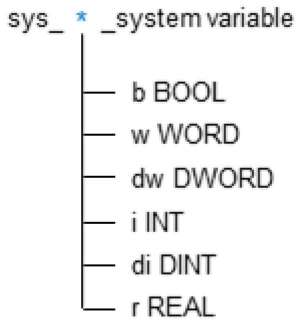
◆ **NOTES**

Proceed from the Menu Bar to "Extras" -> "Options" -> "Compile options" -> "Address range," to specify the address area to be used by the compiler for local variables. Other address areas are used for global variables.

11.1.3 System variables

Use system variables that are prepared in each PLC to read or write special data registers and special internal relays. In the "Variable" dialog box, system variables can be directly inserted into POU body.

- As specified in the prefix table, system variables are prepared for special data registers and special internal relays, in accordance with the following syntax.



- System constants written in upper case can also be usd.
  Example: SYS_CONSTANT, such as SYS_MEMORY_AREA_DT

System variables can be monitored by proceeding from "Monitor" -> "Special relays / special data registers." The name of a system variable is displayed to the extreme right of Comment.

For more details concerning system variables, refer to the Online Help.

11.2 Variable class

Data type	Class	List of global variables	List of local variables			Definition
			Program	FUN	FB	
Global variable	VAR_GLOBAL	●				Global variables (non-hold type)
	VAR_GLOBAL_RETAIN	●				Global variables (hold type)
	VAR_GLOBAL_CONSTANT	●				Global variables (constant)
	VAR_EXTERNAL		●	●	●	Global variables (non-hold type)
	VAR_EXTERNAL_RETAIN		●	●	●	Global variables (hold type)
	VAR_EXTERNAL_CONSTANT		●	●	●	Global variables (constant)
Local variable	VAR		●	●	●	Local variables (non-hold type)
	VAR_RETAIN		●		●	Local variables (hold type)
	VAR_CONSTANT		●	●	●	Local variable (constant)
	VAR_INPUT			●	●	Input variables
	VAR_IN OUT			●	●	Input/output variables
	VAR_OUTPUT			●	●	Output variables
	VAR_OUTPUT_RETAIN				●	Output variables (hold type)

For more details, refer to the Online Help.

11.3 Variable name

Each variable declared in FPWIN Pro7 must have a unique name. By using a meaningful variable name, it becomes easy to maintain the program, and to save time for error detection.

◆ **NOTES**

- **Special characters cannot be used. Example: !, ", $, %, parentheses, etc.**
- **A number cannot be used for the starting character.**
- **An underscore cannot be used for the ending character. More than one underscores cannot be used consecutively.**
- **It is not possible to use an address only for the variable name. Example: R0', 'DT0', 'LD0'**
  It is not possible to use a key word for a variable name. Example: ADD', 'ARRAY', 'INT'

- To increment a variable name by one, proceed from the Menu Bar to "Extras" -> "Option" -> "Program option" -> "Editor" -> "Declaration editor," and enable "Automatically increment variable name when a new declaration is inserted."

  Example:
  Enter "bMotor_on1" as the first variable name.
  Select [ ] and add a new line.
  FPWIN Pro7 automatically inserts "bMotor_on2" as the next variable name.

- It is possible to determine the data type at the time of declaration, based on the prefix of the variable name. For example, bVar1 sets BOOL type to declaration. Proceed from the Menu Bar to "Extras" -> "Option" -> "Program option" -> "Editor" -> "Declaration editor" -> "Prefix," and enable "Auto complete declaration using prefix of variable name."

  Example:
  Enter "iMaximumValue" as the variable name.
  Press the <Tab> key.
  FPWIN Pro7 automatically inserts data type INT in accordance with the prefix table.

For more details, refer to the online Help.

11.4 Data type

11.4.1 Array

"ARRAY" is a group of variables that combines elements of the same data type as consecutive data blocks. This group of variables is handled and declared as a single variable. In a program, it is possible to designate the entire array, or individual array elements.

Declaration

To declare an ARRAY-type variable in POU header, use the following syntax.

ARRAY[A...B,C...D,E...F] OF <Data type>

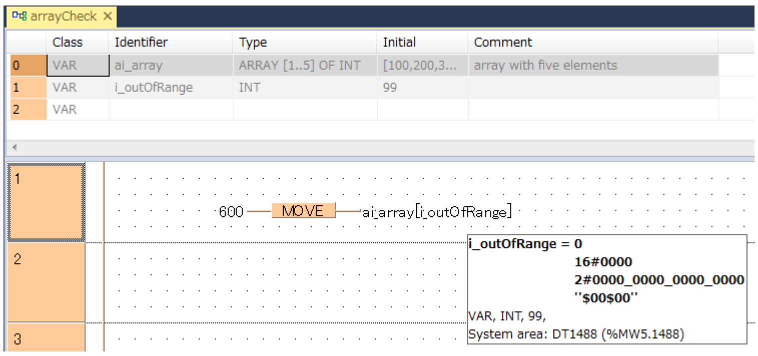
A=	Index of the first element	1-D
B=	Index of the final element	
C=	Index of the first element	2-D (Option)
D=	Index of the final element	
E=	Index of the first element	3-D (Option)
F=	Index of the final element	

Array can be declared as an array variable of 1-D, 2-D, or 3-D. Array has a field for each dimension. The index of element is a positive or negative integer. The first element needs to be smaller than the last element.

NOTES

- **Array may not be used as a variable of another array.**
- **When the index of array is accessed, FPWIN Pro7 does not check the index by matching it with the border of array. Confirm that the index is within the range defined in the POU header.**

Example: ARRAY [1..5] OF INT
In this example, ai_array[99] is outside the range, but no error message is generated.



	Class	Identifier	Type	Initial	Comment
0	VAR	ai_array	ARRAY [1..5] OF INT	[100,200,3...	array with five elements
1	VAR	i_outOfRange	INT	99	
2	VAR				

```
600 ── MOVE ── ai_array[i_outOfRange]
```

```
i_outOfRange = 0
               16#0000
               2#0000_0000_0000_0000
               "$00$00"
VAR, INT, 99,
System area: DT1488 (%MW5.1488)
```

Data types that can be used as array:

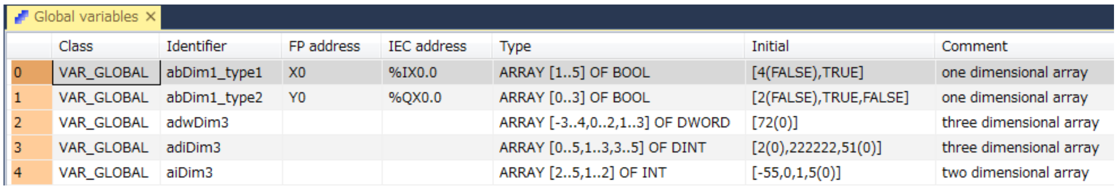
BOOL	BOOL-type variables become active or inactive, depending on the related action qualifier. Default behavior (a status without a qualifier) is N (non-hold).
DATE	Data type DATE is a literal that represents date. The range that can be handled by DATE type is from D#2001-01-01 to D#2099-12-31.
DATE_AND_TIME	Data type DATE_AND_TIME is a literal that represents date and time. The range that can be handled by DATE_AND_TIME type is from DT#2001-01-01-00:00:00 to DT#2099-12-31-23:59:59.
DINT	The values of DOUBLE INTEGER-type variables are integers without decimal points. The range that can be handled by DOUBLE INTEGER type is from -2147483648 to 2147483647.
DWORD	A DOUBLE WORD-type variable consists of 32-bit binary data. It handles the statuses of 32 inputs/outputs in the unit of two words (DOUBLE WORD).
INT	The values of INTEGER-type variables are integers without decimal points. The range that can be handled by INTEGER type is from -32768 to 32767.
REAL	The values of REAL-type variables are floating-point real numbers. The number of significant digits of mantissa is seven. The mantissa has 23 bits, and the exponent has 8 bits. (Based on IEEE754)
STRING	Data type STRING consists of ASCII strings of up to 32767 characters. The maximum number of characters is determined by the memory size of PLC.
TIME	Data type TIME is a literal that represents time length. The range that can be handled by TIME type is from T#0.01 to T#21,474,836.47S in a 32-bit support PLC, and from T#0.01 to T#327.67S in a 16-bit support PLC.
TIME_OF_DAY	Data type TIME_OF_DAY is a literal that represents the time of day. The range that can be handled by TIME_OF_DAY type is from TOD#00:00:00 to TOD#23:59:59.
UDINT	UDINT type is unsigned double-word data, and is an integer without a decimal point. The range that can be handled by UDINT type is from 0 to 4294967295.
UINT	UINT type is a variable of unsigned INTEGER data type, and is an integer without a decimal point. The range of UINT-type values is from 0 to 65535.
WORD	A WORD-type variable consists of 16-bit binary data. The statuses of 16 inputs/outputs are handled in the unit of one word (WORD). The default value of this variable is "0."

◆ **NOTES**

- **Do not directly assign an address unless necessary. Use this when it is necessary to access the input/output (X, Y) of PLC, or a specific memory area.**
- **The compiler assigns an address automatically. This prevents an error caused by double assignment, and enables the automatic refreshment of addresses in the case of change to the PLC model.**

Example 1: individual arrays that are declared in the list of global variables

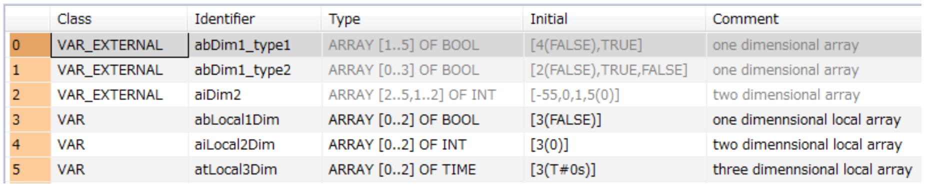
For example, abDim1_type1 is a 1-D array that has five elements. The first element is accessed by using Index 2(abDim1_type1[1]), and the last element is accessed by using Index 5.



Global variables

	Class	Identifier	FP address	IEC address	Type	Initial	Comment
0	VAR_GLOBAL	abDim1_type1	X0	%IX0.0	ARRAY [1..5] OF BOOL	[4(FALSE),TRUE]	one dimensional array
1	VAR_GLOBAL	abDim1_type2	Y0	%QX0.0	ARRAY [0..3] OF BOOL	[2(FALSE),TRUE,FALSE]	one dimensional array
2	VAR_GLOBAL	adwDim3			ARRAY [-3..4,0..2,1..3] OF DWORD	[72(0)]	three dimensional array
3	VAR_GLOBAL	adiDim3			ARRAY [0..5,1..3,3..5] OF DINT	[2(0),222222,51(0)]	three dimensional array
4	VAR_GLOBAL	aiDim3			ARRAY [2..5,1..2] OF INT	[-55,0,1,5(0)]	two dimensional array

Example 2: individual arrays that are declared in the POU header



	Class	Identifier	Type	Initial	Comment
0	VAR_EXTERNAL	abDim1_type1	ARRAY [1..5] OF BOOL	[4(FALSE),TRUE]	one dimensional array
1	VAR_EXTERNAL	abDim1_type2	ARRAY [0..3] OF BOOL	[2(FALSE),TRUE,FALSE]	one dimensional array
2	VAR_EXTERNAL	aiDim2	ARRAY [2..5,1..2] OF INT	[-55,0,1,5(0)]	two dimensional array
3	VAR	abLocal1Dim	ARRAY [0..2] OF BOOL	[3(FALSE)]	one dimennsional local array
4	VAR	aiLocal2Dim	ARRAY [0..2] OF INT	[3(0)]	two dimennsional local array
5	VAR	atLocal3Dim	ARRAY [0..2] OF TIME	[3(T#0s)]	three dimennsional local array

For more details concerning arrays, refer to the Online Help or the Programming Manual.

11.4.2 DUT

DUT (Data Unit Type) consists of several different basic data types.

For more details concerning DUTs, refer to the Online Help or the Programming Manual.

11.5 Use of Variables

11.5.1 Declaration of global variables

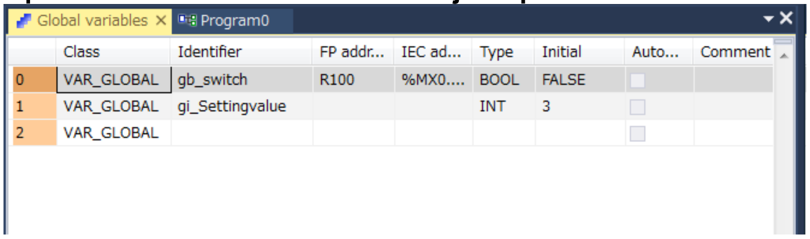
To declare variables using Declaration editor, take the following procedure. Variables can be declared easily in the POU body (by using LD/FBD, ST/IL, or "Variable" pane).

◆ **NOTES**

- **Do not directly assign an address unless necessary. Use this when it is necessary to access the input/output (X, Y) of PLC, or a specific memory area.**
- **The compiler assigns an address automatically. This prevents an error caused by double assignment, and enables the automatic refreshment of addresses in the case of change to the PLC model.**
- **To structure a program, it is recommended to declare all global variables in the list of global variables before programming.**

◆ **PROCEDURE**

1. **Open "Global variables" in the "Project" pane.**



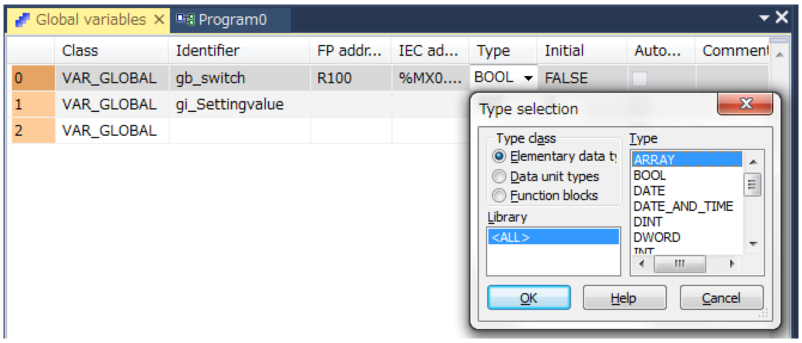
	Class	Identifier	FP addr...	IEC ad...	Type	Initial	Auto...	Comment
0	VAR_GLOBAL	gb_switch	R100	%MX0....	BOOL	FALSE	☐	
1	VAR_GLOBAL	gi_Settingvalue			INT	3	☐	
2	VAR_GLOBAL						☐	

2. **Under "Class," select a variable class.**
3. **Enter a name in the "Variable name."**

   Only in the case of input/output of PLC, or in the case of variables that need to be assigned to specific addresses for the purpose of data exchange concerning operating equipment, take the following two procedures.
   Click here to check the method for automatically assigning, or shifting, the addresses of global variables.
4. **Enter an address in the FP format. Example: X0**

   When you move to the next field, the corresponding IEC address is automatically generated.
   Alternatively, enter the address into the IEC format. Example: %IX0.0
   The respective symbols indicate I = Input, X = Bit, and 0.0 = First input contact of first unit. When you move to the next field, the corresponding IEC address is automatically generated.

5. **Select under "Data type," and open the "Type selection" dialog box.**



	Class	Identifier	FP addr...	IEC ad...	Type	Initial	Auto...	Comment
0	VAR_GLOBAL	gb_switch	R100	%MX0....	BOOL	FALSE		
1	VAR_GLOBAL	gi_Settingvalue						
2	VAR_GLOBAL							

Type selection

Type class: [✓ Elementary data ty] Data unit types, Function blocks

Library: <ALL>

Type: ARRAY, BOOL, DATE, DATE_AND_TIME, DINT, DWORD, INT

- Under "Type class," select a data type.
Elementary data types
Data unit types (DUT)
Function blocks

- From "Library," select a library.

6. **After the setting of the "Type selection" dialog box is completed, press the [OK] button.**

   The default value of the selected data type is automatically displayed. This default value can be changed when necessary.

7. **Set "Automatic declaration of external variable" and "Comment" when necessary.**

   - Select "Automatic declaration of external variable," for automatic insertion into all POU headers.
   - Enter text into the "Comment" field when necessary.

For more details, refer to the online Help.

11.5.2 Automatic assignment of address

Perform automatic assignment using the context menu, or shift the user address in the list of global variables. The command of context menu becomes usable only after the variable class, variable name, and variable type have been entered.

◆ **PROCEDURE**

1. **Select one or more lines in the list of global variables.**
2. **In the context menu, select required operation.**

   All changes are displayed in the dialog box.

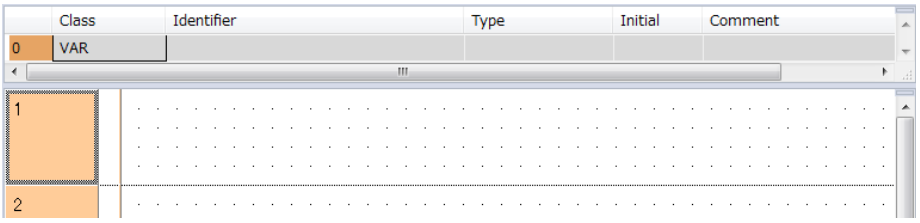
For more details, refer to the online Help.

11.5.3 Declaration of local variables

To declare variables using Declaration editor, take the following procedure. Variables can be declared easily in the POU body (by using LD/FBD, ST/IL, or "Variable" pane).

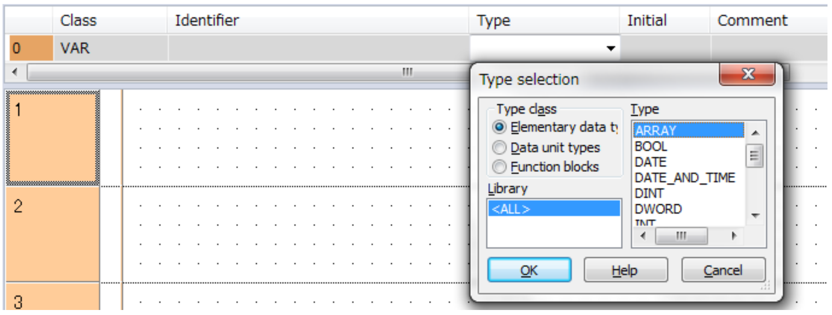
◆ **PROCEDURE**

1. **Open POU Header in the "Project" pane.**



	Class	Identifier	Type	Initial	Comment
0	VAR				

2. **Select a variable class under "Class."**
3. **Enter a name in the "Identifier."**
4. **Select under "Type," and open the "Type selection" dialog box.**



	Class	Identifier	Type	Initial	Comment
0	VAR				

- Under "Type class," select a data type.
Elementary data types
Data unit types (DUT)
Function blocks

- From "Library," select a library.

5. **After the setting of the "Type selection" dialog box is completed, press the [OK] button.**

   The default value of the selected data type is automatically displayed. This default value can be changed when necessary.

6. **Enter text into "Comment" when necessary.**

For more details, refer to the online Help.

12
Glossary

12.1 Glossary

Accumulator

Accumulator stores the intermediate results of IL Instructions. The respective results of instructions are immediately stored following their execution. No input conditions are required for executing the next instructions. The next processes are performed in accordance with the values of Accumulator. The content of Accumulator is overwritten when the subsequent network is processed. Therefore, if it is necessary to call the value later in a different network, it is required to store the value as a variable.

Action Assignment

Action is a sequence created by SFC Editor, and refers to a part of logic that is executed when a step becomes active. Action contains all kinds of logics. Action can be assigned to multiple steps, and can be created by Function Block Diagram (FBD), Ladder Diagram (LD), Instruction List (IL), or Structured Text (ST).

Array

Array refers to an aggregation of data that consecutively lists variables of the same data type. Array becomes a variable itself, and is declared.

Reference:

Array
1-D array
2-D array
3-D array

Compile

When a project is compiled, FPWIN Pro7 converts it into program code that can be read by PLC.

Data Type

Data types in FPWIN Pro7 are categorized into basic data types and user-defined data types. Basic data types are as indicated below.
BOOL, INT, DINT, WORD, DWORD, REAL, TIME, STRING

Declaration

Declaration refers to the definition of a global variable or a local variable.

DUT

DUT stands for "Data Unit Type," which consists of variables that have multiple different data types. Define DUT at first, and subsequently use it in the list of global variables or in the POU header, just as standard data types (BOOL, INT, etc.).

Reference:
Defined System DUT
DUT (DUT that has overlapping elements)

DUT Pool

DUT Pool is placed in Project Navigator, and all DUTs are registered in it. DUT is defined by the user.

Reference:
Defined System DUT
DUT (DUT that has overlapping elements)

F-Instructions

F Instructions are the basic set of instructions for FP-series PLC. F Instructions are executed when the value of EN Input (Enable Input) is "TRUE."

Function Block Diagram(FBD)

Function Block Diagram (FBD) is a graphic language that creates program by connecting logics. Variables of individual POUs are connected as the inputs and outputs of Function Box. This connection indicates the flow of data among variables and the inputs/outputs of Function Box.

Program of Function Block Diagram is structured in a network.

A network of Function Block Diagram is defined by the connection diagram of Function Box.

Function Block

Function Block defines both algorithm and data of a user program. This definition enables the examination of Class. Function Block does not execute process by itself. Instead, several instances of Function Block are created, and these instances are used individually. Individual instances have their respective memories to store data required to execute the functions of Function Block.

The content of the respective memories of Function Block that individual instances have are retained until the relevant instances are executed the next time. These internal memories have effect on the execution of functions that are added, depending on the usage of Function Block.

This means that, even if Function Block instances are executed with the same input variable value, the obtained results are not necessarily the same.

Unlike Function, multiple outputs can be defined for Function Block.

Instances of Function Block can be declared as local variables to be used in a POU. To declare a Function Block instance in a POU means to define the range of use of this instance at the same time.

Function Block Instance

Objects of Function Block have respective data memories of Function Block. These respective data areas are associated to algorithms of Function Block by individual instances.

Function

Function (FUN) is a logic in which the same result is always obtained when it is executed. Function can be defined and used when necessary. Because Function does not access the internal memory, the same operation result is obtained every time when the input parameter is the same. Immediately after Function is defined, it can be called and used from any POU in the user program.

Global Variable

Global variables declare physical addresses in the list of global variables. Global variables can be accessed from all POUs.

Identifier

Identifier (variable name) is the symbol name of a variable.

IEC 60870 International Standard

This standard provides a communication profile for sending basic telecontrol messages between a central telecontrol station and a telecontrol outstation, which uses permanent directly connected data circuits between the central station and the individual outstations.

Input Variable

Input variables are entered as variables required for the operation of Function or Function Block.

Instruction List (IL)

Instruction List (IL) is a text-type programming language that is suitable for the programming of PLC. Instruction List consists of individual instructions, each of which performs one process. In addition to variables that are declared as arguments for operation, the current values of Accumulator are used as implicit factors.

After the execution of instruction, the operation result is stored in Accumulator, and is called when the subsequent instruction is processed.

Instruction List is structured as an aggregation of networks.

Ladder Diagram (LD)

Ladder Diagram (LD) is a graphic language that creates program by connecting logics. Just as Function Block Diagram, variables of individual POUs are connected as the inputs and outputs of Function Box. In addition, it is possible to write BOOL-type connection using coils and contacts. Ladder Diagram expresses the flow of BOOL-type signals.

Program of Ladder Diagram is structured in a network.

A network created by Ladder Diagram is defined by connecting Function Box with the bus line on the left as the start point.

Library Pool

FPWIN Pro7 has four types of libraries. For more details, refer to the Online Help.

Local Variable

Local variables are variables that are effective only in the POU that is defined in the header.

Logic

A program of PLC is defined by the user. The user logic is structured in POU.

Network

Network exists in the POU body, and program is written inside it.

Object

In FPWIN Pro7, all components that are displayed in Project Navigator are called "Objects."

Online

"Online" is a status when PC and PLC are communicating. It is also possible to perform programming in the online status.
Warning! Changing a program in the online status may cause human injury or damage to machinery. Take adequate precautions and perform such changes carefully.

The number of pulses that is set as the target value is outputted. Rotation is in the normal direction if the value is positive, and in the reverse direction if the value is negative.

The number of pulses as the difference between the preset target value and the current value is outputted. The rotation is in the normal direction if the value is larger than the current value, and in the reverse direction if the value is smaller than the current value.

Output Variable

Functions and Function Blocks write operation results into output variables.

P-Instructions

P Instructions are the set of instructions of the FP-series PLC. They are executed only when the execution condition detects leading edge differential (edge detection), and perform the same processes as F Instructions.

Program Organization Unit (POU)

POU is the most basic unit that comprises a user program. Individual POUs can also call other POUs. Note that a cyclic structure cannot be specified.

There are POUs that are defined as standards by default, and POUs that are defined by the user.

In FPWIN Pro7, POU consists of POU header that declares variables, and body in which the content of algorithms of POU is written.
Different types of POUs are prepared to enable a variety of programming.
There are POU types: Program (PRG), Function (FUN), and Function Block (FB).

Program

Program is similar to Function Block that has Function Block instances.

Difference between Program and Function Block:

- Program is positioned at the top of POU, and cannot be called from other POUs.
- When a program is defined, it can be directly specified using input/output variables and other physical addresses.

Project Navigator

Project Navigator displays the structure of Project and all objects contained in it. Example) Library, PLC environment, Task, POU, etc.

Project

Project is positioned at the highest layer in FPWIN Pro7. Project contains all information required by PLC.

RPI (Requested Packet Interval)

RPI: constant cycle

Set the transmission interval for the cyclic communication. Set a value within the communication capacity of the adapter.
The usable RPI range depends on devices.
In the case of FP7, the range is from 0.5 ms to 10 s (in the unit of 0.5 ms).

RTU

Normally an RTU (Remote Terminal Unit or Remote Telemetry Unit) is a substation with little or no PLC functionality. Our PLCs with the IEC 60870-Communicator combine the RTU function and the PLC function to become an intelligent outstation.

Sequential Function Chart (SFC)

SFC is structured by the combination of Steps and Transitions. Conditions for transition from one step to the next step, while one step is being executed, are set in Transition.
By using parallel branches and selection branches, a wide variety of SFC sequences can be created.
By using Ladder Diagram (LD), Function Block Diagram (FBD), Instruction List (IL), and other appropriate programming languages, steps that are created can be graphically connected to perform programming.

Structured Text

This is a text language where general structures can be used. ST is a high-level language that can describe complex programs and control structures. It can be used in all PLCs, and the amount of resources that are used does not change, such as steps, labels, and calls, compared to other programming languages.

Supervisory Control System

This system can be used to control large water and sewage treatment plants, gas and energy suppliers.

Task

Tasks define the order of execution of the program. POU contains the logic that defines what to do. By assigning the program to tasks, the order of execution is defined.

Variable

Variables are the symbol names that associate inputs and outputs to the internal memory of PLC. Variables are categorized into global variables and local variables.

Action Pool

The Action Pool is displayed under POU in the "Project" pane, when a program is created using SFC. All actions that are programmed in this POU are registered in the Action Pool.

Record of changes

Manual No.	Date	Record of Changes
WUME-FPWINPRO7-01	Jun. 2019	First Edition
WUME-FPWINPRO7-02	Feb. 2023	2nd Edition Supported OSs reviewed.
WUME-FPWINPRO7-02	Apr. 2024	3rd Edition Change in Corporate name.

Panasonic Industry Co., Ltd.

1006, Oaza Kadoma, Kadoma-shi, Osaka 571-8506, Japan
https://industry.panasonic.com/

Please visit our website for inquiries and about our sales network.

April, 2024 WUME-FPWINPRO7-03