Smarter. Greener. Together.

Industrial Automation Headquarters

Delta Electronics, Inc.
Taoyuan Technology Center
No.18, Xinglong Rd., Taoyuan City,
Taoyuan County 33068, Taiwan
TEL: 886-3-362-6301 / FAX: 886-3-371-6301

Asia

Delta Electronics (Jiangsu) Ltd.
Wujiang Plant 3
1688 Jiangxing East Road,
Wujiang Economic Development Zone
Wujiang City, Jiang Su Province,
People's Republic of China (Post code: 215200)
TEL: 86-512-6340-3008 / FAX: 86-769-6340-7290

Delta Greentech (China) Co., Ltd.
238 Min-Xia Road, Pudong District,
ShangHai, P.R.C.
Post code : 201209
TEL: 86-21-58635678 / FAX: 86-21-58630003

Delta Electronics (Japan), Inc.
Tokyo Office
2-1-14 Minato-ku Shibadaimon,
Tokyo 105-0012, Japan
TEL: 81-3-5733-1111 / FAX: 81-3-5733-1211

Delta Electronics (Korea), Inc.
1511, Byucksan Digital Valley 6-cha, Gasan-dong,
Geumcheon-gu, Seoul, Korea, 153-704
TEL: 82-2-515-5303 / FAX: 82-2-515-5302

Delta Electronics Int'l (S) Pte Ltd
4 Kaki Bukit Ave 1, #05-05, Singapore 417939
TEL: 65-6747-5155 / FAX: 65-6744-9228

Delta Electronics (India) Pvt. Ltd.
Plot No 43 Sector 35, HSIIDC
Gurgaon, PIN 122001, Haryana, India
TEL : 91-124-4874900 / FAX : 91-124-4874945

Americas

Delta Products Corporation (USA)
Raleigh Office
P.O. Box 12173,5101 Davis Drive,
Research Triangle Park, NC 27709, U.S.A.
TEL: 1-919-767-3800 / FAX: 1-919-767-8080

Delta Greentech (Brasil) S.A
Sao Paulo Office
Rua Itapeva, 26 - 3° andar Edificio Itapeva One-Bela Vista
01332-000-São Paulo-SP-Brazil
TEL: +55 11 3568-3855 / FAX: +55 11 3568-3865

Europe

Deltronics (The Netherlands) B.V.
Eindhoven Office
De Witbogt 20, 5652 AG Eindhoven, The Netherlands
TEL: 31-40-2592850 / FAX: 31-40-2592851

DVP-4949220-03

*We reserve the right to change the information in this catalogue without prior notice.

20141031

PMSoft **User Manual**

PMSoft **User Manual**

www.delta.com.tw/ia

Smarter. Greener. Together.

PMSoft User Manual

Revision History

Version	Revision	Date
First version	The first version was published.	2013/05/30
Second version	Chapter 12, Chapter 13, and Appendix B are added.	2014/04/21
Third version	1. Section 13.5.5 is updated. 2. Section 13.5.7, section 13.5.8, and Appendix C are added.	2014/10/31

PMSoft User Manual

Table of Contents

Chapter 1 Introduction of the Software

Chapter 2 Starting and Setting

Chapter 3 Program Organization Units

Chapter 4 Symbols

Chapter 5 Function Block

Chapter 6 Ladder Diagram

Chapter 7 Instruction List

Chapter 8 G-codes and Electronic Cams

Chapter 9 Project Management

Chapter 10 Managing Passwords

Chapter 11 Online Functions

Chapter 12 Continuous Function Chart

Appendix A Installing a USB Driver

Appendix B Delta-defined Parameter Table

Appendix C Overwriting .dfb Files in Delta Libraries

Chapter 1 Introduction of the Software

Table of Contents

Note

Graphic representations

The graphic representations used in the manual are listed in the following table.

Graphic representation	Description
🖱	Clicking the left mouse button
🖱	Clicking the right mouse button
🖱2	Double-clicking the left mouse button
↑ 🖱	Pressing and holding the left mouse button, and then moving the mouse without releasing the button
⌨	Typing with a keyboard
①	Operating sequence (The graphic representation is used when an operating sequence is mentioned. For example, 🖱① and ⌨② .)
❶	Number used with a picture
⚠	Point for attention (The item mentioned may be related to damage to equipment, property, or a human body.)

Trademark declaration

- The products and trademarks which do not belong to Delta Electronics, Inc. belong to the companies which produce and declare them.

1.1 Introduction of PMSoft and System Requirements

PMSoft is a software development tool for Delta's new generation motion controllers. In addition to basic programming functions and interfaces, PMSoft also contains many convenient functions and tools. The multilingual environment and the friendly user interface provide users with an efficient development environment.

1.1.1 Characteristics

- Users can create program organization units, declaring global symbols, and declaring local symbols.
- It supports two programming languages. They are ladder diagrams (LD), and instruction lists (IL).
- It supports electronic cams and G-codes. It provides a good interface for designing motion control.
- It provides simulators, displays an x-axis/y-axis/z-axis path, provides monitoring modes, and trace motion instructions. Users can debug the program created conveniently.
- It supports a large number of applied instructions.
- It provides basic functions such as copying objects, cutting objects, pasting objects, undoing actions, and printing projects.
- It supports traditional Chinese, simplified Chinese, and English.
- Users can search for/replace networks or symbols.
- The project management adopts an interface which uses a hierarchical tree structure.
- It provides many convenient functions such as making comments, activating/inactivating networks, managing devices and symbols, simulation, and etc.
- Users can import/export projects.
- It provides several password setting and data protection mechanism.
- It supports COMMGR, a new generation communication manager.

1.1.2 System Requirements

Before using PMSoft, users have to make sure that an operating system meets the requirements below.

Item	System requirement	
Operating system	Windows 2000/NT/Me/XP/Vista/7	
CPU	Pentium 1.5 G or above	
Memory	256 MB or above (A memory having a capacity of 512 MB or above is recommended.)	
Hard disk drive	Capacity : 500 MB or above	
CD-ROM drive	For installing software It is optionally required.	
Monitor	Resolution: 16-bit color quality or above	
Keyboard/Mouse	A general keyboard/mouse, or devices compatible with Windows	
Printer	A printer with a driver for Windows (It is used to print a project, and is optionally required.)	
RS-232 port	For connecting to a motion controller	One of them is used, but a motion controller which is connected must have a corresponding port. (*1)
USB port	For connecting to a motion controller	
Ethernet port	For connecting to a motion controller	
Communication software	COMMGR version 0.53 or above must be installed on a computer. (*2)	

Item	System requirement
Models which are supported	AH20MC-5A, AH10PM-5A, AH05PM-5A, DVP-20PM00D, DVP-20PM00M, and DVP-10PM00M

***1. PMSoft supports several ways in which a computer is connected to a motion controller. Users have to make sure of the port and the mode supported by a motion controller before a computer is connected to the motion controller.**

***2. Please refer to section 1.2 for more information about COMMGR.**

***3. The functions and specifications mentioned above are only applicable to PMSoft version 2.04 or above. The older versions are not equipped with the complete functions.**

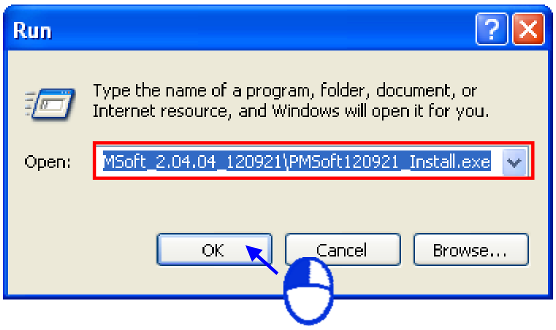
1.1.3 Installing PMSoft

If an older version of PMSoft has been installed on a computer, users have to uninstall it before installing PMSoft. (Pleases refer to section 1.1.4 for more information about uninstalling PMSoft.)

(1) Start a Windows 2000/NT/Me/XP/Vista/7 operating system.

(2) Put a PMSoft CD in the CD-ROM drive, or download the installation program from http://www.delta.com.tw/ch/index.asp. (Before the installation program downloaded from the website is installed, it has to be decompressed.)

(3) Click **Start**, and then click **Run…** to open the **Run** window. Specify a path which denotes the installation file in the **Open** box, and then click **OK**. Users can also double-click the installation file to execute the installation program.

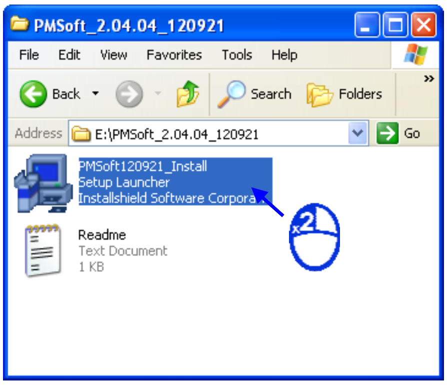


Run

Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: MSoft_2.04.04_120921\PMSoft120921_Install.exe

OR



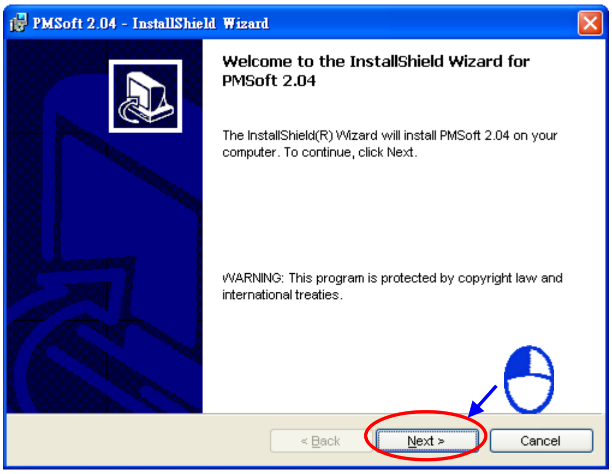
PMSoft_2.04.04_120921

Address E:\PMSoft_2.04.04_120921

PMSoft120921_Install
Setup Launcher
Installshield Software Corpora

Readme
Text Document
1 KB

(4) After the **InstallShield Wizard** window appears, click **Next**.



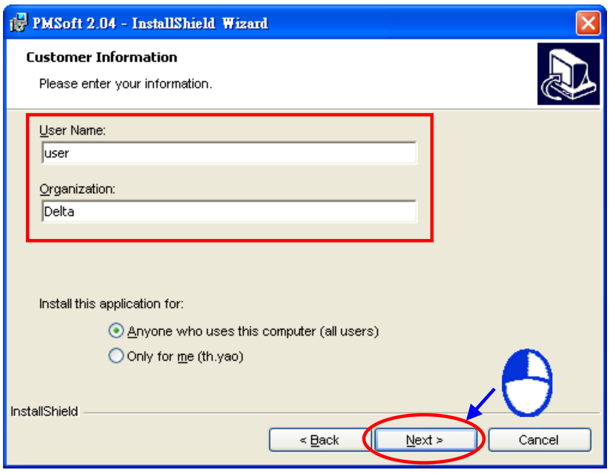
PMSoft 2.04 - InstallShield Wizard

Welcome to the InstallShield Wizard for PMSoft 2.04

The InstallShield(R) Wizard will install PMSoft 2.04 on your computer. To continue, click Next.

WARNING: This program is protected by copyright law and international treaties.

(5) Type related information in the **User Name** box and the **Organization** box, and then click **Next**.



PMSoft 2.04 - InstallShield Wizard

Customer Information

Please enter your information.

User Name: user

Organization: Delta

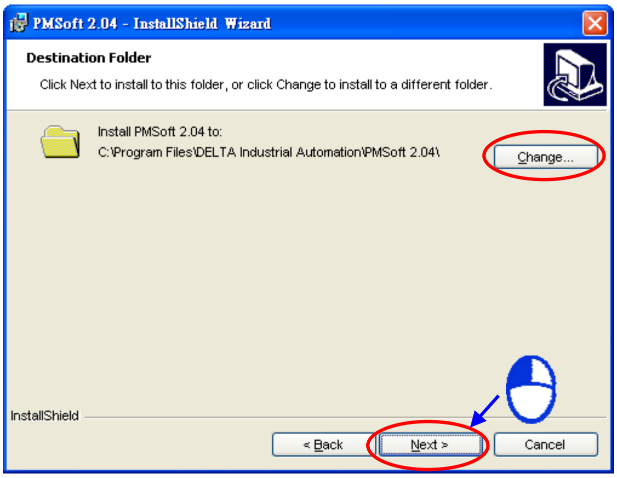
Install this application for:

● Anyone who uses this computer (all users)

○ Only for me (th.yao)

InstallShield

(6) Leave the default path unchanged, or click **Change…** to change the path. Click **Next** to proceed to the next step.



PMSoft 2.04 - InstallShield Wizard

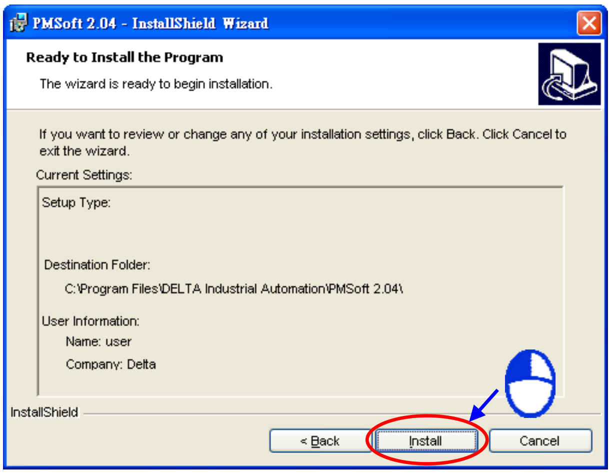
Destination Folder

Click Next to install to this folder, or click Change to install to a different folder.

Install PMSoft 2.04 to:
C:\Program Files\DELTA Industrial Automation\PMSoft 2.04\

InstallShield

(7) Check the installation information, and then click **Install**.



PMSoft 2.04 - InstallShield Wizard

Ready to Install the Program

The wizard is ready to begin installation.

If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

Current Settings:

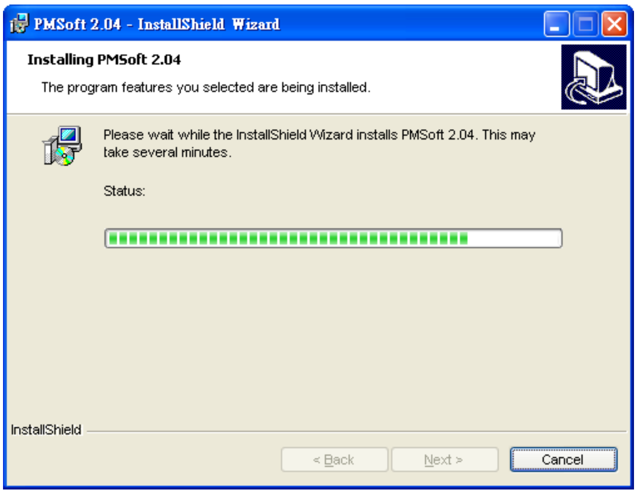
Setup Type:

Destination Folder:
C:\Program Files\DELTA Industrial Automation\PMSoft 2.04\

User Information:
Name: user
Company: Delta

InstallShield

(8) After PMSoft is installed, shortcuts to the program are created on the desktop and the **Start** menu. Click **Finish** to complete the installation.



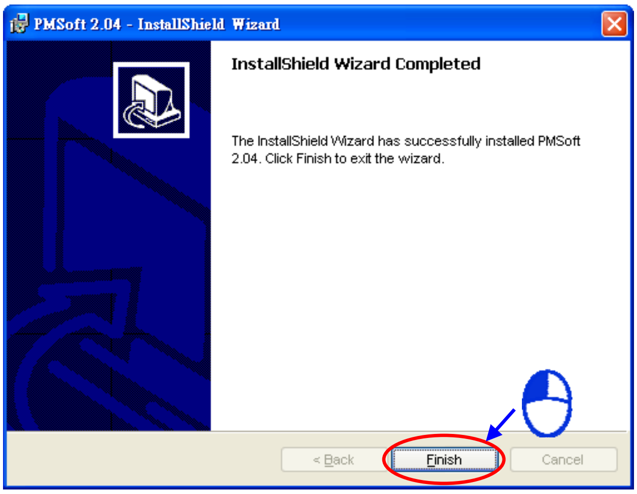
PMSoft 2.04 - InstallShield Wizard

Installing PMSoft 2.04

The program features you selected are being installed.

Please wait while the InstallShield Wizard installs PMSoft 2.04. This may take several minutes.

Status:



PMSoft 2.04 - InstallShield Wizard

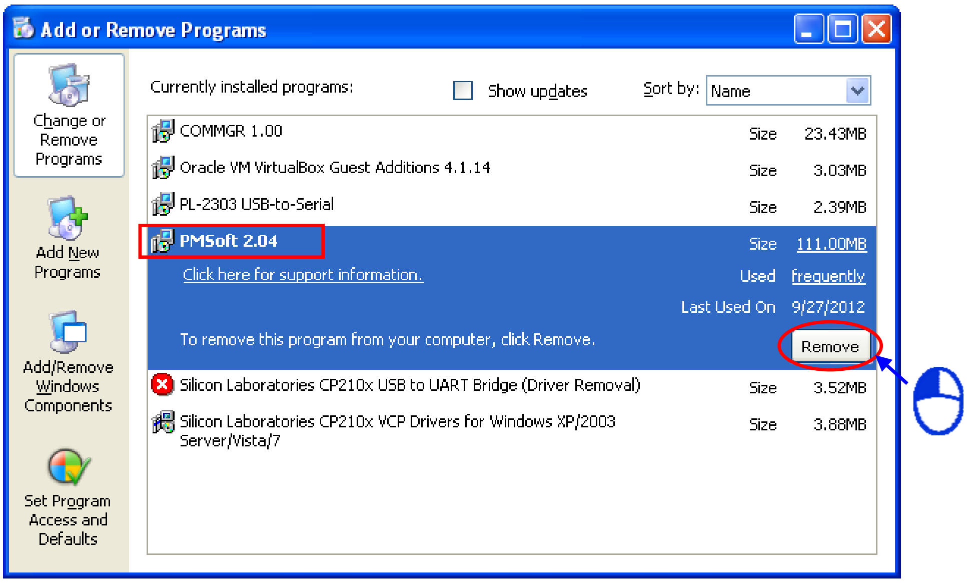
InstallShield Wizard Completed

The InstallShield Wizard has successfully installed PMSoft 2.04. Click Finish to exit the wizard.

1.1.4 Uninstalling PMSoft

(1) There are two methods of uninstalling PMSoft.

- Method 1: Open the **Control Panel** window, and click **Add or Remove Programs**. In the **Currently installed programs** box, click **PMSoft x.xx**, and then click **Remove**. **x.xx** denotes the version of the software.



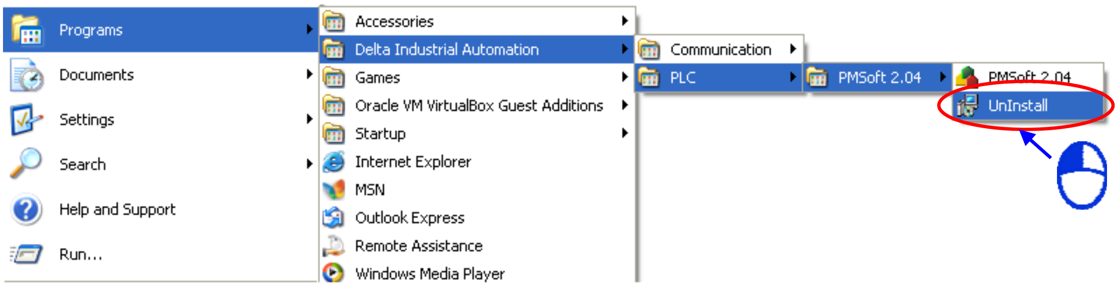
Add or Remove Programs

Currently installed programs: □ Show updates Sort by: Name

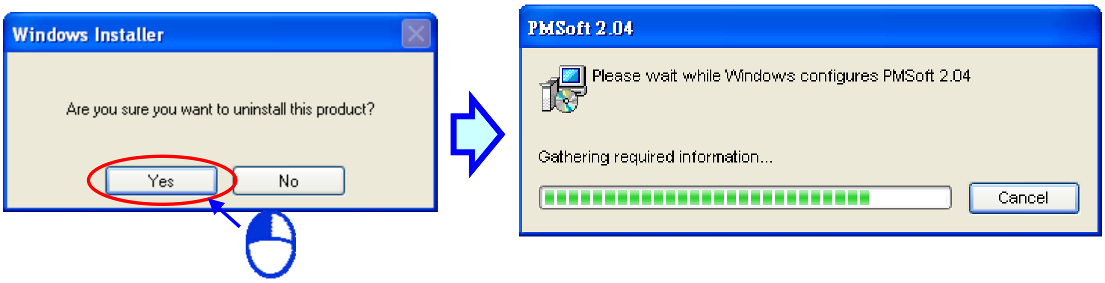
Program		
COMMGR 1.00	Size	23.43MB
Oracle VM VirtualBox Guest Additions 4.1.14	Size	3.03MB
PL-2303 USB-to-Serial	Size	2.39MB
PMSoft 2.04	Size	111.00MB
Click here for support information.	Used	frequently
	Last Used On	9/27/2012
To remove this program from your computer, click Remove.	Remove	
Silicon Laboratories CP210x USB to UART Bridge (Driver Removal)	Size	3.52MB
Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7	Size	3.88MB

Change or Remove Programs
Add New Programs
Add/Remove Windows Components
Set Program Access and Defaults

- Method 2: **Start**>**Programs**>**Delta Industrial Automation**>**PLC**>**PMSoft x.xx**>**Uninstall**



(2) After users click **Yes**, PMSoft will be removed.



Windows Installer

Are you sure you want to uninstall this product?

PMSoft 2.04

Please wait while Windows configures PMSoft 2.04

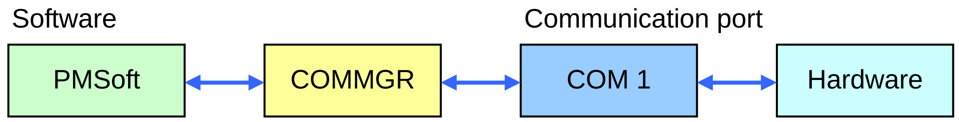
Gathering required information...

1.2 Introduction of COMMGR

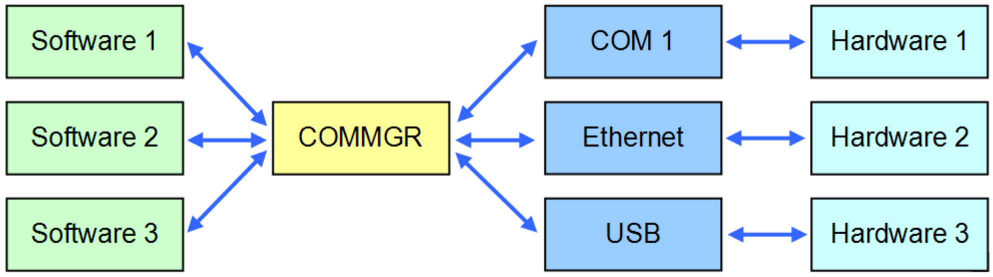
COMMGR is a new generation communication management tool developed by Delta Electronics, Inc.. It functions as a communication bridge between Delta software and hardware.

1.2.1 Working Mode of COMMGR

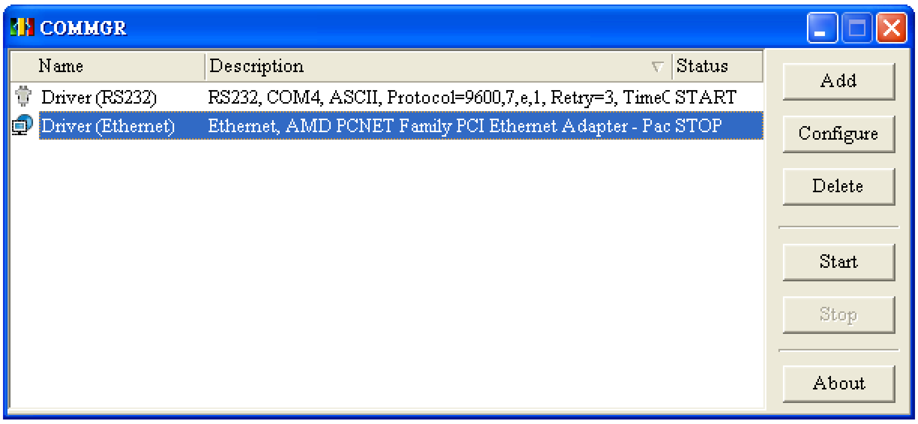
Users can create communication parameters which must be set on the management list in COMMGR in advance. The communication parameters which have been created in advance are called a driver. Users can start or stop a driver in COMMGR. If a driver is started, a connection will be created automatically. After the users specify a driver in PMSoft, a communication will be carried out.



In addition to PMSoft, other software communicating with hardware through COMMGR can operate simultaneously. COMMGR manages all communication commands, and makes software connect to hardware.



The **COMMGR** window and the management list are shown below. Drivers which are named by users are displayed in the **Name** column, parameters related to the drivers are displayed in the **Description** column, and the statuses of the drivers are displayed in the **Status** column.



Name	Description	Status
Driver (RS232)	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeC	START
Driver (Ethernet)	Ethernet, AMD PCNET Family PCI Ethernet Adapter - Pac	STOP

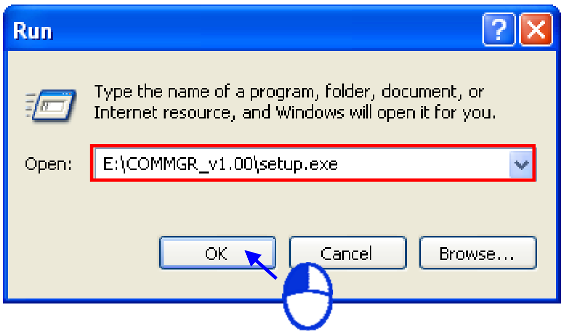
1.2.2 Installing COMMGR

COMMGR is software independent of PMSoft. It must be installed and uninstalled separately. If an older version of COMMGR has been installed on a computer, users have to uninstall it before installing COMMGR. (Pleases refer to section 1.2.3 for more information about uninstalling COMMGR).

(1) Start a Windows 2000/NT/Me/XP/Vista/7 operating system.

(2) Put a COMMGR CD in the CD-ROM drive, or download the installation program from http://www.delta.com.tw/ch/index.asp. (Before the installation program downloaded from the website is installed, it has to be decompressed.)

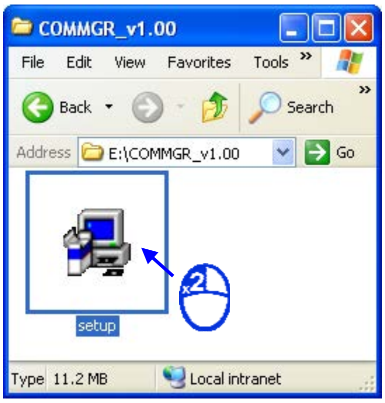
(3) Click **Start**, and then click **Run…** to open the **Run** window. Specify a path which denotes the installation file in the **Open** box, and then click **OK**. Users can also double-click the installation file to execute the installation program.



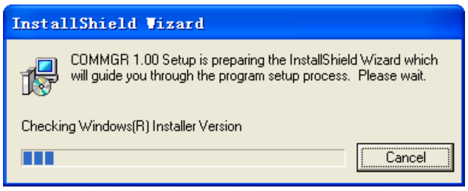
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: E:\COMMGR_v1.00\setup.exe

OR

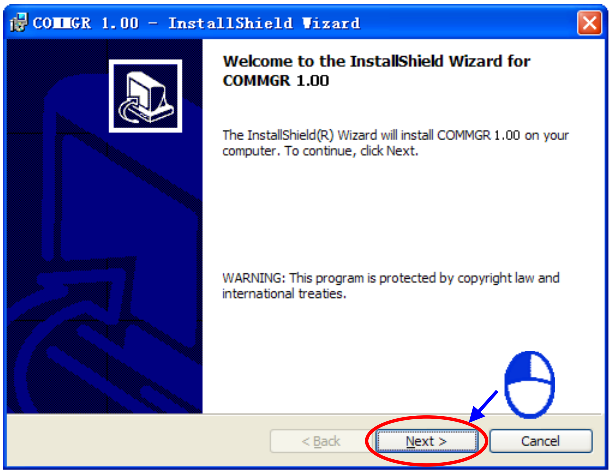


(4) After the **InstallShield Wizard** window appears, click **Next**.



COMMGR 1.00 Setup is preparing the InstallShield Wizard which will guide you through the program setup process. Please wait.

Checking Windows(R) Installer Version

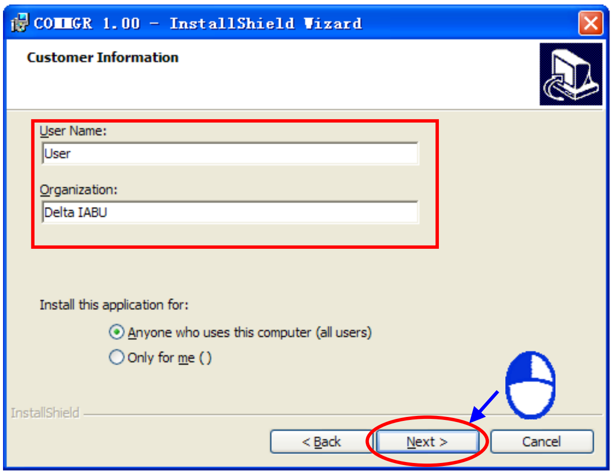


Welcome to the InstallShield Wizard for COMMGR 1.00

The InstallShield(R) Wizard will install COMMGR 1.00 on your computer. To continue, click Next.

WARNING: This program is protected by copyright law and international treaties.

(5) Type related information in the **User Name** box and the **Organization** box, and then click **Next**.



COMMGR 1.00 – InstallShield Wizard

Customer Information

User Name: User

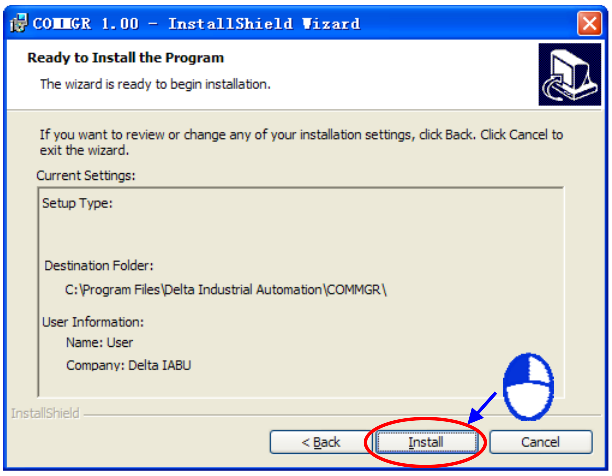
Organization: Delta IABU

Install this application for:

● Anyone who uses this computer (all users)

○ Only for me ()

(6) Check the installation information, and then click **Install**.



COMMGR 1.00 – InstallShield Wizard

Ready to Install the Program

The wizard is ready to begin installation.

If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

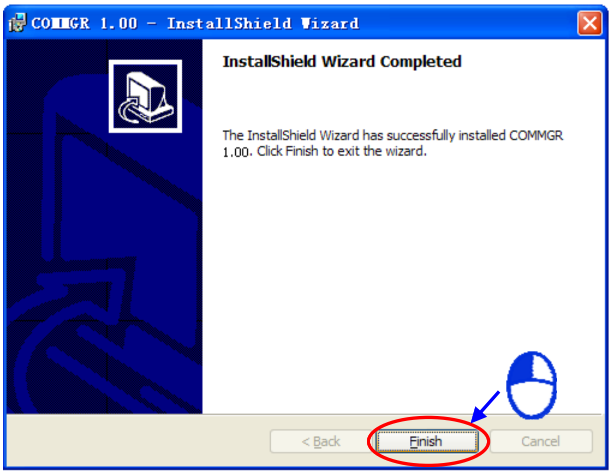
Current Settings:

Setup Type:

Destination Folder:
C:\Program Files\Delta Industrial Automation\COMMGR\

User Information:
Name: User
Company: Delta IABU

(7) After COMMGR is installed, a shortcut to the program is created on the **Start** menu. Click **Finish** to complete the installation.



COMMGR 1.00 – InstallShield Wizard

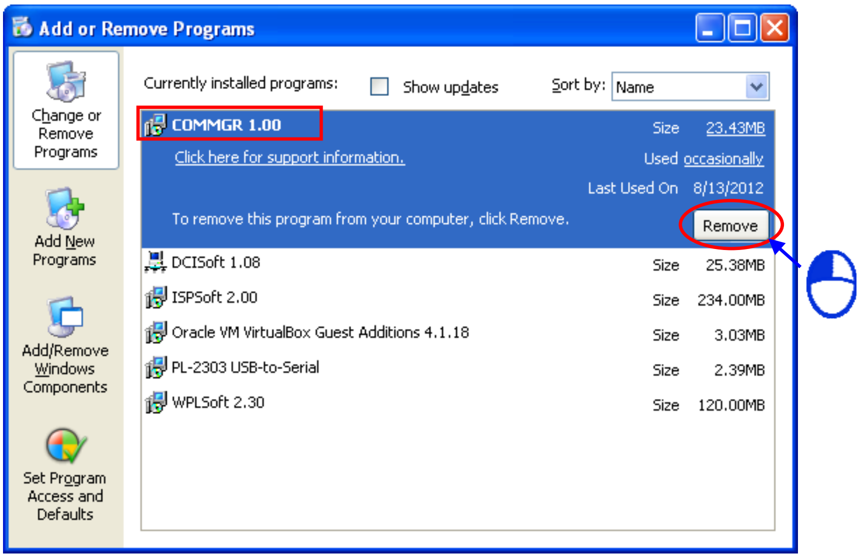
InstallShield Wizard Completed

The InstallShield Wizard has successfully installed COMMGR 1.00. Click Finish to exit the wizard.

1.2.3 Uninstalling COMMGR

(1) There are two methods of uninstalling COMMGR.

- Method 1: Open the **Control Panel** window, and click **Add or Remove Programs**. In the **Currently installed programs** box, click **COMMGR x.xx.xx**, and then click **Remove**. **x.xx** denotes the version of the software.

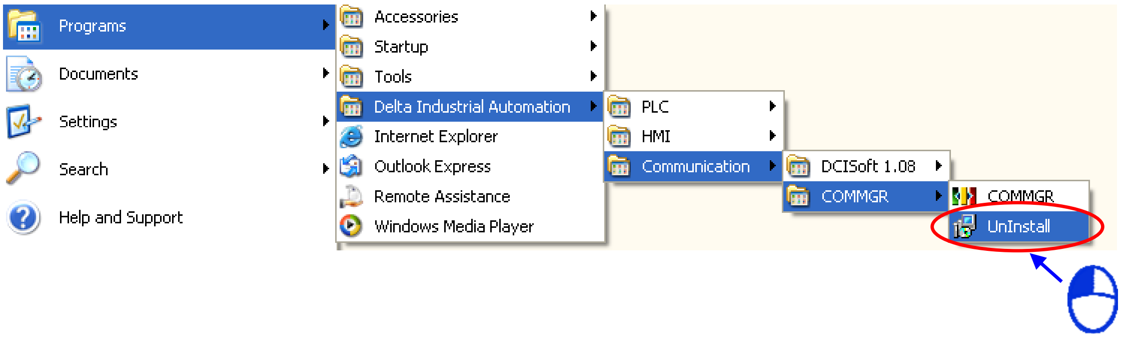


Add or Remove Programs

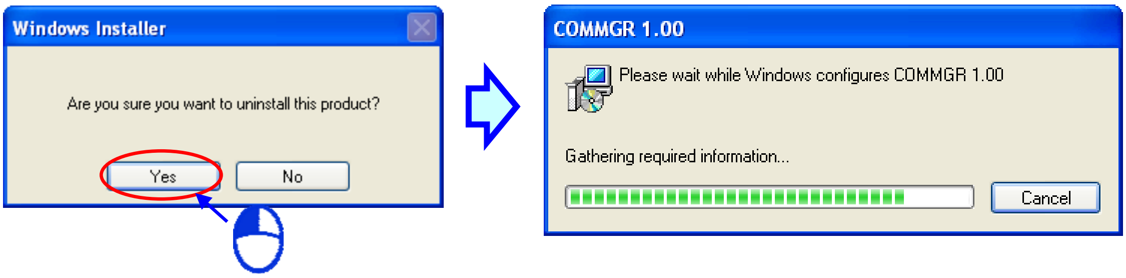
Currently installed programs:	☐ Show updates	Sort by: Name	
COMMGR 1.00		Size	23.43MB
Click here for support information.		Used	occasionally
		Last Used On	8/13/2012
To remove this program from your computer, click Remove.			Remove
DCISoft 1.08		Size	25.38MB
ISPSoft 2.00		Size	234.00MB
Oracle VM VirtualBox Guest Additions 4.1.18		Size	3.03MB
PL-2303 USB-to-Serial		Size	2.39MB
WPLSoft 2.30		Size	120.00MB

Change or Remove Programs
Add New Programs
Add/Remove Windows Components
Set Program Access and Defaults

- Method 2: **Start**>**Programs**>**Delta Industrial Automation**>**Communication**>**COMMGR**>**Uninstall**



(2) After users click **Yes**, COMMGR will be removed.



Windows Installer

Are you sure you want to uninstall this product?

COMMGR 1.00

Please wait while Windows configures COMMGR 1.00

Gathering required information...

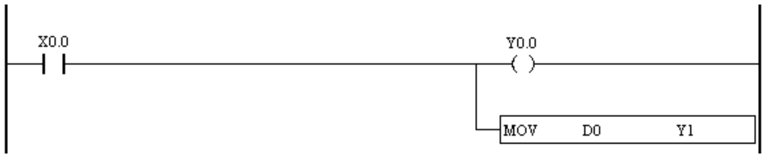
1.3 Important Points about Writing a Program for an AH500 Series Motion Controller

AH20MC-5A, AH10PM-5A, and AH05PM-5A are medium types of motion control modules. The system frameworks of the AH500 series motion control modules are slightly different from those of the DVP series motion controllers. An AH500 series motion control module can be used independently. It can also function as an extension module because it can be used with an AH500 series CPU module. Users have to refer to related manuals for more information before they develop a project.

The important points about writing a program in PMSoft are listed below.

- Some M devices and some D devices in a DVP series motion controller are special relays and special data registers. The special relays and the special data registers in an AH500 series motion controller are SM devices and SR devices. Users can use SM9192~SM16383 by themselves. The SM devices assigned to symbols can function as general registers. The system does not assign SM devices until the M devices are assigned.
- There are W devices in an AH500 series motion controller. They function as D devices. Users can use the W devices by themselves. The system does not assign W devices to symbols until the D devices are assigned to symbols.
- The X/Y devices in DVP series motion controllers are represented by X0, Y1, and etc. The X/Y devices which are used as word devices in AH500 series motion controllers are represented by X0, Y0, and etc.. Besides, the X/Y devices which are used as bit devices in AH500 series motion controllers are represented by X0.0, Y0.0, and etc. X0.0 represents the first bit (the lowest bit) in X0, and X0.15 represents the last bit (the highest bit) in X0.

In an AH500 series motion controller, there are 256 X devices which are used as bit devices, and 256 Y devices which are used as bit devices. They are X0.0~X15.15, and Y0.0~Y15.15. In the example below, the states of Y1.0~Y1.15 will change after the value in D0 is moved to Y1.



The devices in an AH500 series motion controller are listed in the following table.

Device type	Manipulating the bits		Manipulating the words		Remark
	Supporting	Range	Supporting	Range	
X	✓	X0.0~X15.15	✓	X0~X15	
Y	✓	Y0.0~Y15.15	✓	Y0~Y15	
M	✓	M0~M4095		---	
S	✓	S0~S1023		---	
P	✓	P0~P255		---	
SP	✓	SP0~SP16383		---	
D		---	✓	D0~D9999	
W		---	✓	W0~W65535	
C		C0~C255		---	A counter is ON when the number of times a particular event or process has occurred conforms to the setting value, and a counter is OFF when the number of times a particular event or process has occurred does not conform to the setting value.

Device type	Manipulating the bits		Manipulating the words		Remark
	Supporting	Range	Supporting	Range	
C		---	✓	C0~C255	The value stored in a counter is the present value of the counter.
T	✓	T0~T255		---	A timer is ON when the time interval which is measured conforms to the setting value, and a timer is OFF when the time interval which is measured does not conform to the setting value.
T		---	✓	T0~T255	The value stored in a timer is the present value of the timer.
V		---	✓	V0~V5	
Z		---	✓	Z0~Z7	Double word (32 bits)
SR		---	✓	SR0~SR16383	
SM	✓	SM0~SM16383		---	

Additional remark

If the bits in the X/Y devices which are word devices in an AH500 series motion controller are manipulated, the memory blocks that the X/Y devices occupy are manipulated. However, the memory blocks that the T/C devices used as bit devices occupy are different from the memory blocks that the T/C devices used as word devices occupy. If a timer is used as a bit device, the timer is ON when the time interval which is measured conforms to the setting value, and the timer is OFF when the time interval which is measured does not conform to the setting value. If a timer is used as a word device, the value stored in the timer is the present value of the timer. If a counter is used as a bit device, the counter is ON when the number of times a particular event or process has occurred conforms to the setting value, and the counter is OFF when the number of times a particular event or process has occurred does not conform to the setting value. If a counter is used as a word device, the value stored in the counter is the present value of the counter.

MEMO

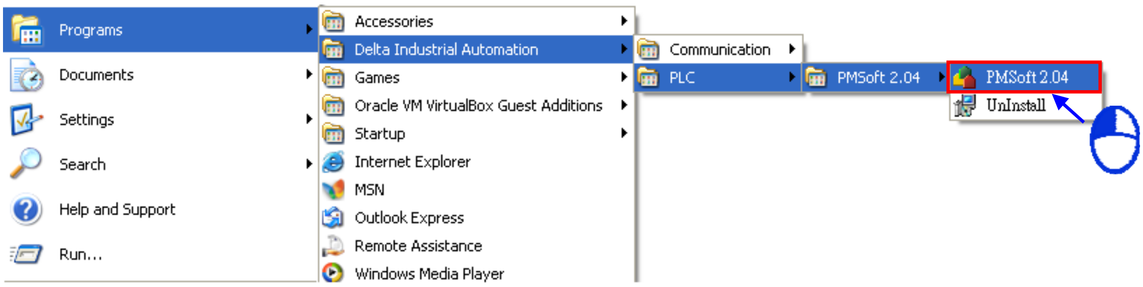
Chapter 2 Starting and Setting

Table of Contents

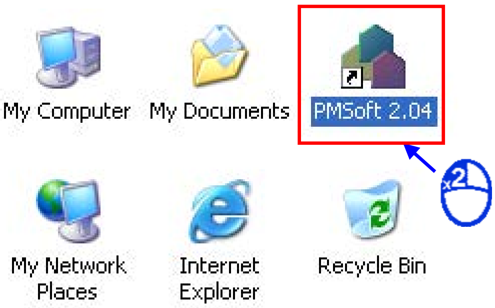
2.1 Guide to Starting PMSoft and Introduction of the Environment

2.1.1 First Step of Entering PMSoft

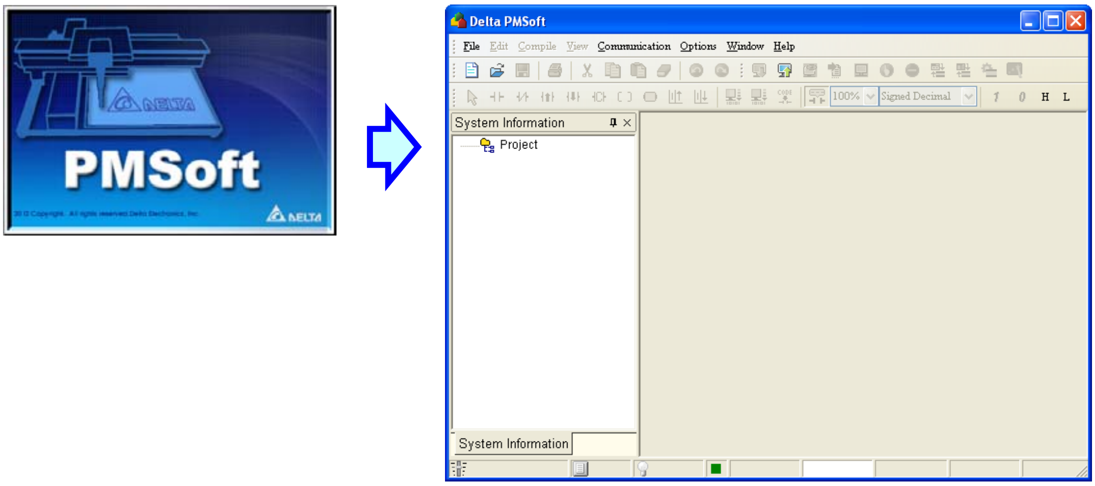
After PMSoft is installed, shortcuts to the program will be created on the desktop and the **Start** menu. Users can click the shortcut on the **Start** menu or double-click the shortcut on the desktop to start PMSoft. Besides, several PMSoft windows are allowed. The users can start PMSoft again in the same way.



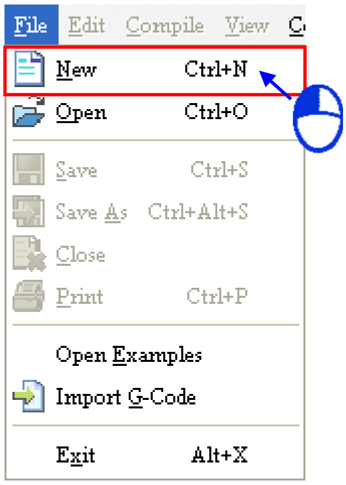
OR



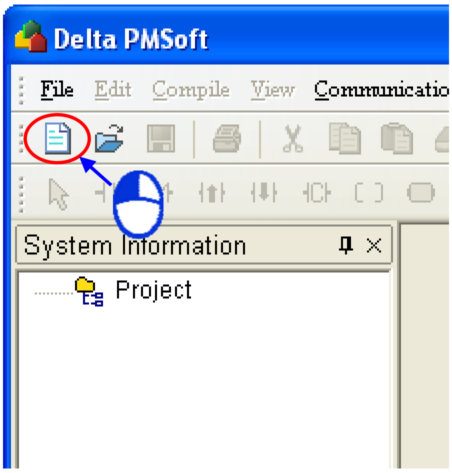
After the welcome screen disappears, the **Delta PMSoft** window will appear. There are basic functions available to the users.



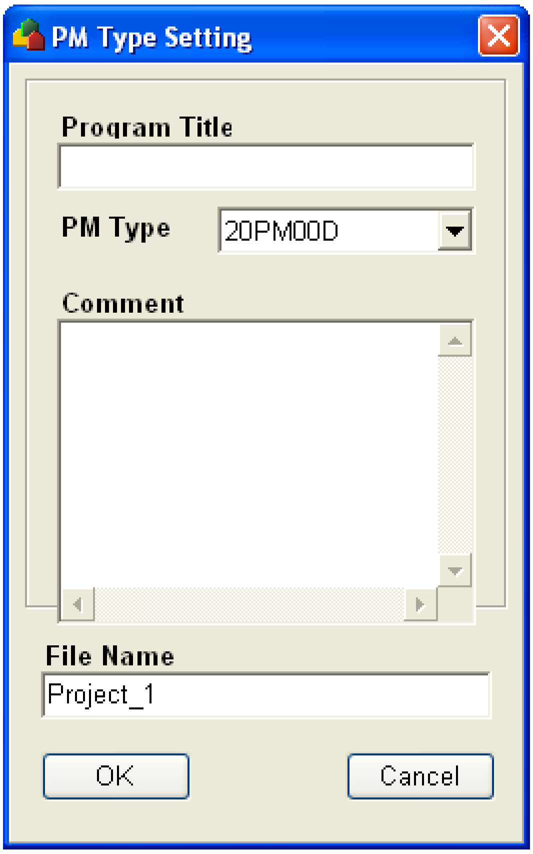
After **New** on the **File** menu, or 🗋 on the toolbar is clicked, a new project is created.



OR



In the **PM Type Setting** window, type a program title in the **Program Title** box, and select a model in the **PM Type** drop-down list box. The users can type a comment in the **Comment** box, if necessary. Finally, type a file name in the **File Name** box. Click **OK** after a file name is typed. The program title typed in the **Program Title** box is similar to a comment. It is not displayed in another place. The users can type a program title in the **Program Title** box, if necessary. The file name typed in the **File Name** box is the name with which the project is saved. It is also a project name.



PM Type Setting

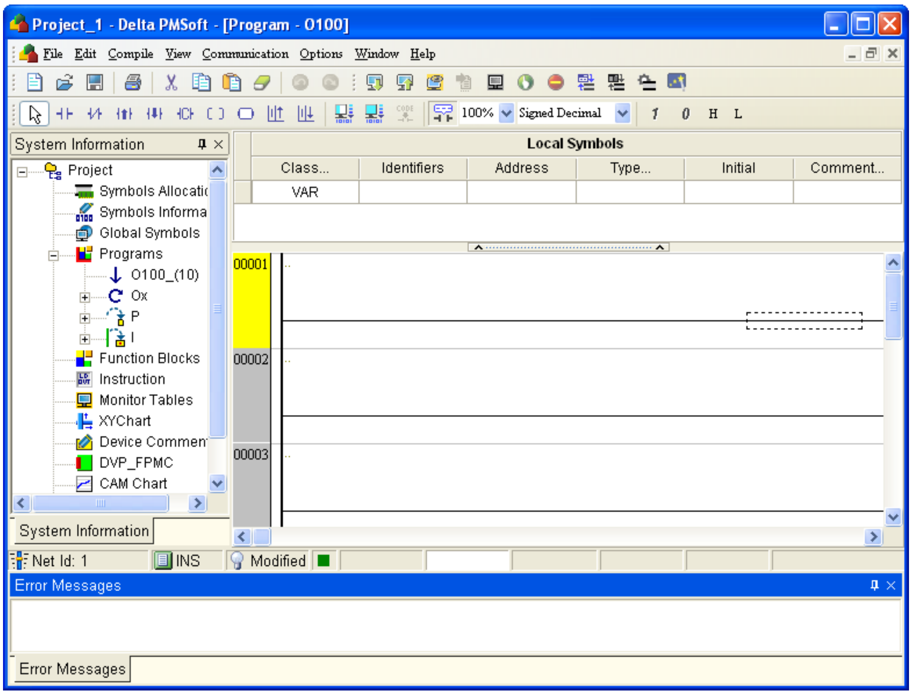
Program Title

PM Type: 20PM00D

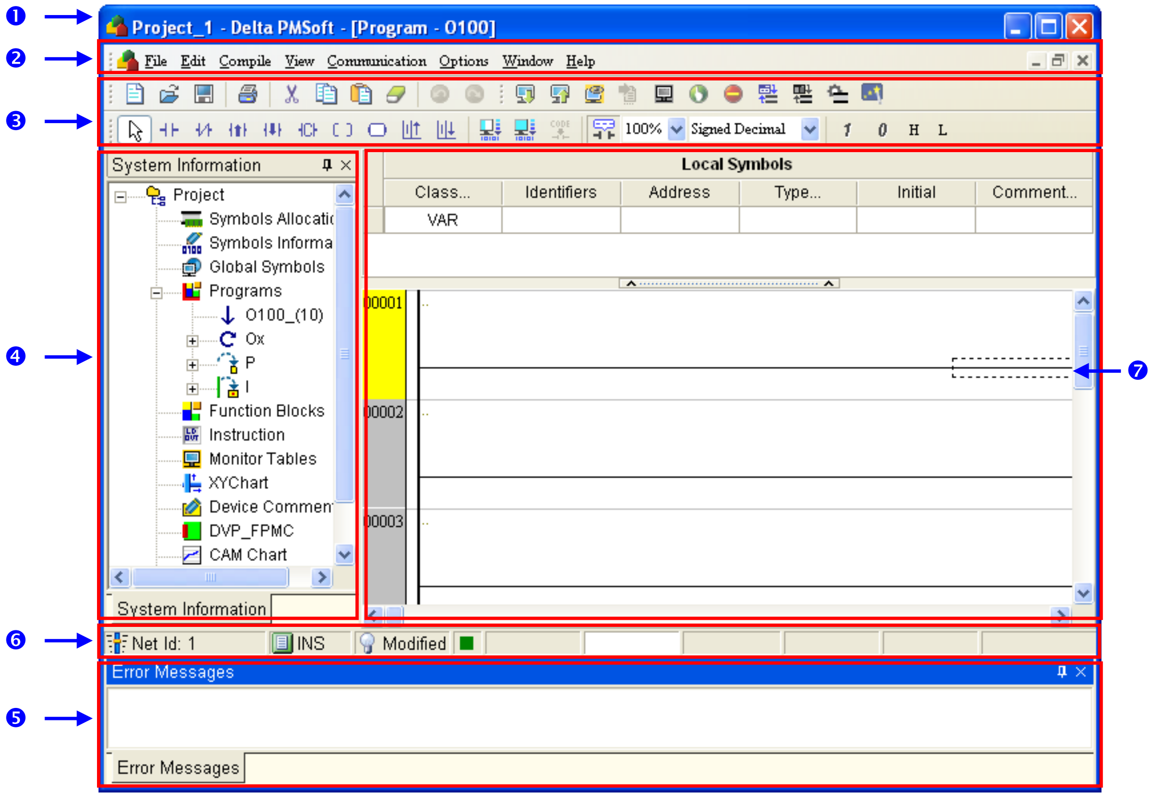
Comment

File Name: Project_1

After the project is created successfully, a system information area will appear at the left side of the main screen. The relation between the items listed in the system information area is represented by a hierarchical tree structure. If the system information area does not appear, the users can click **View System Information** on the **View** menu. If the users click **View Error Message Box** on the **View** menu, an error message area will appear. The **O100** window will appear in the main working area after the project is created successfully.



The main screen of PMSoft is shown below.



❶ Window title: It displays the project name or the file name, and the program which is edited.

❷ Menu bar: There are eight menus.

❸ Toolbar: There are three toolbars.

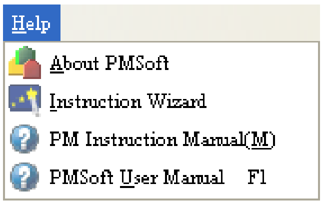
❹ System information area: It adopts an interface which uses a hierarchical tree structure.

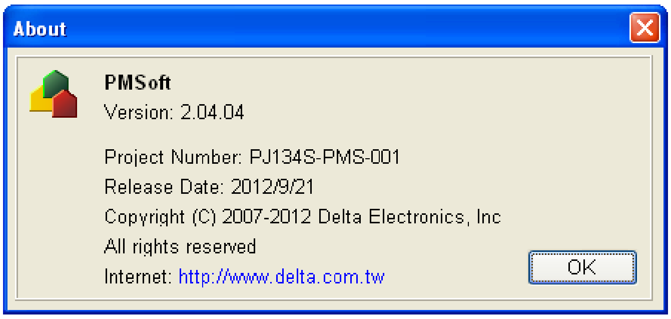
❺ Error message area: The messages related to the compiling of a program are displayed in this area.

❻ Status bar: The information about the editing of a program and a connection is displayed.

❼ Working area: A program editing area, a symbol table, a device table, and etc. are displayed in this area.

When using PMSoft, the users can click an item on the **Help** menu to get help.




About

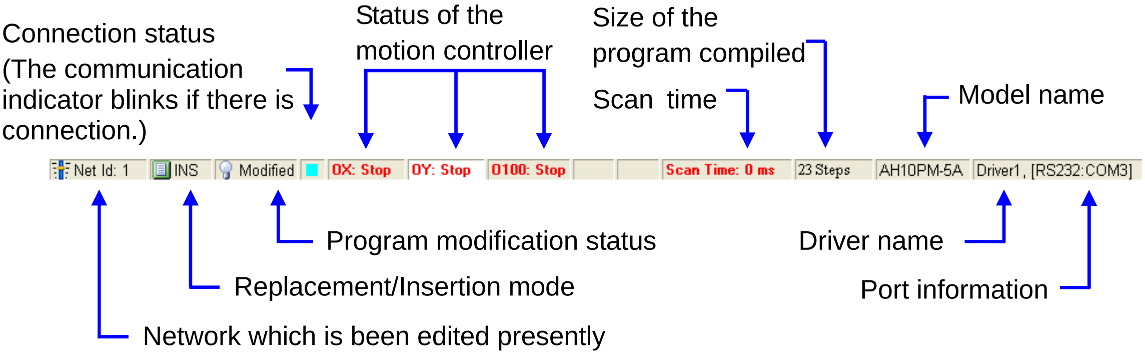
PMSoft
Version: 2.04.04

Project Number: PJ134S-PMS-001
Release Date: 2012/9/21
Copyright (C) 2007-2012 Delta Electronics, Inc
All rights reserved
Internet: http://www.delta.com.tw

- **About PMSoft**: Users can get the information about the version of the software, and the date when the software is released.
- **Instruction Wizard**: The **Application Instruction** window leads users to type an instruction.
- **PM Instruction Manual (M)**: Users can get the information about the instructions and the registers.
- **PMSoft User Manual**: Users can get the usage of the software. (Users can also press F1 on the keyboard to get the usage of the software.)

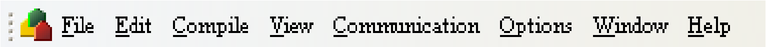
2.1.2 Status Bar

The status bar displays all kinds of working states, including the network which is edited presently, the replacement/insertion mode, the program modification status, the connection status, the status of the motion controller (execution/stop/error), the scan time, and the size of the program compiled.

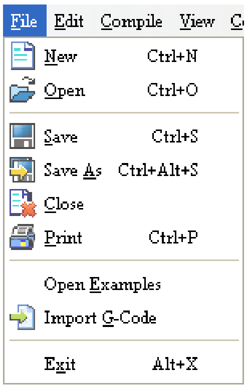

Connection status
(The communication indicator blinks if there is connection.)
Status of the motion controller
Size of the program compiled
Scan time
Model name
Net Id: 1
INS
Modified
0X: Stop
0Y: Stop
0100: Stop
Scan Time: 0 ms
23 Steps
AH10PM-5A
Driver1, [RS232:COM3]
Program modification status
Driver name
Replacement/Insertion mode
Port information
Network which is been edited presently

2.1.3 Menu Bar

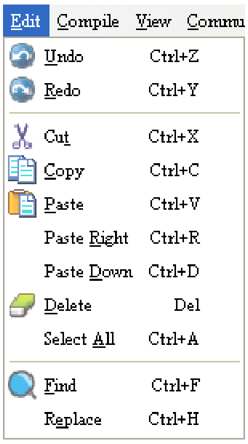
There are eight menus in the menu bar. The items on the menus vary with the editing work carried out. The items shown in grayscale can not be clicked. A brief introduction of the menu bar is presented here, and a more detailed introduction of the menu bar will be presented in the following chapters.



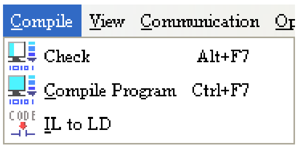
- **File**: It provides the function of accessing projects.



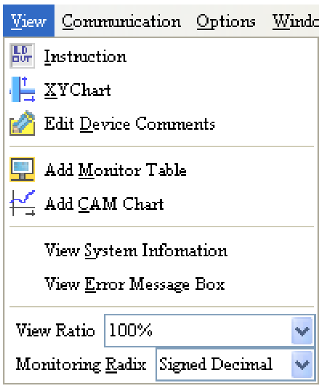
- **Edit**: It provides the function of editing a project.



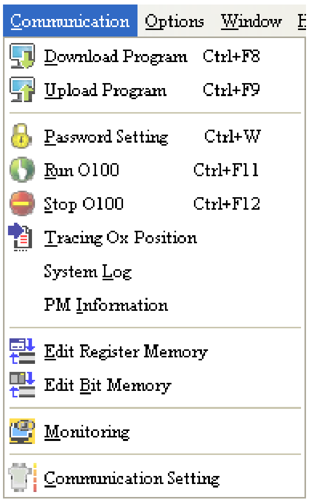
- **Compile**: It provides the function of checking syntax, the function of transforming a program into an execution code, and the function of transforming an instruction list into a ladder diagram.



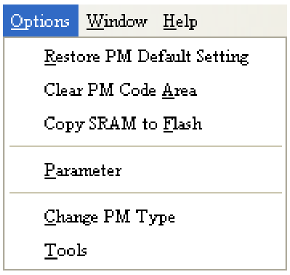
- **View**: Users can zoom in or zoom out the working area, and open some function windows.



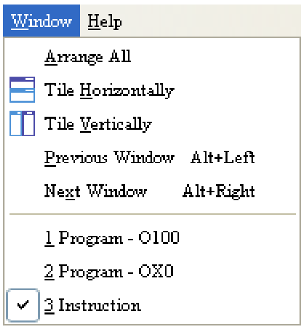
- **Communication**: The items on the **Communication** menu are functions related to a motion controller.



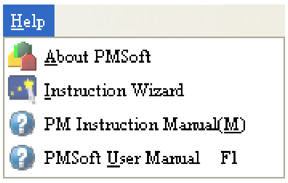
- **Options**: Users can set the memory of a motion controller, and motion control parameters.



- **Window**: Users can manage the windows in the working area

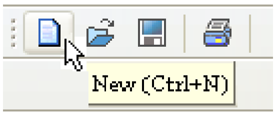


- **Help**: It provides help to users.

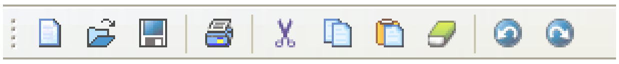


2.1.4 Toolbars

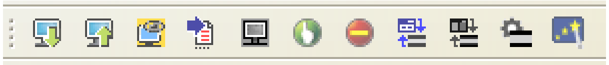
There are three toolbars. If a mouse cursor is moved to an icon and stays there for a while, the function represented by the icon and the shortcut to the function will be displayed.



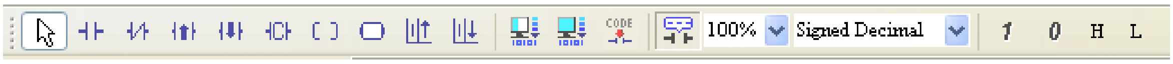
- **Standard toolbar**: It provides the functions related to processing a project.



- **Fast toolbar**: It provides the functions related to operating a motion controller.

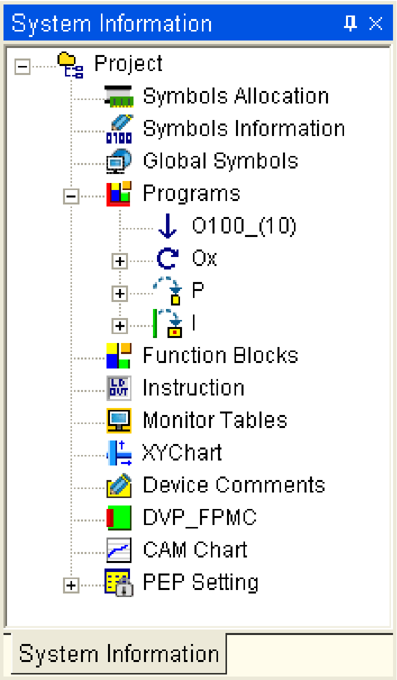


- **PMSoft toolbar**: It provides the functions related to editing a program.

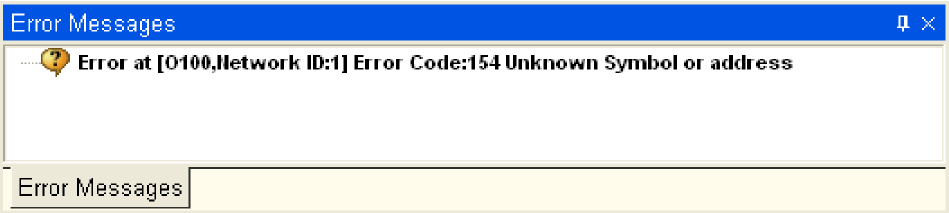


2.1.5 System Information Area and Error Message Area

The contents of the project management area are items related to project development, including device information, symbols, programs, and function blocks. The system information area adopts an interface which uses a hierarchical tree structure. The system information area appears at the left side of the main screen. If users want to close the area, they can click ☒ in the upper right corner of the area. After **View System Information** on the **View** menu is clicked, the system information area will appear. After the users double-click or right-click an item in the system information area, the corresponding window or the corresponding context menu will appear.



The error messages related to the compiling of a program are displayed in the error message area. The error message area appears at the bottom of the main screen. After **View Error Message Box** on the **View** menu is clicked, the error message area will appear. If users want to close the area, they can click ☒ in the upper right corner of the area.

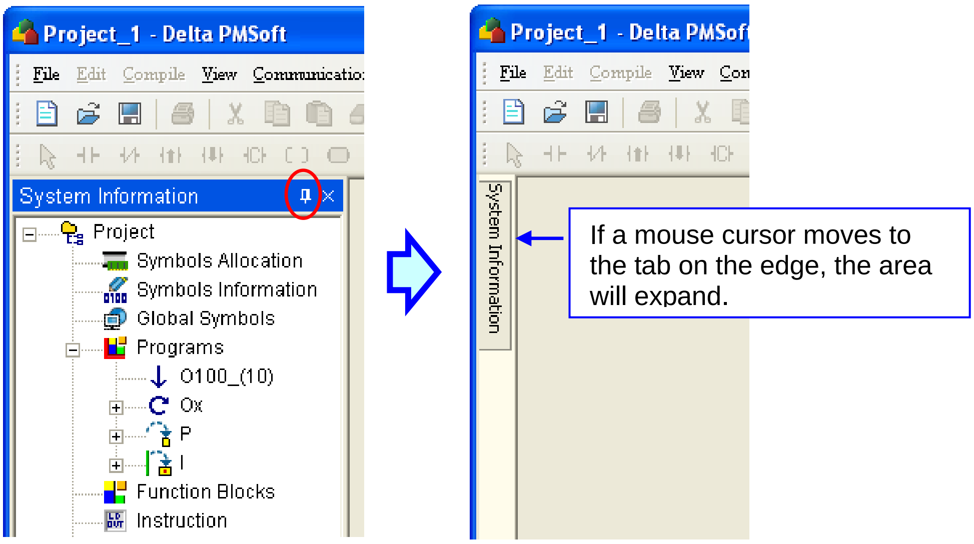


Error Messages

Error at [O100,Network ID:1] Error Code:154 Unknown Symbol or address

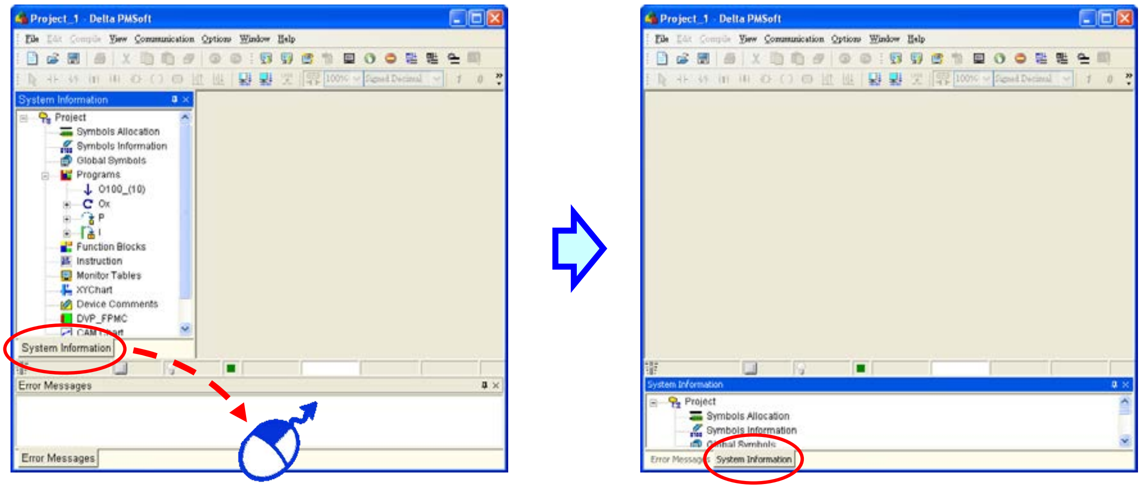
Error Messages

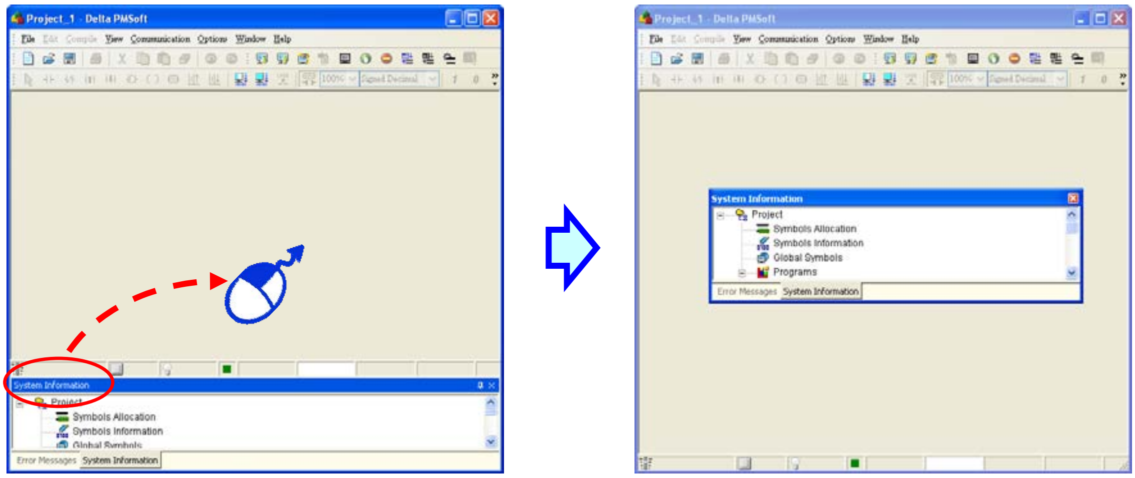
Users can change the way in which the system information area/error message area is displayed by clicking 📌 in the upper right corner of the area. If the icon is ⊣, the area will be hidden automatically. After the mouse cursor leaves the area for a while, the area will contract. If the mouse cursor moves to the tab on the edge, the area will expand.



If a mouse cursor moves to the tab on the edge, the area will expand.

Besides, if the system information area/error message area is not automatically hidden, the users can press the left mouse button while the mouse cursor hovers over the tab, and move the mouse cursor to any position/the other area while holding the left mouse button down. If the users want to drag the whole area, they have to press the left mouse button while the mouse cursor hovers over the title bar, and moves the mouse cursor while holding the left mouse button down.

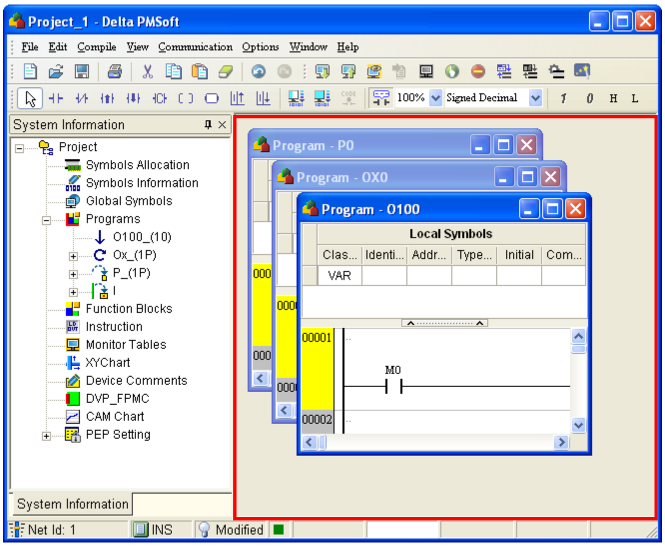




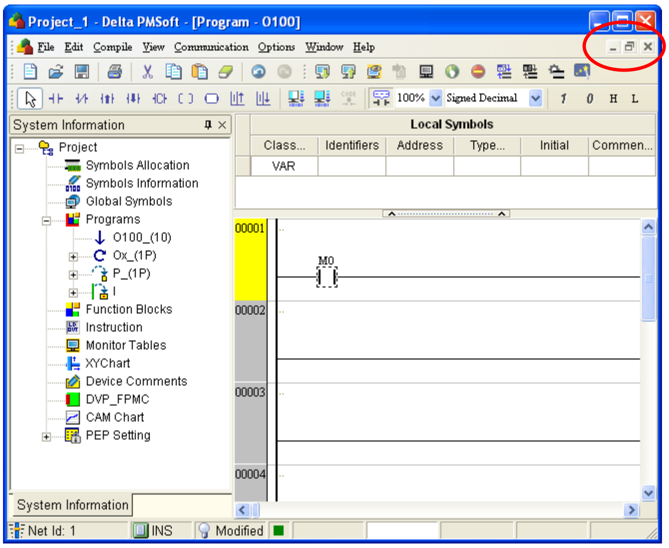
2.1.6 Working Area

All windows are displayed in this area. If a window is maximized or minimized, it is maximized or minimized in this area. If a window is maximized, the status buttons for the window appear at the right side of the menu bar.

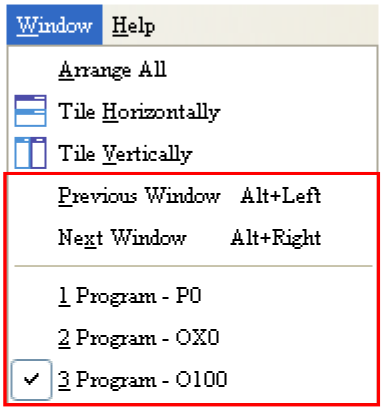
- **Display of windows**



- **A window is maximized.**

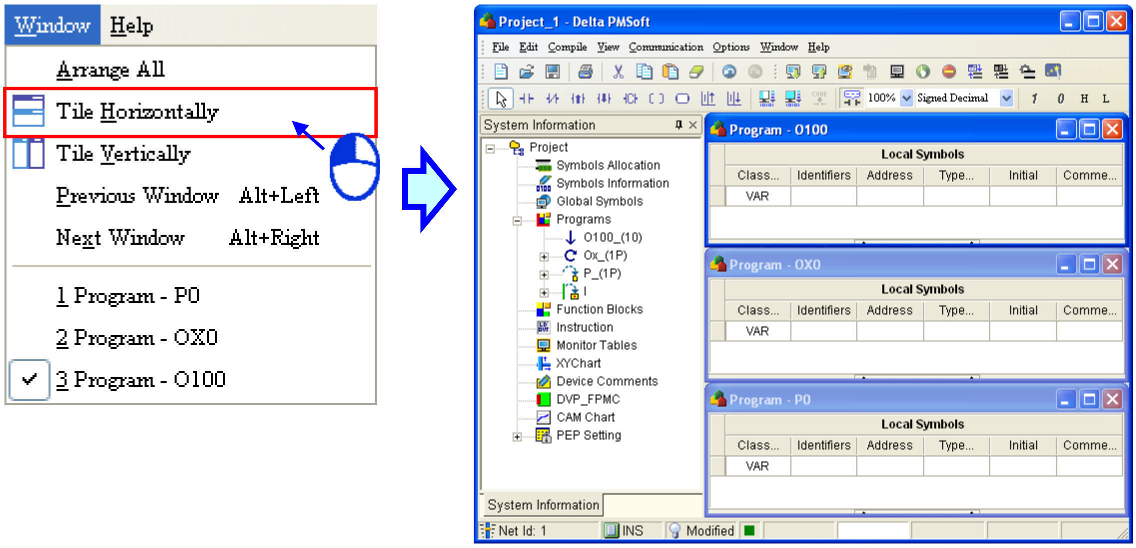


Users can manage the windows in the working area and switch among the windows in the working area through the items on the **Window** menu.

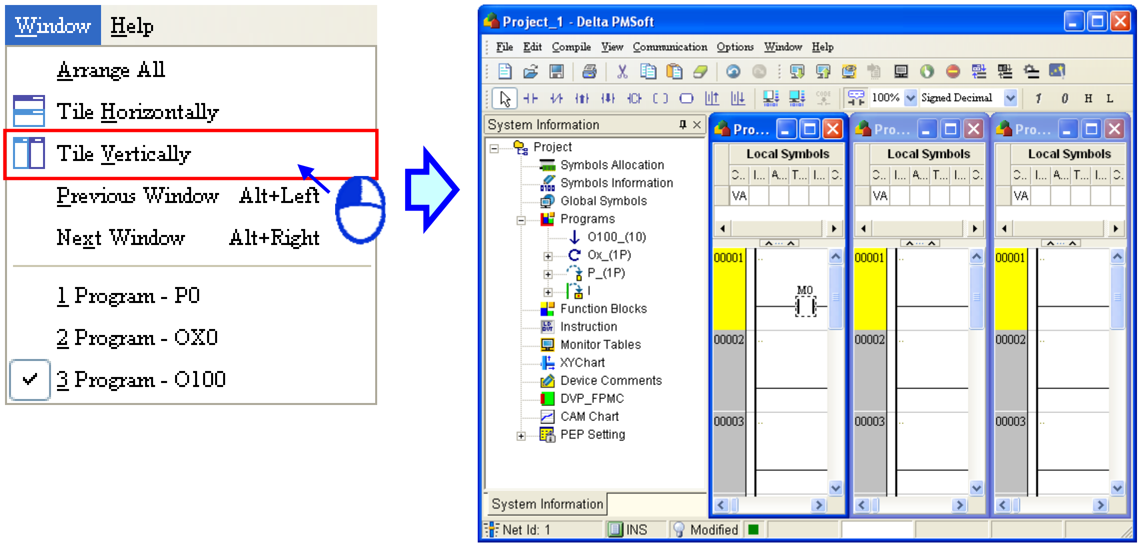


Selecting an item to switch among the windows

- **Arrange All**: After the item is clicked, all the windows will be tiled horizontally or vertically according to the previous shape they took on.
- **Tile Horizontally**: After the item is clicked, all the windows will be tiled vertically, but take on a horizontal shape. The present window will be the topmost window.



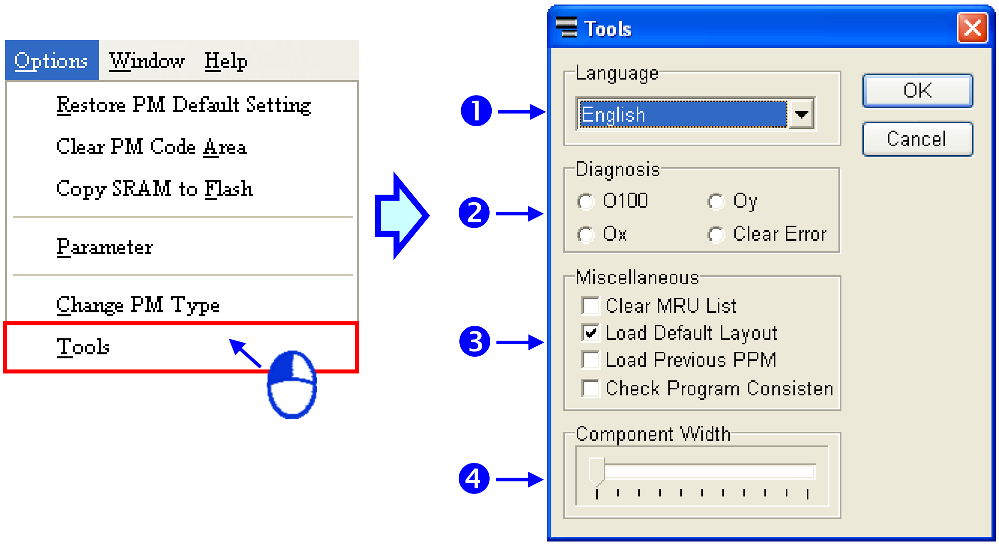
- **Tile Vertically**: After the item is clicked, all the windows will be tiled horizontally, but take on a vertical shape. The present window will be at the leftmost side.



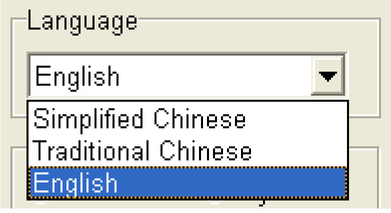
2.2 Basic Settings in PMSoft

2.2.1 Tools

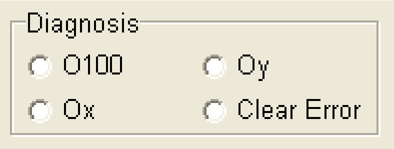
PMSoft allows users to set the operating environment. After users click **Tools** on the **Options** menu, the **Tools** window will appear. There are four sections in the **Tools** window. They are the **Language** section, the **Diagnosis** section, the **Miscellaneous** section, and the **Component Width** section. The four sections are described below.



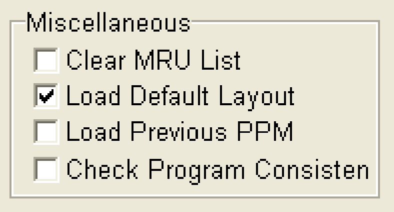
❶ **Language**: Users can select **Traditional Chinese**, **Simplified Chinese**, and **English** in the drop-down list box. After the users select a language, they can click **OK**.



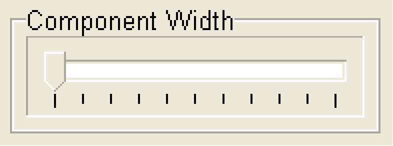
❷ **Diagnosis**: Users can select the **O100** option button, the **Ox** option button, the **Oy** option button, and the **Clear Error** option button. If an error occurs during the operation of a motion controller, the system will not stop running. For the sake of safety, users can select an option button. If an error occurs in the program which is diagnosed, the system will stop running, and an error message will appear. If the users want to restart the system, they have to clear the error flag.



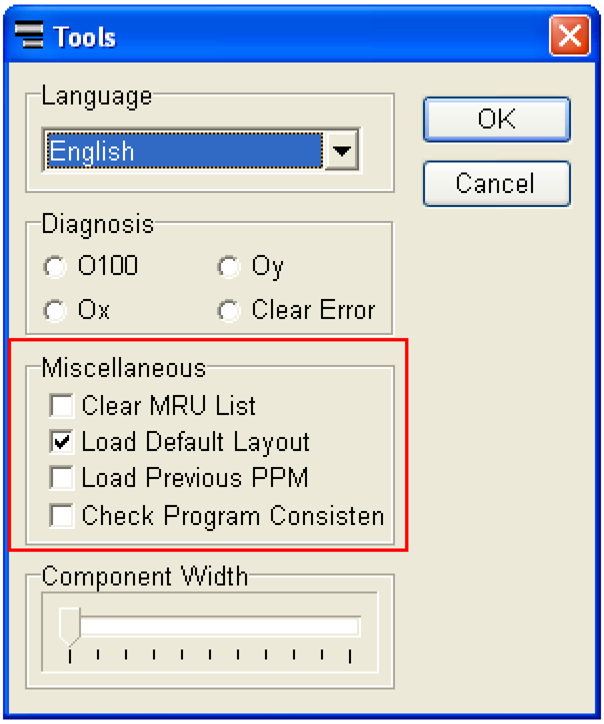
❸ **Miscellaneous**: Users can select the **Clear MRU List** checkbox, the **Load Default Layout** checkbox, the **Load Previous PPM** checkbox, and the **Check Program Consistency** checkbox. These checkboxes are described below.



❹ **Component Width**: Users can set the widths of the components on the screen. If the slider is slid leftward, the components on the screen will appear larger. If the slider is slid rightward, the components on the screen will appear smaller.



- The **Miscellaneous** section in the **Tools** window



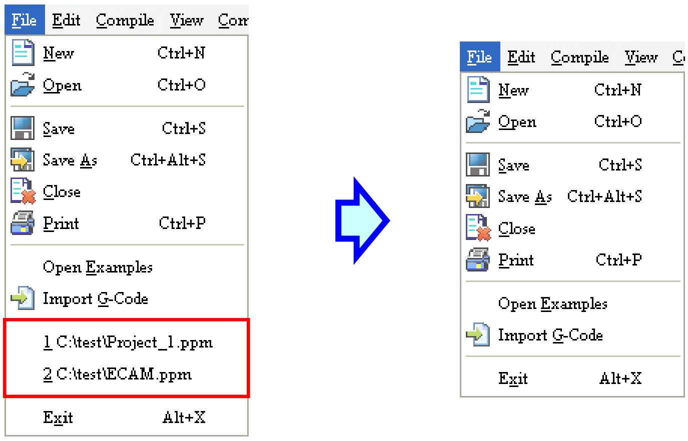
Language: English

Diagnosis
○ 0100 ○ Oy
○ Ox ○ Clear Error

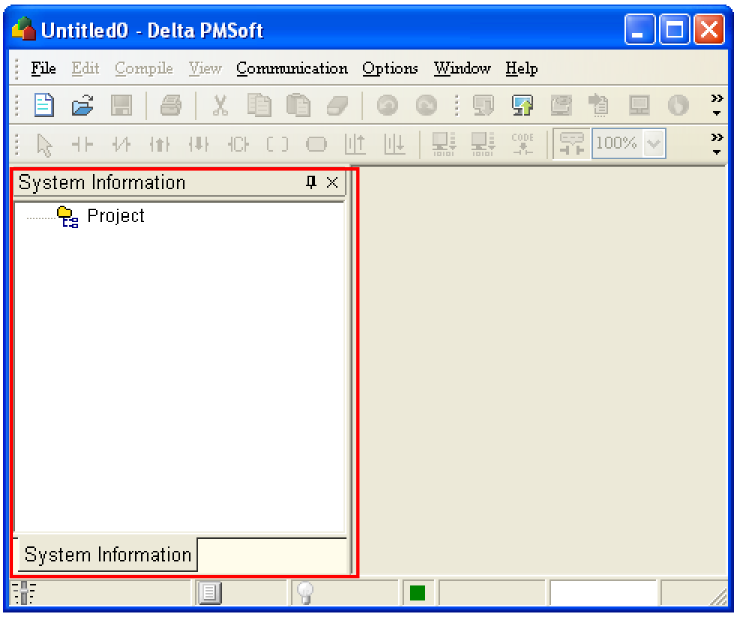
Miscellaneous
□ Clear MRU List
☑ Load Default Layout
□ Load Previous PPM
□ Check Program Consisten

Component Width

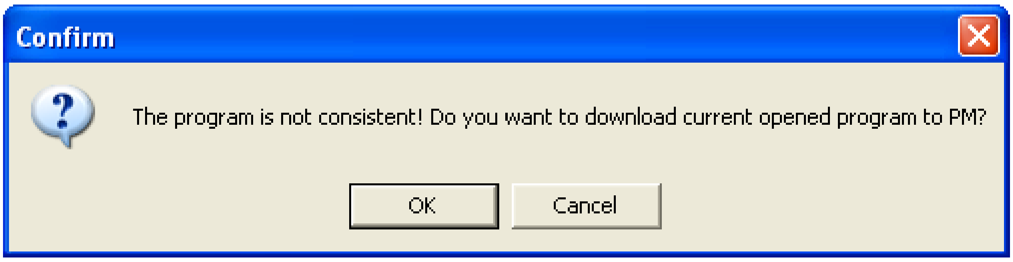
➢ **Clear MRU List**: If users select the **Clear MRU List** checkbox, and click **OK**, the MRU list on the **File** menu will be cleared.



➢ **Load Default Layout**: If the **Load Default Layout** checkbox is selected, the system information area appears whenever PMSoft is started.



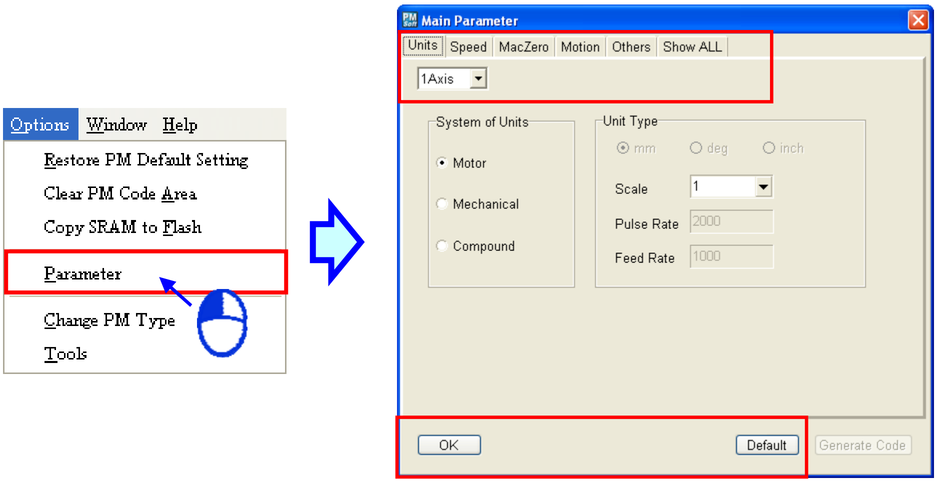
- **Load Previous PPM**: If the **Load Previous PPM** checkbox is selected, the previous project file (*.PPM) is opened whenever PMSoft is started.
- **Check Program Consistency**: If the **Check Program Consistency** checkbox is selected, PMSoft will automatically check whether the program created in PMSoft is the same as the program in the motion controller connected to PMSoft after **Monitoring** on the **Communication** menu is clicked. If the program created in PMSoft is the same as the program in the motion controller connected to PMSoft, the **Confirm** window shown below will appear.



The program is not consistent! Do you want to download current opened program to PM?

2.2.2 Parameters

After users click **Parameters** on the **Options** menu, the **Main Parameter** window will appear. There are six tabs corresponding to six types of parameters in the **Main Parameter** window. The users can select an axis which they want to set. Please refer to DVP-PM Application Manual for more information about setting the parameters. After the users finish setting the parameters, they can click **OK**, and close the window. If the users click ☒ in the upper right corner of the window, the present setting will be cancelled, and the window will be closed. If the users click **Default**, the parameters will be restored to the factory setting. After the users finishes setting the paramters, they can decide whether to select the **Paramter** checkbox in the **Data Transfer** window. The values of the parameters are not saved in the project file. If the users want to donwload the values of the paramters several times, they have to set the paramters before downloading the values of the parameters, or transform the values of the paramters into instruction codes. The parameter setting interface simplifies the setting of the axis parameters. The parameters are described below.



1Axis

System of Units: ● Motor ○ Mechanical ○ Compound

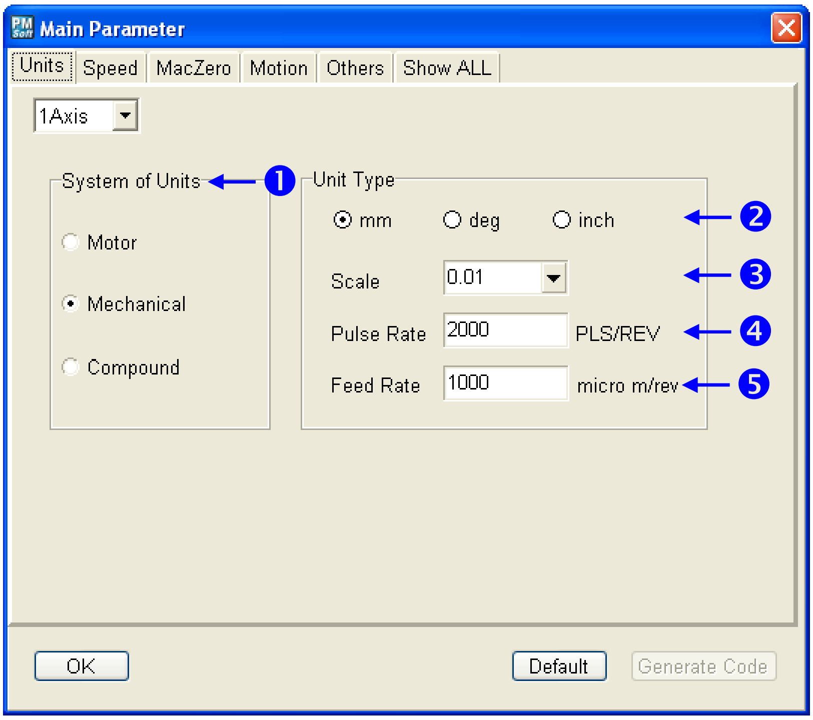
Unit Type: ● mm ○ deg ○ inch

Scale: 1

Pulse Rate: 2000

Feed Rate: 1000

- **Units**: PMSoft provides three system units. Please refer to DVP-PM Application Manual for more information about setting the system units



1Axis

System of Units: ○ Motor ● Mechanical ○ Compound

Unit Type: ● mm ○ deg ○ inch

Scale: 0.01

Pulse Rate: 2000 PLS/REV

Feed Rate: 1000 micro m/rev

❶ Users can choose s system unit. If the **Motor** option button is selected, a pulse is a unit. If the **Mechanical** option button is selected, a millimeter, a degree, or an inch is a unit. If the **Compound** option button is selected, a millimeter, a degree, or an inch is a position unit, and a pulse is a speed unit.

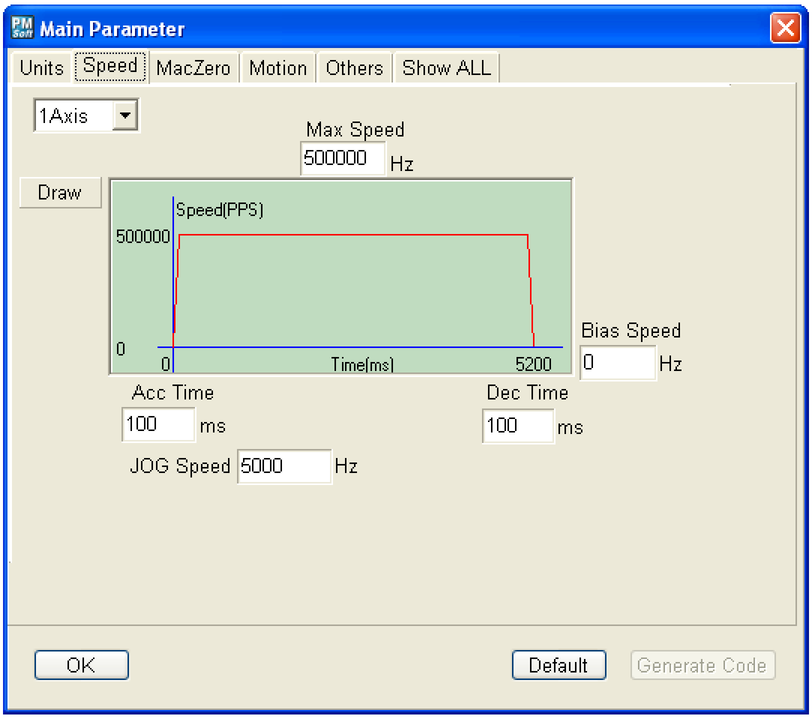
❷ Users can select the **mm** option button, the **deg** option button, or the **inch** option button.

❸ All the information about the positions set must be multiplied by the scale selected.

❹ Users can set the number of pulses it takes for an axis to rotate once (resolution).

❺ Users can set the distance for which the controlled device moves (lead).

- **Speed**: Users can set the speed parameters needed for controlling motion. After the users finish setting the speed parameters, they can click **Draw** to refresh the line chart about the relation between the speed and the time. Please refer to DVP-PM Application Manual for more information about setting speed parameters.



1Axis

Max Speed 500000 Hz

Bias Speed 0 Hz

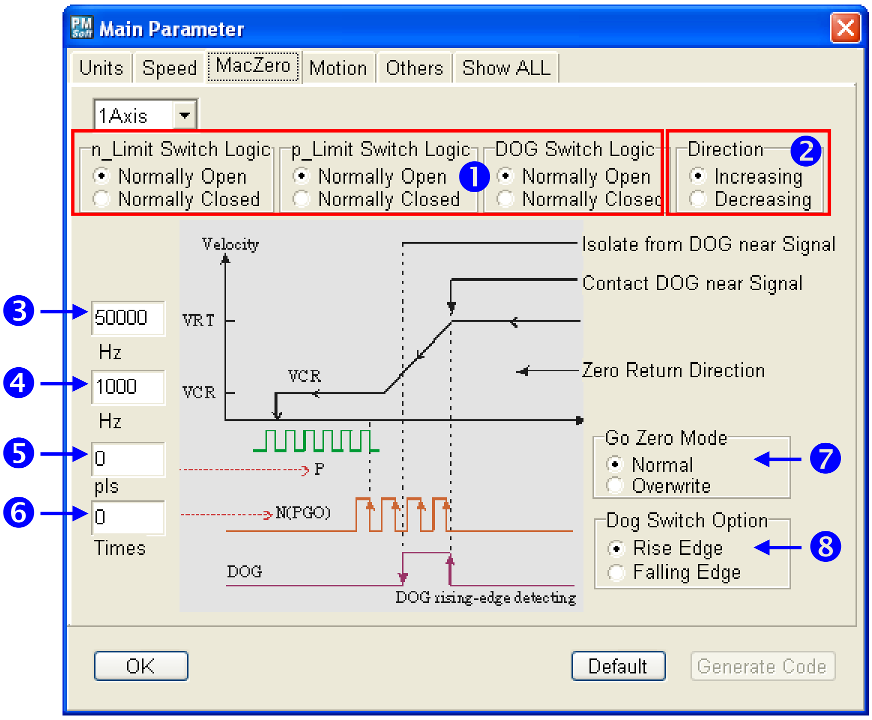
Acc Time 100 ms

Dec Time 100 ms

JOG Speed 5000 Hz

- **Max Speed**: Users can set the maximum speed at which an axis rotates.
- **Bias Speed**: Users can set the start-up speed at which an axis rotates. The minimum start-up speed is 0 Hz.
- **JOG Speed**: Users can set a JOG speed.
- **Acc Time**: Users can set the time it takes for the start-up speed to increase to the maximum speed.
- **Dec Time**: Users can set the time it takes for the maximum speed to decrease to the start-up speed.

- **MacZero**: The parameters related to returning to zero are described below. Please refer to DVP-PM Application Manual for more information about setting the parameters.



1Axis

n_Limit Switch Logic: ● Normally Open ○ Normally Closed

p_Limit Switch Logic: ● Normally Open ○ Normally Closed

DOG Switch Logic: ● Normally Open ○ Normally Closed

Direction: ● Increasing ○ Decreasing

50000 Hz

1000 Hz

0 pls

0 Times

Go Zero Mode: ● Normal ○ Overwrite

Dog Switch Option: ● Rise Edge ○ Falling Edge

❶ Users can set the polarity of the negative limit switch, the polarity of the positive limit switch, and the polarity of the DOG switch.

❷ When the motor returns to zero, it rotates in the positive direction, or in the negative direction.

❸ **VRT**: Speed of returning to zero

❹ **VCR**: Users can set the deceleration of returning to zero. When the motor returns to zero, it rotates at the speed of returning to zero (VRT). After the DOG signal is triggered, the motor will begin to decelerate the return to zero.

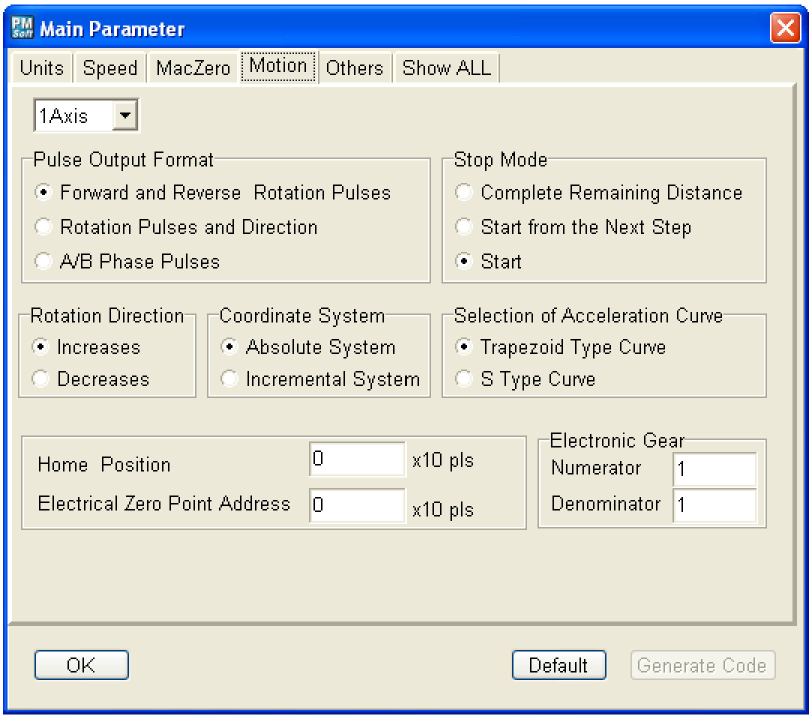
❺ **P**: The motor will stop rotating after a certain number of pulses are generated. After the DOG is signal is triggered, the motor will not be at zero until P pulses are generated.

❻ **N**: The motor will stop rotating after a certain number of zero signals (PG0) are generated by the encoder of the motor. After the DOG is signal is triggered, the motor will not be at zero until N zero signals (PG0) are generated.

❼ Users can select a mode of returning to zero mode. If the **Normal** option button in the **Go Zero Mode** section is selected, the motor will stop rotating after N zero signals and P pulses are generated. If the **Overwrite** option button in the **Go Zero Mode** section is selected, the motor will stop rotating after N zero signals or P pulses are generated.

❽ If the **Rise Edge** option button in the **Dog Switch Option** section is selected, the circuit is rising edge-triggered. If the **Falling Edge** option button in the **Dog Switch Option** section is selected, the circuit is falling edge-triggered.

- **Motion:** The parameters related to motion are described below. Please refer to DVP-PM Application Manual for more information about setting the parameters.



1Axis

Pulse Output Format
● Forward and Reverse Rotation Pulses
○ Rotation Pulses and Direction
○ A/B Phase Pulses

Stop Mode
○ Complete Remaining Distance
○ Start from the Next Step
● Start

Rotation Direction
● Increases
○ Decreases

Coordinate System
● Absolute System
○ Incremental System

Selection of Acceleration Curve
● Trapezoid Type Curve
○ S Type Curve

Home Position: 0 x10 pls
Electrical Zero Point Address: 0 x10 pls

Electronic Gear
Numerator: 1
Denominator: 1

➢ **Pulse Output Format**: There are three types of pulse outputs.

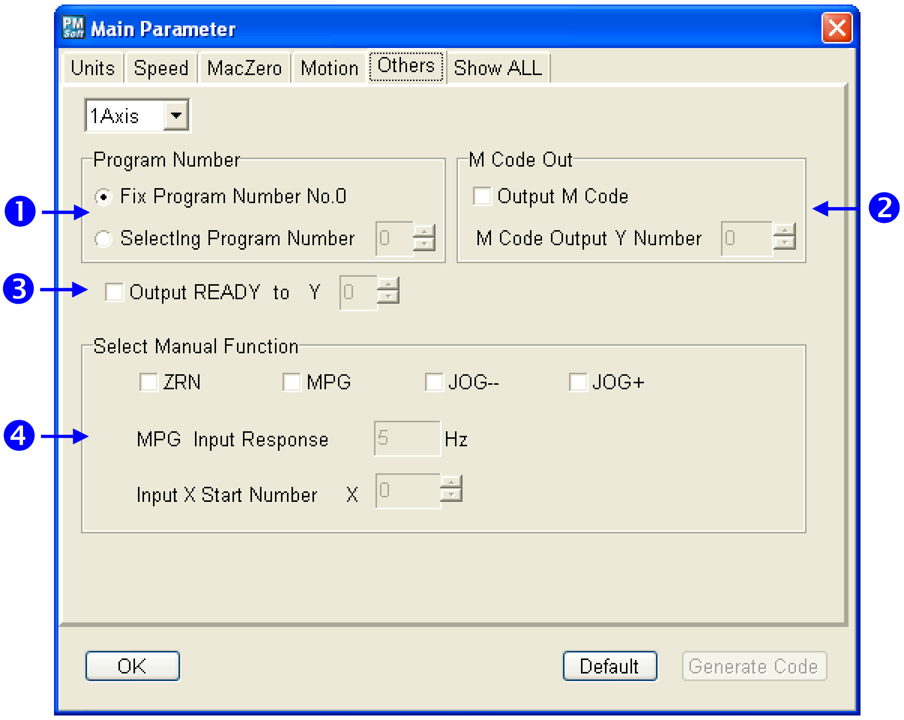
➢ **Stop Mode**: Mode of restarting the motion control after the motion control pauses

➢ **Rotation Direction**: If the motor rotates in the positive direction, the present address increases. If the motor rotates in the negative direction, the present address decreases.

➢ **Coordinate System**: An absolute coordinate is the distance from 0. If the target position is larger than the present position, the motor will rotate in the positive position. If the target position is less than the present position, the motor will rotate in the negative position. An incremental coordinate is the distance from the present position. If an incremental coordinate is a positive, the motor will rotate in the positive direction. If an incremental coordinate is a negative, the motor will rotate in the negative direction.

- **Selection of Acceleration Curve**: Users can select the **Trapezoid Type Curve** option button, or the **S Type Curve** option button.
- **Home Position**: Users can set a machine zero point.
- **Electrical Zero Point Address**: Users can set an electrical zero point.
- **Electronic Gear**: Users can set a gear ration. They can type a numerator and a denominator in the **Electronic Gear** section.

- **Others**: Users can select a motion program number (Ox0~Ox99), set the output of M-codes, and the manual function. Please refer to DVP-PM Application Manual for more information about setting the parameters.



1Axis

Program Number
● Fix Program Number No.0
○ SelectIng Program Number 0

M Code Out
□ Output M Code
M Code Output Y Number 0

□ Output READY to Y 0

Select Manual Function
□ ZRN □ MPG □ JOG-- □ JOG+
MPG Input Response 5 Hz
Input X Start Number X 0

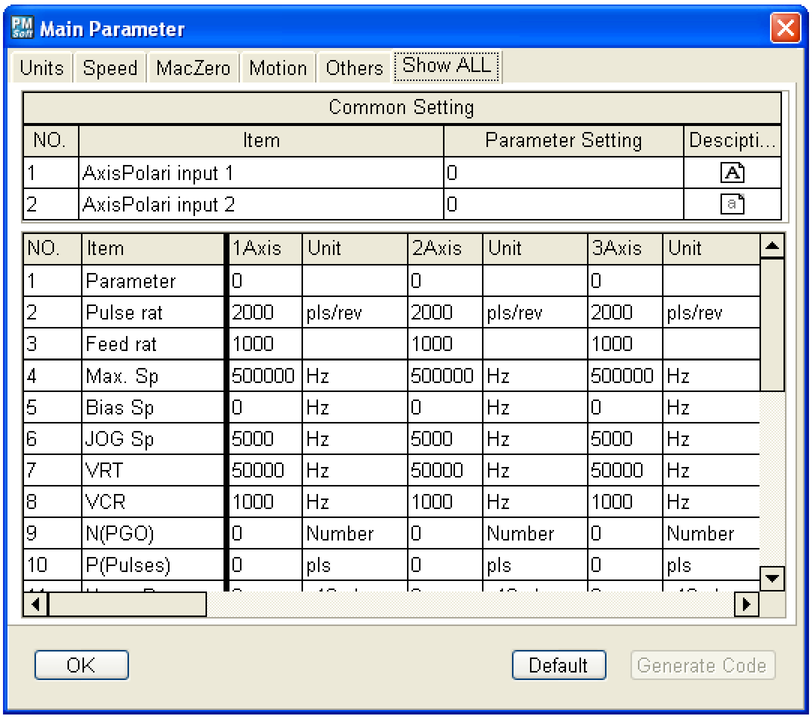
❶ Users can select a motion subroutine which can be triggered by an external signal.

❷ When a M-code is executed, the M-code is displayed in the corresponding Y device.

❸ If the motion subroutine is ready, the corresponding output device will be ON.

❹ Users can type a **frequency** in the **MPG Input Response** box, or select a number in the **Input X Start Number** box. After the **ZRN** checkbox, the **JOG-** checkbox, and the **JOG+** checkbox are selected, and 0 in the **Input X Start Number** box is selected, the motor will return to zero if X0 is ON, the motor will jog in the positive direction if X1 is ON, and the motor will jog in the negative direction if X2 is ON.

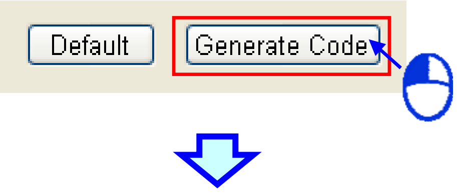
- **Show All**: Motion parameters and system operation parameters are displayed on this page. Users can change the values of the parameters in the tables. After the users click A for a parameter, they can see the description of the parameter. The parameters in the tables correspond to special data registers. The parameters are shown below.

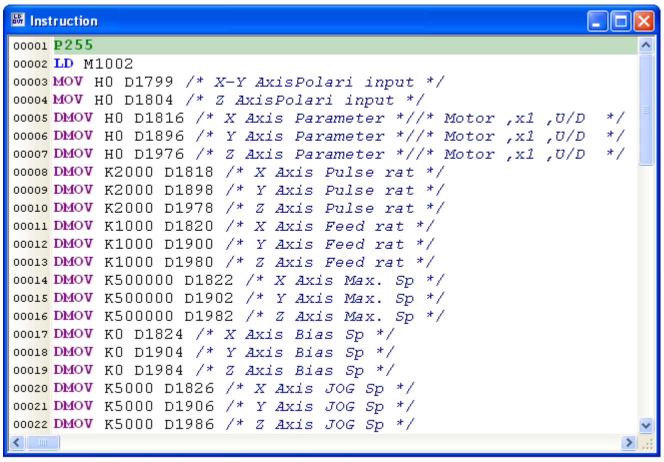


Common Setting			
NO.	Item	Parameter Setting	Descipti...
1	AxisPolari input 1	0	
2	AxisPolari input 2	0	

NO.	Item	1Axis	Unit	2Axis	Unit	3Axis	Unit
1	Parameter	0		0		0	
2	Pulse rat	2000	pls/rev	2000	pls/rev	2000	pls/rev
3	Feed rat	1000		1000		1000	
4	Max. Sp	500000	Hz	500000	Hz	500000	Hz
5	Bias Sp	0	Hz	0	Hz	0	Hz
6	JOG Sp	5000	Hz	5000	Hz	5000	Hz
7	VRT	50000	Hz	50000	Hz	50000	Hz
8	VCR	1000	Hz	1000	Hz	1000	Hz
9	N(PGO)	0	Number	0	Number	0	Number
10	P(Pulses)	0	pls	0	pls	0	pls

The **Generate Code** button will be displayed only if the **Instruction** window is opened. The function of the code generated is that the values of the parameters for the axes are moved to special data registers by means of the instruction MOV or DMOV. The code is placed in a program section. After users click **Generate Code**, a code will be generated automatically, and will be placed in the **P255** section in the **Instruction** window. The users can cut the code, and paste it into another program section (the **O100** section, the Ox section, or the P section).

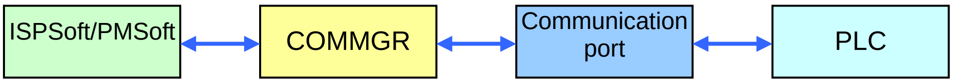




```
P255
LD  M1002
MOV  H0 D1799 /* X-Y AxisPolari input */
MOV  H0 D1804 /* Z AxisPolari input */
DMOV  H0 D1816 /* X Axis Parameter *//* Motor ,x1 ,U/D  */
DMOV  H0 D1896 /* Y Axis Parameter *//* Motor ,x1 ,U/D  */
DMOV  H0 D1976 /* Z Axis Parameter *//* Motor ,x1 ,U/D  */
DMOV  K2000 D1818 /* X Axis Pulse rat */
DMOV  K2000 D1898 /* Y Axis Pulse rat */
DMOV  K2000 D1978 /* Z Axis Pulse rat */
DMOV  K1000 D1820 /* X Axis Feed rat */
DMOV  K1000 D1900 /* Y Axis Feed rat */
DMOV  K1000 D1980 /* Z Axis Feed rat */
DMOV  K500000 D1822 /* X Axis Max. Sp */
DMOV  K500000 D1902 /* Y Axis Max. Sp */
DMOV  K500000 D1982 /* Z Axis Max. Sp */
DMOV  K0 D1824 /* X Axis Bias Sp */
DMOV  K0 D1904 /* Y Axis Bias Sp */
DMOV  K0 D1984 /* Z Axis Bias Sp */
DMOV  K5000 D1826 /* X Axis JOG Sp */
DMOV  K5000 D1906 /* Y Axis JOG Sp */
DMOV  K5000 D1986 /* Z Axis JOG Sp */
```

2.3 Communication Setting

The communication between ISPSoft/PMSoft and a Delta PLC is shown below. The communication manager **COMMGR** is a communication interface between ISPSoft/PMSoft and a PLC. This section introduces how to create a connection between ISPSoft/PMSoft and a PLC, and complete a basic test.

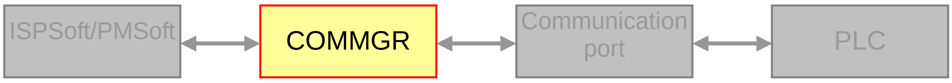


***1. Please refer to section 1.2 for more information about the installation of COMMGR.**

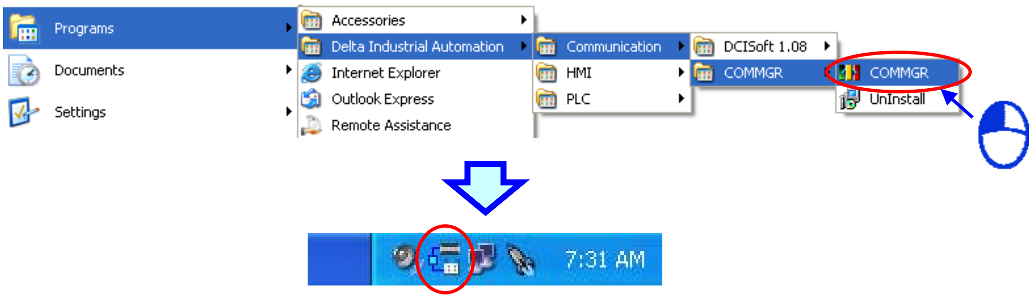
***2. Different connections are applicable to different software. Please refer to the following sections for more information.**

***3. COMMGR is used with PMSoft version 2.04 (and above), or ISPSoft version 2.0 (and above). An older version of PMSoft or an older version of ISPSoft still communicates with a PLC in a traditional way.**

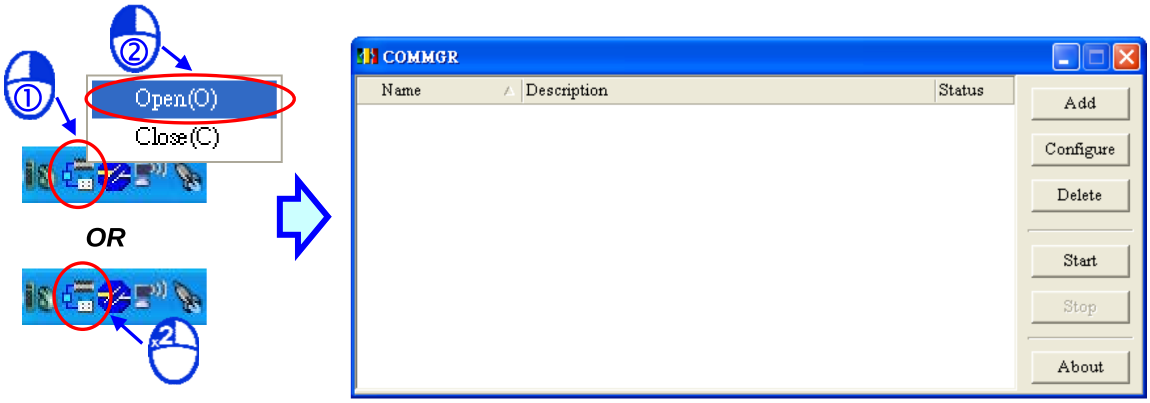
2.3.1 Starting / Closing COMMGR



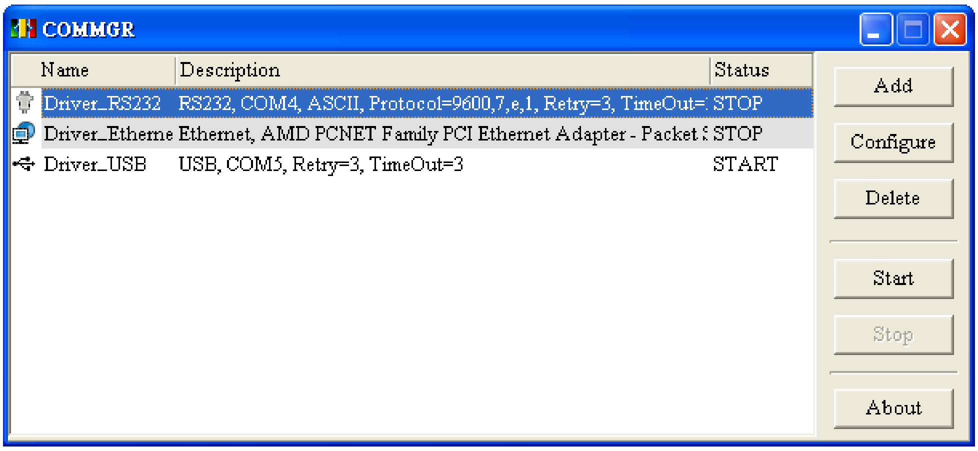
After COMMGR is installed on a computer successfully, a shortcut to COMMGR will be created on the **Start** menu. Users can click the shortcut on the **Start** menu to start COMMGR. After the installation of COMMGR is complete, the users have to start it by themselves. However, whenever the computer is restarted, the system starts COMMGR automatically, and the icon representing COMMGR is displayed on the system tray. If the icon representing COMMGR is not displayed on the system tray, the users can start COMMGR by clicking the shortcut on the **Start** menu.



After COMMGR is started successfully, the icon representing COMMGR will be displayed on the system tray. The users can open the **COMMGR** window by double-clicking the icon. They can also open the **COMMGR** window by right-clicking the icon, and clicking **Open** on the context menu.



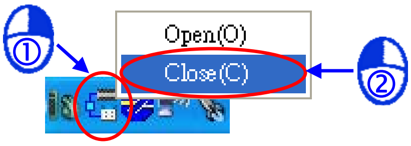
The **COMMGR** window is shown below. The drivers created are listed in the window. ISPSoft/PMSoft connects to COMMGR by means of specifying a driver. The users can manage the drivers through the buttons at the right side of the window. Please refer to the section below for more information about managing drivers.



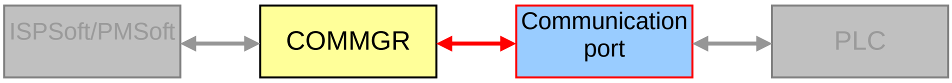
COMMGR

Name	Description	Status
Driver_RS232	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=	STOP
Driver_Etherne	Ethernet, AMD PCNET Family PCI Ethernet Adapter - Packet S	STOP
Driver_USB	USB, COM5, Retry=3, TimeOut=3	START

The users can close the **COMMGR** window by clicking ☒ or ▁ in the upper right corner of the window. However, the icon representing COMMGR is still displayed on the system tray. If the users want to close COMMGR completely, they can right-click the icon displayed on the system tray, and click **Close** on the context menu.



2.3.2 Managing Drivers



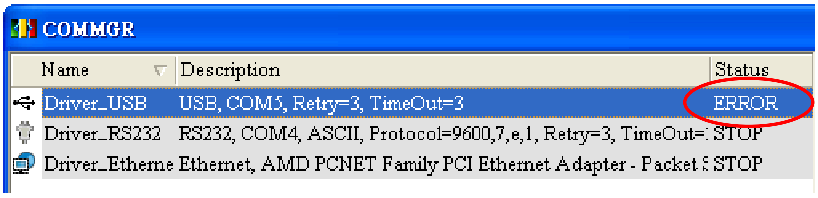
ISPSoft/PMSoft

COMMGR

Communication port

PLC

The drivers listed in the **COMMGR** window connect programs and communication ports. If the status of a driver displayed in the **COMMGR** window is **START**, COMMGR connects to the communication port specified by the driver. Whenever the computer is restarted, COMMGR starts the driver automatically. However, if COMMGR can not connect to the communication port specified by a driver, COMMGR automatically stops the driver, the status of the driver displayed in the window is **ERROR**, and the icon representing COMMGR on the system tray is marked with a red cross.

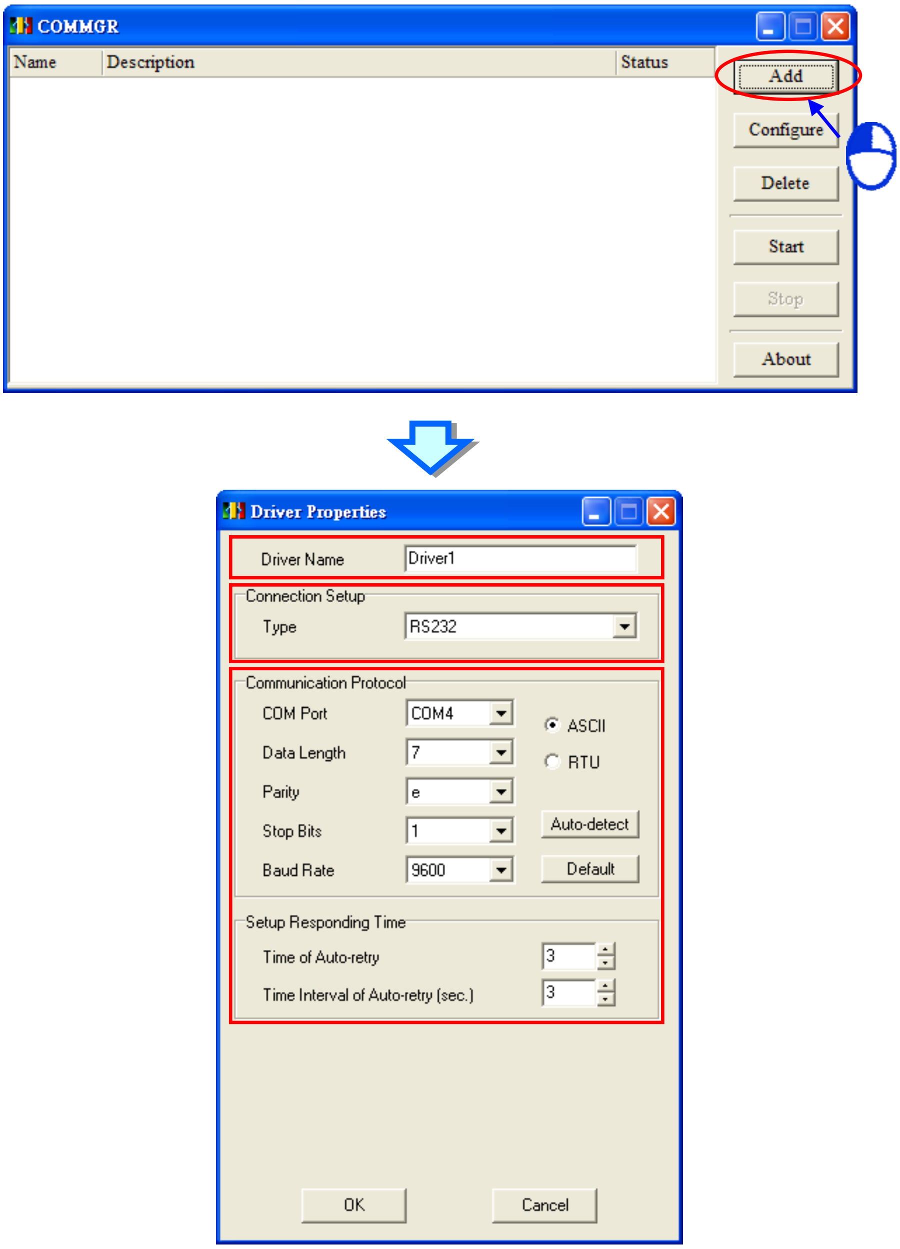


COMMGR

Name	Description	Status
Driver_USB	USB, COM5, Retry=3, TimeOut=3	ERROR
Driver_RS232	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=	STOP
Driver_Etherne	Ethernet, AMD PCNET Family PCI Ethernet Adapter - Packet S	STOP

2.3.3 Creating a Connection—Creating a Driver

Click **Add** in the **COMMGR** window to open the **Driver Properties** window.



Driver Name: Driver1

Connection Setup

Type: RS232

Communication Protocol

COM Port: COM4

Data Length: 7

Parity: e

Stop Bits: 1

Baud Rate: 9600

● ASCII

○ RTU

Setup Responding Time

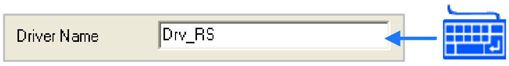
Time of Auto-retry: 3

Time Interval of Auto-retry (sec.): 3

The steps of creating a driver are as follows.

(1) **Driver name**

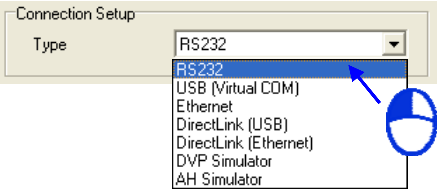
Users can type a driver name in the **Driver Name** box. ISPSoft/PMSoft specifies the driver which will be used according to the driver name typed. It is suggested that users should set identifiable driver names for the drivers created.



Driver Name: Drv_RS

(2) **Connection type**

Users can select a connection type in the **Type** drop-down list box. The connection types supported by COMMGR are as follows.



Connection Setup

Type: RS232

- **RS-232**

  A computer communicates with a PLC through a communication port on the computer. Owing to the fact that an AH500 series motion control module converts USB to RS232, RS232 will be selected if the USB port of an AH500 series motion control module is used.

- **USB (Virtual COM)**

  A computer can connect to a Delta PLC equipped with a USB port through a USB cable. However, users have to make sure that a USB driver is installed on a computer before the computer connects to a PLC equipped with a USB port. Please refer to appendix A for more information about installing a USB driver.

- **Ethernet**

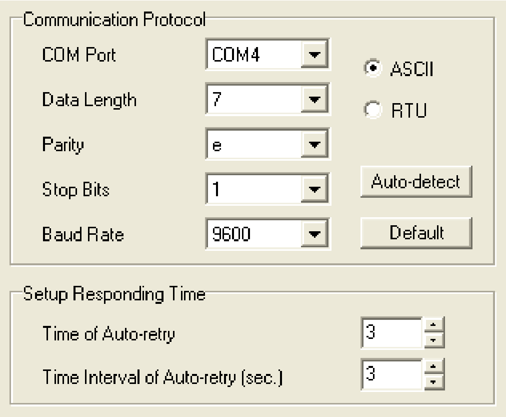
  A computer communicates with a PLC through an Ethernet port on the computer.

- **DirectLink (USB) & DirectLink (Ethernet)**

  They are the connection functions provided by Delta human-machine interfaces (HMI). If a PLC connects to a HMI normally, a computer can connect to the HMI through a USB cable or Ethernet, and connect to the PLC indirectly. Please refer to manuals for Delta human-machine interfaces for more information about setting a connection.

(3) **Communication parameters**

Communication parameters are set according to the connection type selected. Different connection types have different communication parameters. The setting of the parameters for the different connection types is described in the following sections.



Communication Protocol

COM Port: COM4

Data Length: 7

Parity: e

Stop Bits: 1

Baud Rate: 9600

● ASCII

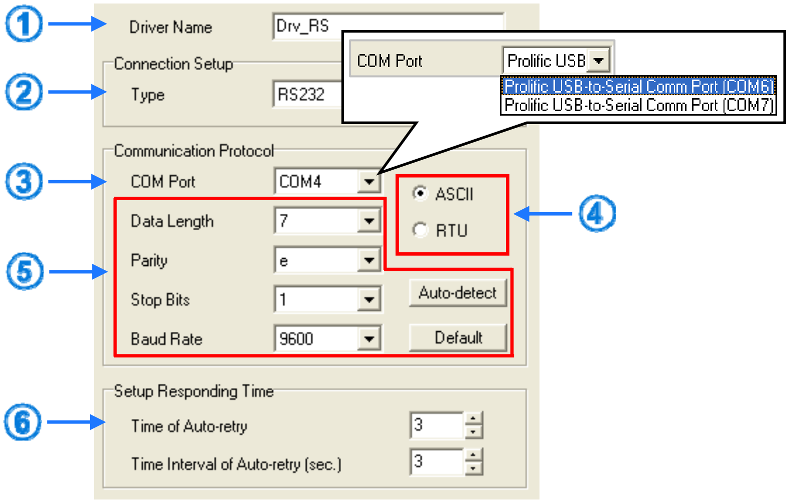
○ RTU

Setup Responding Time

Time of Auto-retry: 3

Time Interval of Auto-retry (sec.): 3

2.3.4 Setting Communication Parameters for RS232



Driver Name: Drv_RS

Connection Setup

Type: RS232

COM Port: Prolific USB
- Prolific USB-to-Serial Comm Port (COM6)
- Prolific USB-to-Serial Comm Port (COM7)

Communication Protocol

COM Port: COM4

Data Length: 7

Parity: e

Stop Bits: 1

Baud Rate: 9600

● ASCII ○ RTU

Setup Responding Time

Time of Auto-retry: 3

Time Interval of Auto-retry (sec.): 3

① Users can type a driver name in the **Driver Name** box.

② Select **RS232** in the **Type** drop-down list box in the **Connection Setup** section.

③ Select a RS232 communication port in the **COM Port** drop-down list box. Each item in the **COM Port** drop-down list box is composed of a device name and a communication port number. The communication ports in the **COM Port** drop-down list box are the same as the communication ports in the **Device Manager** window.

④ The communication format can be **ASCII** or **RTU**.

⑤ The communication protocol for exchanging data through the communication port selected must be the same as the communication protocol for exchanging data through a communication port on a device connected. If users click **Default**, all communication parameters will return to the default values.

If users do not know the communication protocol for exchanging data through a communication port on a device connected, the users can connect the device to a RS232 communication port selected with a RS232 cable, and click **Auto-detect** to automatically detect the communication protocol. If the communication protocol is detected successfully, the related communication parameters in the **Driver Properties** window are set. However, when the communication protocol is detected automatically, the **COM Port** parameter and the **ASCII/RTU** parameter are not detected. As a result, the users have to set the **COM Port** parameter and the **ASCII/RTU** parameter before clicking **Auto-detect**.

⑥ Users can select the number of times the sending of a command is retried if a connection error occurs in the **Time of Auto-retry** box, and select an interval of retrying the sending of a command in the **Time Interval of Auto-retry** box.

***1. When the Driver Properties window is opened, the information about the communication ports in the Device Manager window is retrieved once. However, the information in the COM Port drop-down list box will not be updated. If a device is added to the computer system after the Driver Properties window is opened, the device will not be displayed in the COM Port drop-down list box. Users have to close the Driver Properties window, and open it again.**

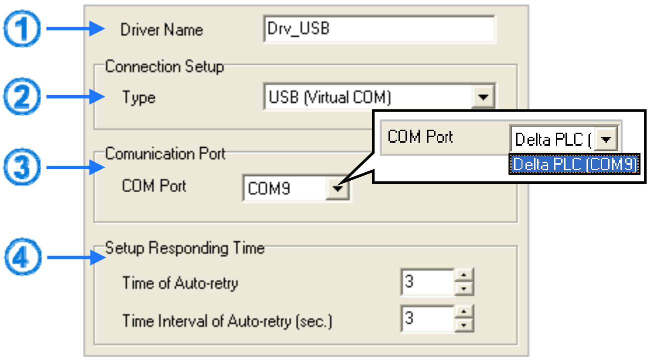
***2. If the USB port of an AH500 series motion control module is used, RS232 in the Type drop-down list box must be selected. Please refer to section 2.3.3 for more information.**

2.3.5 Setting Communication Parameters for USB (Virtual COM)

If the USB port of an AH500 series CPU module is used, USB in the **Type** drop-down list box must be selected. Users have to make sure of the items below before opening the **Driver Properties** window.

(a) A USB driver is installed on the computer. Please refer to appendix A for more information about installing a USB driver.

(b) The computer is connected to the PLC through a USB cable. The computer and the PLC operate normally.



Driver Name: Drv_USB

Connection Setup

Type: USB (Virtual COM)

Comunication Port

COM Port: COM9

COM Port: Delta PLC (

Delta PLC (COM9)

Setup Responding Time

Time of Auto-retry: 3

Time Interval of Auto-retry (sec.): 3

① Users can type a driver name in the **Driver Name** box.

② Select **USB (Virtual COM)** in the **Type** drop-down list box in the **Connection Setup** section.

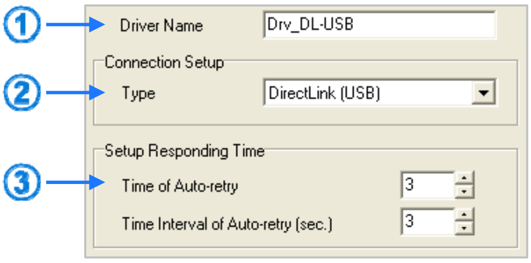
③ Select a communication port in the **COM Port** drop-down list box. If users have made sure of the two items above, the PLC which is connected and its communication port will be displayed in the **COM Port** drop-down list box.

④ Users can select the number of times the sending of a command is retried if a connection error occurs in the **Time of Auto-retry** box, and select an interval of retrying the sending of a command in the **Time Interval of Auto-retry** box.

***1. Please refer to appendix A or technical documents for more information about installing the USB drivers for AH500 series motion control modules and CPU modules.**

***2. If the USB port of an AH500 series motion control module is used, RS232 in the Type drop-down list box must be selected. Please refer to section 2.3.3 for more information.**

2.3.6 Setting Communication Parameters for DirectLink (USB)



Driver Name: Drv_DL-USB

Connection Setup

Type: DirectLink (USB)

Setup Responding Time

Time of Auto-retry: 3

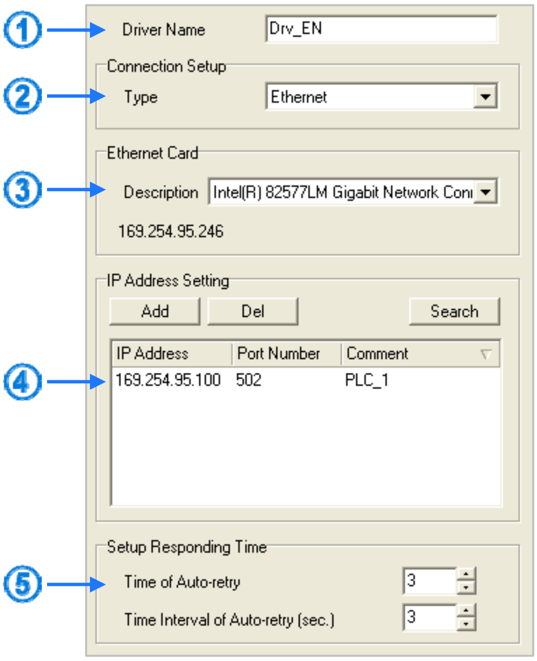
Time Interval of Auto-retry (sec.): 3

① Users can type a driver name in the **Driver Name** box.

② Select **DirectLink (USB)** in the **Type** drop-down list box in the **Connection Setup** section.

③ Users can select the number of times the sending of a command is retried if a connection error occurs in the **Time of Auto-retry** box, and select an interval of retrying the sending of a command in the **Time Interval of Auto-retry** box.

2.3.7 Setting Communication Parameters for Ethernet



Driver Name: Drv_EN

Connection Setup

Type: Ethernet

Ethernet Card

Description: Intel(R) 82577LM Gigabit Network Con

169.254.95.246

IP Address Setting

IP Address	Port Number	Comment
169.254.95.100	502	PLC_1

Setup Responding Time

Time of Auto-retry: 3

Time Interval of Auto-retry (sec.): 3

① Users can type a driver name in the **Driver Name** box.

② Select **Ethernet (USB)** in the **Type** drop-down list box in the **Connection Setup** section.

③ Select a network interface card in the **Description** drop-down list box. The IP address assigned to the network interface card selected is displayed in the lower left corner of the **Ethernet Card** section.

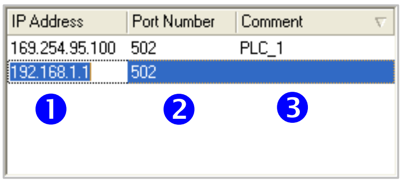
④ Owing to the characteristics of Ethernet, a computer can communicate with all the devices on a network. Users can create IP addresses of the devices connected to this driver in the **IP Address Setting** section.

➢ After users click **Add** to add a new IP address to the list of IP addresses in the **IP Address Setting** section, they can type related information in the **IP Address** column, the **Port Number** column, and the **Comment** column.

❶ Users can type the IP addresses of the devices connected in this column.

❷ Users can type the communication port numbers specified.

❸ Users can type comments in this column.



IP Address	Port Number	Comment
169.254.95.100	502	PLC_1
192.168.1.1	502	

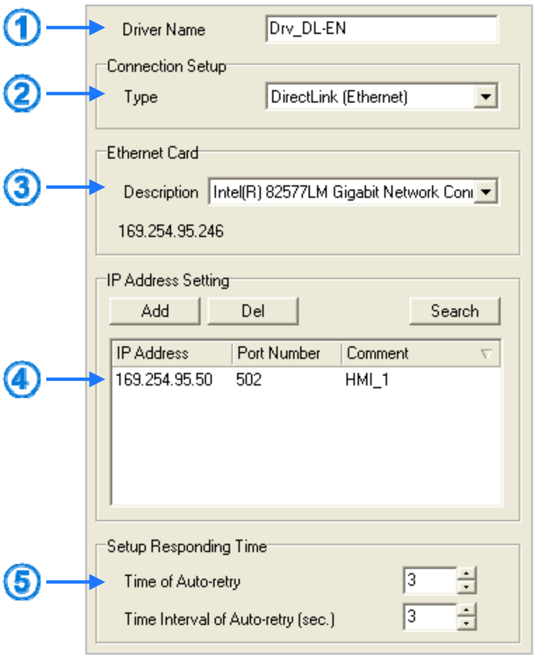
➢ After users select an IP address, they can click **Del** or press Delete on the keyboard to delete the IP address from the list.

➢ Some devices such as DVPEN01-SL and IFD9506 support the **Search** function. If these devices are in the same domain as the network interface card selected, they will be displayed in the **IP Address Setting** section after users click **Search** to search for IP addresses.

⑤ Users can select the number of times the sending of a command is retried if a connection error occurs in the **Time of Auto-retry** box, and select an interval of retrying the sending of a command in the **Time Interval of Auto-retry** box.

***. When the Driver Properties window is opened, the information about the network interface cards in the computer is retrieved once. However, the information in the Description drop-down list box will not be updated. If a network interface card is added to the computer system after the Driver Properties window is opened, the network interface card will not be displayed in the Description drop-down list box. Users have to close the Driver Properties window, and open it again.**

2.3.8 Setting Communication Parameters for DirectLink (Ethernet)



Driver Name: Drv_DL-EN

Connection Setup

Type: DirectLink (Ethernet)

Ethernet Card

Description: Intel(R) 82577LM Gigabit Network Conı

169.254.95.246

IP Address Setting

IP Address	Port Number	Comment
169.254.95.50	502	HMI_1

Setup Responding Time

Time of Auto-retry: 3

Time Interval of Auto-retry (sec.): 3

① Users can type a driver name in the **Driver Name** box.

② Select **DirectLink (Ethernet)** in the **Type** drop-down list box in the **Connection Setup** section.

③ Select a network interface card in the **Description** drop-down list box. The IP address assigned to the network interface card selected is displayed in the lower left corner of the **Ethernet Card** section.

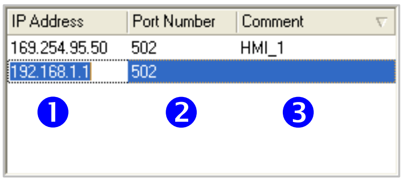
④ Owing to the characteristics of Ethernet, a computer can communicate with all the devices on a network. Users can create IP addresses of the devices connected to this driver in the **IP Address Setting** section.

- After users click **Add** to add a new IP address to the list of IP addresses in the **IP Address Setting** section, they can type related information in the **IP Address** column, the **Port Number** column, and the **Comment** column.

  ❶ Users can type the IP addresses of the devices connected in this column.

  ❷ Users can type the communication port numbers specified.

  ❸ Users can type comments in this column.



IP Address	Port Number	Comment
169.254.95.50	502	HMI_1
192.168.1.1	502	

- After users select an IP address, they can click **Del** or press Delete on the keyboard to delete the IP address from the list.
- Delta human-machine interfaces support the **Search** function. After users click **Search**, the

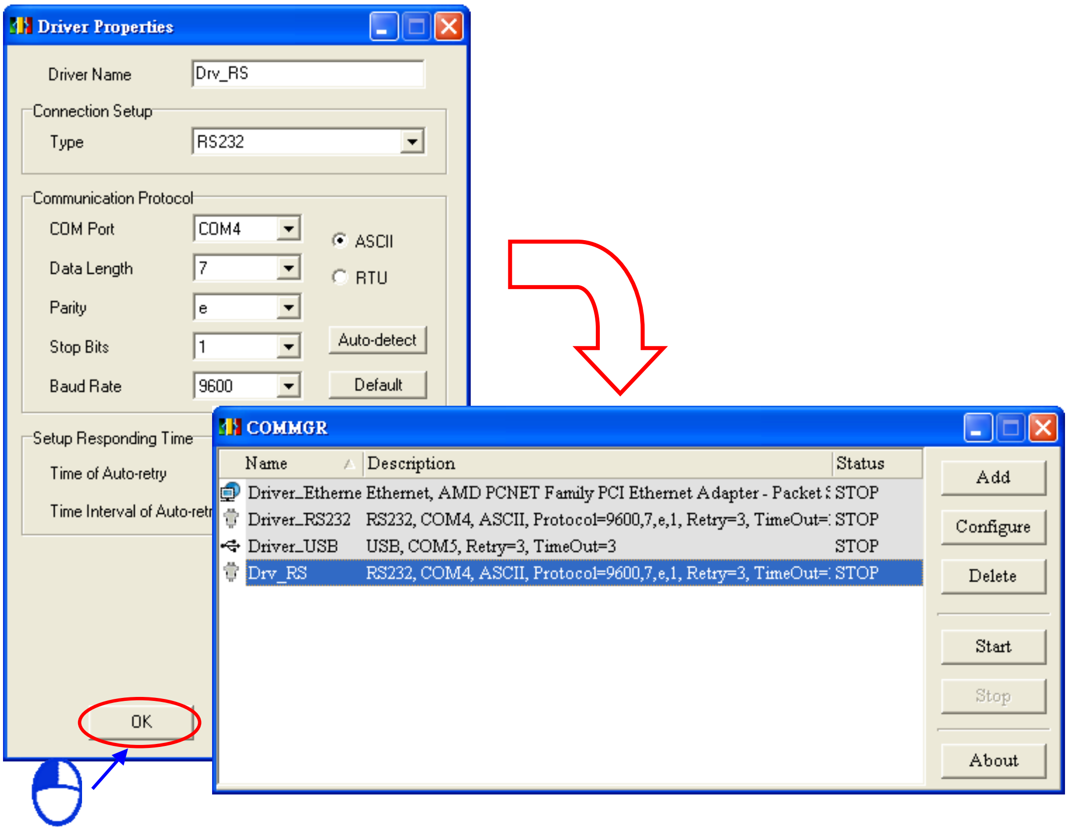
Delta human-machine interfaces in the same domain as the network interface card selected will be searched for, and the results will be displayed in the **IP Address Setting** section.

⑤ Users can select the number of times the sending of a command is retried if a connection error occurs in the **Time of Auto-retry** box, and select an interval of retrying the sending of a command in the **Time Interval of Auto-retry** box.

***. When the Driver Properties window is opened, the information about the network interface cards in the computer is retrieved once. However, the information in the Description drop-down list box will not be updated. If a network interface card is added to the computer system after the Driver Properties window is opened, the network interface card will not be displayed in the Description drop-down list box. Users have to close the Driver Properties window, and open it again.**

2.3.9 Completing the Setting of the Parameters for a Driver

After users complete the setting of the communication parameters for a driver and click **OK**, the values of the parameters for the driver will be displayed in the COMMGR window. Creating a driver is equivalent to creating a connection. The users can start or stop the driver according to the actual requirement. Please refer to the following section for more information about starting/stopping a driver.



Driver Properties

Driver Name: Drv_RS

Connection Setup

Type: RS232

Communication Protocol

COM Port: COM4

Data Length: 7

Parity: e

Stop Bits: 1

Baud Rate: 9600

● ASCII ○ RTU

Setup Responding Time

Time of Auto-retry

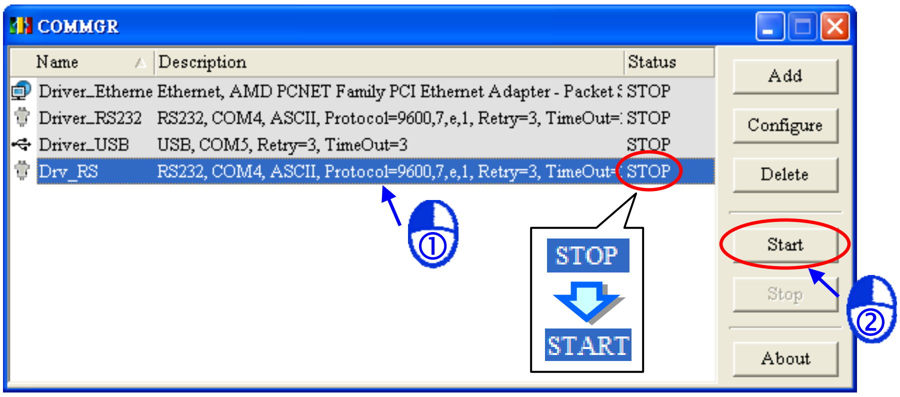
Time Interval of Auto-retr

COMMGR

Name	Description	Status
Driver_Etherne	Ethernet, AMD PCNET Family PCI Ethernet Adapter - Packet S	STOP
Driver_RS232	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=:	STOP
Driver_USB	USB, COM5, Retry=3, TimeOut=3	STOP
Drv_RS	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=:	STOP

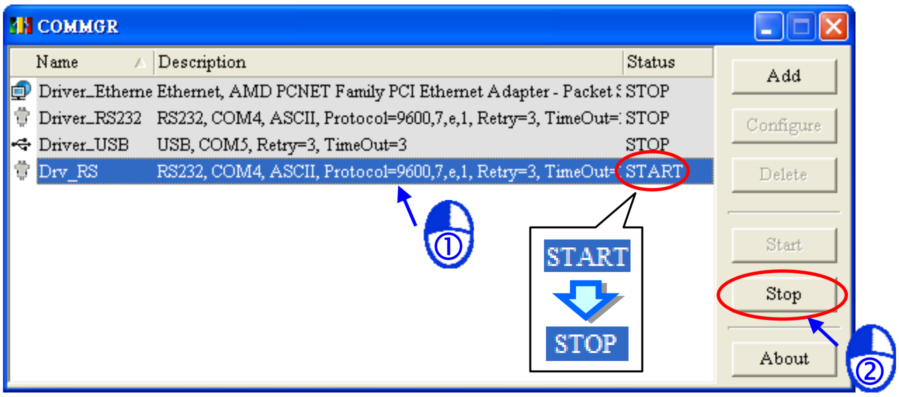
2.3.10 Starting / Stopping a Driver

If users want to start a driver, they have to connect a PLC to a communication port, select the driver in the **COMMGR** window, and click **Start**. If the status of the driver displayed in the window becomes **START**, the driver is started successfully. Please refer to section 2.3.12 for more information about connecting a PLC to a communication port.



Name	Description	Status
Driver_Etherne	Ethernet, AMD PCNET Family PCI Ethernet Adapter - Packet S	STOP
Driver_RS232	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=:	STOP
Driver_USB	USB, COM5, Retry=3, TimeOut=3	STOP
Drv_RS	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=	STOP

If users want to stop a driver, they can select the driver in the **COMMGR** window, and click **Stop**. If the status of the driver displayed in the window becomes **STOP**, the driver stops.



Name	Description	Status
Driver_Etherne	Ethernet, AMD PCNET Family PCI Ethernet Adapter - Packet S	STOP
Driver_RS232	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=:	STOP
Driver_USB	USB, COM5, Retry=3, TimeOut=3	STOP
Drv_RS	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=	START

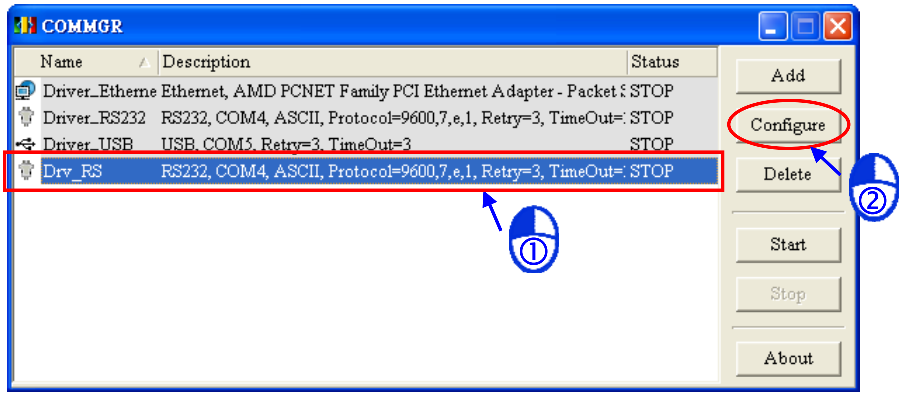
Additional remark

When users start or stop a RS232 driver, they have to notice the following items.

- If the same communication port number is assigned to different drivers, only one of these drivers can be started.
- If the communication port number specified does not exist, e.g. the cable converting USB to RS232 is removed, the driver can not be started.
- If a driver to which a communication port is assigned is started, software which is not connected through COMMGR can not use the communication port assigned to the driver. If the software wants to use the communication port, users have to stop the driver.
- If a communication port is used by software not connected through COMMGR, a driver to which the communication port is assigned can not be started. If users want to start the driver, they have to close the software, or the software has to use another communication port.

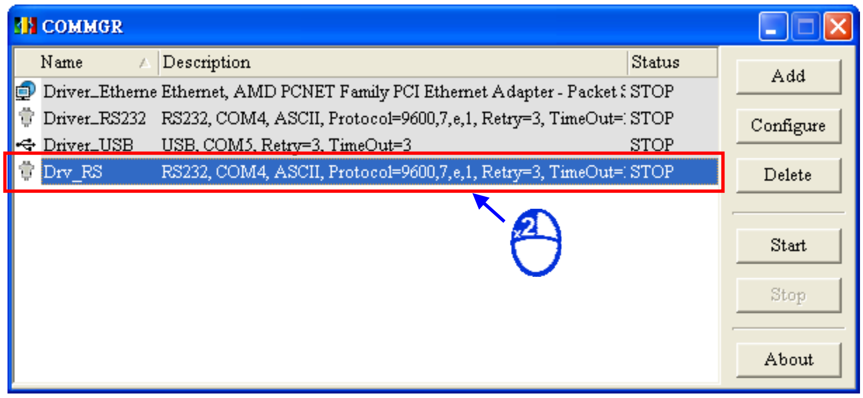
2.3.11 Configuring / Deleting a Driver

If users want to modify the values of the parameters for a driver, they have to stop the driver, and click **Configure**, or double-click the driver to open the **Driver Properties** window. The users can set the parameters in the **Driver Properties** window according to the descriptions in the previous sections.



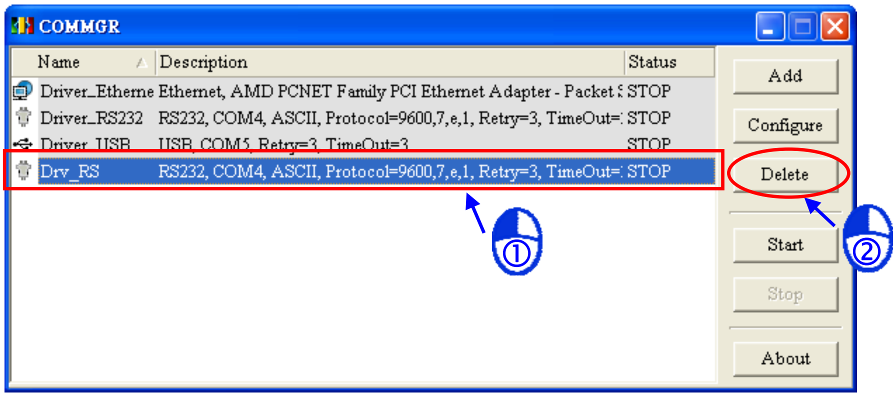
Name	Description	Status
Driver_Etherne	Ethernet, AMD PCNET Family PCI Ethernet Adapter - Packet S	STOP
Driver_RS232	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=	STOP
Driver_USB	USB, COM5, Retry=3, TimeOut=3	STOP
Drv_RS	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=	STOP

OR



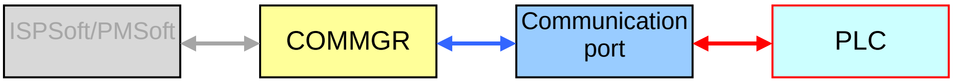
Name	Description	Status
Driver_Etherne	Ethernet, AMD PCNET Family PCI Ethernet Adapter - Packet S	STOP
Driver_RS232	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=	STOP
Driver_USB	USB, COM5, Retry=3, TimeOut=3	STOP
Drv_RS	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=	STOP

If users want to delete a driver, they have to make sure that the driver stops, select the driver, and click **Delete**, or press Delete on the keyboard to delete the driver.



Name	Description	Status
Driver_Etherne	Ethernet, AMD PCNET Family PCI Ethernet Adapter - Packet S	STOP
Driver_RS232	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=	STOP
Driver_USB	USB, COM5, Retry=3, TimeOut=3	STOP
Drv_RS	RS232, COM4, ASCII, Protocol=9600,7,e,1, Retry=3, TimeOut=	STOP

2.3.12 Connecting a PLC and a Communication Port



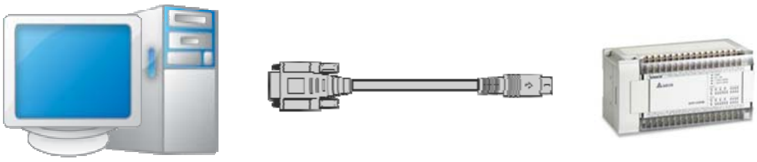
After all the setting is complete, users can connect a PLC to the communication port specified through a communication cable. ISPSoft/PMSoft can be connected to a motion controller in two ways. If a motion controller is regarded as an independent motion controller, and a computer is connected to the motion controller directly, the connection is called a direct connection. If a motion controller is an extension module in an AH500 system, a computer is connected to the CPU module in the AH500 system, and the computer is connected to the motion controller by means of a backplane in the AH500 system, the connection is called an indirect connection. Different connection types must be used with different software. Some common ways to connect a PLC to a communication port, and some points for attention are listed below. Please refer to operation manuals for more information about connecting PLCs to communication ports.

- Direct connection

  (1) DVP series motion controller (RS232)

  A computer is connected to a DVP series motion controller through a Delta communication cable. The connection type that the driver created uses is RS232.

  Note: Users have to make sure that the communication protocol for exchanging data through a driver is the same as the communication protocol for exchanging data through a communication port on a motion controller before they connect the driver to the motion controller.



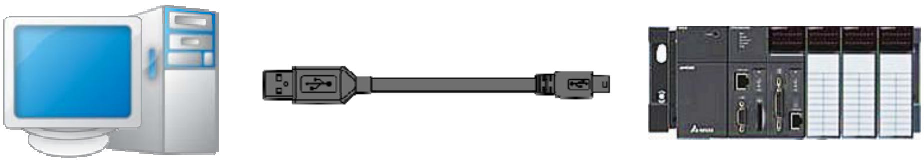
  (2) AH500 series motion control module (USB)

  A computer is connected to an AH500 series motion control module through a USB cable. Owing to the fact that an AH500 series motion control module converts USB to RS232, the connection type that the driver created uses must be RS232 if the USB port of an AH500 series motion control module is used.

  Note:

  (a) Users have to make sure that the USB driver for an AH500 series motion control module has been installed on a computer. Please refer to appendix A for more information.

  (b) Users have to make sure that the communication protocol for exchanging data through a driver is the same as the communication protocol for exchanging data through a communication port on an AH500 series motion control module before they connect the driver to the AH500 series motion control module.



  (3) AH500 series motion control module or DVP series motion controller equipped with the function card DVP-FPMC (Ethernet)

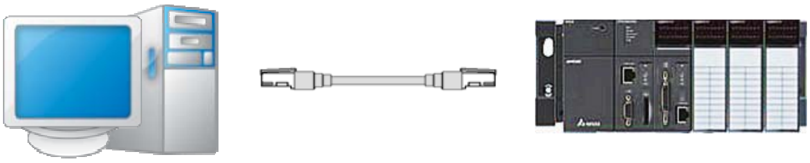
  An AH500 series motion control module is equipped with an Ethernet port to which a RJ45 cable can be connected. DVP series motion controller equipped with the function card DVP-FPMC can be connected to a computer through a network cable. A computer can connect to an AH500 series motion control module or a DVP series motion controller in two ways. One way to connect a computer and an AH500 series motion control module/a DVP series motion controller is connecting them to the same domain by means of a switching hub. The other way to connect a computer and an AH500 series motion control module/a DVP series motion controller is connecting them by means of a network cable (no jumper wire is needed). The connection type that the driver created uses is Ethernet.

  Note:

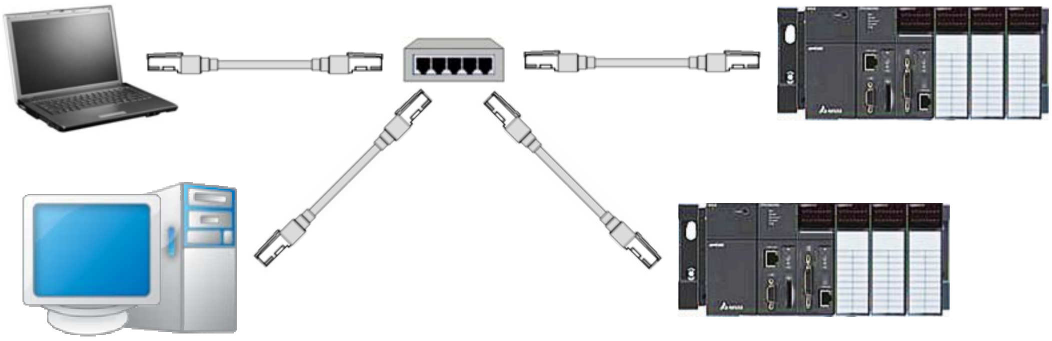
  (a) Before users connect an AH500 series motion control module/a DVP series motion controller to a computer, they have to make sure that the network created is normal, and the computer and the AH500 series motion control module/DVP series motion controller are in the same domain.

(b) Users have to make sure that the values of the Ethernet parameters in an AH500 series motion control module/DVP-FPMC are correct. The default values of the Ethernet parameters in an AH500 series motion control module/DVP-FPMC are shown below. If users want to change the IP address of an AH500 series motion control module/a DVP series motion controller, or the port number of an AH500 series motion control module/a DVP series motion controller, they have to move values to special data registers in the AH500 series motion control module/DVP series motion controller by means of instructions.

	Value
IP address	192.168.0.100
Port number	1024



OR



- Indirect connection

  (1) AH500 series CPU module (RS232)

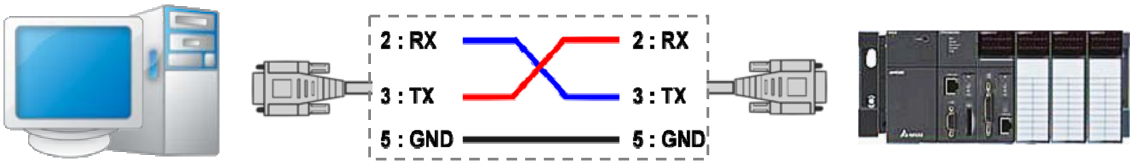
  Owing to the fact that an AH500 series CPU module is quipped with standard communication ports, users have to connect an AH500 series CPU module to a computer with a RS232 cable depicted in the figure below, or with an adapter into which jumper wires are built. The connection type that the driver created uses is RS232.

  Note:

  (a) Before users connect the AH500 series CPU module in an AH500 system to a computer, they have to make sure that the power supply modules, the CPU module, the network modules, and the backplanes in the AH500 system are installed properly.

  (b) Users have to make sure that the communication protocol for exchanging data through a driver is the same as the communication protocol for exchanging data through a communication port on an AH500 series CPU module before they connect the driver to the AH500 series CPU module.

  (c) The communication ports on an AH500 series CPU module can be RS232 communication ports, RS485 communication ports, or RS422 communication ports. Users have to make sure that a communication port on an AH500 series CPU module is a RS232 port before they connect the communication port to a computer.



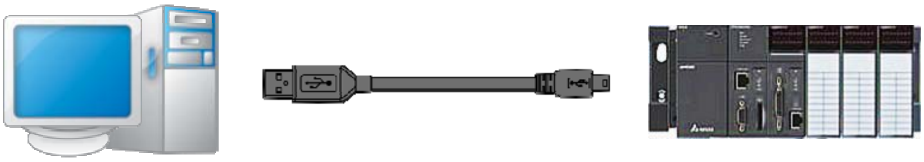
(2) AH500 series CPU module (USB)

A computer is connected to an AH500 series CPU module through a USB cable. An AH500 series CPU module is equipped with a type B mini USB interface. The connection type that the driver created uses must be USB (Virtual COM).

Note:

(a) Before users connect the AH500 series CPU module in an AH500 system to a computer, they have to make sure that the power supply modules, the CPU module, the network modules, and the backplanes in the AH500 system are installed properly.

(b) Users have to make sure that the USB driver for an AH500 series CPU module has been installed on a computer. Please refer to appendix A for more information.



(3) AH500 series CPU module (Ethernet)

An AHCPU5xx-EN series CPU module is quipped with an Ethernet port to which a RJ45 cable can be connected, and can be connected to a computer through a network cable. A computer can connect to an AHCPU5xx-EN series CPU module in two ways. One way to connect a computer and an AHCPU5xx-EN series CPU module is connecting them to the same domain by means of a switching hub. The other way to connect a computer and an A AHCPU5xx-EN series CPU module is connecting them by means of a network cable (no jumper wire is needed). The connection type that the driver created uses is Ethernet. Please refer to the figure in (3) above for more information.

Note:

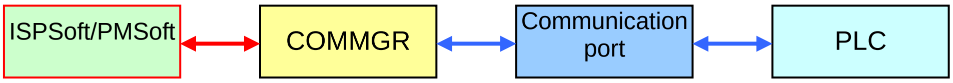
(a) Before users connect the AH500 series CPU module in an AH500 system to a computer, they have to make sure that the power supply modules, the CPU module, the network modules, and the backplanes in the AH500 system are installed properly.

(b) Before users connect an AHCPU5xx-EN series CPU module to a computer, they have to make sure that the network created is normal, and the computer and the AHCPU5xx-EN series CPU module are in the same domain.

(c) Users have to make sure that the values of the Ethernet parameters in an AHCPU5xx-EN series CPU module are correct. The default values of the Ethernet parameters in an AHCPU5xx-EN series CPU module are shown below. I

	Value
IP addressing	Static IP address
IP address	192.168.1.1
Subnet mask	255.255.255.0
Gateway address	192.168.1.1
Port number	502

2.3.13 Creating a Connection Between ISPSoft/PMSoft and COMMGR

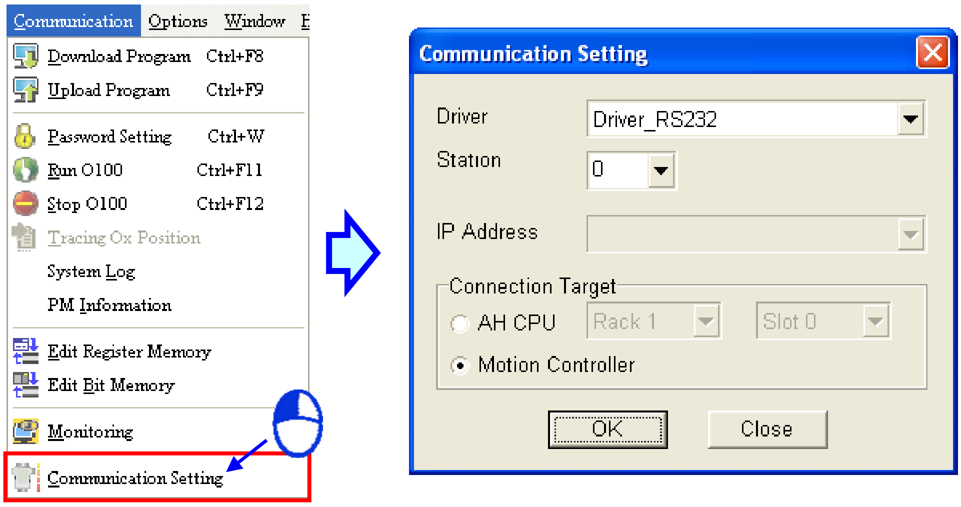


After drivers are created and started in COMMGR, users can specify a driver in ISPSoft/PMSoft. After the setting is complete, a connection between ISPSoft/PMSoft and COMMGR is created. If software directly connects to a motion controller, the software is PMSoft. If software indirectly connects to a motion controller through an AH500 series CPU module, the software is ISPSoft and PMSoft.

2.3.14 Using PMSoft (Direct Connection)

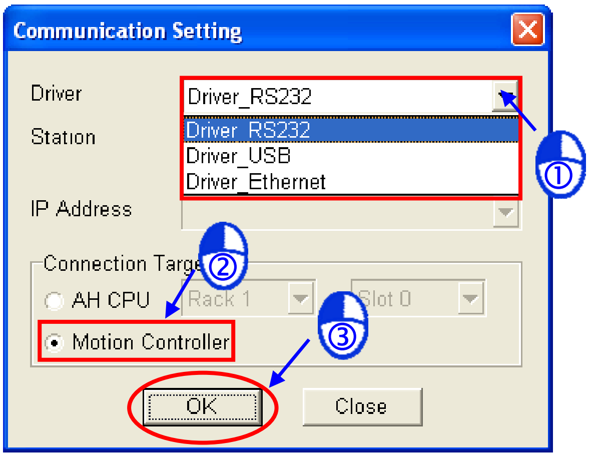
A direct connection is applicable to Delta DVP series motion controllers, and AH500 series motion control modules. After users directly connect a computer to a motion controller, and complete all the setting, they have to follow the steps described below.

(1) Start PMSoft, and click **Communication Setting** on the **Communication** menu.



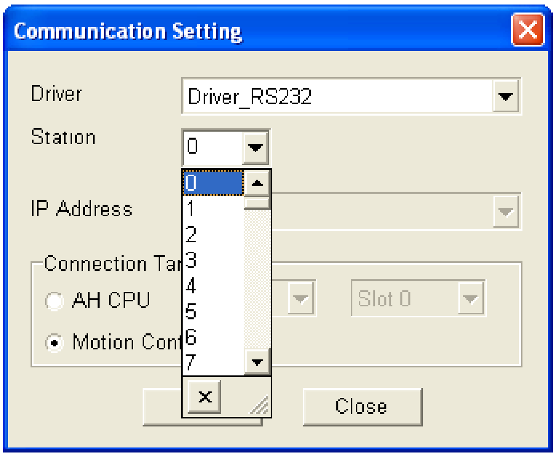
Driver: Driver_RS232
Station: 0
IP Address:
○ AH CPU Rack 1 Slot 0
● Motion Controller

(2) Select a driver in the **Driver** drop-down list box. Before users create a connection between PMSoft and a motion controller, they have to make sure that the driver is started in COMMGR. Select the **Motion Controller** option button, and click **OK**. The communication setting varies with the driver selected.



Driver: Driver_RS232
○ AH CPU Rack 1 Slot 0
● Motion Controller

- **RS232 and USB**
  Users have to select the station address of the motion controller connected to the computer in the **Station** drop-down list box. If the station address selected is 0, a broadcast communication will be carried out.

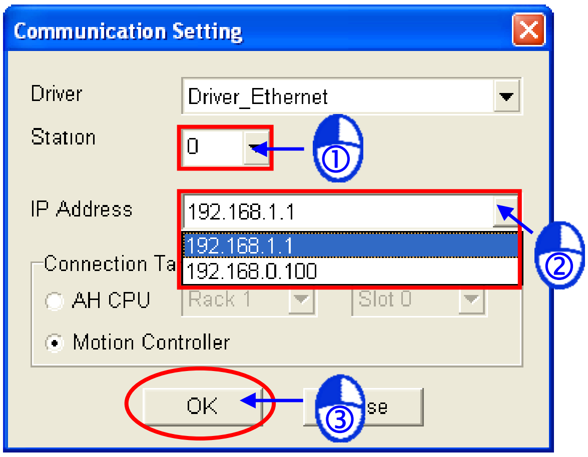


Communication Setting

Driver: Driver_RS232

Station: 0

- **Ethernet**
  Users have to select the station address of the motion controller connected to the computer in the **Station** drop-down list box. If the station address selected is 0, a broadcast communication will be carried out. The users also have to select the IP address created in COMMGR in the **IP Address** drop-down list box.



Communication Setting

Driver: Driver_Ethernet

Station: 0

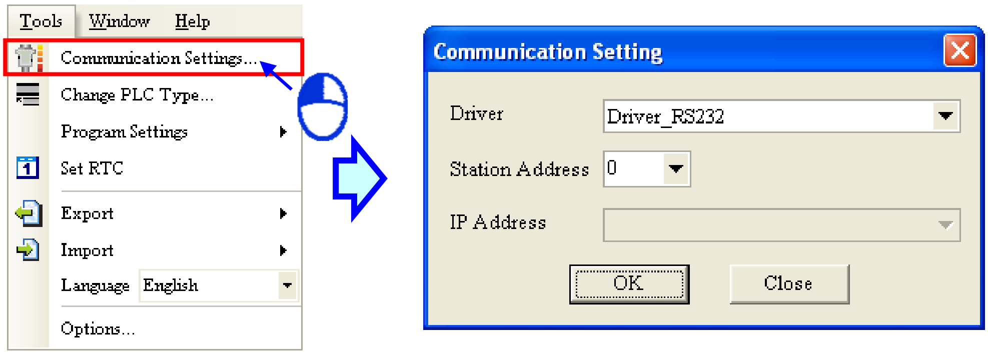
IP Address: 192.168.1.1

2.3.15 Using ISPSoft and PMSoft (Indirect Connection)

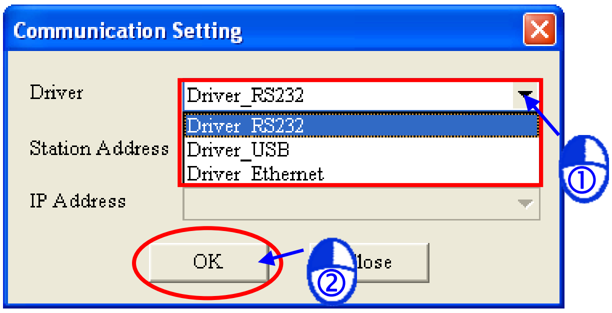
An indirect connection is applicable to AH500 series motion control modules (AH20MC-5A, AH10PM-5A, and AH05PM-5A). After users connect a computer to the CPU module in an AH500 system, and complete all the setting, the CPU module will communicate with the motion control module in the AH500 system through a backplane. The motion control module is regarded as an extension module in the AH500 system, and therefore needs to be set by means of ISPSoft. If the users want to write a motion program, or upload/download a program, they have to use PMSoft. The use of PMSoft will be introduced in the following chapters.

- Setting ISPSoft

  (1) Start ISPSoft, and click **Communication Settings...** on the **Tools** menu.

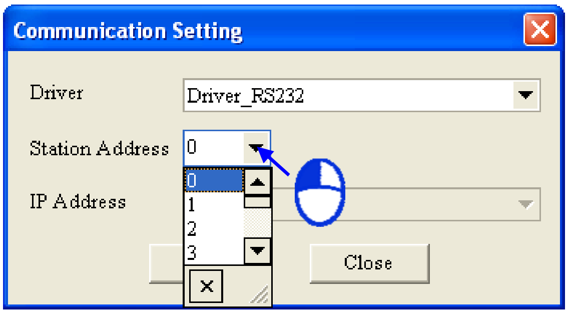


(2) Select a driver in the **Driver** drop-down list box. Before users create a connection between ISPSoft and an AH500 series CPU module, they have to make sure that the driver is started in COMMGR. After the users click **OK**, the communication setting is complete. The communication setting varies with the driver selected.



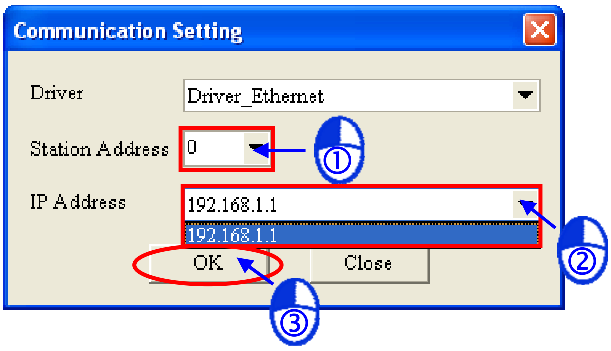
➢ **RS232 and USB**

Users have to select the station address of the CPU module connected to the computer in the **Station Address** drop-down list box. If the station address selected is 0, a broadcast communication will be carried out.



➢ **Ethernet**

Users have to select the station address of the CPU module connected to the computer in the **Station Address** drop-down list box. If the station address selected is 0, a broadcast communication will be carried out. The users also have to select the IP address created in COMMGR in the **IP Address** drop-down list box.



Driver: Driver_Ethernet

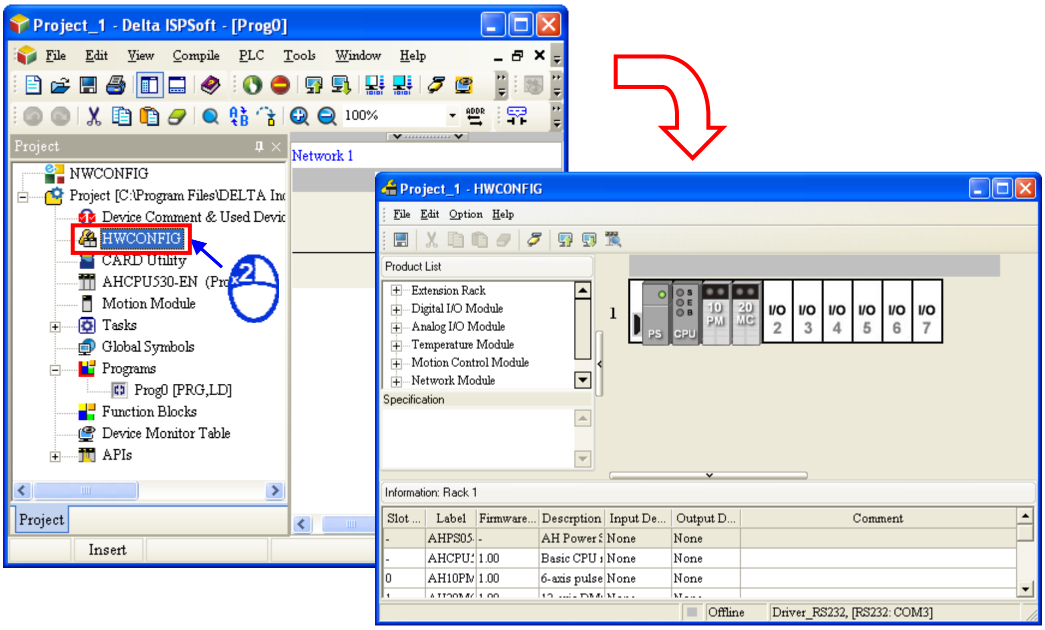
Station Address: 0

IP Address: 192.168.1.1

- Setting HWCONFIG

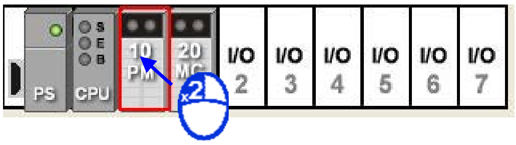
  HWCONFIG is a hardware configuration tool for Delta PLCs. It helps users configure the hardware in a system, set the parameters in the system, and download the setting values to the CPU module and the modules in the system. If a motion controller functions as an extension module in an AH500 system, it needs to be configured and set by means of HWCONFIG.

  (1) Start HWOCONFIG in ISPSoft, and configure the hardware architecture which is need according to the actual AH500 system. Please refer to ISPSoft Users Manual for more information.



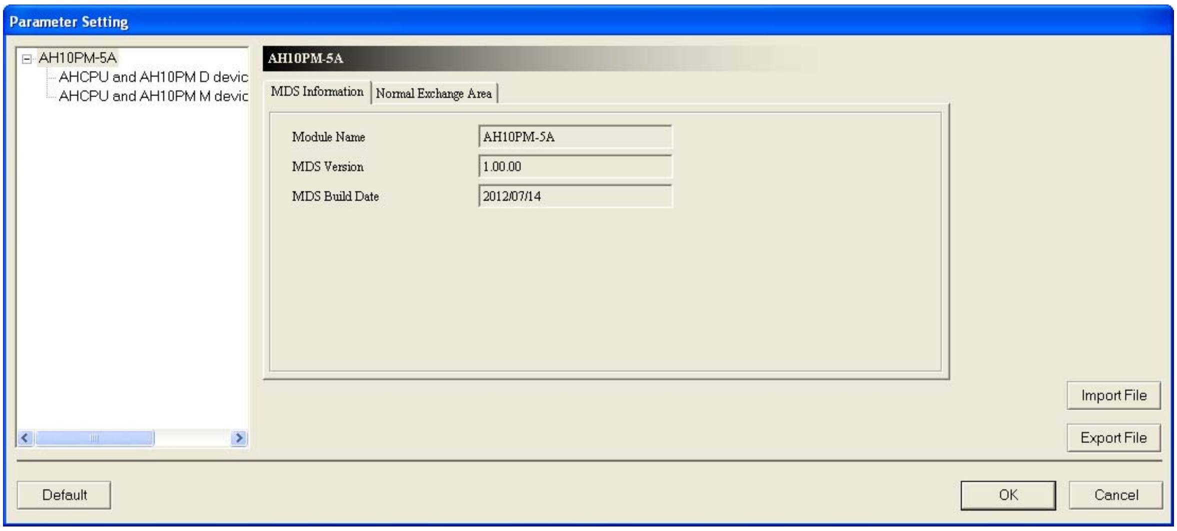
Slot ...	Label	Firmware...	Descrption	Input De...	Output D...	Comment
-	AHPS05·	-	AH Power S	None	None	
-	AHCPU:	1.00	Basic CPU 1	None	None	
0	AH10PM	1.00	6-axis pulse	None	None	

  (2) After users configure the system needed, they have to set the modules in the system. After the users double-click a motion control module in HWCONFIG, the **Parameter Setting** window will appear. The users can set the parameters in the **Parameter Setting** window. After the setting of the parameters is complete, the users can download the values of the parameters to special data registers in the motion control module. The parameters which need to be set do not vary with the motion control module (AH20MC-5A/AH10PM-5A/AH05PM-5A) selected.



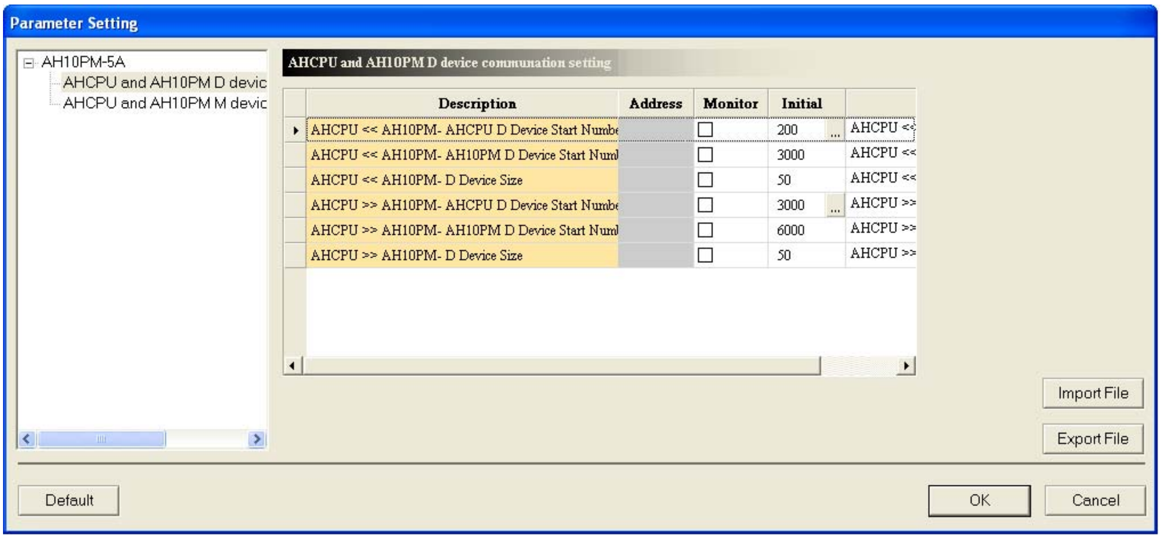
The page shown below displays the name of the motion control module double-clicked in

HWCONFIG, the version of the MDS for the motion control module, and the date when the MDS is created.



Module Name: AH10PM-5A
MDS Version: 1.00.00
MDS Build Date: 2012/07/14

The page shown below displays the data registers involved in data exchange between the CPU module connected to the computer and the motion control module double-clicked in HWCONFIG. The device addresses are typed in the **Initial** cells.



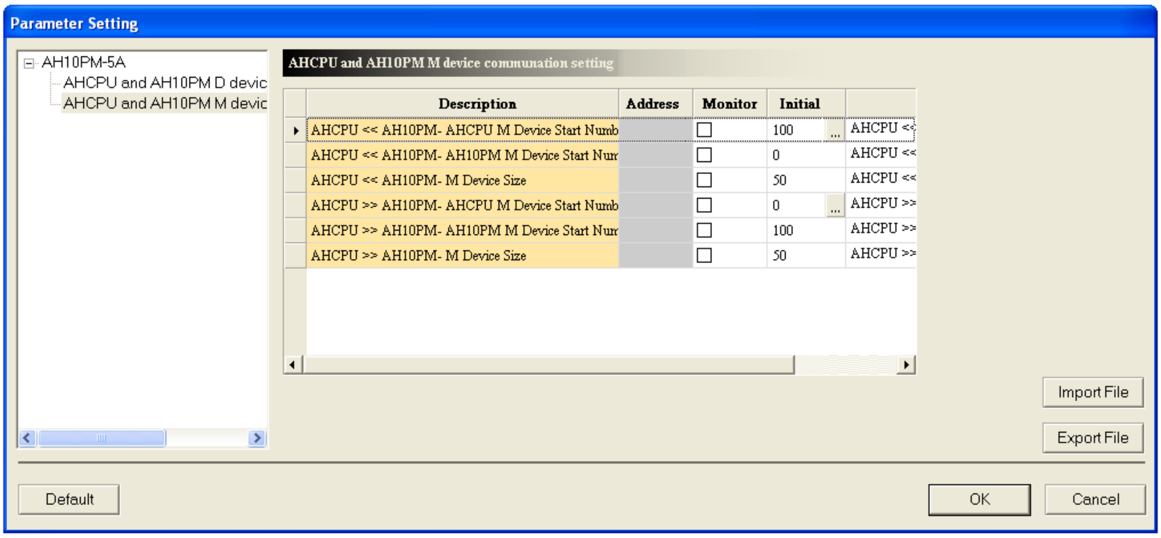
Description	Address	Monitor	Initial	
AHCPU << AH10PM- AHCPU D Device Start Numbe		□	200	AHCPU <<
AHCPU << AH10PM- AH10PM D Device Start Num		□	3000	AHCPU <<
AHCPU << AH10PM- D Device Size		□	50	AHCPU <<
AHCPU >> AH10PM- AHCPU D Device Start Numbe		□	3000	AHCPU >>
AHCPU >> AH10PM- AH10PM D Device Start Num		□	6000	AHCPU >>
AHCPU >> AH10PM- D Device Size		□	50	AHCPU >>

- **AHCPU<<AH10PM - AHCPU D Device Start Number**: Before the motion control module sends data to the CPU module, users have to set a starting data register in the CPU module. For example, the data sent by the motion control module will be stored in the devices starting from D200 in the CPU module.
- **AHCPU<<AH10PM - AH10PM D Device Start Number**: Before the motion control module sends data to the CPU module, users have to set a starting data register in the motion control module. For example, the data in the device starting from D3000 in the motion control module will be sent.
- **AHCPU<<AH10PM – D Device Size**: Users have to set the length of the data which will be sent. For example, the data in 50 data registers will be sent.
- **AHCPU>>AH10PM - AHCPU D Device Start Number**: Before the CPU module sends data to the motion control module, users have to set a starting data register in the CPU module. For example, the data in the devices starting from D3000 will be sent.
- **AHCPU>>AH10PM - AH10PM D Device Start Number**: Before the CPU module

sends data to the motion control module, users have to set a starting data register in the motion control module. For example, the data sent by the CPU module will be stored in the devices starting from D6000 in the motion control module.

- **AHCPU>>AH10PM – D Device Size**: Users have to set the length of the data which will be sent. For example, the data in 50 data registers will be sent.

The page shown below displays the relays involved in data exchange between the CPU module connected to the computer and the motion control module double-clicked in HWCONFIG. The device addresses are typed in the **Initial** cells.



Parameter Setting

AHCPU and AH10PM M device communation setting

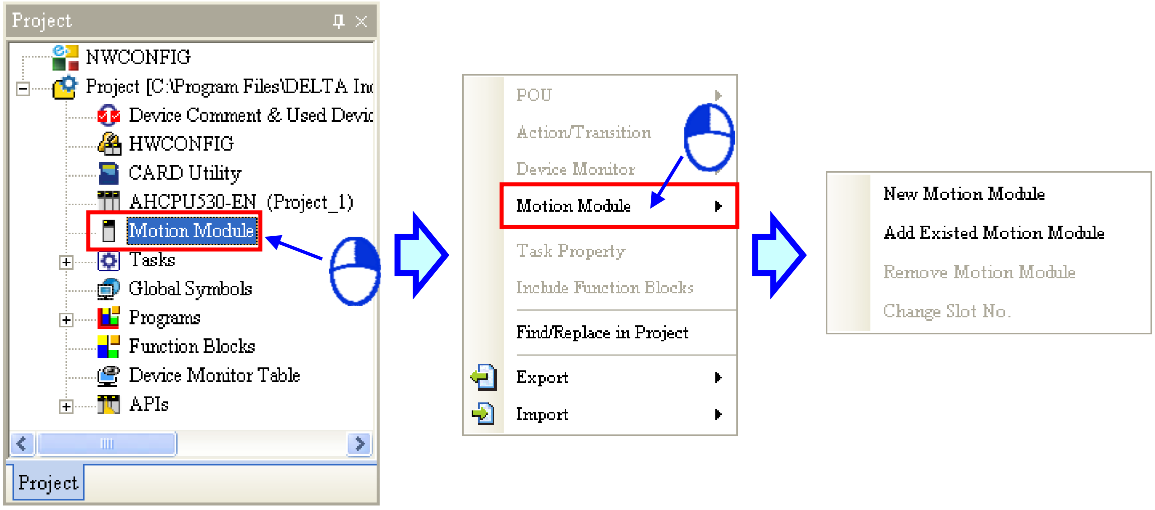
Description	Address	Monitor	Initial	
AHCPU << AH10PM- AHCPU M Device Start Numb		□	100	AHCPU <<
AHCPU << AH10PM- AH10PM M Device Start Nur		□	0	AHCPU <<
AHCPU << AH10PM- M Device Size		□	50	AHCPU <<
AHCPU >> AH10PM- AHCPU M Device Start Numb		□	0	AHCPU >>
AHCPU >> AH10PM- AH10PM M Device Start Nur		□	100	AHCPU >>
AHCPU >> AH10PM- M Device Size		□	50	AHCPU >>

- **AHCPU<<AH10PM - AHCPU M Device Start Number**: Before the motion control module sends data to the CPU **module**, users have to set a starting relay in the CPU module. For example, the data sent by the motion control module will be stored in the devices starting from M100 in the CPU module.
- **AHCPU<<AH10PM - AH10PM M Device Start Number**: Before the motion control module sends data to **the** CPU module, users have to set a starting relay in the motion control module. For example, the data in the device starting from M0 in the motion control module will be sent.
- **AHCPU<<AH10PM – M Device Size**: Users have to set the length of the data which will be sent. For example, the data in 50 relays will be sent.
- **AHCPU>>AH10PM - AHCPU M Device Start Number**: Before the CPU module sends data to the motion control module, users have to set a starting relay in the CPU module. For example, the data in the devices starting from M0 will be sent.
- **AHCPU>>AH10PM - AH10PM M Device Start Number**: Before the CPU module sends data to the motion control module, users have to set a starting relay in the motion control module. For example, the data sent by the CPU module will be stored in the devices starting from M100 in the motion control module.
- **AHCPU>>AH10PM – M Device Size**: Users have to set the length of the data which will be sent. For example, the data in 50 relays will be sent.

- Setting the project management area in ISPSoft

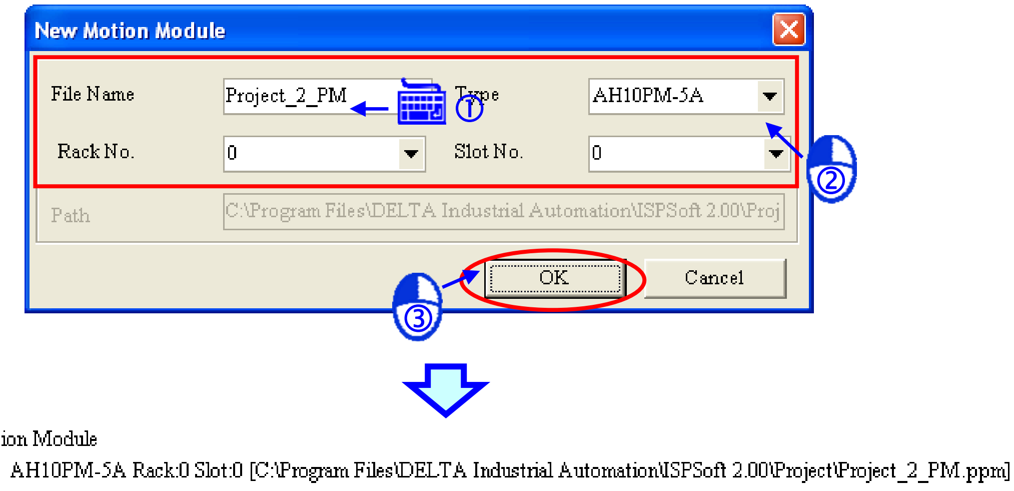
  Users can manage the projects in the project management area in ISPSoft. If users want to use a motion controller as an extension module in an AH500 system, they have to create a motion project in the project management area in ISPSoft.

  (1) After users start ISPSoft, they can see **Motion Module** in the project management area. After the users right-click **Motion Module**, a context menu will appear. After the users point to **Motion Module** on the context menu, they can click **New Motion Module** or **Add Existed Motion Module** on the menu which appears.



- **New Motion Module**

If the users want to create a new motion project, they have to click **New Motion Module** on the menu which appears. After the users click **New Motion Module** on the menu which appears, the **New Motion Module** window will appear. The users have to type a file name in the **File Name** box, select a model in the **Type** drop-down list box, select a rack number in the **Rack No.** drop-down list box, and select a slot number in the **Slot No.** drop-down list box. After the users click **OK**, the model selected will be added to the project management area, and the motion project created will be saved in the folder where the ISPSoft project created is saved.

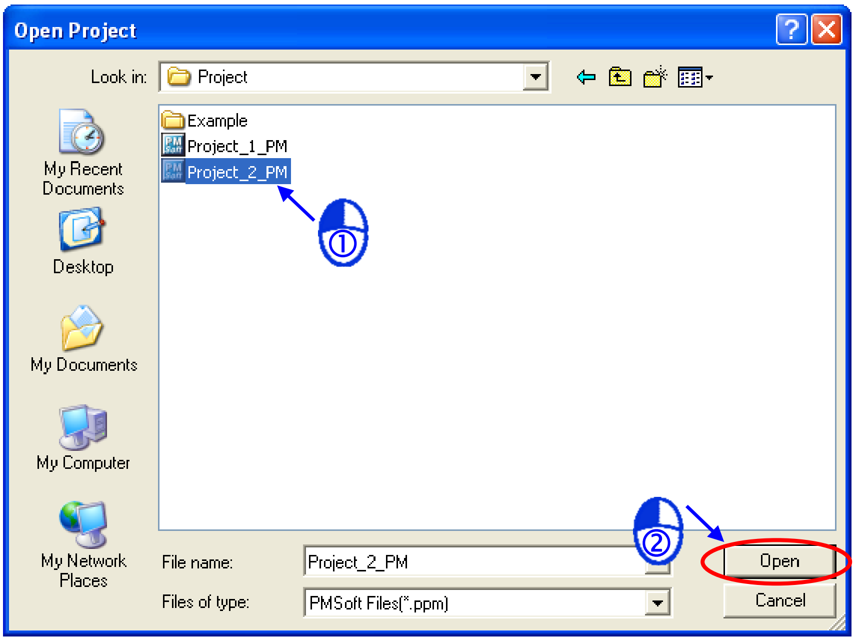


File Name: Project_2_PM
Type: AH10PM-5A
Rack No.: 0
Slot No.: 0
Path: C:\Program Files\DELTA Industrial Automation\ISPSoft 2.00\Proj

AH10PM-5A Rack:0 Slot:0 [C:\Program Files\DELTA Industrial Automation\ISPSoft 2.00\Project\Project_2_PM.ppm]

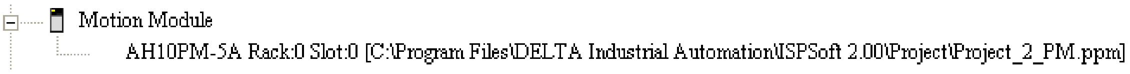
- **Add Existed Motion Module**

If the users want to add a motion project which has been created to the project management area, they have to click **Add Existed Motion Module** on the menu which appears. After the users click **Add Existed Motion Module** on the menu which appears, the **Open Project** window will appear. The users have to click a motion project in the **Open Project** window. After the users click **Open**, the motion project will be added to the project management area, and will be saved in the folder where the ISPSoft project created is saved.



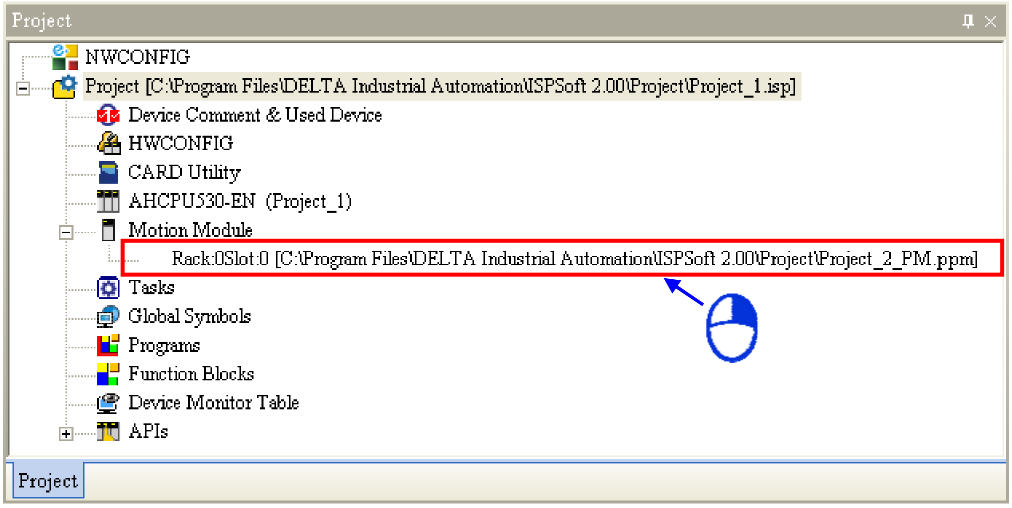
Example
Project_1_PM
Project_2_PM

File name: Project_2_PM
Files of type: PMSoft Files(*.ppm)

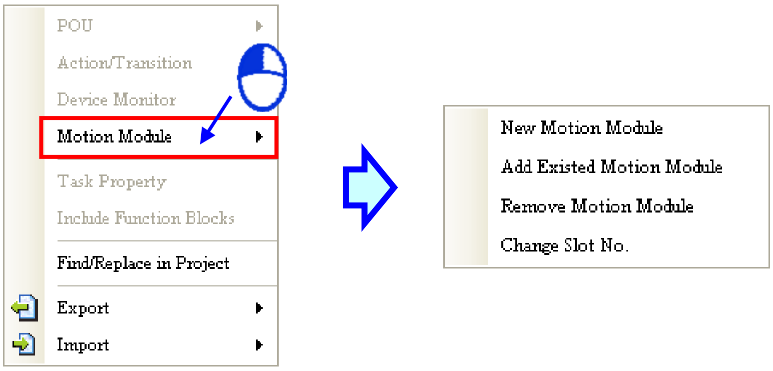


Motion Module
AH10PM-5A Rack:0 Slot:0 [C:\Program Files\DELTA Industrial Automation\ISPSoft 2.00\Project\Project_2_PM.ppm]

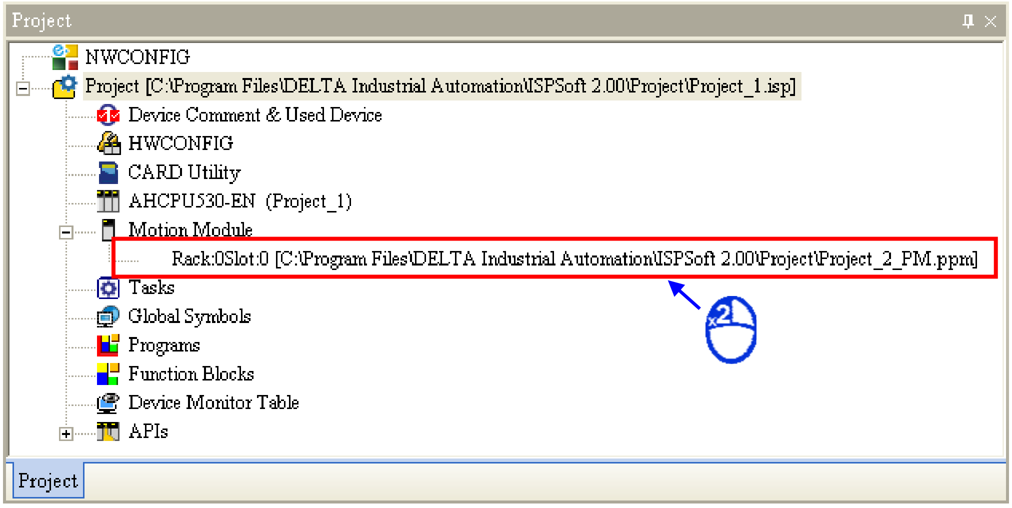
(2) After the users right-click the motion project added, a context menu will appear. After the users point to Motion Module on the context menu, they can click **New Motion Module**, **Add Existed Motion Module**, **Remove Motion Module**, or **Change Slot No.** on the menu which appears. Every slot number in an AH500 system must be unique, and the slot numbers in an AH500 system must be the same as the slot numbers used in HWCONFIG.

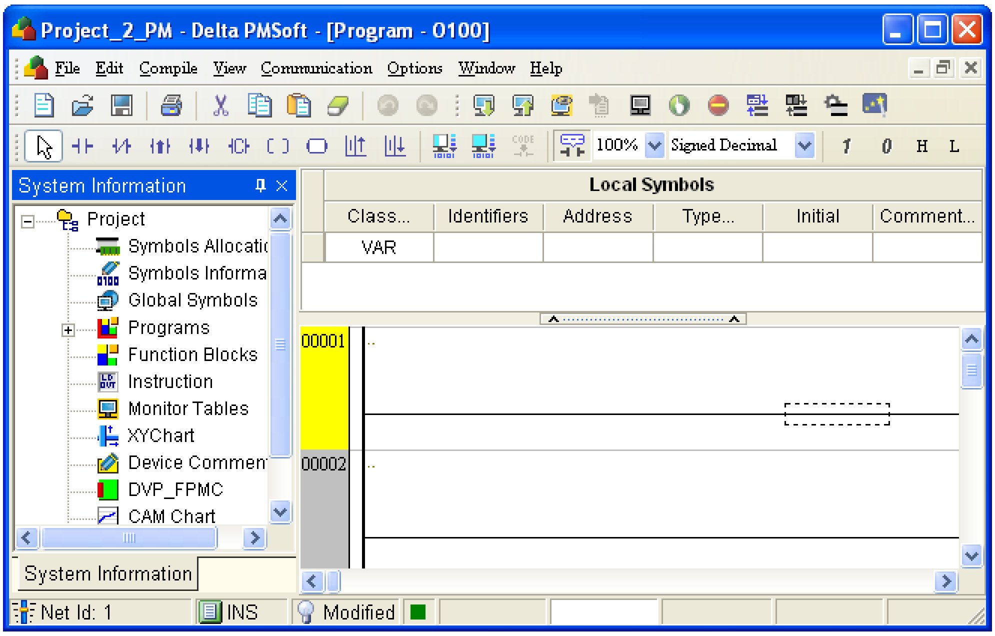


NWCONFIG
Project [C:\Program Files\DELTA Industrial Automation\ISPSoft 2.00\Project\Project_1.isp]
Device Comment & Used Device
HWCONFIG
CARD Utility
AHCPU530-EN (Project_1)
Motion Module
Rack:0Slot:0 [C:\Program Files\DELTA Industrial Automation\ISPSoft 2.00\Project\Project_2_PM.ppm]
Tasks
Global Symbols
Programs
Function Blocks
Device Monitor Table
APIs



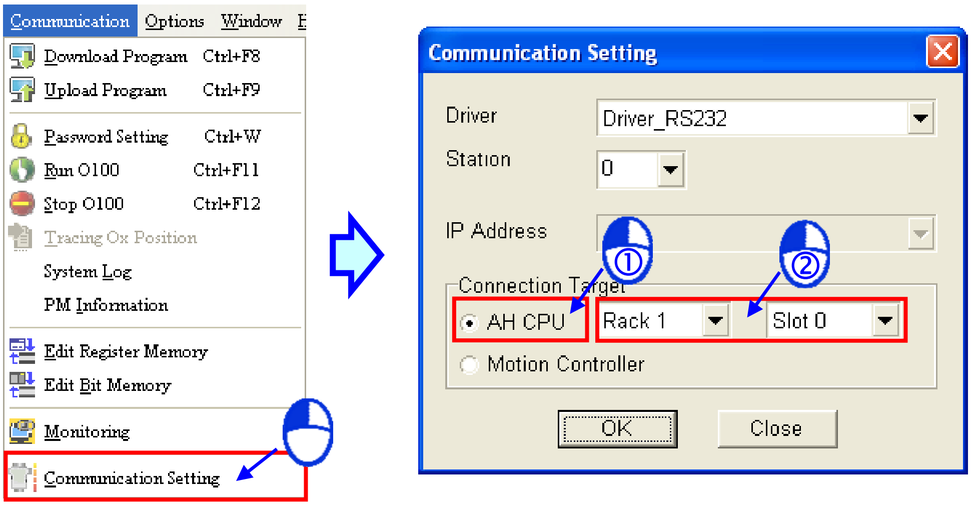
(3) After the users double-click the motion project added, PMSoft will be started. The users can not click **New**, **Open**, **Save As**, and **Close** on the **File** menu in PMSoft.


Project
NWCONFIG
Project [C:\Program Files\DELTA Industrial Automation\ISPSoft 2.00\Project\Project_1.isp]
Device Comment & Used Device
HWCONFIG
CARD Utility
AHCPU530-EN (Project_1)
Motion Module
Rack:0Slot:0 [C:\Program Files\DELTA Industrial Automation\ISPSoft 2.00\Project\Project_2_PM.ppm]
Tasks
Global Symbols
Programs
Function Blocks
Device Monitor Table
APIs
Project



(4) After the users click **Communication Setting** on the **Communication** menu in PMSoft, the

Communication Setting window will appear. The users have to select the **AH CPU** option button, select the correct rack number, and select the correct slot number.



Communication Setting

Driver: Driver_RS232

Station: 0

IP Address:

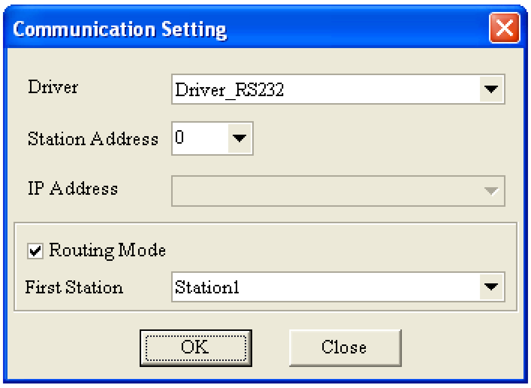
Connection Target

● AH CPU Rack 1 Slot 0

○ Motion Controller

- Routing

  If the ISPSoft project activated is a project in the group created, the **Routing Mode** checkbox will appear in the **Communication Setting** window in ISPSoft. If the **Routing Mode** checkbox in the **Communication Setting** window in ISPSoft is selected, the motion project added to the project management area will bring the setting to PMSoft. Please refer to chapter 2 and chapter 16 in ISPSoft User Manual for more information about creating a group of projects in ISPSoft, routing, and creating a network.



Communication Setting

Driver: Driver_RS232

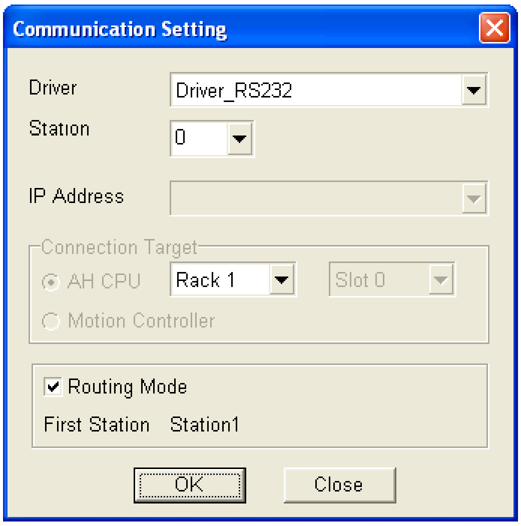
Station Address: 0

IP Address:

☑ Routing Mode

First Station: Station1

Communication setting in ISPSoft



Communication Setting

Driver: Driver_RS232

Station: 0

IP Address:

Connection Target

● AH CPU: Rack 1, Slot 0

○ Motion Controller

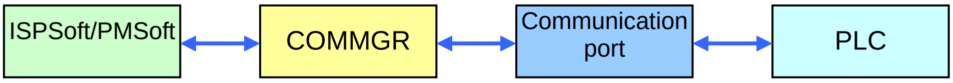
☑ Routing Mode

First Station: Station1

Communication setting in PMSoft

2.3.16 Practical Connection Test

After users complete the steps described above, they can check whether the computer communicates with the PLC normally by means of a simple test.



The users have to make sure of the items below before the PLC communicates with the computer.

(a) The users have to make sure that COMMGR has been started normally, and the status of the driver which will be specified is **START**.

(b) The users have to make sure that the parameters related to the driver which will be specified are set correctly. If the connection type the driver uses is RS232, the users have to make sure that the communication protocol for exchanging data through the driver is the same as the communication protocol for exchanging data through a communication port on the PLC.

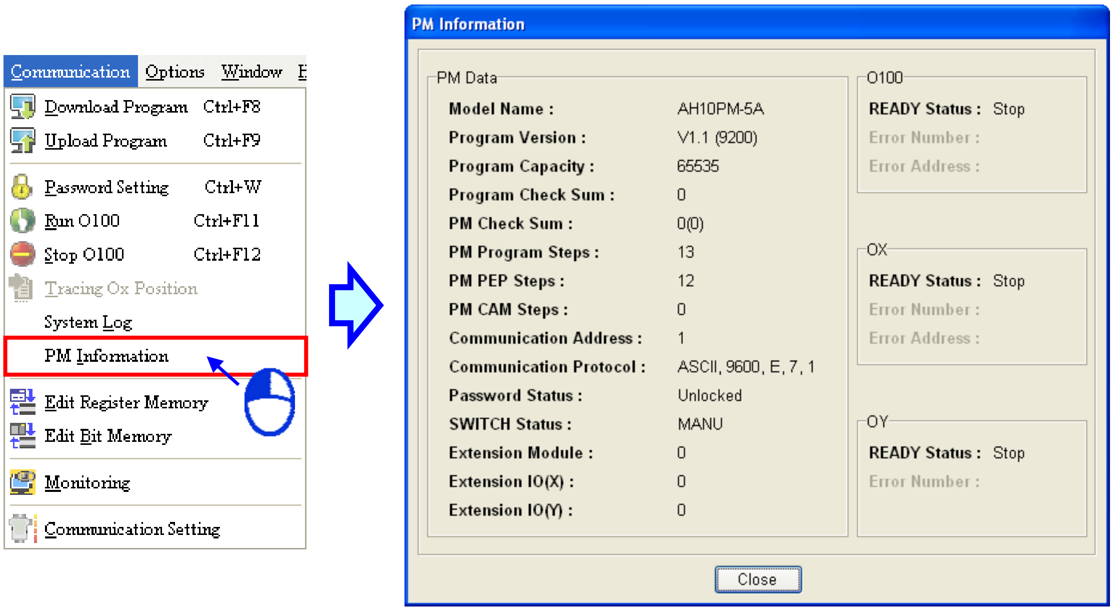
(c) The users have to make sure that the communication port used, e.g. a network interface card in the computer, a switching hub, or a serial port, is in a normal state.

(d) The users have to make sure that the driver, the station address, and the IP address which are specified in the **Communication Setting** window in PMSoft or ISPSoft are correct. If an IP address is specified, the users have to check whether the IP address of the PLC is the same as the IP address specified.

(e) The users have to make sure that the PLC is correctly connected to the computer through a communication cable, power is supplied to the PLC, and the PLC operates normally.

After the users make sure of the items above, they can click **PM Information** on the

Communication menu in PMSoft. If the computer communicates with the PLC normally, PMSoft will retrieve information from the PLC, and the information retrieved from the PLC will be displayed in the **PM Information** window.



PM Information

PM Data

Model Name :	AH10PM-5A
Program Version :	V1.1 (9200)
Program Capacity :	65535
Program Check Sum :	0
PM Check Sum :	0(0)
PM Program Steps :	13
PM PEP Steps :	12
PM CAM Steps :	0
Communication Address :	1
Communication Protocol :	ASCII, 9600, E, 7, 1
Password Status :	Unlocked
SWITCH Status :	MANU
Extension Module :	0
Extension IO(X) :	0
Extension IO(Y) :	0

O100

READY Status : Stop
Error Number :
Error Address :

OX

READY Status : Stop
Error Number :
Error Address :

OY

READY Status : Stop
Error Number :

Chapter 3 Program Organization Units

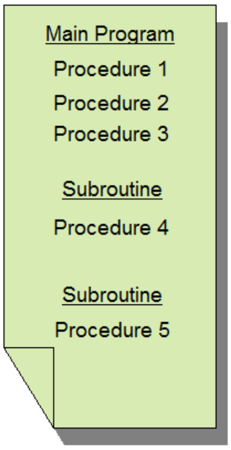
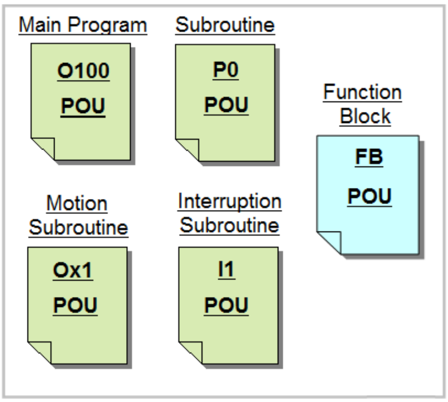
Table of Contents

3.1 Knowing Program Organization Units (POUs)

3.1.1 Program Architecture

Program organization units are the basic elements of the program in a PLC. The characteristic of the program architecture in PMSoft is that a program is divided into several units. These units are called program organization units.

In the classic architecture shown below, the source code for a PLC is composed of all procedures, including subroutines. If the size of a program becomes larger, the maintenance of the program and the debugging of the program will be burdens. In the program architecture in PMSoft, a program is divided into several units according to functions or characteristics. It is convenient to develop and maintain a program. Besides, owing to the fact that program organization units are modular, different program organization units can be developed by different designers. It benefits the distribution of manpower and the execution of a project.

Classic architecture	Program architecture in PMSoft
 Main Program Procedure 1 Procedure 2 Procedure 3 Subroutine Procedure 4 Subroutine Procedure 5	 Main Program O100 POU Subroutine P0 POU Function Block FB POU Motion Subroutine Ox1 POU Interruption Subroutine I1 POU

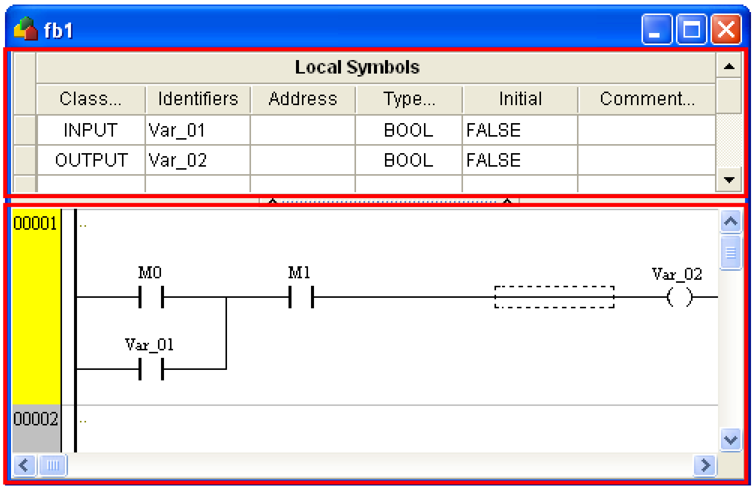
3.1.2 Types

There are two types of POUs in PMSoft. They are POUs which are programs, and POUs which are function blocks.

- **POUs which are programs**
  In PMSoft, the main program, the motion programs, the subroutines, and the interrupt subroutines are POUs which are programs. The motion programs are activated by the main program. The subroutines are also called by the main program. The subroutines have different functions. Please refer to DVP-PM Application Manual for more information.
- **POUs which are function blocks**
  Static symbols can be declared in a function block. If static symbols are declared in a function block, the values of the symbols will be retained after the operation in the function block is performed. Owing to the fact that the operation is performed on the values memorized in the function block and the values input, the values output may be different from the values output last time even if the values input are the same as the values input last time. Besides, a function block can call another function block.

3.1.3 POUs in PMSoft

The structure of a POU in PMSoft is shown below. It is composed of two areas. A local symbol table is at the upper part of the structure, and a program editing area is at the lower part of the structure. Please refer to chapter 4 for more information about creating symbol tables and programs.



fb1

Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
INPUT	Var_01		BOOL	FALSE	
OUTPUT	Var_02		BOOL	FALSE	

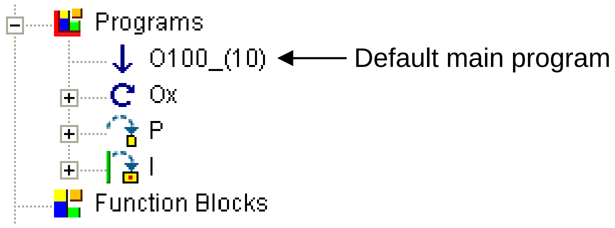
Local symbol table

Program editing area

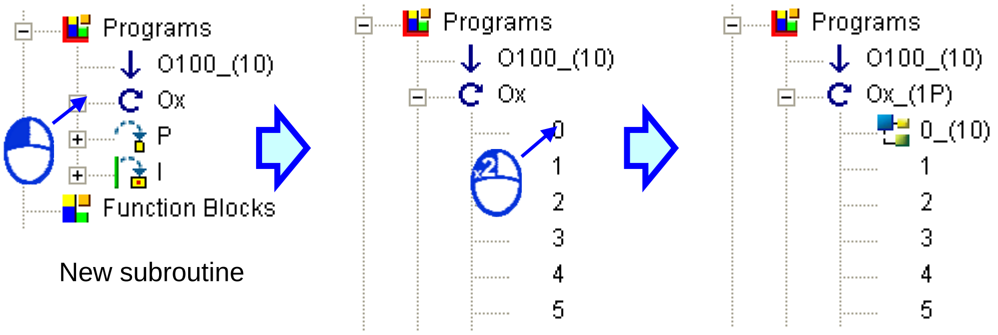
3.2 Managing POUs

3.2.1 Adding a POU

(1) Adding a new POU which is a program: Users have to select an item which they want to create in the system information area. A default main program is created in PMSoft. If the users want to create another POU, they have to expand a program section, and double-click a number.

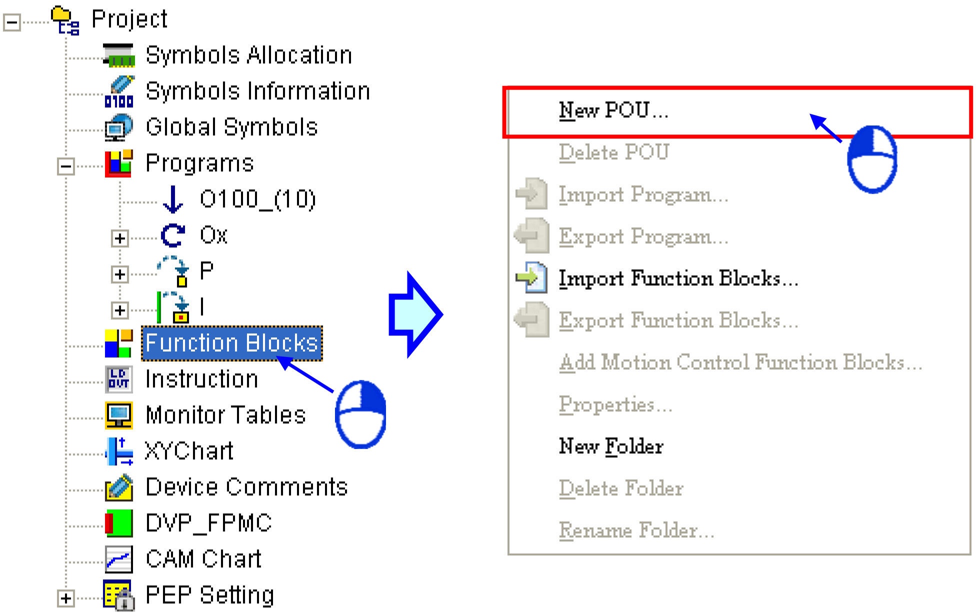


Default main program

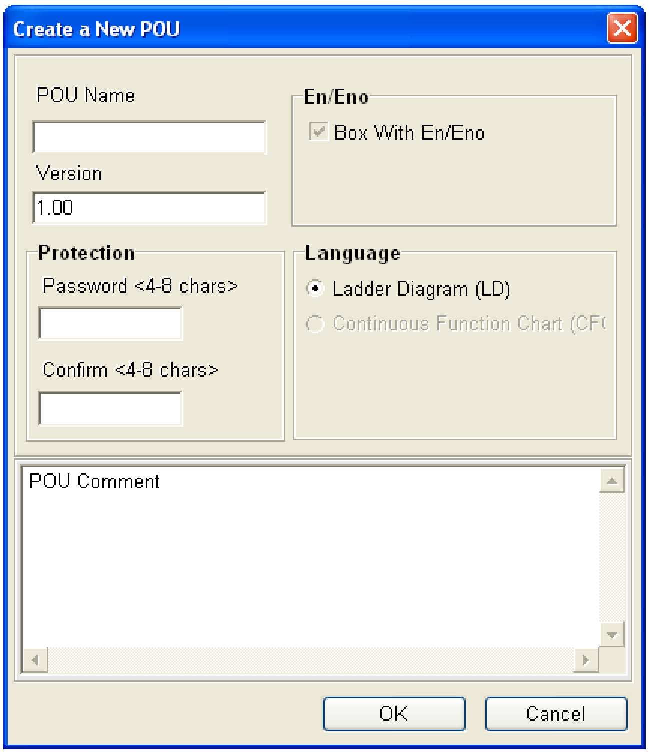


New subroutine

(2) Adding a new POU which is a function block: Users have to click **Function Blocks**, right-click **Function Blocks**, and click **New POU...** on the context menu.



The users have to set the attributes of the POU they want to create in the **Create a New POU** window. After the setting of the attributes is complete, they have to click **OK**.


Create a New POU

POU Name

Version
1.00

En/Eno
☑ Box With En/Eno

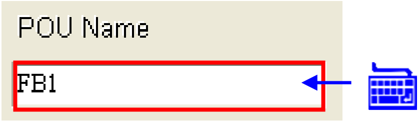
Protection
Password <4-8 chars>

Confirm <4-8 chars>

Language
● Ladder Diagram (LD)
○ Continuous Function Chart (CF(

POU Comment

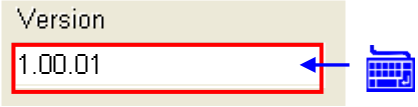
- **POU Name**: Users can type a POU name in this box.
  - Special marks such as (,), ~, !, @, #, $, %, ^, &, and * can not be used.
  - Every POU name in a project must be unique.
  - The POU name typed in this box is case-insensitive.
  - A POU name is composed of twenty characters at most.



POU Name

FB1

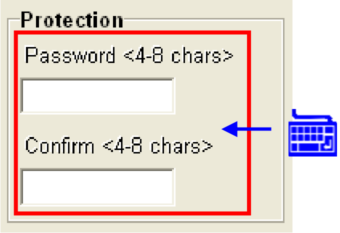
- **Version**: Users can type a version number in this box. A version number is composed of numbers and decimal points. Two decimal points at most can be typed in this box.



Version

1.00.01

- **Protection**: If **users** want to set a password, the password typed in the **Password <4-8 chars>** box must be the same as the password typed in the **Confirm <4-8 chars>** box.

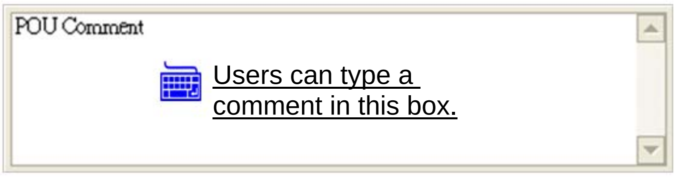


Protection

Password <4-8 chars>

Confirm <4-8 chars>

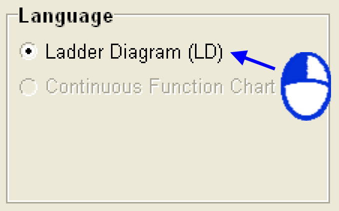
- **POU Comment**: Users can type a comment in this box.



POU Comment

Users can type a comment in this box.

- **Language**: Users can select a programming language in this section. Only the **Ladder Diagram (LD)** option button can be selected now.

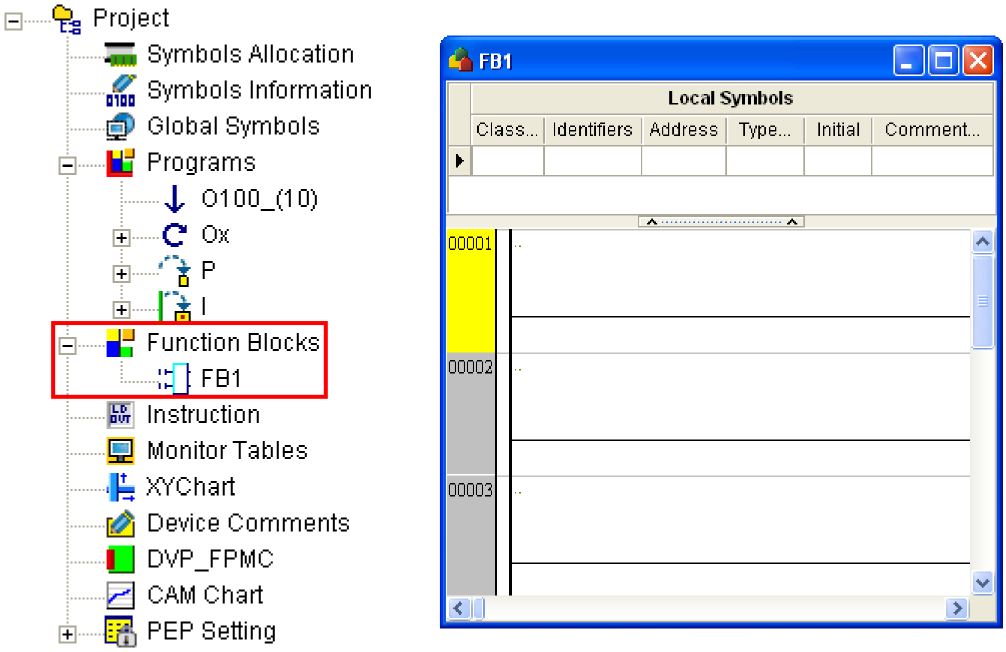


Language

● Ladder Diagram (LD)

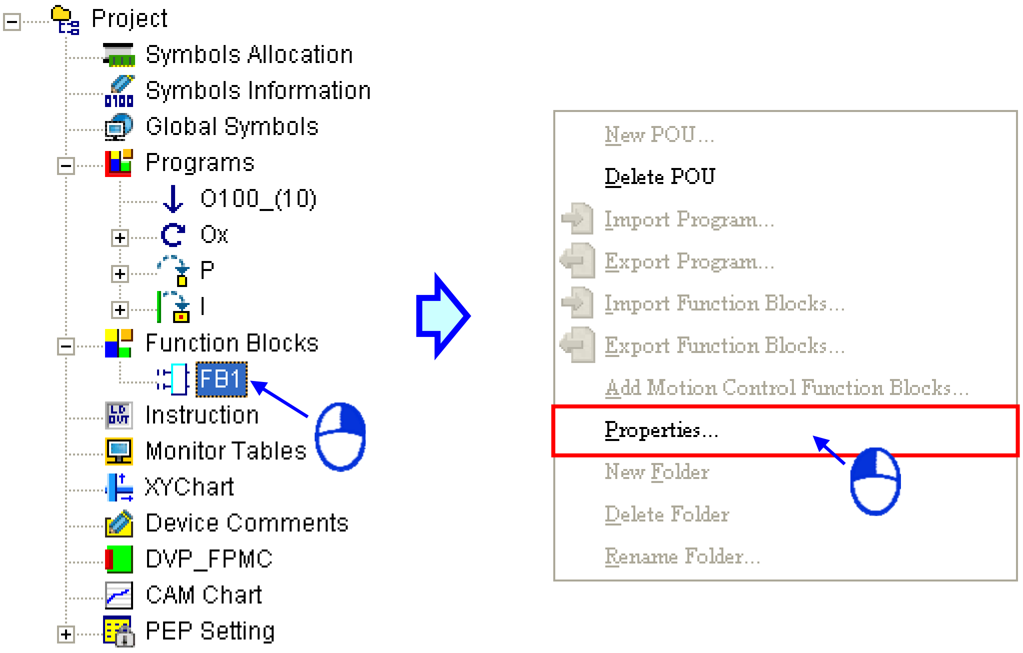
○ Continuous Function Chart

After the users click **OK**, a program editing window will appear, and the new POU will appear in the system information area.



3.2.2 Changing the Attributes of a POU Which Is a Function Block

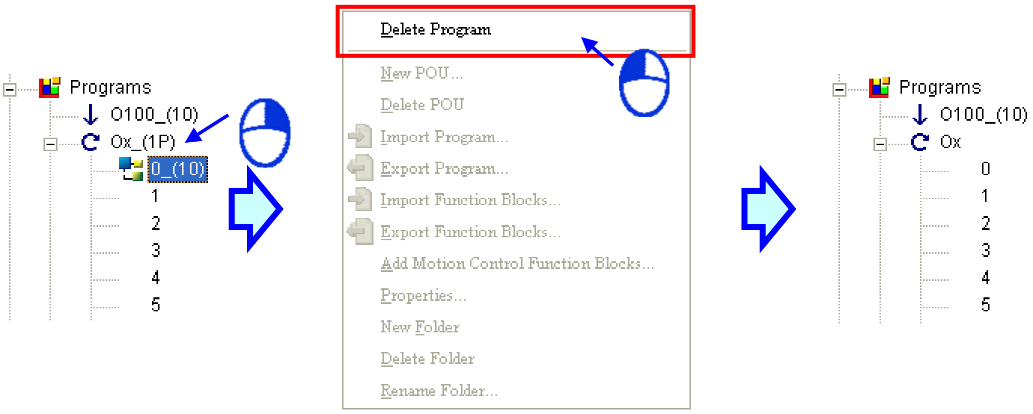
(1) If users want to modify the attributes of a POU which is a function block, they have to right-click the POU, and click **Properties...** on the context menu.



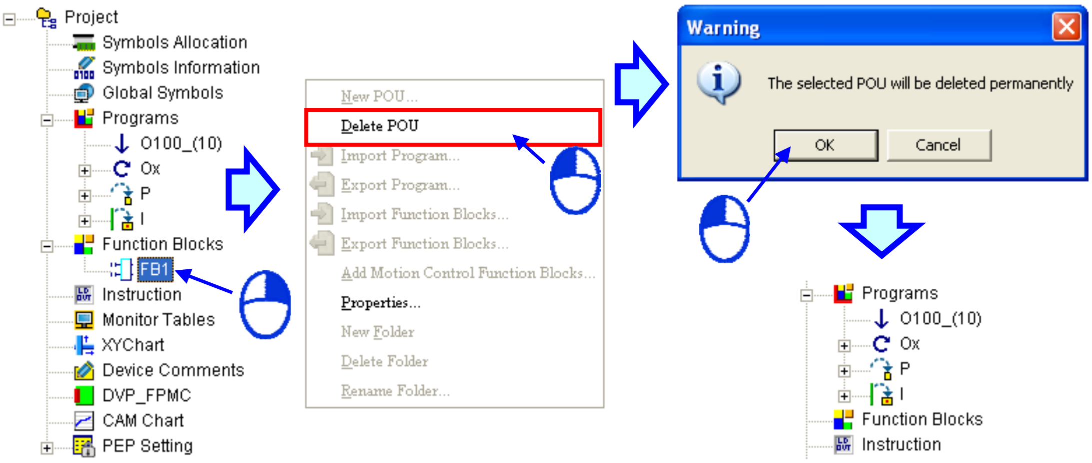
(2) The users can follow the description in the previous section, and change the attributes of the POU. They can not change the programming language used. Besides, the users can not modify the attributes of a POU which is a program.

3.2.3 Deleting a POU

(1) Deleting a POU which is a program: If users want to delete a POU which is a program, they have to right-click the POU, and click **Delete Program** on the context menu.



(2) Deleting a POU which is a function block: If users want to delete a POU which is a function block, they have to right-click the POU, click **Delete POU** on the context menu, and click **OK** in the **Warning** window.

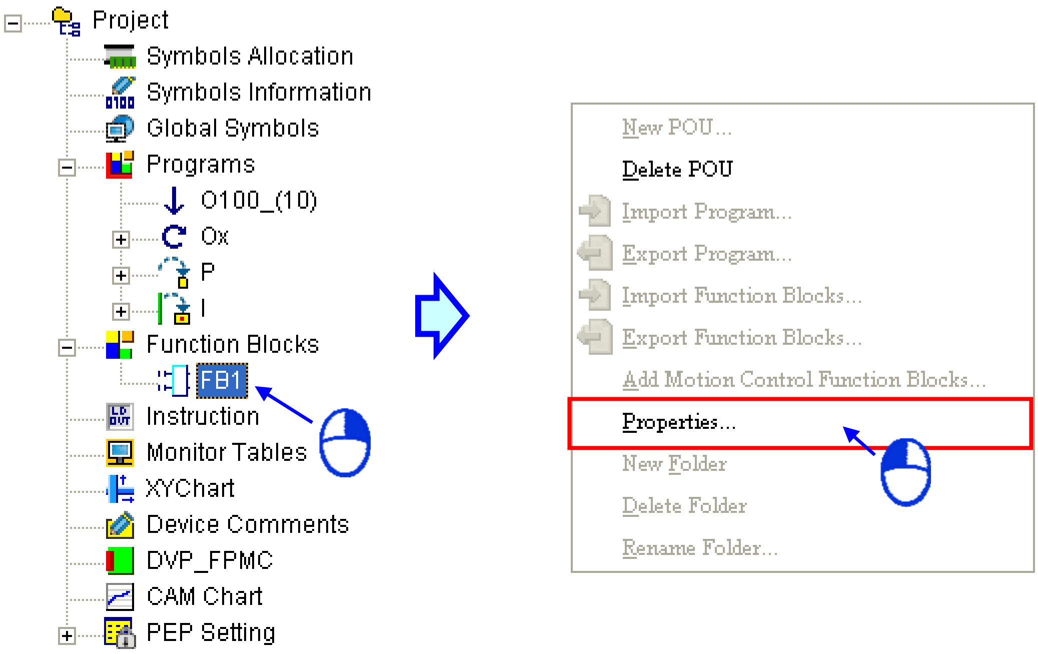


The selected POU will be deleted permanently

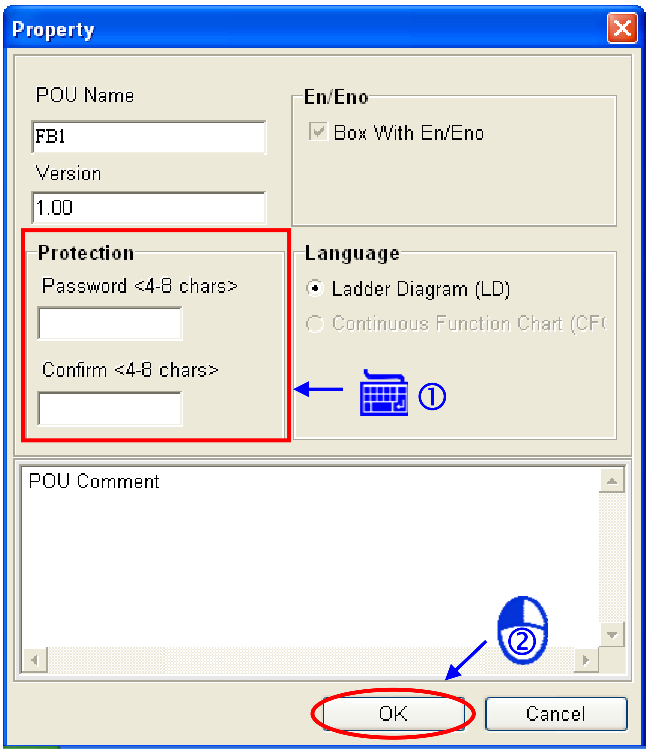
3.2.4 Managing the Password Protecting a POU Which Is a Function Block

If some parts of a program need to be protected, users can write the parts in a POU which is a function block, and protect the POU by a password. A PM password and a PEP password will be introduced in chapter 10. A POU password protects a POU which is a function block in a PMSoft project. A PM password and a PEP password protect the program in a motion controller.

(1) If users want to protect a POU which is a function block by a password, they have to right-click the POU, and click **Properties...** on the context menu.



The password typed in the **Password <4-8 chars>** box must be the same as the password typed in the **Confirm <4-8 chars>** box. After users type passwords in the boxes, they have to click **OK**.


Property

POU Name: FB1

Version: 1.00

En/Eno: ☑ Box With En/Eno

Protection

Password <4-8 chars>

Confirm <4-8 chars>

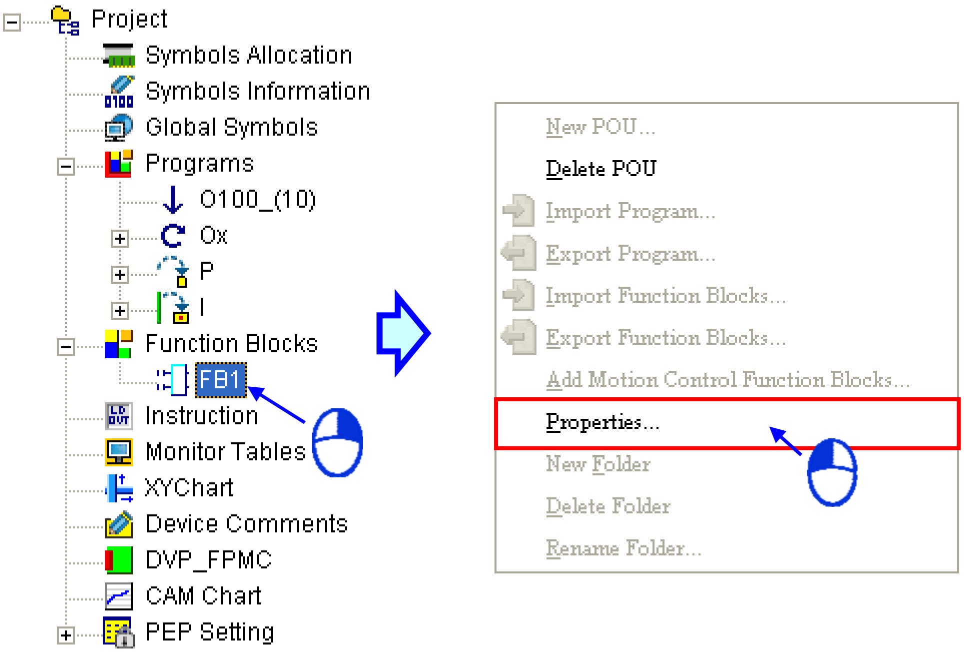
Language: ● Ladder Diagram (LD) ○ Continuous Function Chart (CF(

POU Comment

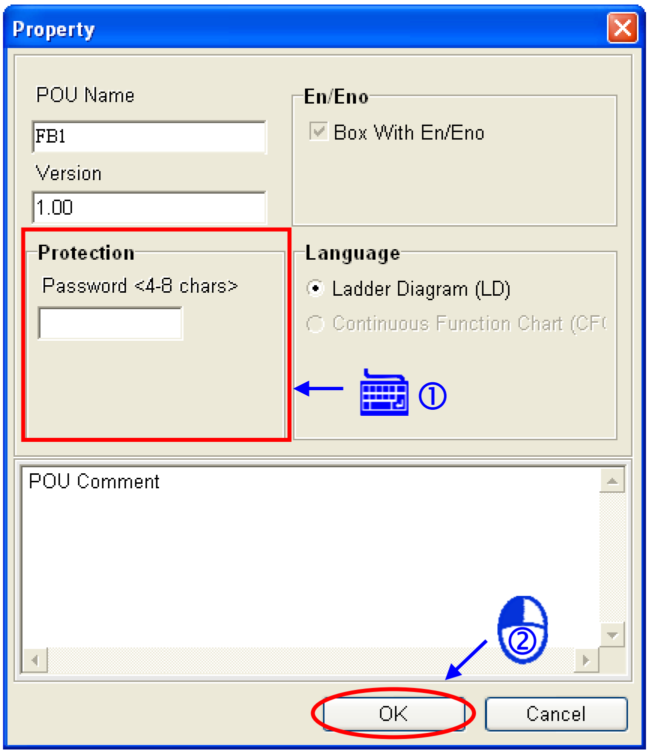
***1. Users can set a password when they add a new POU.**

***2. A POU which is a program is protected only when it is uploaded from a motion controller, or downloaded to a motion controller. Please refer to chapter 10 for more information.**

(2) If users want to remove the password which protects a POU which is a function block, they have to right-click the POU, and click **Properties...** on the context menu.



After the users type the correct password, they have to click **OK**.


Property

POU Name: FB1

Version: 1.00

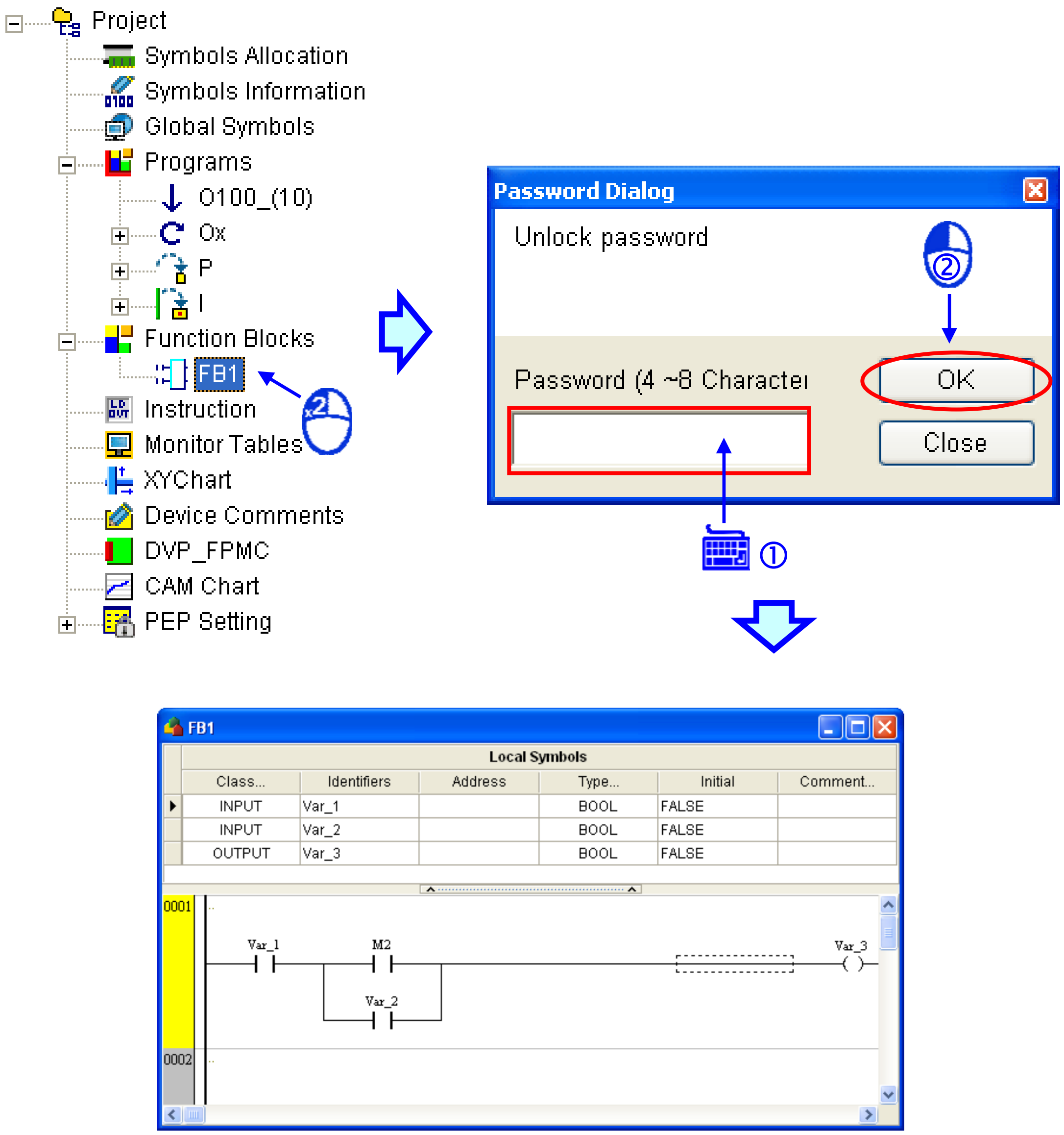
En/Eno: ☑ Box With En/Eno

Protection: Password <4-8 chars>

Language: ● Ladder Diagram (LD) ○ Continuous Function Chart (CF(

POU Comment

(3) If a POU which is a function block is protected by a password, the **Password Dialog** window will appear after users double-click the POU in the system information area. After the users type the correct password in the **Password Dialog** window, they can view the contents of the POU.



Password Dialog

Unlock password

Password (4 ~8 Characte

FB1

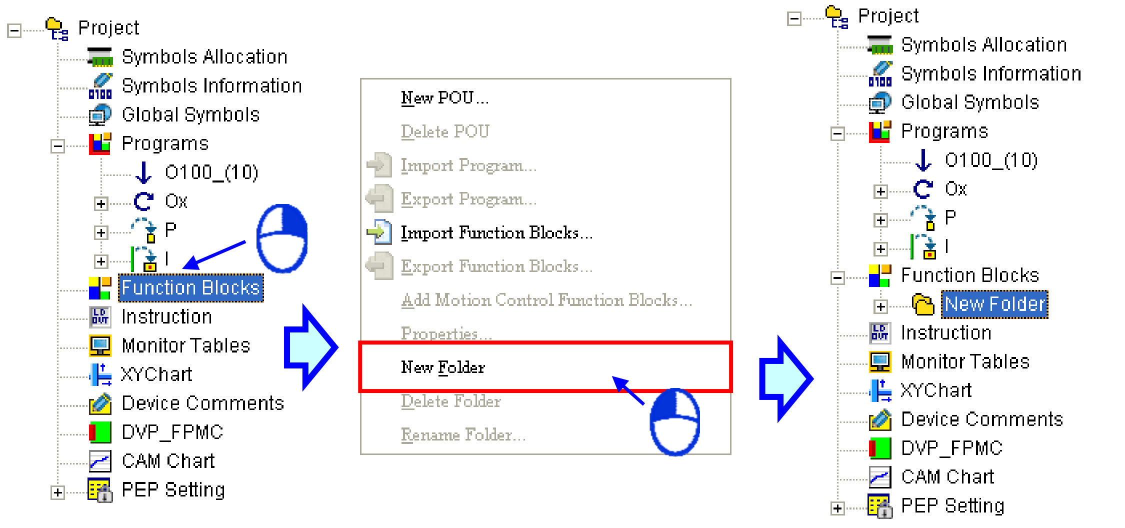
Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
INPUT	Var_1		BOOL	FALSE	
INPUT	Var_2		BOOL	FALSE	
OUTPUT	Var_3		BOOL	FALSE	

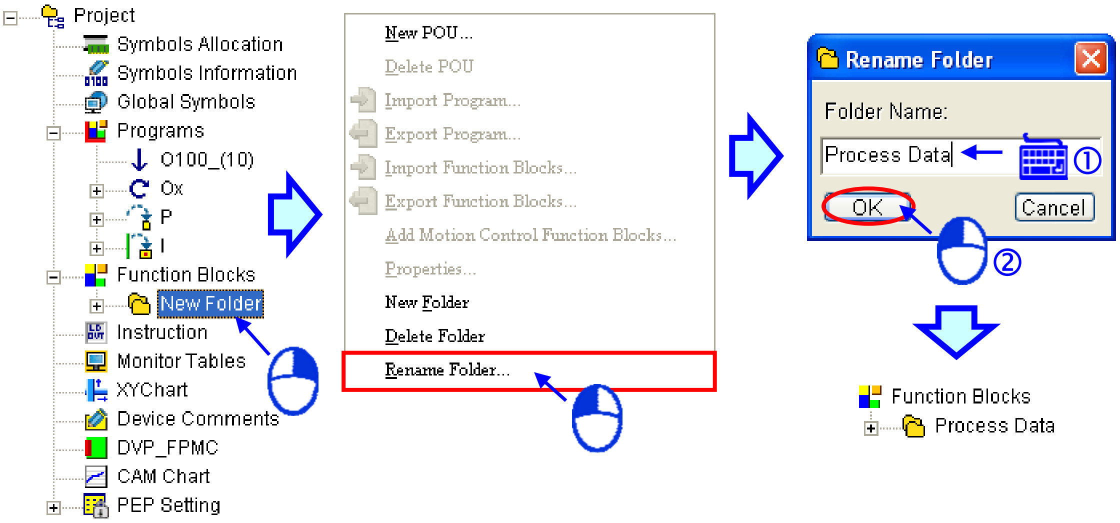
3.2.5 Managing POUs Which Are Function Blocks

Users can manage the function blocks in PMSoft by adding folders to **Function Blocks** in the system information area. A folder added contains a global symbol table, and the global symbol table is applicable to the function block definitions in the folder and the subfolders in the folder.

(1) If users want to add a folder to **Function Blocks** in the system information area, they have to right-click **Function Blocks**, and click **New Folder** on the context menu.

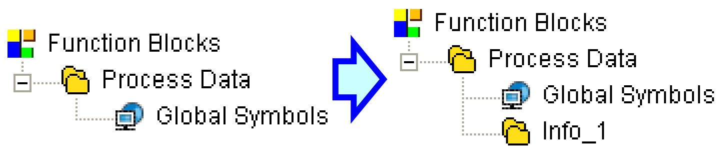


(2) If the users want to change the name of the folder added, they have to right-click **New Folder**, click **Rename Folder…** on the context menu, type a folder name in the **Rename Folder** window, and click **OK**.

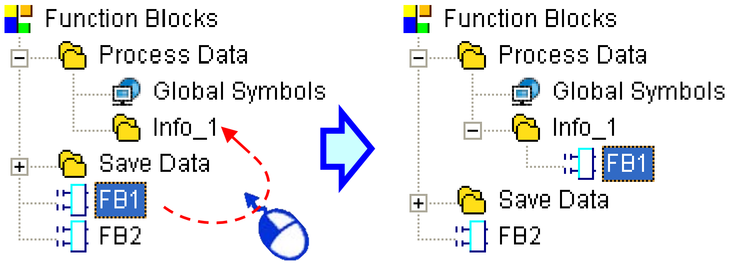


Folder Name: Process Data

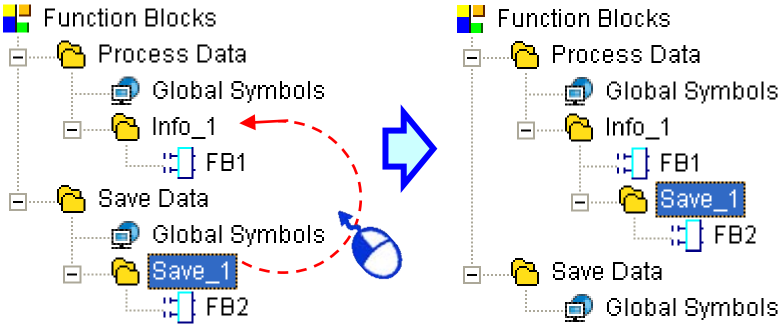
A global symbol table is in the folder added. Please refer to section 5.2.2 for more information. The users can add a subfolder to the folder in the way described above.

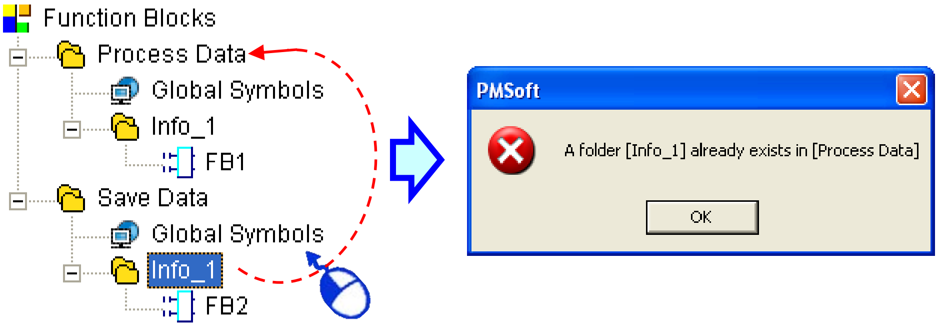


(3) The users can add a new POU which is a function block to the folder in the way described in section 3.2.1. They can also drag a POU which has been created to the folder.



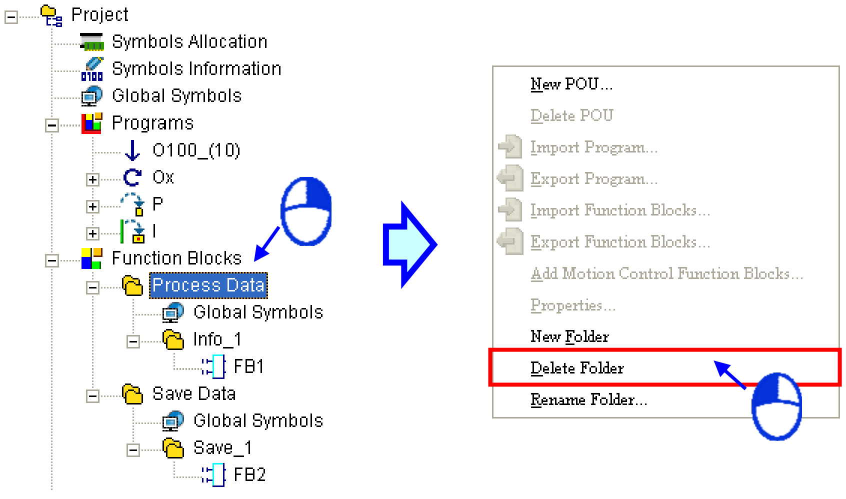
(4) If a subfolder is dragged to another folder, the POUs contained in the subfolder will also be dragged. If a subfolder is dragged to a folder containing a subfolder whose name is the same as the subfolder dragged, a warning message will appear. Besides, owing to the fact that a folder contains a global symbol, it can not be dragged.

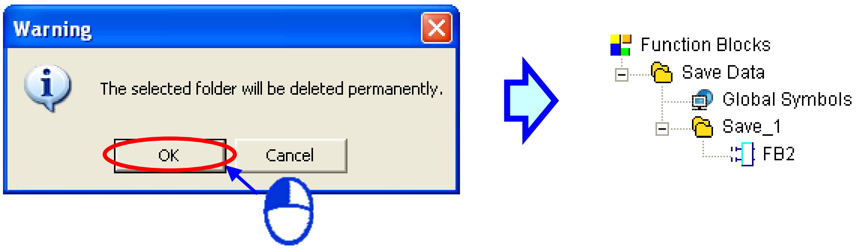




A folder [Info_1] already exists in [Process Data]

(5) If the users want to delete a folder/subfolder, they have to right-click the folder/subfolder, click **Delete Folder** on the context menu, and click **OK** in the **Warning** window. After a folder is deleted, the items in the folder will also be deleted. The global symbol table in a folder can not be deleted alone. If the users want to delete the global symbol table in a folder, they have to delete the folder.

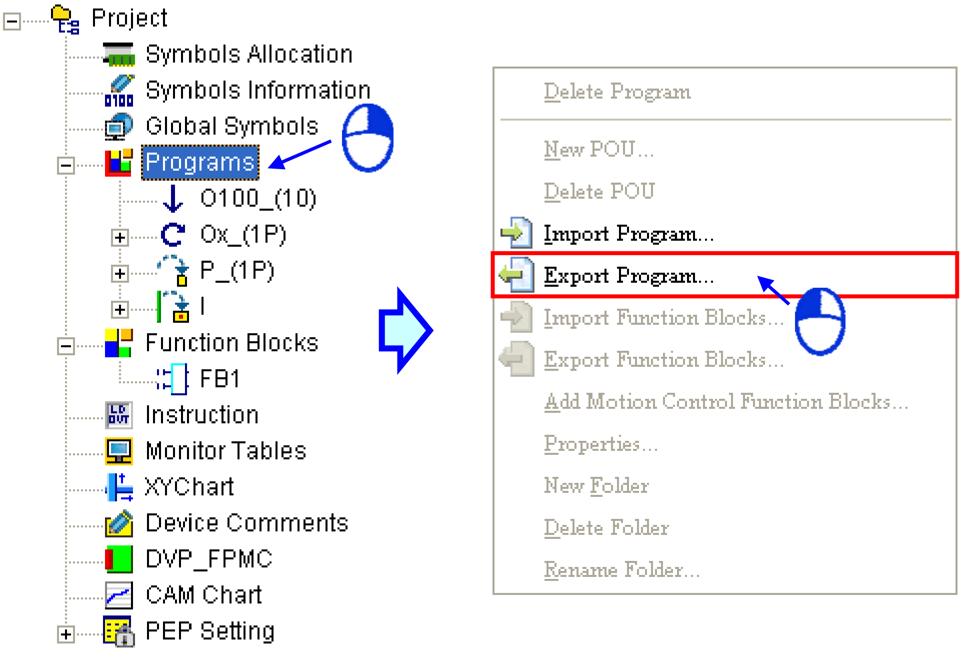



Warning

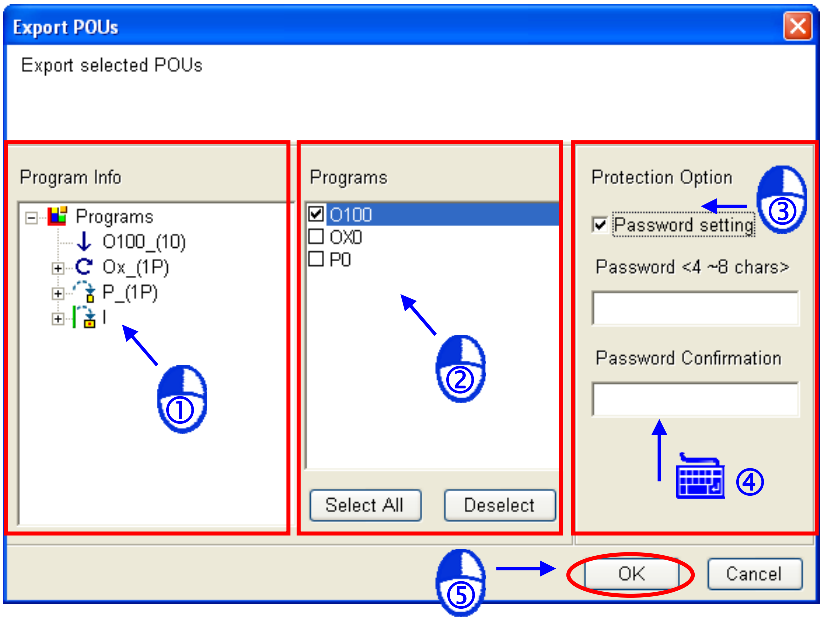
The selected folder will be deleted permanently.

3.2.6 Exporting POUs Which Are Programs

PMSoft allows users to export POUs which are programs, and import POUs which are programs. The users can edit the same POU which is a program in different projects. If the users want to export POUs which are programs, they have to right-click **Programs** in the system information area, and click **Export Program...** on the context menu.



After the users click **Export Program…** on the context menu, the **Export POUs** window will appear. The users can expand the program sections in the **Program Info** box, and select the POUs which they want to export in the **Programs** box. If the users click **Select All**, all the POUs in the **Programs** box will be selected. If the users click **Deselect**, all the POUs in the **Programs** box will not be selected. If the users set a password when they export POUs, they will be asked to type the password next time the POUs are imported. After the users click **OK** in the **Export POUs** window, they have to type a file name, and select a file path in the **Export Programs** window. The POUs exported are saved as a .mpu file.



Export POUs

Export selected POUs

Program Info
- Programs
  - O100_(10)
  - Ox_(1P)
  - P_(1P)
  - I

Programs

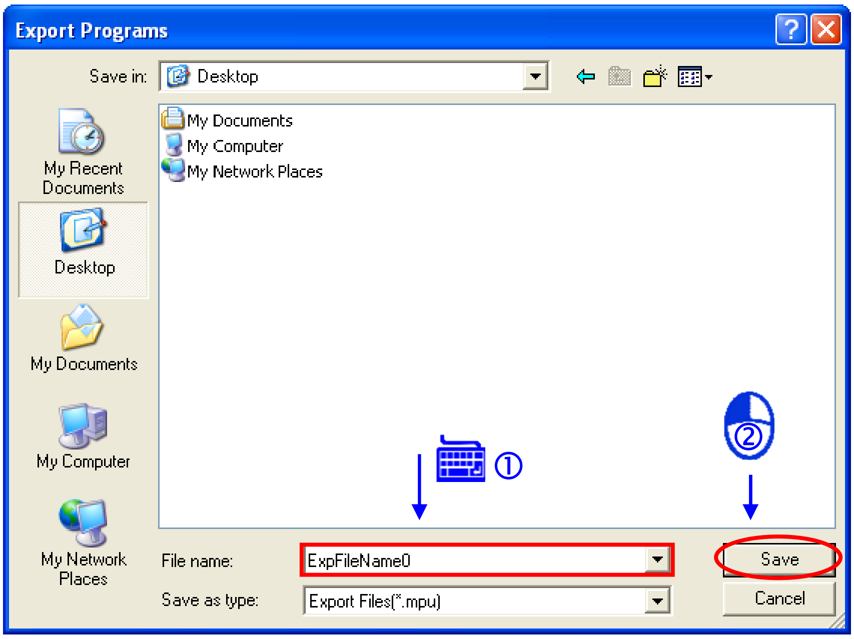
☑ O100
☐ OX0
☐ P0

Protection Option

☑ Password setting

Password <4 ~8 chars>

Password Confirmation



Export Programs

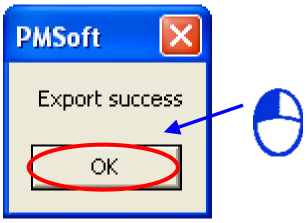
Save in: Desktop

My Documents
My Computer
My Network Places

File name: ExpFileName0

Save as type: Export Files(*.mpu)

After the user click **Save** in the **Export Programs** window, the POUs selected in the **Export POUs** window will be exported.

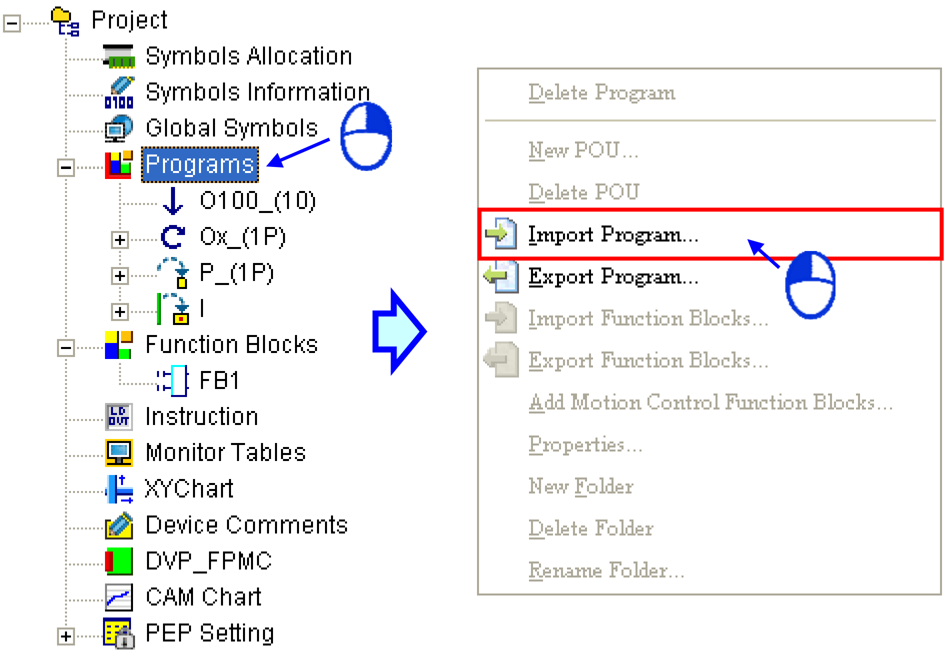


PMSoft

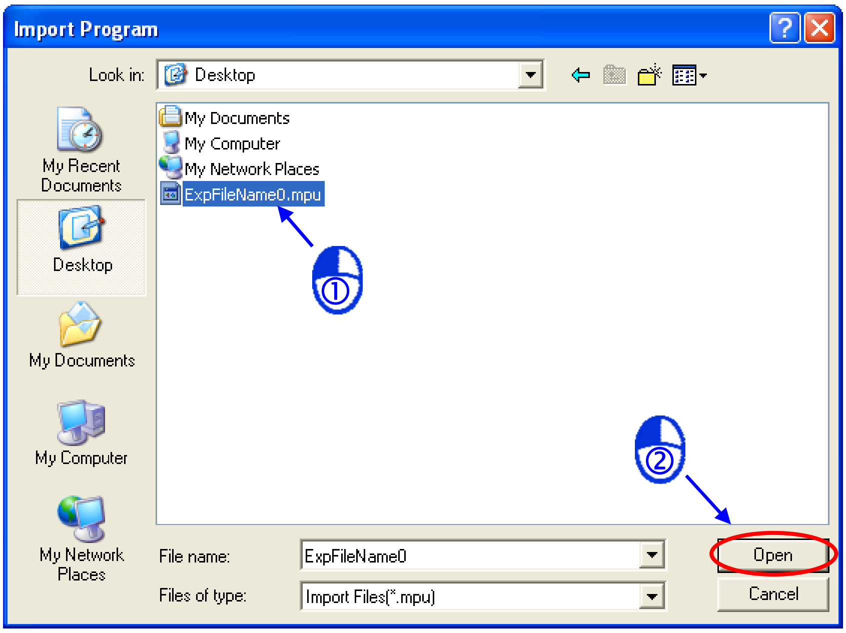
Export success

3.2.7 Importing POUs Which Are Programs

If users want to import POUs which are programs, they have to right-click **Programs** in the system information area, and click **Import Program...** on the context menu.



(1) After the users select the file which they want to import in the **Import Program** window, they have to click **Open**.



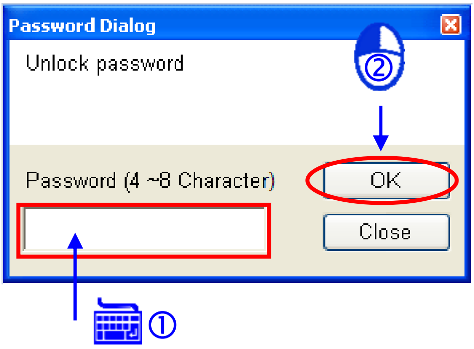
Import Program

Look in: Desktop

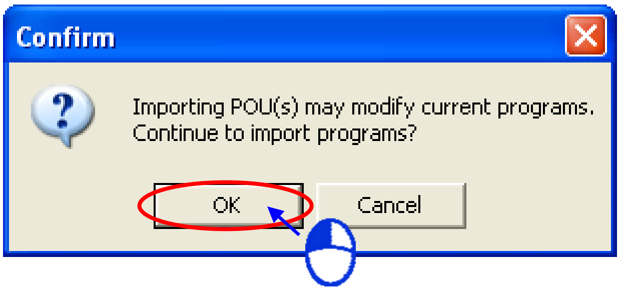
File name: ExpFileName0

Files of type: Import Files(*.mpu)

(2) If the file selected is protected by a password, the users have to type the correct password, and click **OK** in the **Password Dialog** window.



(3) After the users click **OK** in the **Password Dialog** window, the **Confirm** window will appear. If the users decide to import the file, they have to click **OK** in the **Confirm** window.



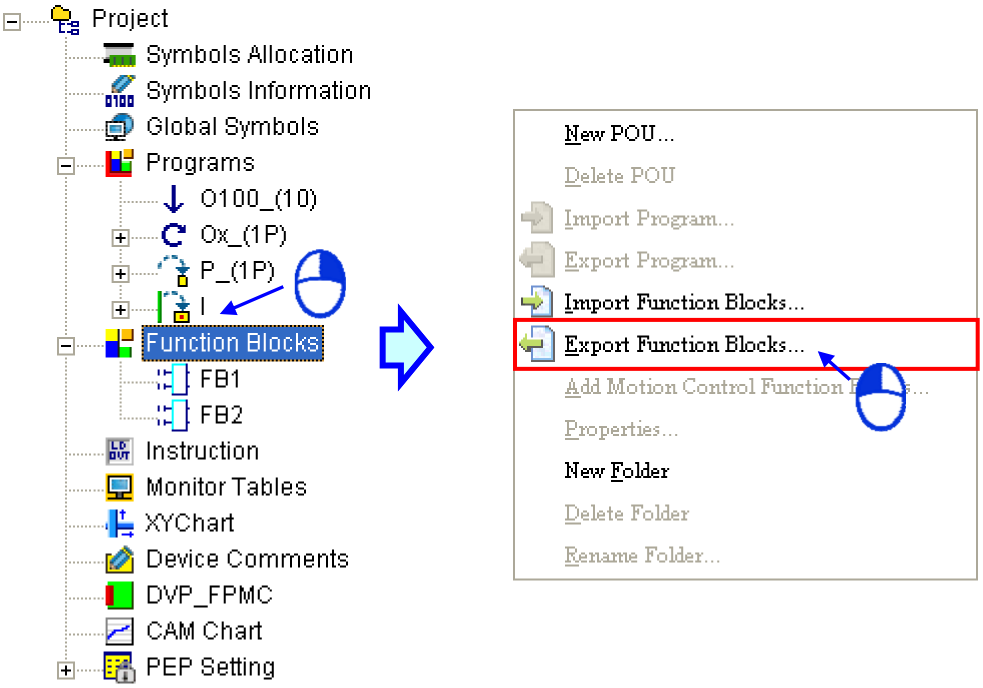
Importing POU(s) may modify current programs.
Continue to import programs?

***1. POU numbers can not be changed. If a POU number imported is the same as a POU number in PMSoft, the POU number in PMSoft will be overwritten.**

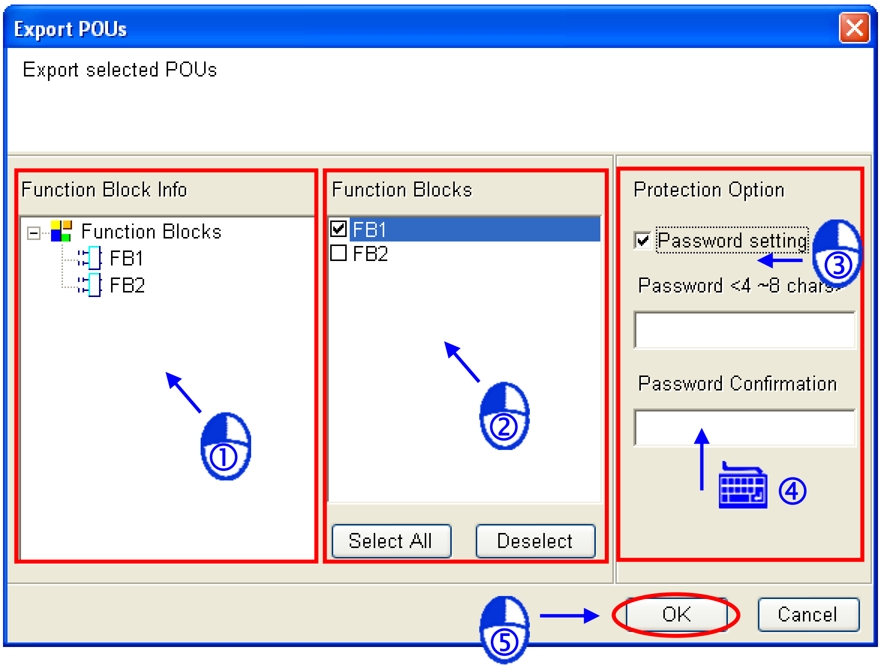
***2. If the POUs which are imported do not include a POU created in PMSoft, the POU will not be overwritten. For example, if Ox0 and Ox1 are created in PMSoft, and another Ox0 is imported, Ox1 in PMSoft will remain.**

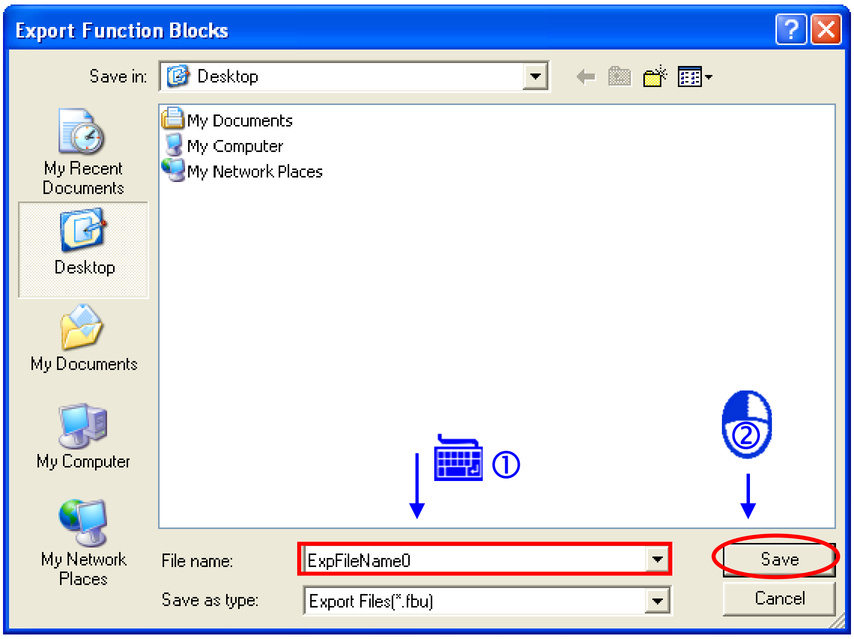
3.2.8 Exporting POUs Which Are Function Blocks

PMSoft allows users to export POUs which are function blocks, and import POUs which are function blocks. The users can edit the same POU which is a function block in different projects. If a POU which is a function block is exported, the folder in which the POU is contained, and the global symbol table which is contained in the folder will also be exported. If the users want to export POUs which are function blocks, they have to right-click **Function Blocks** in the system information area, and click **Export Function Blocks...** on the context menu.



After the users click **Export Function Blocks…** on the context menu, the **Export POUs** window will appear. The users can expand the function block sections in the **Function Block Info** box, and select the POUs which they want to export in the **Function Blocks** box. If the users click **Select All**, all the POUs in the **Function Blocks** box will be selected. If the users click **Deselect**, all the POUs in the **Function Blocks** box will not be selected. If the users set a password when they export POUs, they will be asked to type the password next time the POUs are imported. After the users click **OK** in the **Export POUs** window, they have to type a file name, and select a file path in the **Export Function Blocks** window. The POUs exported are saved as a .fbu file.





Export Function Blocks

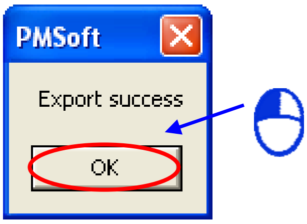
Save in: Desktop

My Documents
My Computer
My Network Places

File name: ExpFileName0

Save as type: Export Files(*.fbu)

After the user click **Save** in the **Export Function Blocks** window, the POUs selected in the **Export POUs** window will be exported.

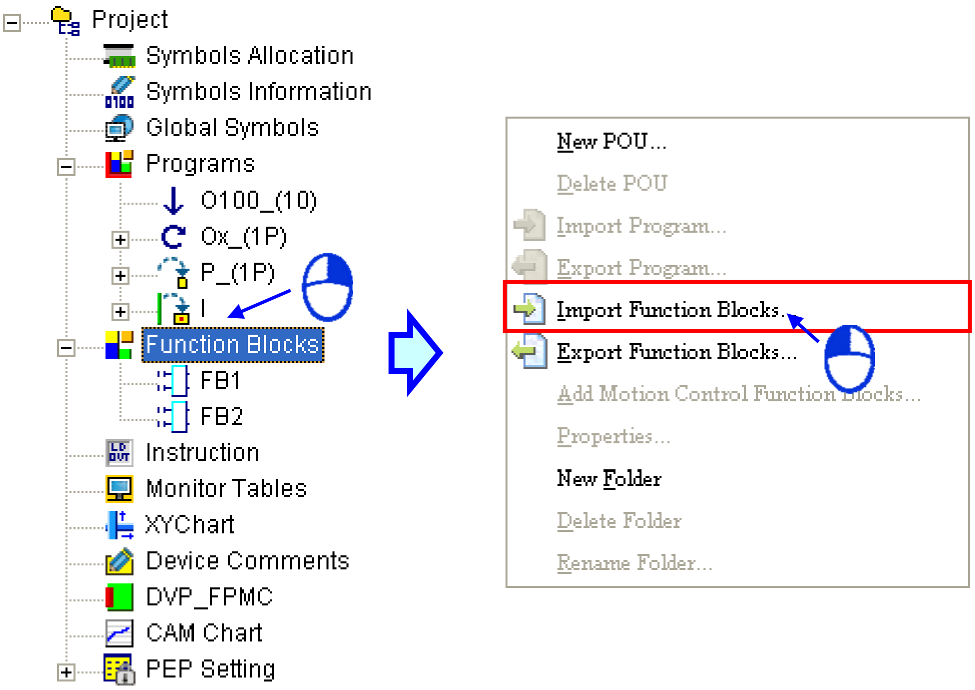


PMSoft

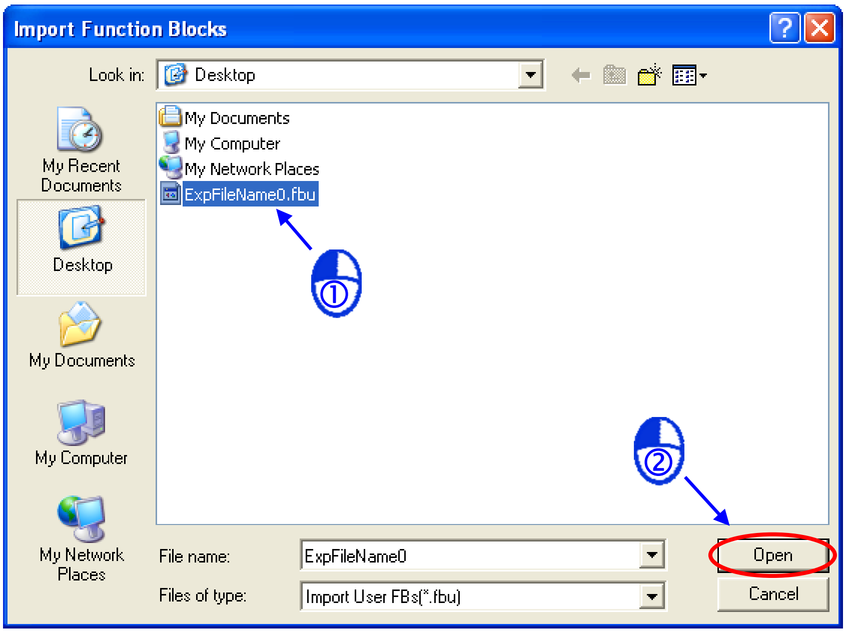
Export success

3.2.9 Importing POUs Which Are Function Blocks

If POUs which are function blocks are imported, the folders in which the POUs are contained, and the global symbol tables which are contained in the folders will also be imported. If users want to import POUs which are function blocks, they have to right-click **Function Blocks** in the system information area, and click **Import Function Blocks...** on the context menu.



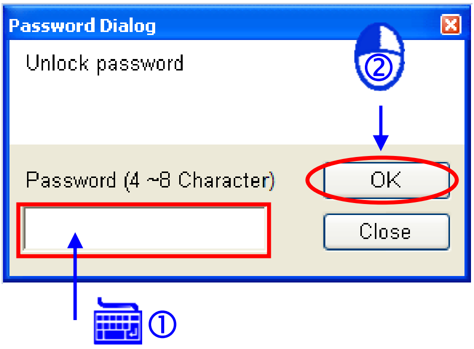
(1) After the users select the file which they want to import in the **Import Function Blocks** window, they have to click **Open**.



File name: ExpFileName0

Files of type: Import User FBs(*.fbu)

(2) If the file selected is protected by a password, the users have to type the correct password, and click **OK** in the **Password Dialog** window.



*. If the POUs which are imported include a POU created in PMSoft, the POU can not be imported.

Chapter 4 Symbols

Table of Contents

4.1 Introduction of Symbols

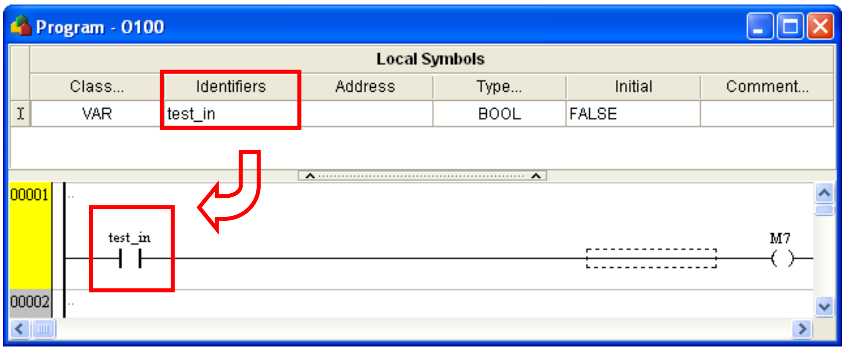
During the process of developing a traditional program for a PLC, it generally takes much time to manage device addresses. Besides, managing or debugging the program in a big project is a burden on users. As a result, the concept of symbols in a high-level programming language is introduced into IEC 61131-3. A device in a PLC can be represented by a symbol, and a device can be automatically assigned to a symbol. The time of assigning devices is saved, a program is more readable, and the efficiency of developing a program increases. Variables in PMSoft are called symbols. As a result, variables are the same as symbols in terms of meaning in this manual.

4.1.1 Application of Symbols and Creation of Identifiers

A symbol has to be declared before it is used. There are two types of symbols. They are global symbols and local symbols. The global symbols in a project can be used in all the POUs in the project, and the local symbols in a project can only be used in the POU in which the local symbols are declared. Besides, the identifier of a local symbol in a POU can be the same as the identifier of a local symbol in another POU. However, if the identifier of a local symbol declared in a POU is the same as the identifier of a global symbol, the system will automatically regard the local symbol declared in the POU as a local symbol.

The regulations of creating the identifier of a symbol are as follows.

- G-codes do not support symbols.
- A symbol can not be declared repeatedly in a symbol table.
- An identifier is composed of 30 characters at most, and a Chinese character occupies two characters.
- The identifier of a symbol is case-insensitive. A symbol can be composed of underlines, English letters, numerals, and Chinese characters. Spaces can not be used.
- A symbol can not be composed of constants, or be a number preceded by “K”, “KK”, “H”, “HH”, or “F”. If a symbol begins with “DD”, and “DD” is followed by a decimal value, the symbol represents a 32-bit data registers, and is an illegal symbol.
- The identifier of a symbol can not be a name reserved by the system, e.g. an instruction code, a device name, or a name given a special significance. However, if a name reserved by the system is a part of the identifier of a symbol, the identifier is a legal identifier. For example, “M0” is an illegal identifier, but “_M0” is a legal identifier.
- Underlines can be used, but they can not be used continuously. For example, “INPUT_CH0” is a legal identifier, but “INPUT__CH0” and “INPUT_CH0_” are illegal identifiers. An underline can not be put at the end of an identifier
- Special marks can not be used. For example, *, #, ?, \, %, @, and etc. can not be used.
- If a device is assigned to a symbol, users have to make sure that the device is in the device range allowed.
- If an index symbol is used, the index register modifying it must be in the range of V0~V7, orZ0~Z7.
- A folder added to **Function Blocks** in the system information area contains a global symbol table, and the global symbol table is applicable to the function block definitions in the folder and the subfolders in the folder. Please refer to section 5.2.2 for more information.



Program - 0100

Local Symbols

	Class...	Identifiers	Address	Type...	Initial	Comment...
I	VAR	test_in		BOOL	FALSE	

4.1.2 Classes

In terms of functions, symbols can be classified into four classes. The characteristics of these four classes are described below.

- **VAR - General symbol**

  The symbols of this class are for general operations only. The significance of a symbol of this class depends on the data type of the symbol.

- **INPUT - Symbol used as an input pin of a function block**

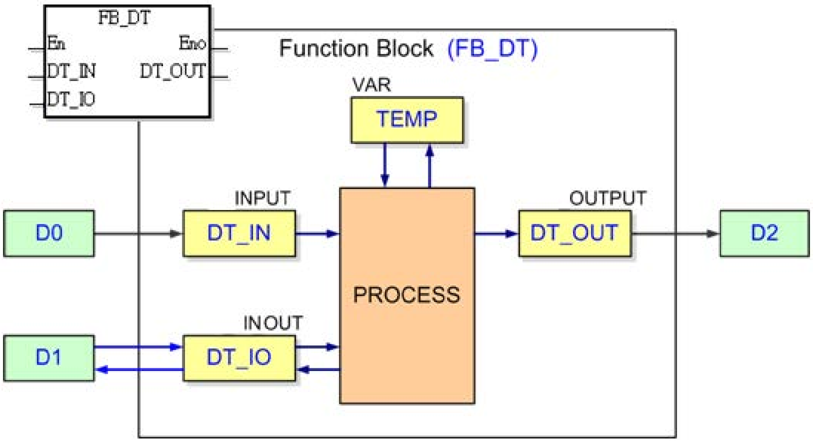
  A symbol of this class is used as an input pin of a function block. It can only be declared in the function block. If a function block is called, the symbols of this class can receive the input values sent by the caller. Besides, in a ladder diagram, the symbols of this class are put at the left sides of function blocks, and the pins which receive the input values sent by the caller are assigned to the symbols of this class.

- **OUTPUT - Symbol used as an output pin of a function block**

  A symbol of this class is used as an output pin of a function block. It can only be declared in a function block. After the execution of a function block is complete, the operation result will be sent to the caller through the symbols of this class. Besides, in a ladder diagram, the symbols of this class are put at the right sides of function blocks, and the pins which send the operation results to the caller are assigned to the symbols of this class.

- **INOUT - Symbol used as a feedback pin of a function block**

  A symbol of this class is used as a feedback pin of a function block. It can only be declared in the function block. Please refer to the following example. When the function block is called, the caller sends the value in D1 to DT_IO, which is a symbol of the INOUT class. After the operation comes to an end, the final value of DT_IO is sent to D1. Besides, in a ladder diagram, the symbols of this class are put at the left sides of function blocks.



4.1.3 Data Types

The data type of a symbol determines the significance of the value of symbol. Suppose there are two symbols VAR_1 and VAR_2. The data type of VAR_1 is BOOL, and the data type of VAR_2 is WORD. If VAR_1 and VAR_2 are used in a program, VAR_1 will represent a contact, and VAR_1 will represent a 16-bit device which can be involved in arithmetic or data transfer.

The data types supported by PMSoft are listed below.

Data type	Description	Program	Function block
BOOL	Boolean data type A Boolean value represents the state of a contact	✓	✓
WORD	16-bit value 16-bit data can be stored.	✓	✓

Data type	Description	Program	Function block
DWORD	32-bit value 32-bit data can be stored.	✓	✓
LWORD	64-bit value 64-bit data can be stored.	✓	✓
FLOAT	32-bit value It is applicable to floating-point number instructions.	✓	✓
COUNTER	16-bit counter value or 32-bit counter value It represents a counter.	✓	✓
TIMER	16-bit timer value It represents a timer.	✓	✓
ARRAY	Array If a symbol is declared, the size of an array and an array type must be specified. (An array is composed of 256 elements at most.)	✓	✓
Function Block	It represents a function block definition. (*)	✓	✓

***. A symbol representing a function block definition has a special significance. Please refer to chapter 5 for more information.**

4.1.4 Assigning a Device to a Symbol and Setting the Initial Value of a Symbol

A device is assigned to a symbol according to the data type of the symbol. Users can set the initial value of a symbol. After the program in a project is downloaded to a motion controller, the initial values of the symbols will be written into the devices assigned to the symbols if the program is scanned for the first time.

The principles of assigning devices to symbols are as follows.

- Users can assign devices to the global symbols and the local symbols declared in the POUs which are programs. The system can also automatically assign devices to the global symbols and the local symbols declared in the POUs which are programs.
- If a local symbol declared in a function block is not a symbol of the VAR class, the system will automatically assign a device to the symbol, and users can not assign a device to the symbol.
- The devices assigned by the system are usable devices. (Users can set a range of devices which can be assigned automatically.)
- If a symbol is declared, the device assigned to the symbol, the data type of the symbol, and the initial value of the symbol must be compatible with one another.

The relation between the data types and the device types which can be assigned is described below.

Data type	AH500		DVP	
	Device assigned by users	Device assigned by the system	Device assigned by users	Device assigned by the system
BOOL	Contact M/SM or bit in the device X/Y (*3)	Contact M/SM	Contact M/X/Y	Contact M
WORD	D/W/X/Y/SR/V	D/W	D/V	D
DWORD	D/W/X/Y/SR/Z	D/W	D/Z	D
LWORD	D/W/X/Y/SR	D/W	D	D
FLOAT	D/W/X/Y/SR	D/W	D	D
COUNTER	C	C	C	C
TIMER	T	T	T	T

Data type	AH500		DVP	
	Device assigned by users	Device assigned by the system	Device assigned by users	Device assigned by the system
ARRAY	The devices assigned to a symbol whose data type is ARRAY depend on the array type specified. An array is composed of the devices starting from the device assigned by users or the system, and the number of devices in an array conforms to the size of the array.			

***1. Please refer to section 4.2.8 for more information about setting a range of devices which can be assigned automatically.**

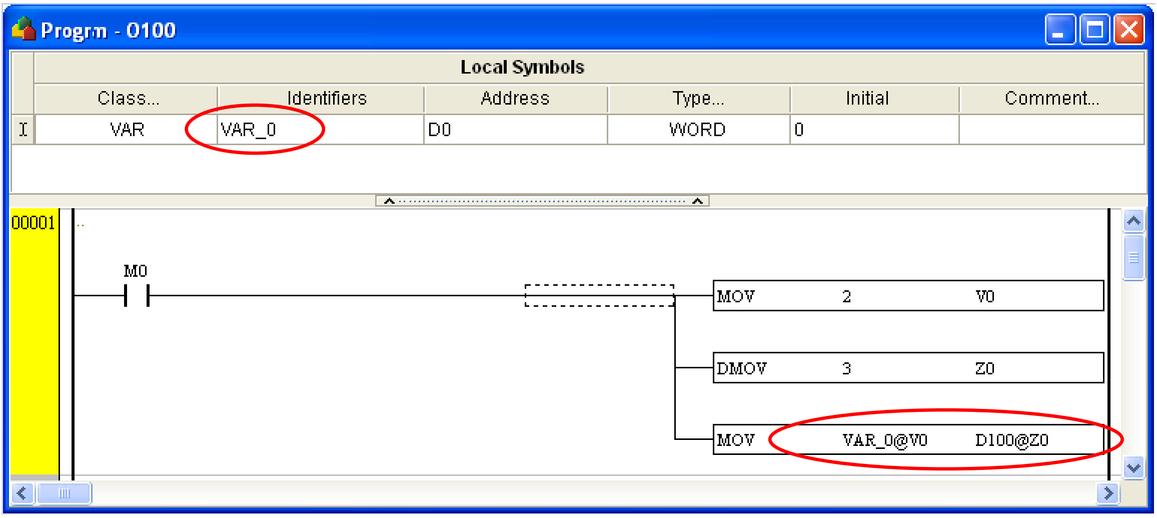
***2. A symbol representing a function block definition has a special significance. Please refer to chapter 5 for more information.**

***3. X0.0 and Y0.1 are bits in the word devices X and Y. Please refer to section 1.3 for more information.**

4.1.5 Modifying a Symbol with an Index Register

Users are allowed to use index registers in PMSoft. There are two types of index registers. The V devices are like general data registers in that they are 16-bit data registers. The users can write data into the V devices and read data from the V devices freely. If a V device is used as a general register, it can only be used in a 16-bit instruction. The Z devices are 32-bit data registers. If a Z device is used as a general register, it can only be used in a 32-bit instruction. The modification of a symbol by an index register is represented by format is **<u>Identifier@Index register</u>**. If a V/Z device is used to modify an operand, it can be used in a 16-bit instruction or a 32-bit instruction.

Please refer to the program below. The device assigned to VAR_0 is D0. The data stored in an index register indicates the offset for the object which the index register modifies. If the value in the index register V0 is 2, VAR_0@V0 indicates that 2 is added to the device address (D0) assigned to VAR_0, that is, VAR_0@V0 represents D2. If M0 is ON, the value in V0 will be 2, the value in Z0 will be 3, and the value in D2 will be moved to D103.

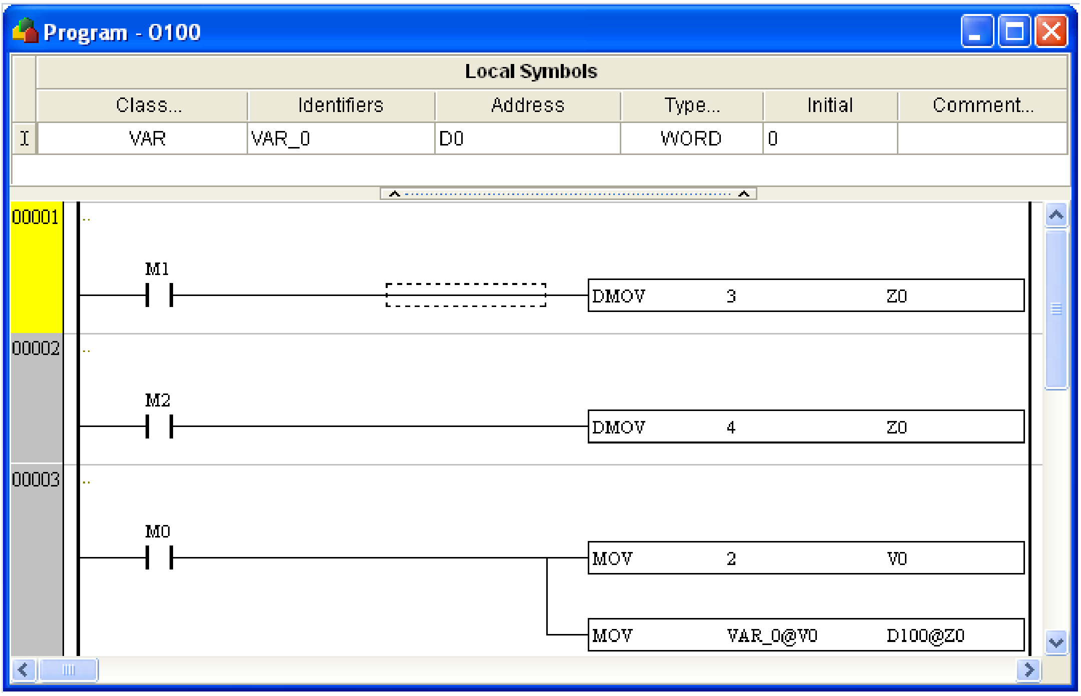


Local Symbols

	Class...	Identifiers	Address	Type...	Initial	Comment...
I	VAR	VAR_0	D0	WORD	0	

```
00001
   M0
---| |----------------+--[MOV   2         V0     ]
                      |
                      +--[DMOV  3         Z0     ]
                      |
                      +--[MOV   VAR_0@V0  D100@Z0]
```

Besides, if the value in an index register is changed, the device which actually operates differs from the original device. As a result, if the original device is not used in the program, the final value in the original device is retained. In the figure below, if the value in Z0 is 3, the value in D2 will be moved to D103. If the value in Z0 is changed from 3 to 4, the value in D2 will be moved to D104, and the value in D103 will remain unchanged.



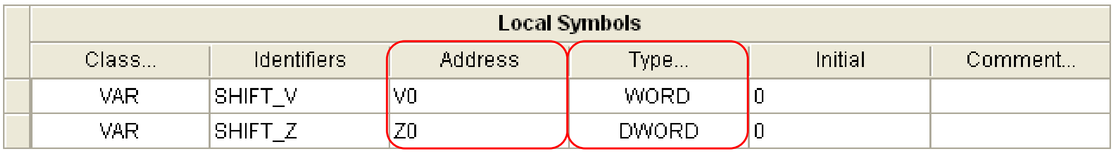
Program - 0100

Local Symbols					
Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	VAR_0	D0	WORD	0	

***1. The data stored in an index register indicates the offset for the device which the index register modifies. If the system automatically assigns a device to a symbol, the use of an index register to modify the symbol will cause the program to be executed incorrectly because users do not know which device is assigned to the symbol.**

***2. Bit devices can only be modified by index registers in some applied instruction. Please refer to DVP-PM Application Manual for more information.**

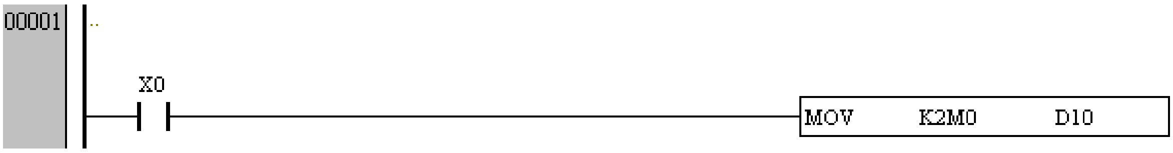
If the users want to assign index registers to symbols, they have to specify device addresses and data types.



Local Symbols					
Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	SHIFT_V	V0	WORD	0	
VAR	SHIFT_Z	Z0	DWORD	0	

4.1.6 Combining Bit Devices into a Word Devices

In a DVP series motion controller, the X/Y/M/S devices are bit devices. Some applied instructions allowed the bit devices to be combined into word devices. A word device composed of bit devices is represented by KnX/KnY/KnM/KnS. If n is 1, four bit devices will be used. As a result, n must be in the range of 1 to 1 if a 16-bit instruction is used, and n must be in the range of 1 to 8 if a 32-bit instruction is used. For example, K4M0 indicates that M0~M15 are used. In the program below, if X0 is ON, the values in M0~M7 will be moved to bit 0~bit 7 in D10, and the values of bit 8~bit 15 in D10 will be 0.



***1. Please refer to DVP-PM Application Manual for more information about the applied instructions which allow bit devices to be combined into word devices.**

***2. Different applied instructions may place different restrictions on the modification of a bit device by Kn. Please refer to DVP-PM Application Manual for more information.**

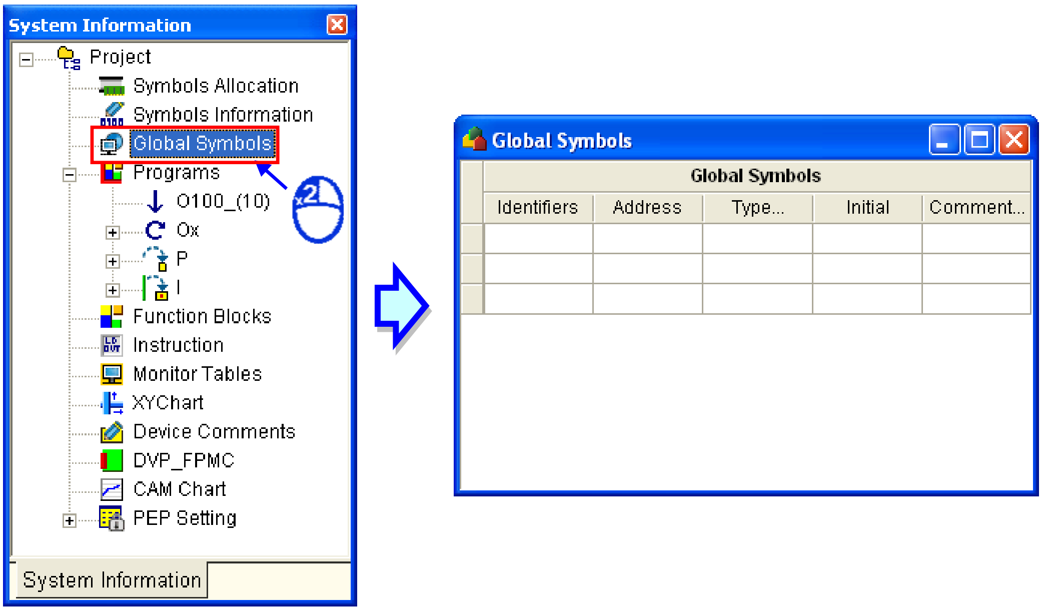
In an AH500 series motion controller, the X/Y devices can not be combined into word devices. Some applied instructions allow the M/S/SM devices to be combined into word devices.

4.2 Managing the Symbols in PMSoft

4.2.1 Symbol Tables

- **Global symbol table**

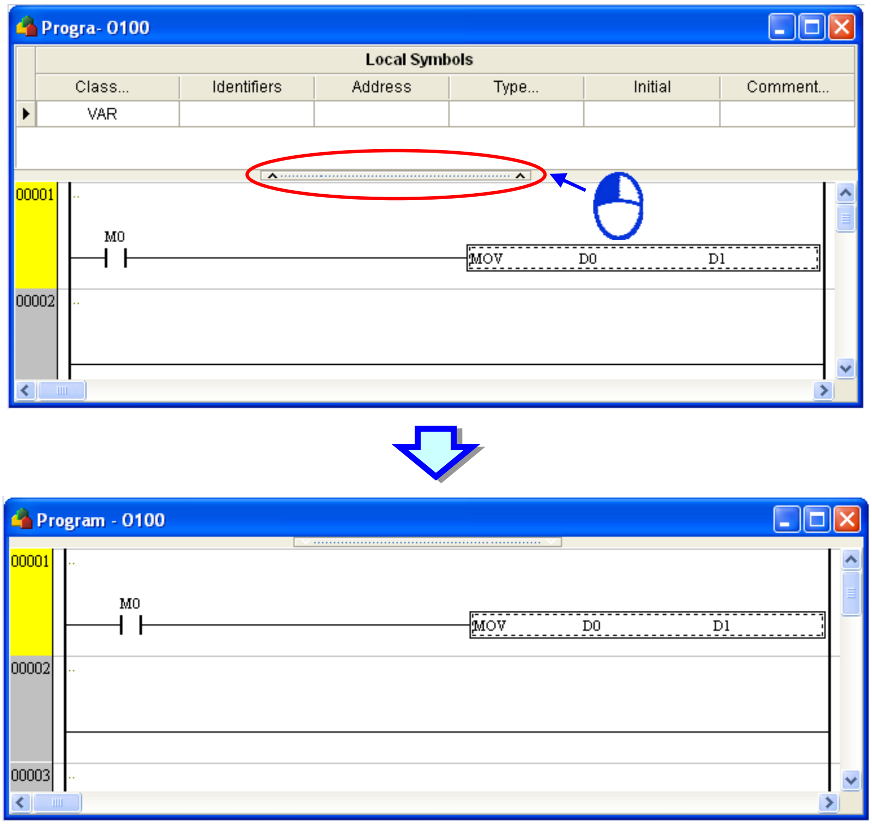
  After users double-click **Global Symbols** in the system information area, the **Global Symbols** window will appear.



Global Symbols				
Identifiers	Address	Type...	Initial	Comment...

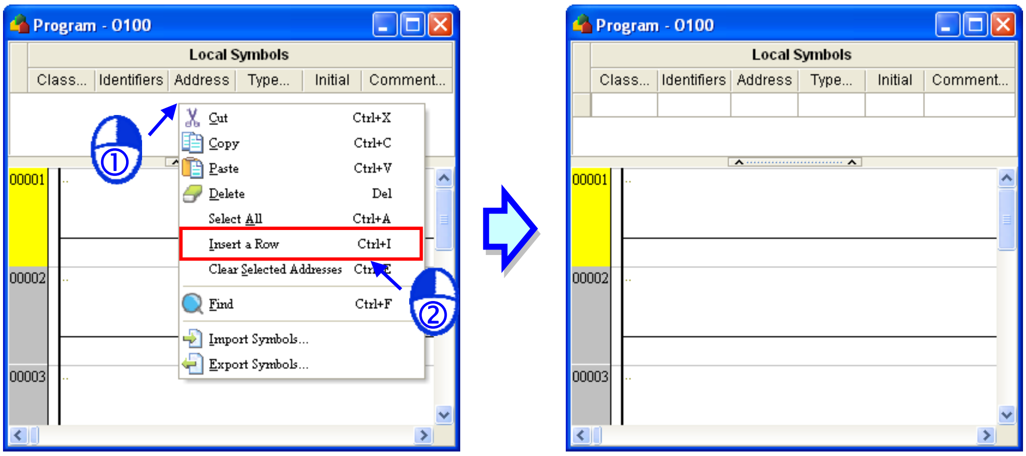
- **Local symbol table**

  In PMSoft, a local symbol table is at the top of the window for a POU. After users click the button under a local symbol table, the local symbol table will be hidden. If the users click the button again, the local symbol table will be displayed.

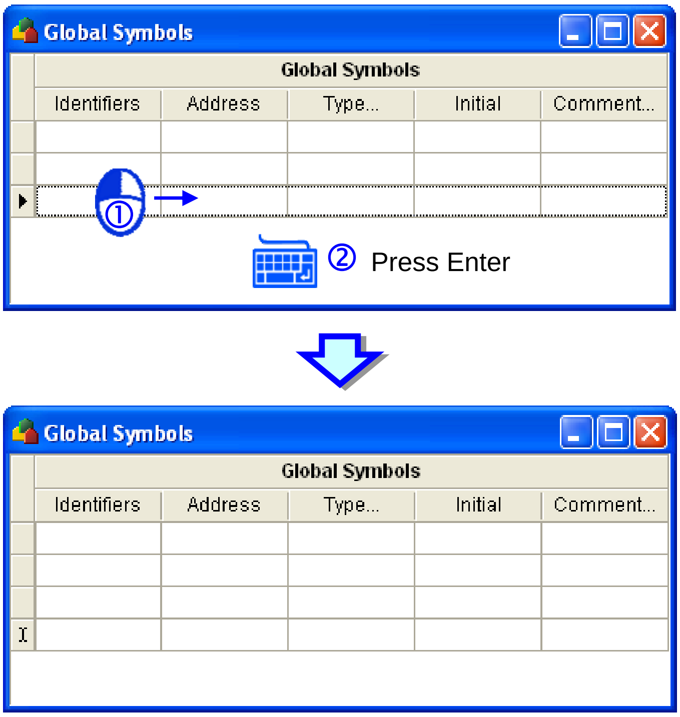


4.2.2 Adding a Symbol

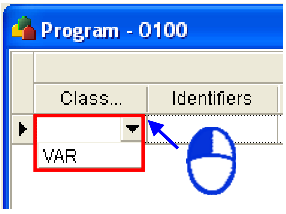
(1) After users open the **Global Symbols** window or the window for a POU for the first time, the system will create a table in the window. If the users want to insert a row in the table, they have to right-click a cell in the table, and click **Insert a Row** on the context menu.

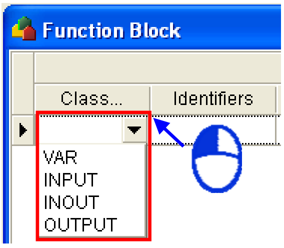


If the users click a cell in the last row, and press Enter on the keyboard, they can also insert a row in the table.



(2) The users can select a class in a **Class…** drop-down list cell, or type a class in a **Class…** drop-down list cell. The items which can be selected in a **Class…** drop-down list box vary with the POU created. Please refer to section 4.1.2 for more information.

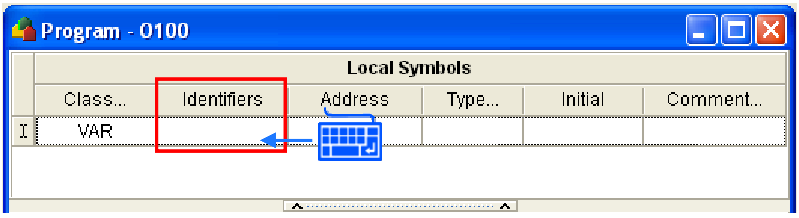




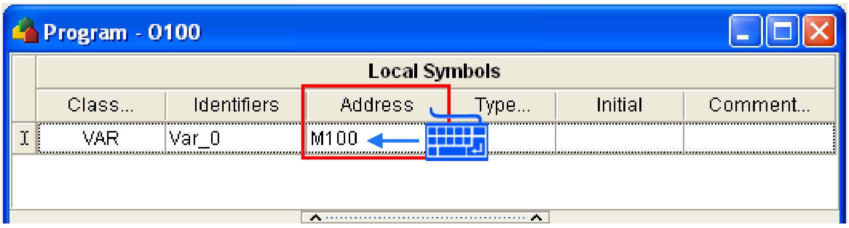
***1. VAR is the only item which can be selected in the Class… drop-down list cell for a local symbol in the window for a POU which is a program. There is no Class… column in a global symbol table.**

***2. The items which can be selected in the Class drop-down list box for a local symbol in a POU which is a function block are VAR, INPUT, INOUT, and OUTPUT.**

(3) The users can type an identifier in an **Identifiers** cell. Please refer to section 4.1.1 for more information about creating an identifier.



(4) If the **Address** cell for a symbol is blank, the system will automatically assign a device to the symbol. If the users want to assign a device to a symbol, they have to type a device address in the **Address** cell for the symbol. Please refer to section 4.1.4 for more information.



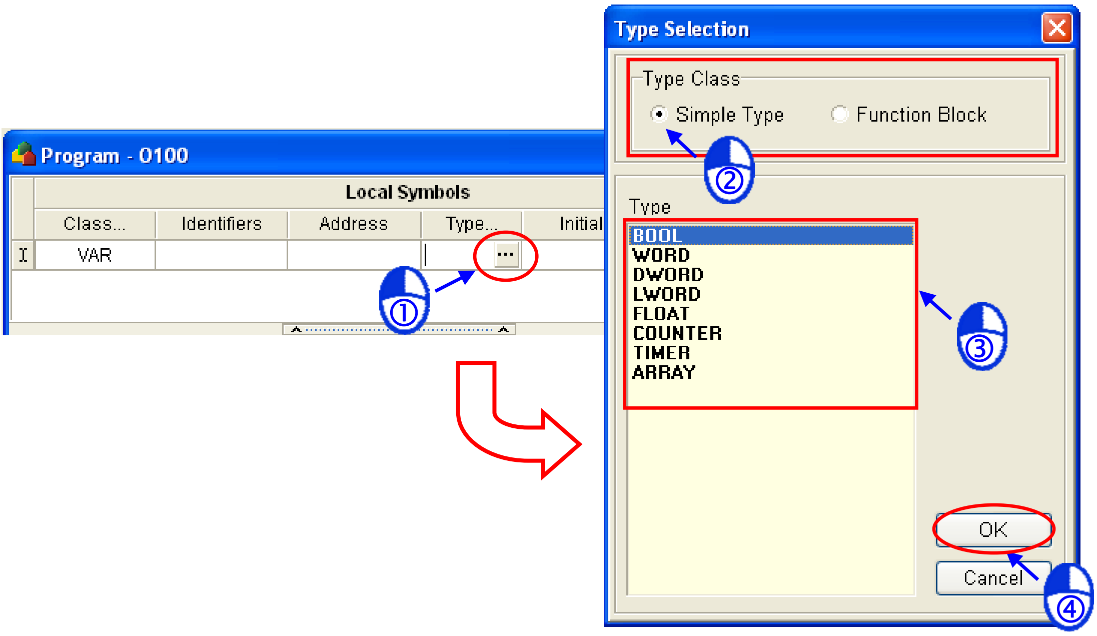
Program - 0100

Local Symbols					
Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	Var_0	M100			

***1. If the data type of a symbol is a function block, user can not type a device address in the Address cell for the symbol.**

***2. If a symbol is declared in the local symbol table in the window for a POU which is a function block, and the symbol declared is a symbol of the INPUT/INOUT/OUTPUT class, users can not type a device address in the Address cell for the symbol, otherwise an error message will appear after the program created is compiled.**

(5) After the users click the button at the right side of a **Type…** cell, the **Type Selection** window will appear. After the users set a data type, they have to click **OK**. Please refer to section 4.1.3 for more information about data types.



Program - 0100

Local Symbols				
Class...	Identifiers	Address	Type...	Initial
VAR			...	

Type Selection

Type Class

● Simple Type ○ Function Block

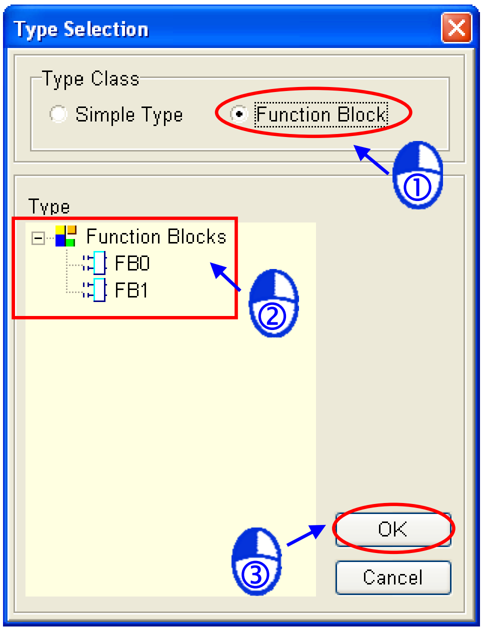
Type

BOOL
WORD
DWORD
LWORD
FLOAT
COUNTER
TIMER
ARRAY

The setting about the **Function Block** option button, and the setting about **ARRAY** in the **Type** box are described below.

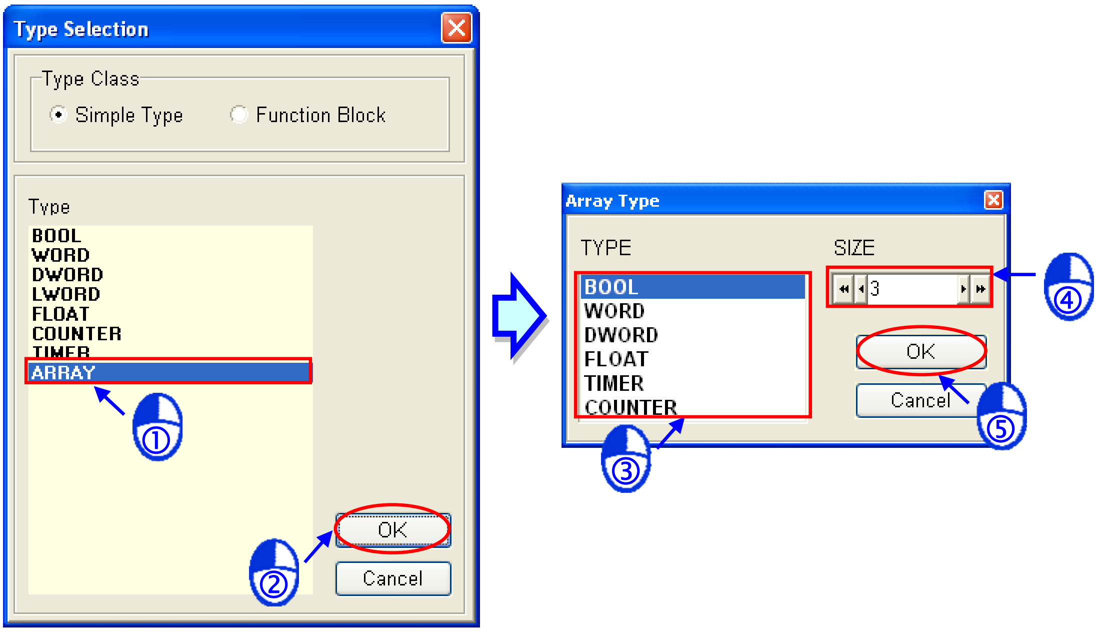
- **Function Block**

  After the users select the **Function Block** option button in the **Type Class** section, they have to select a function block definition in the **Type** box. Declaring a symbol whose data type is a function block is equivalent to declaring an instance of the function block definition which is selected. Please refer to chapter 5 for more information about function blocks.

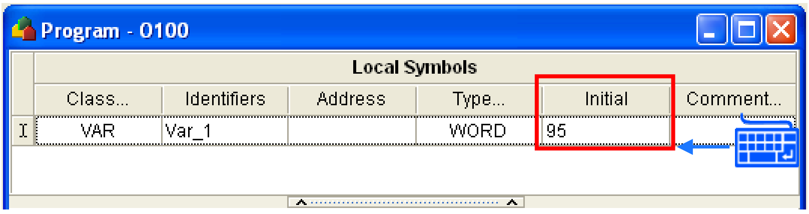


- **ARRAY**

  If **Array** in the **Type** box is selected, the **Array Type** window will appear after **OK** is clicked. The users have to select an array type, and set the size of the array created. The size of an array must be in the range of 1 element to 256 elements. The users can click ◂ or ▸ to decrease or increase the digit in the ones place of the setting value, and they can click ◂◂ or ▸▸ to decrease or increase the digit in the tens place of the setting value.



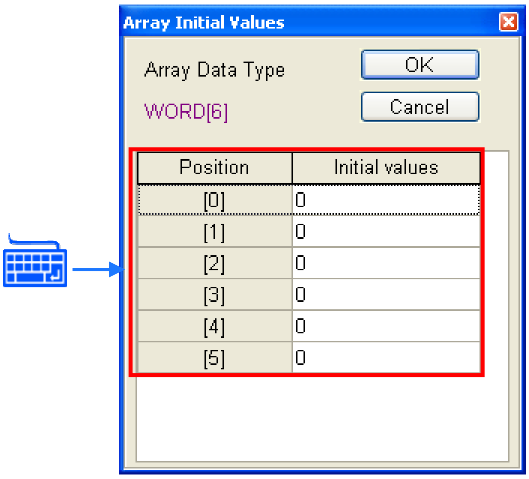
(6) The users can type the initial value of a symbol in the **Initial** cell for the symbol. The initial value typed must be compatible with the data type of the symbol. If the data type of a symbol is BOOL, the initial value of the symbol must be TRUE or FALSE. If the data type of a symbol is WORD or DWORD, the initial value of the symbol can be preceded by 16#, which indicates that the value following it is a hexadecimal value. For example, 16#2A0 represents a hexadecimal value. If the value in an Initial cell is not preceded by 16#, the value is a decimal value.



Local Symbols

	Class...	Identifiers	Address	Type...	Initial	Comment...
I	VAR	Var_1		WORD	95	

If the data type of a symbol is ARRAY, the **Array Initial Values** window will appear after the users click the **Initial** cell for the symbol. In the **Array Initial Values** window, the users have to set the initial values of the elements in the array created.



Array Data Type

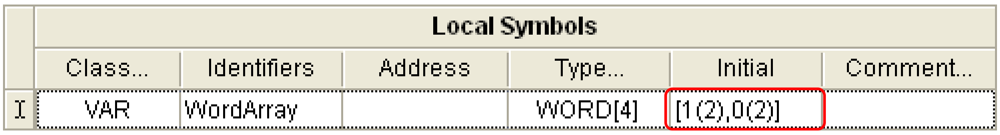
WORD[6]

Position	Initial values
[0]	0
[1]	0
[2]	0
[3]	0
[4]	0
[5]	0

*. **If the data type of a symbol is a function block, users can not type a value in the Initial cell for the symbol, otherwise an error message will appear after the program created is compiled.**

The users can also type values in the **Initial** cell for a symbol whose data type is ARRAY. They have to pay attention to the following points.

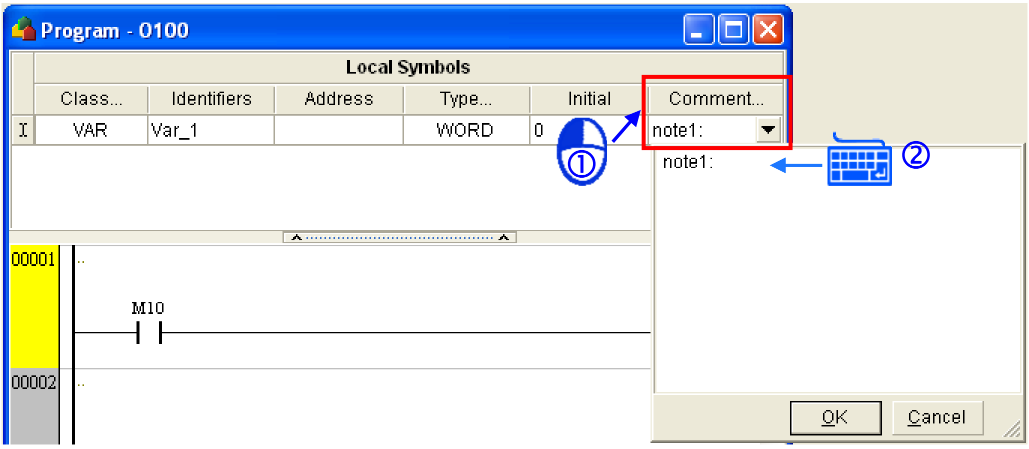
- Values are put in square brackets
- The values in the brackets are separated by commas, e.g. [1,2,3,4].
- If a value is repeated, the repetition can be represented by "Value (Number of times the value is repeated)". (The array shown in the figure below is composed of four word devices. 1(2) indicates that the initial value in the first word device is 1, and the initial value in the second word device is 1. 0(2) indicates that the initial value in the third word device is 0, and the initial value in the fourth word device is 0.)



Local Symbols

	Class...	Identifiers	Address	Type...	Initial	Comment...
I	VAR	WordArray		WORD[4]	[1(2),0(2)]	

(7) After the users click a **Comment...** cell, they can type a comment in the box which appears.



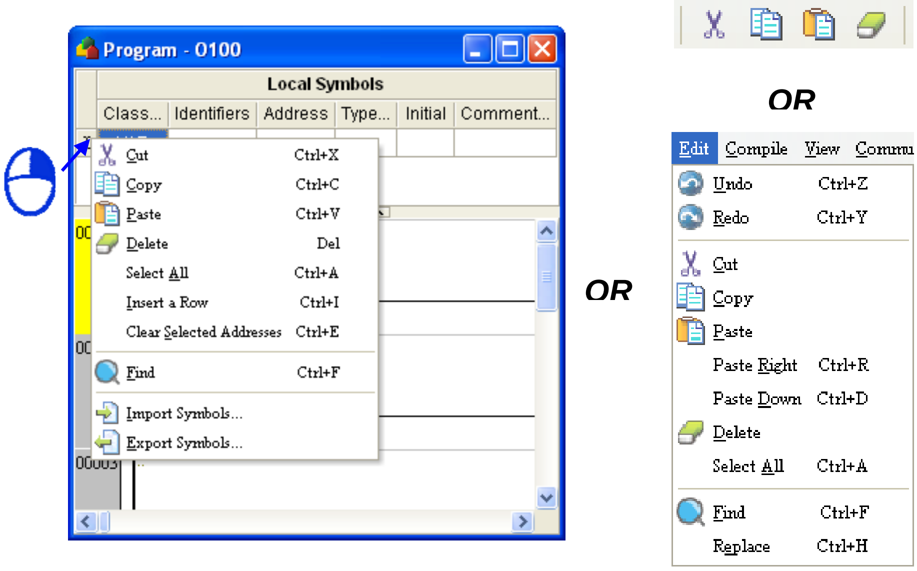
Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	Var_1		WORD	0	note1:

note1:

4.2.3 Modifying a Symbol and Editing a Symbol Table

After users click a cell for a symbol, they can modify the setting value in the cell according to the description in section 4.2.2.

Besides, the users can edit a symbol in a symbol table by means of the standard toolbar, the context menu which appears after the users right-click the symbol, or the **Edit** menu. Please refer to the following table for more information.



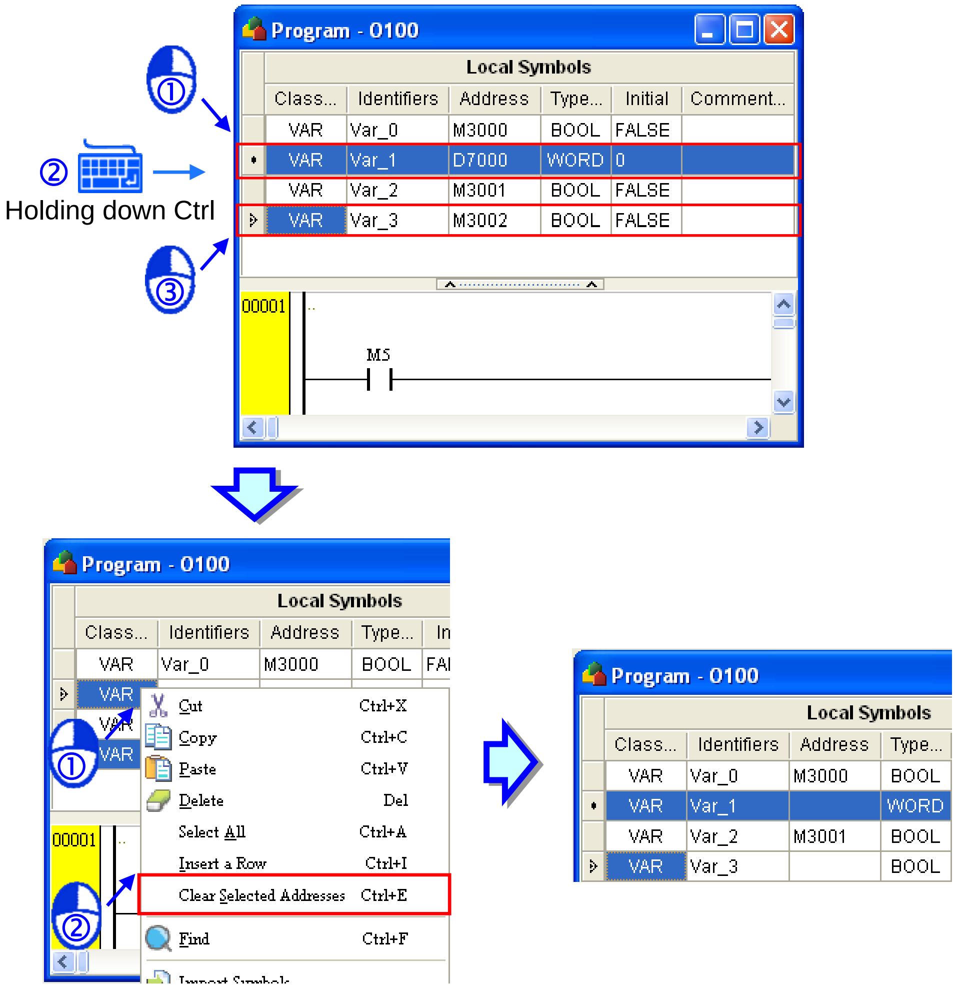
***. After users right-click a symbol, a context menu will appear.**

Item	Function
Cut	Cutting the symbol selected
Copy	Copying the symbol selected
Paste	Pasting an object which has been copied or cut on the present symbol table
Delete	Deleting the symbol selected
Select All	Selecting all the symbols in the symbol table
Insert a Row	Inserting a row above the row selected
Clear Selected Addresses	Clearing the device address selected
Find	Opening the **Find/Replace** window
Import Symbols…	Importing the symbol table which was exported

Item	Function
Export Symbols	Exporting the symbol table

4.2.4 Clearing Device Addresses and Selecting Multiple Symbols

The device addresses selected in a symbol table can be cleared, whether the device addresses selected are device addresses assigned by users or device addresses assigned by the system. After users right-click a device address in a symbol table, and click **Clear Selected Addresses** on the context menu, the device address will be cleared. If the users hold down Ctrl on the keyboard while they click device addresses, right-click a device address selected, and click **Clear Selected Addresses** on the context menu, the device addresses selected will be cleared. If the users click at the start of a selection, scroll to the end of the selection, hold down Shift on the keyboard while they click where they want the selection to end, right-click a device address selected, and click **Clear Selected Addresses** on the context menu, the device addresses selected will be cleared. The users can also clear device addresses by means of **Symbols Allocation** in the system information area. Please refer to section 4.2.8 fore more information.

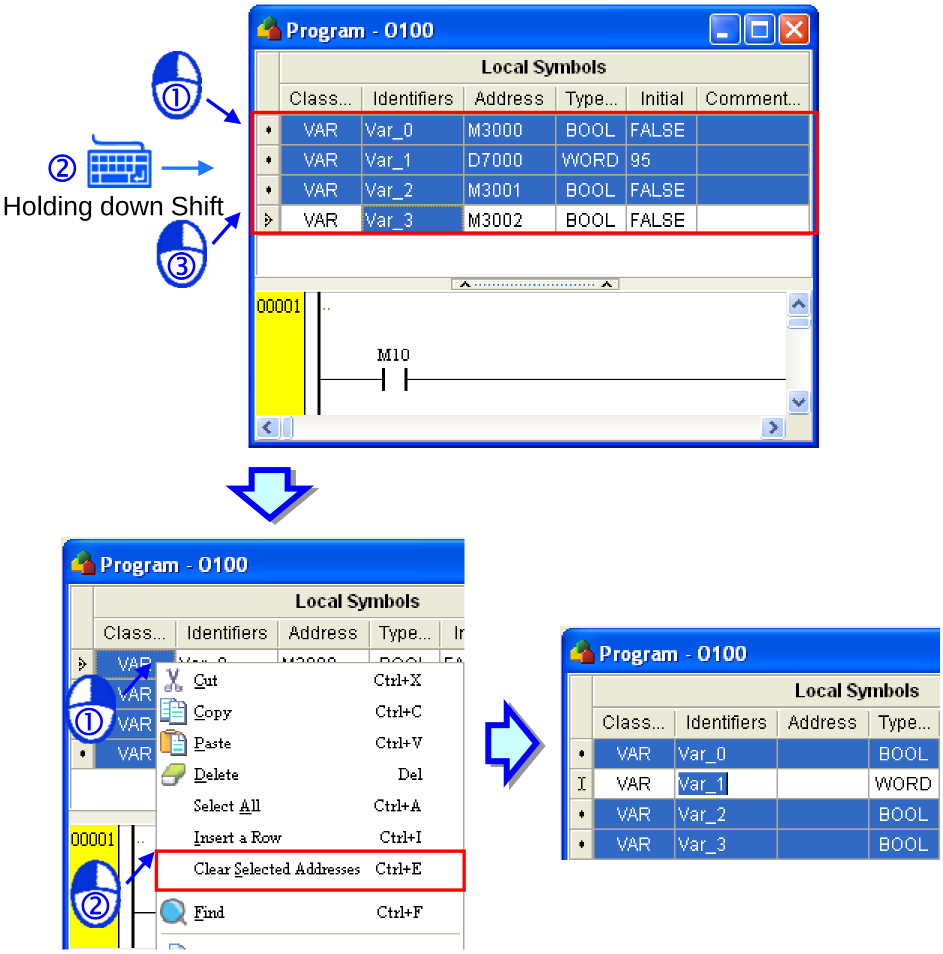


Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	Var_0	M3000	BOOL	FALSE	
VAR	Var_1	D7000	WORD	0	
VAR	Var_2	M3001	BOOL	FALSE	
VAR	Var_3	M3002	BOOL	FALSE	

Local Symbols

Class...	Identifiers	Address	Type...
VAR	Var_0	M3000	BOOL
VAR	Var_1		WORD
VAR	Var_2	M3001	BOOL
VAR	Var_3		BOOL



Program - 0100

Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	Var_0	M3000	BOOL	FALSE	
VAR	Var_1	D7000	WORD	95	
VAR	Var_2	M3001	BOOL	FALSE	
VAR	Var_3	M3002	BOOL	FALSE	

00001 M10

Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Delete	Del
Select All	Ctrl+A
Insert a Row	Ctrl+I
Clear Selected Addresses	Ctrl+E
Find	Ctrl+F

Program - 0100

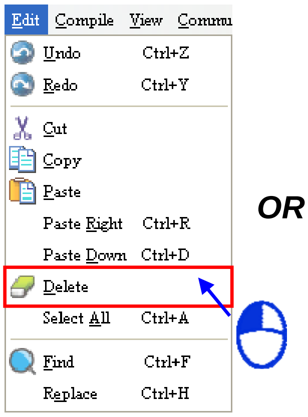
Local Symbols

Class...	Identifiers	Address	Type...
VAR	Var_0		BOOL
VAR	Var_1		WORD
VAR	Var_2		BOOL
VAR	Var_3		BOOL

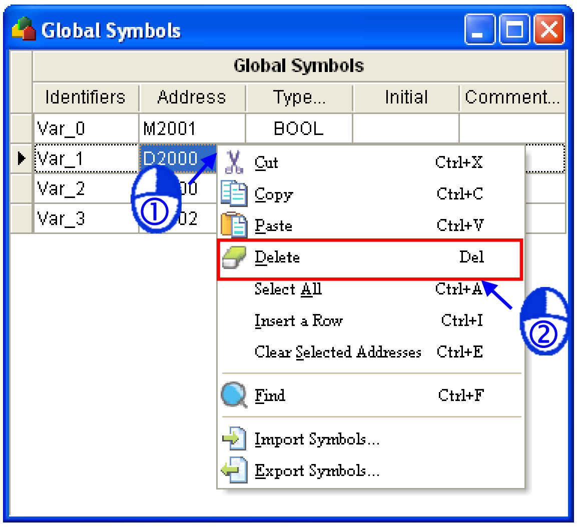
4.2.5 Deleting Symbols

If users want to delete a row in a symbol table, they have to click a cell in the row, and delete the row in a way described below. If the users want to delete several rows in a symbol table, they have to select the rows in a way described in section 4.2.4, and delete the rows in a way described below.

- Click **Delete** on the **Edit** menu.
- Click on the **standard** toolbar.
- Right-click a symbol **selected**, and then click **Delete** on the context menu.
- Press Delete on the **keyboard**.



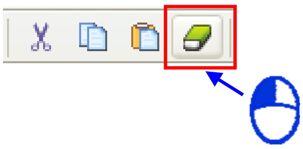
OR



Global Symbols

Identifiers	Address	Type...	Initial	Comment...
Var_0	M2001	BOOL		
Var_1	D2000			
Var_2	00			
Var_3	02			

OR

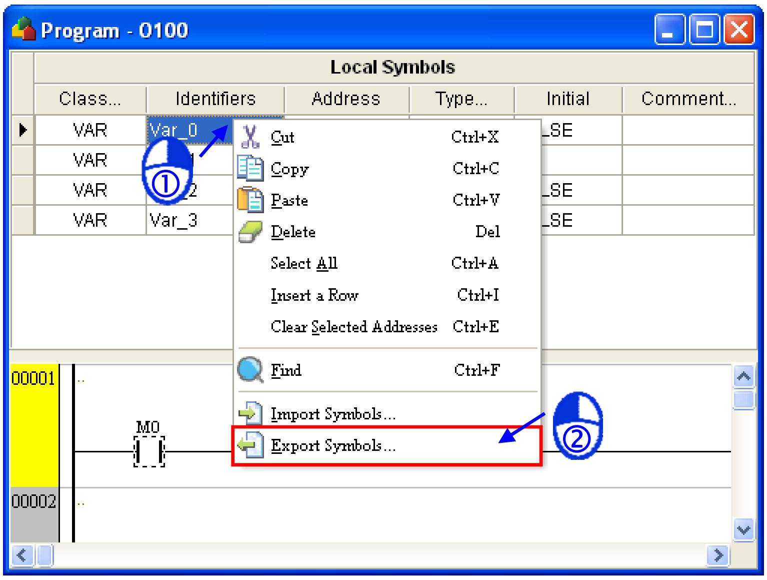


OR

Press Delete on the keyboard.

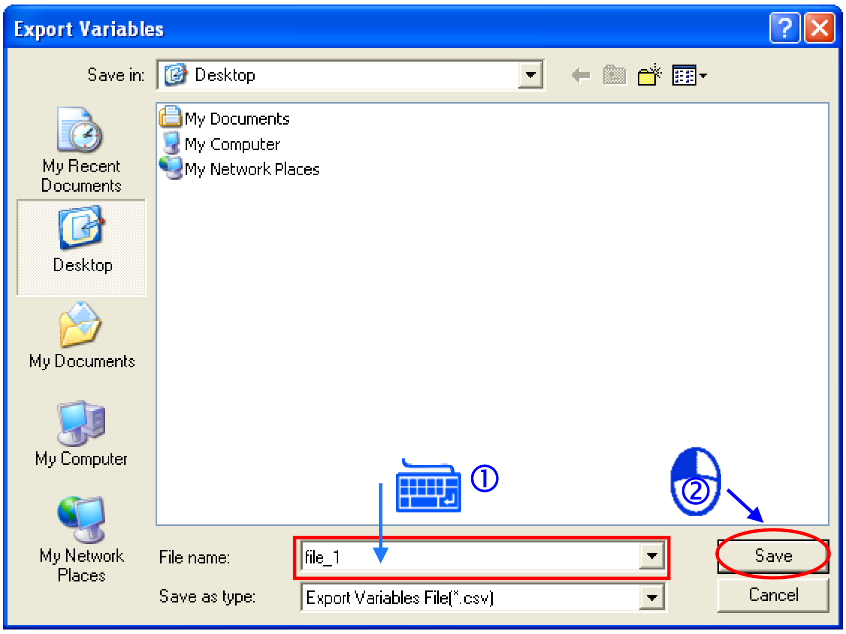
4.2.6 Exporting / Importing a Symbol Table

If users want to export a symbol table, they have to right-click the symbol table, click **Export Symbols…** on the context menu, type a file name in the **Export Variables** window, select a file path in the **Export Variables** window, and click **Save** in the **Export Variables** window.

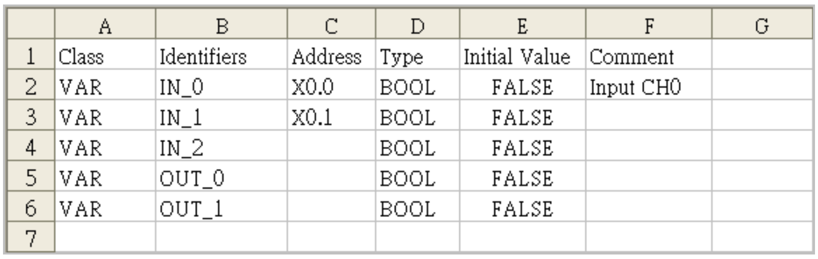


Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	Var_0			_SE	
VAR	1				
VAR	2			_SE	
VAR	Var_3			_SE	

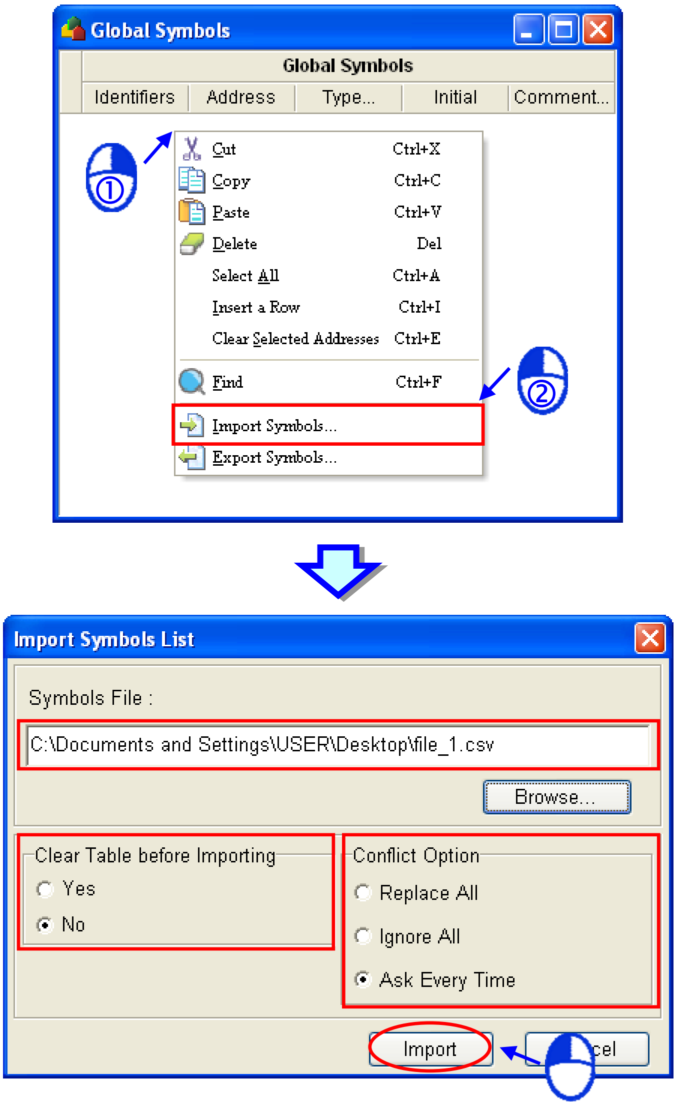


After the symbol table is exported, it will be saved as a .csv file. The .csv file can be edited through Microsoft Excel. Besides, the users can add symbols to the .csv file, and then import the .csv file. The users can easily create symbols by means of Microsoft Excel.



	A	B	C	D	E	F	G
1	Class	Identifiers	Address	Type	Initial Value	Comment	
2	VAR	IN_0	X0.0	BOOL	FALSE	Input CH0	
3	VAR	IN_1	X0.1	BOOL	FALSE		
4	VAR	IN_2		BOOL	FALSE		
5	VAR	OUT_0		BOOL	FALSE		
6	VAR	OUT_1		BOOL	FALSE		
7							

If the users want to import a symbol table, they have to right-click a symbol table, click **Import Symbols…** on the context menu, select a file path in the **Import Symbols List** window, select an option button in the **Clear Table before Importing** section, select an option button in the **Conflict Option** section, and click **Import**.



Symbols File : C:\Documents and Settings\USER\Desktop\file_1.csv

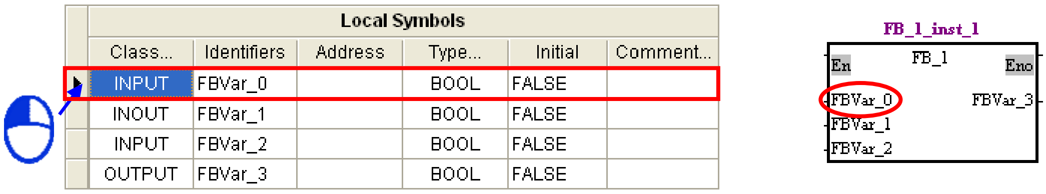
Clear Table before Importing: ○ Yes ● No

Conflict Option: ○ Replace All ○ Ignore All ● Ask Every Time

A global symbol table can be imported into another global symbol table or a local symbol table. A local symbol table can be imported into another local symbol table or a global symbol table. However, after the local symbol table in a POU which is a function block is exported, the symbols which are symbols of the INPUT class, the symbols which are symbols of the OUTPUT class, and the symbols which are symbols of INOUT class in the local symbol table can not be imported into a global symbol table or the local symbol in a POU which is a program. Besides, an error message occurs when the local symbol table is imported.

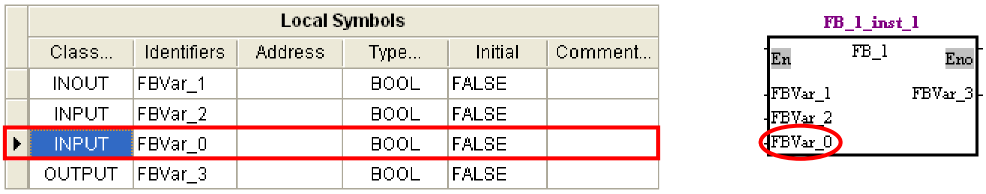
4.2.7 Arranging Symbols

In the local symbol table in a function block, the order in which the symbols of the INPUT class, the symbols of the INOUT class, and the symbols of the OUTPUT class are arranged affects the order in which the operands in the function block are arranged. If users want to move a symbol in the local symbol table in a function block upwards, they have to click the symbol, and press Alt and ↑ on the keyboard at the same time. If the users want to move a symbol in the local symbol in a function block downwards, they have to click the symbol, and press Alt and ↓ on the key board at the same time.



Local Symbols					
Class...	Identifiers	Address	Type...	Initial	Comment...
INPUT	FBVar_0		BOOL	FALSE	
INOUT	FBVar_1		BOOL	FALSE	
INPUT	FBVar_2		BOOL	FALSE	
OUTPUT	FBVar_3		BOOL	FALSE	

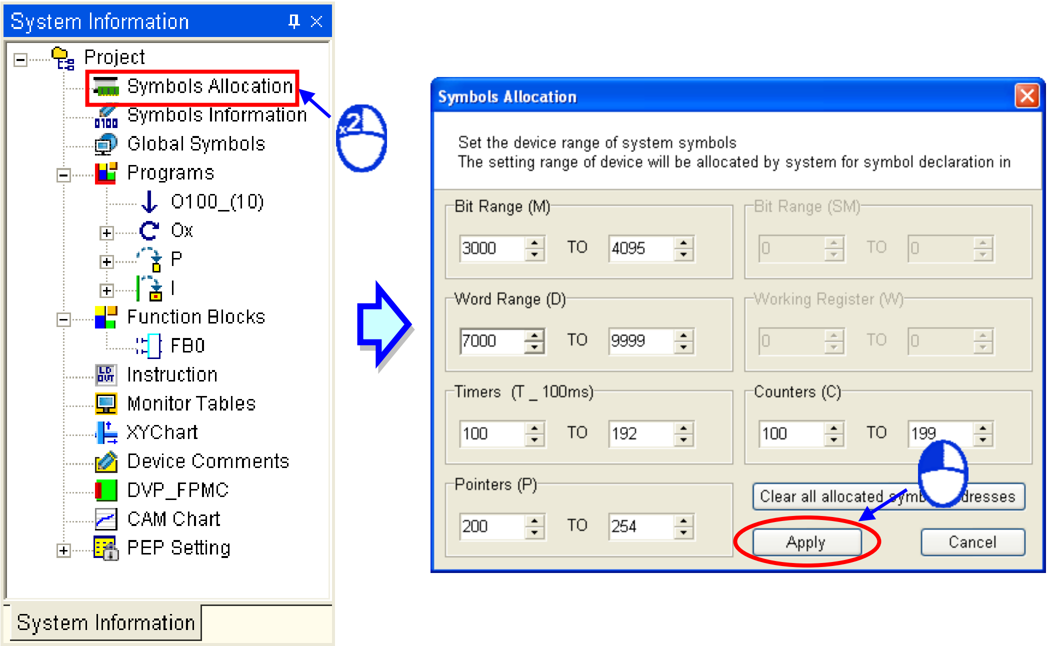
Press Alt+↓ twice.



Local Symbols					
Class...	Identifiers	Address	Type...	Initial	Comment...
INOUT	FBVar_1		BOOL	FALSE	
INPUT	FBVar_2		BOOL	FALSE	
INPUT	FBVar_0		BOOL	FALSE	
OUTPUT	FBVar_3		BOOL	FALSE	

4.2.8 Symbols Allocation

Users can set device ranges in a project. When the program in the project is compiled, the system assigns devices in the device ranges to the symbols in the project. If the number of symbols in the project is larger than the number of devices in the device ranges, an error message occurs when the program in the project is compiled. Please refer to section 4.1.4 for more information about assigning a device to a symbol. After users double-click **Symbols Allocation** in the system information area, the **Symbols Allocation** window will appear. After the users set device ranges in the **Symbols Allocation** window, they have to click **Apply**.



Symbols Allocation

Set the device range of system symbols
The setting range of device will be allocated by system for symbol declaration in

Bit Range (M): 3000 TO 4095

Bit Range (SM): 0 TO 0

Word Range (D): 7000 TO 9999

Working Register (W): 0 TO 0

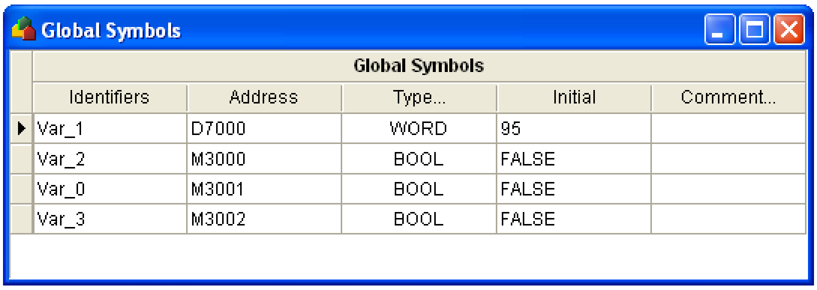
Timers (T _ 100ms): 100 TO 192

Counters (C): 100 TO 199

Pointers (P): 200 TO 254

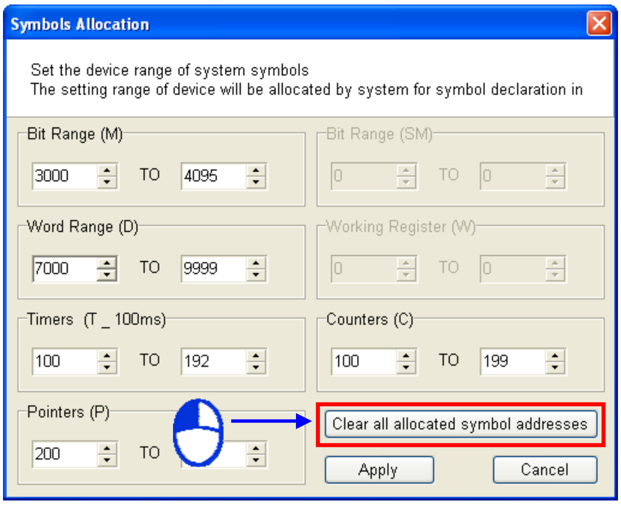
*. **The device types that an AH500 series motion controller has are different from the device types that a DVP series motion controller has. The device types which can not be used are shown in grayscale.**

If the users click **Clear all allocated symbol addresses** in the **Symbols Allocation** window, the device addresses assigned to the global symbols and the local symbols will be cleared.



Global Symbols

Identifiers	Address	Type...	Initial	Comment...
Var_1	D7000	WORD	95	
Var_2	M3000	BOOL	FALSE	
Var_0	M3001	BOOL	FALSE	
Var_3	M3002	BOOL	FALSE	



Symbols Allocation

Set the device range of system symbols
The setting range of device will be allocated by system for symbol declaration in

Bit Range (M): 3000 TO 4095

Bit Range (SM): 0 TO 0

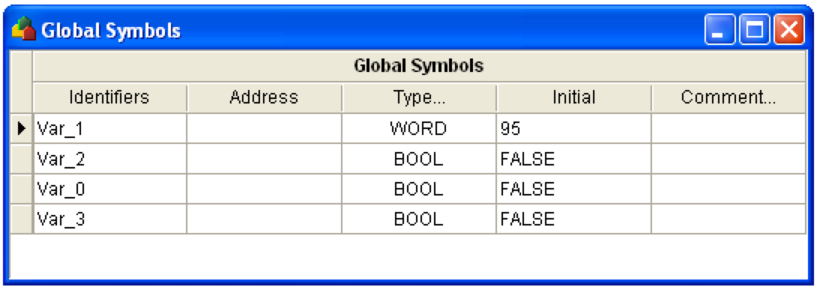
Word Range (D): 7000 TO 9999

Working Register (W): 0 TO 0

Timers (T _ 100ms): 100 TO 192

Counters (C): 100 TO 199

Pointers (P): 200 TO



Global Symbols

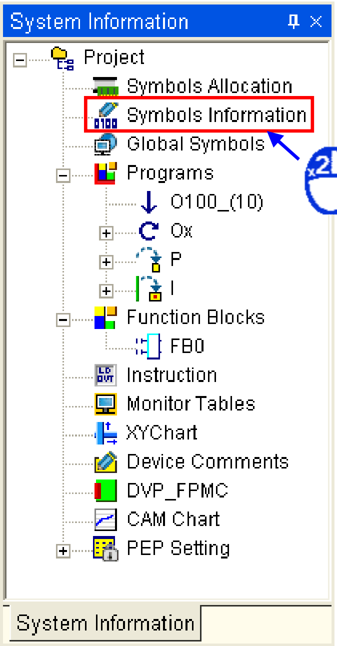
Identifiers	Address	Type...	Initial	Comment...
Var_1		WORD	95	
Var_2		BOOL	FALSE	
Var_0		BOOL	FALSE	
Var_3		BOOL	FALSE	

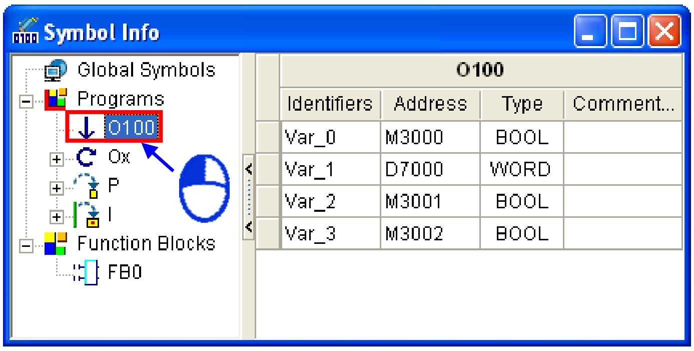
***1. There are restrictions on setting device ranges. For example, users can only set timers which take 100 milliseconds as a unit of measurement for time, the D registers which can be used start from D2000, the M devices which can be used start from M2000, the C devices which can be used are 16-bit counters. If a device address typed is not correct, the system will modify it automatically.**

***2. Only AH500 series motion modules are equipped with W devices and SM devices. Besides, D devices are assigned before W devices are assigned, and M devices are assigned before SM devices are assigned.**

4.2.9 Symbols Information

After users declare symbols, they can view the information about the symbols declared in the POUs created by means of **Symbol Information** in the system information area. After the users double-click **Symbol Information** in the system information area, the **Symbol Info** window will appear. After the users click an item at the left side of the window, they can view symbols created, the device addresses assigned to the symbols, the data types of the symbols, and the comments on the symbols.





O100

Identifiers	Address	Type	Comment...
Var_0	M3000	BOOL	
Var_1	D7000	WORD	
Var_2	M3001	BOOL	
Var_3	M3002	BOOL	

MEMO

Chapter 5 Function Block

Table of Contents

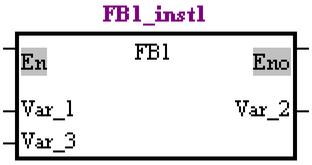
5.1 Knowing Function Blocks

Function blocks (abbreviated as FB) play an important role in the writing of the program in a PLC. Owing to the characteristics and advantages of function blocks, PMSoft provides much support and many functions for function blocks.

5.1.1 Introduction of Function Blocks

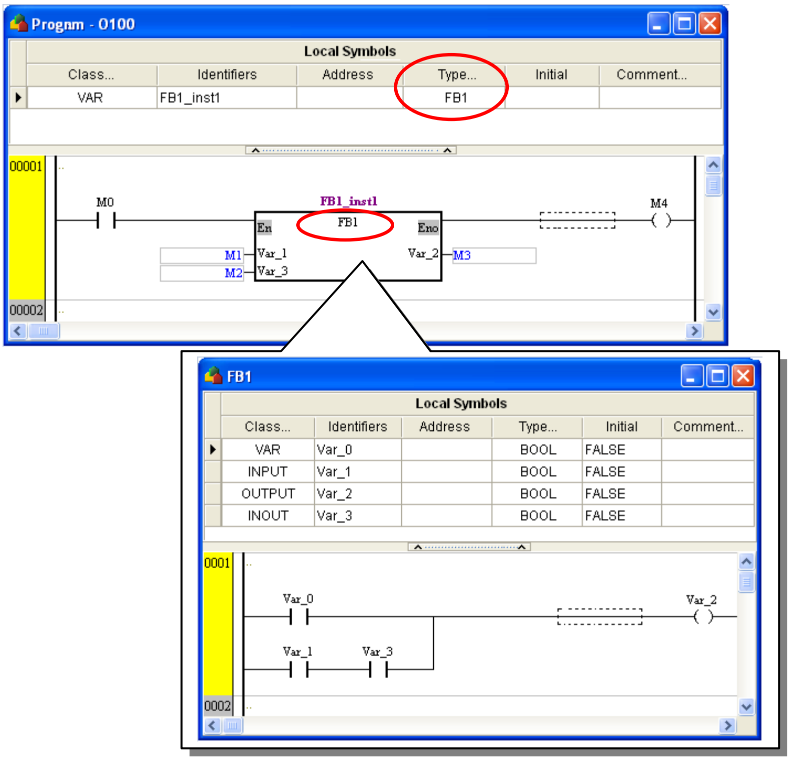
A function block is a component in a program which performs an operation. It is a type of POU, but it can not operate by itself. After a POU which is a program calls a function block, and sends the parameters related to the function block, the function of the function block will be executed. After the execution of the function block is complete, the internal operation result will be sent to the device or symbol specified by the superior POU (caller).

A function block is shown below. The appearance of a function block is similar to that of an applied instruction in that there are input pins and output pins. The usage of a function block is also similar to that of an applied instruction. However, before a POU calls a function block, users have to declare a symbol whose data type is a function block in the POU (caller). Besides, a function block can call another function block.



A function block is a memory unit. Every function block is assigned a substantial memory block where the values of the internal symbols are stored. Owing to the fact that the values of the symbols in a function block will be retained after the function block is executed, every execution result is affected by the last execution result. In other words, the output values may be different even if the input values are the same.

An application example of a function block is shown in the figure below. There is a symbol whose data type is a function block in the local symbol table in the superior POU (caller). The name in the **Type…** cell is the same as the name of the function block. Please refer to the following sections for more information about the usage of a function block.



Prognn - 0100

Local Symbols

	Class...	Identifiers	Address	Type...	Initial	Comment...
▸	VAR	FB1_inst1		FB1		

FB1

Local Symbols

	Class...	Identifiers	Address	Type...	Initial	Comment...
▸	VAR	Var_0		BOOL	FALSE	
	INPUT	Var_1		BOOL	FALSE	
	OUTPUT	Var_2		BOOL	FALSE	
	INOUT	Var_3		BOOL	FALSE	

5.1.2 Characteristics and Advantages of Function Blocks

Owing to the fact that function blocks present characteristics and advantages that traditional programming of a PLC does not have, function blocks are supported by IEC 61131-3.

- **Modular design**

  A large program is divided into several subroutines, and the subroutines are created as function blocks. The function blocks are arranged, and called by POUs which are programs.

- **Reusable**

  Once a function block is created, it can be used repeatedly as long as users conform to the rule of using the function block.

- **Highly portable**

  After a function block is created, it can be used in the original project. Besides, after the function block is exported, it can be imported into another project. As a result, users can gradually create their own function blocks.

- **Function blocks can be maintained conveniently.**

  The program in a function block is an independent module. If an error occurs in a function block, or the function of a function block does not meet the actual requirement, users can just modify the program in the function block, and do not need to debug or modify the whole program in the project created.

- **The readability of a program is increased.**

  Users can write a complex program or a program which will be used repeatedly in a function block, and the function block can be called by the original program. The structure of the original program becomes more simplified, and the program becomes more readable.

- **Highly confidential**

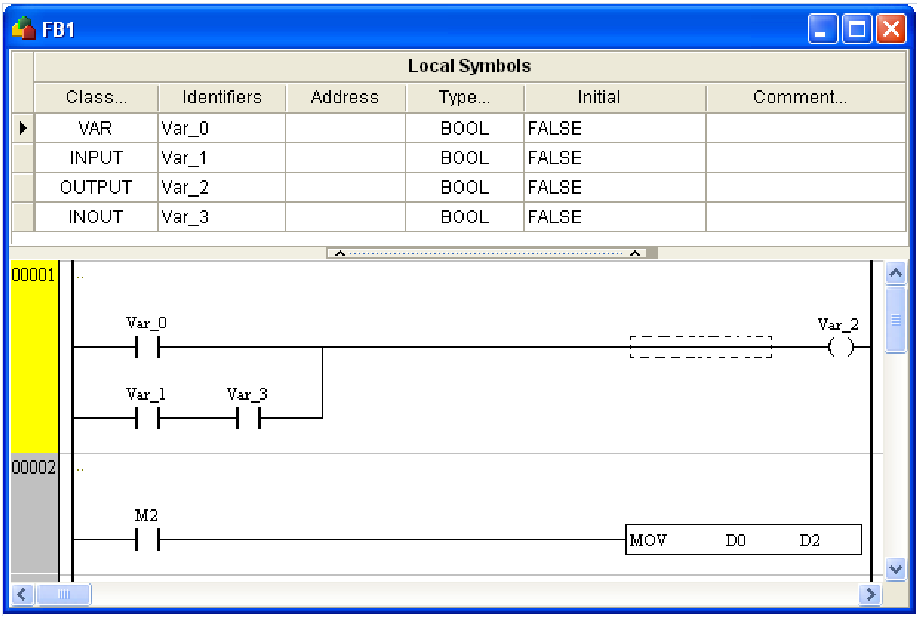
  After users set a password for a function block created, they can create their own core technology in the function block without care, and provide the function block for other users. Besides, system suppliers or software developers can provide function blocks which have specific functions for customers.

- **Highly efficient**

  The application of function blocks provides a highly efficient development environment in terms of project development. Owing to the fact that function block are modular, developers and manufacturers can take part in the development of a project together, and the use of human resources is more flexible.

5.2 Structure of the Function Blocks in PMSoft

Please refer to the following figure. The internal structure of a function block is similar to that of a POU which is a program. It is composed of a local symbol table and a program. A function block is edited in much the same way as a POU which is a program is edited.



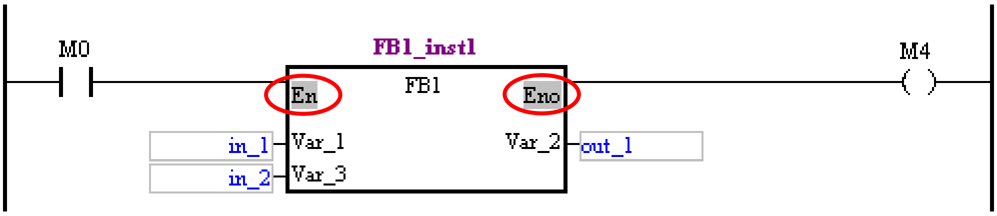
FB1

Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	Var_0		BOOL	FALSE	
INPUT	Var_1		BOOL	FALSE	
OUTPUT	Var_2		BOOL	FALSE	
INOUT	Var_3		BOOL	FALSE	

5.2.1 En Pin of a Function Block

A superior POU (caller) calls a function block in a way similar to the way in which an applied instruction is executed. Whether a function block is executed depends on the logic state sent to the En pin of the function block. If the logic state sent to the En pin of a function block is ON, the function block will be executed. If the logic state sent to the En pin of a function block is OFF, the function block will not be executed. The Eno pin of a function block sends the logic state sent to the En pin of the function block.



5.2.2 Symbols in a Function Block

Function blocks are similar to POUs which are programs in that users can create local symbols in the function blocks. Global symbols can also be used in a function block. However, if global symbols are used in a function block in a project, the function block may not be portable. If the function block is imported into another project, the project may not have the same global symbols.

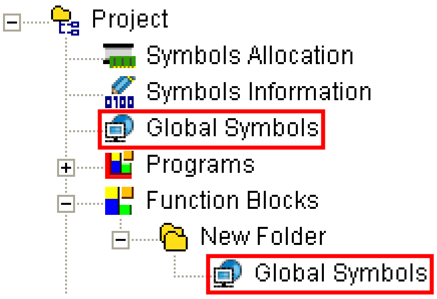
Users can add folders to **Function Blocks** in the system information area in PMSoft. A folder added contains a global symbol table. After a global symbol table, and the folder in which the global symbol table is exported, they can be imported into another project.

The symbols related to function blocks are described below. Please refer to chapter 4 for more information.

- Symbol class

Class	Description
VAR	The symbols of the VAR class are for the operations in function blocks only. After a function block is executed, the values of the symbols of the VAR class in the function block will be retained. Besides, users can assign device addresses to the symbols of the VAR class in a function block.
INPUT	A symbol of this class is used as an input pin of a function block. It receives the value of the operand specified by the superior POU (caller). When a function block is executed, the value of the operand specified by the caller is copied into the function block. Besides, the system automatically assigns device addresses to the symbols of the INPUT class in a function block. Users can not assign device addresses to the symbols of the INPUT class in a function block.
OUTPUT	A symbol of the OUTPUT class is used as an output pin of a function block. After the execution of a function block is complete, the operation result will be sent to the operand specified by the superior POU (caller) through the symbol of the OUTPUT class in the function block. Besides, the system automatically assigns device addresses to the symbols of the OUTPUT class in a function block. Users can not assign device addresses to the symbols of the OUTPUT class in a function block.
INOUT	A symbol of the INOUT class is used as a feedback pin of a function block. When a function block is called, the value of an operand specified by the superior POU (caller) is sent to the function block through the symbol of the INOUT type in the function block. After the function block is executed, the operation result is sent to the operand specified by the superior POU (caller). Besides, the system automatically assigns device addresses to the symbols of the INOUT class in a function block. Users can not assign device addresses to the symbols of the INOUT class in a function block.

- The data types supported by a function block are BOOL, WORD, DWORD, LWORD, FLOAT, COUNTER, TIMER, ARRAY, and function blocks.
- Every symbol whose data type is a function block is assigned a P device. Besides, the system automatically assigns device addresses to the local symbols in a function block instance according to the data types of the local symbols.
- The global symbol table contained in a folder added to **Function Blocks** in the system information area in a project is edited in the same way as the global symbol table in the project is edited. However, the global symbol table contained in a folder is only applicable to the function block definitions in the folder and the subfolders in the folder. As a result, the global symbol table contained in a folder can be regarded as a local variable table in the table.



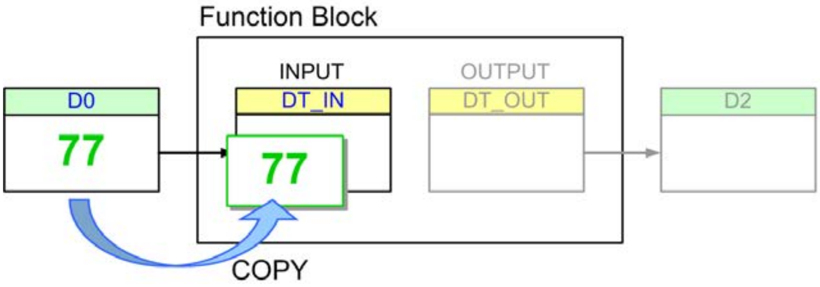
- Global symbol tables are contained in the folder added to **Function Blocks** in the system information area, and are not contained in the subfolders added to **Function Blocks** in the system information area.

5.2.3 Input/Output Pins of a Function Block

If the class of a symbol in a function block is INPUT or OUTPUT, the symbol will be the input interface or the output interface of the function block. When a POU calls a function block, it sends the value in a device to the input pin of the function block. After the function block is executed, the operation result will be sent to a device in the POU through the output pin of the function block. Please refer to the following figures.

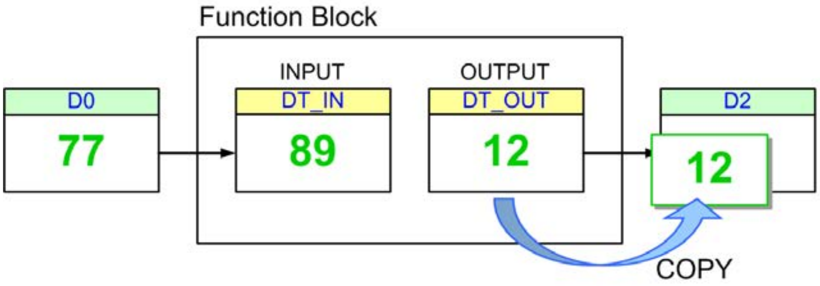
- **Before a function block is executed**

  The caller assigns D0 to DT_IN, the input pin of the function block. When the function block is called, the system sends the present value in D0 to DT_IN. (DT_IN is a symbol of the INPUT type.)

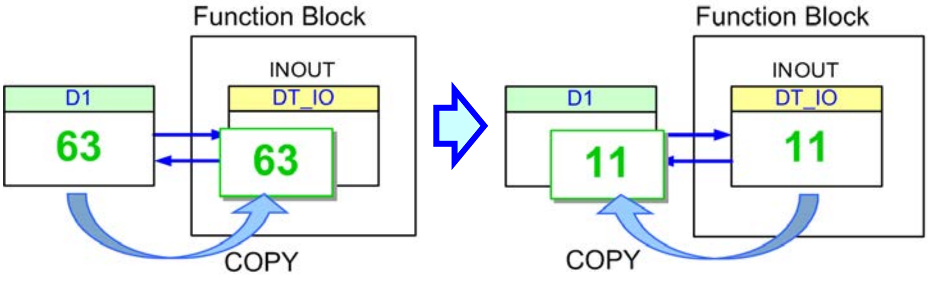


- **After a function block is executed**

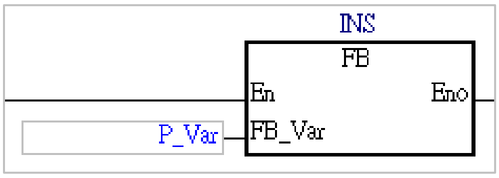
  After the function block is executed, the value of DT_IN will become 89. The value in D0 is unchanged. Besides, DT_OUT sends the final operation result to D2, a device specified by the caller. Even if the value in D2 is overwritten during the execution of the program, the value of DT_OUT will remain unchanged. (DT_ OUT is a symbol of the OUTPUT class.)

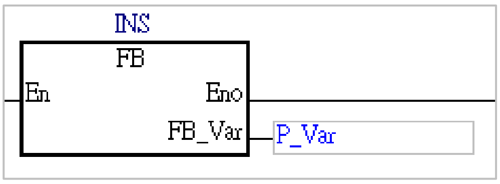


Although the value of the symbol of the INOUT class will be sent to the device that the caller assigns to the symbol after the function block is executed, the present value in the device specified by the caller is sent to the symbol before the symbol sends the final operation result to the device.



Please refer to the function blocks below. FB_Var is the pin of the function block, and P_Var is the operand that the superior POU (caller) assigns to FB_Var. The data type of P_Var and that of FB_Var must conform to the basic principles described below.





- **Basic principles for the general data types**

(1) The relation between the device that the caller assigns to FB_Var and the data type of FB_Var is shown below. Please pay attention to * in the table. If FB_Var is the input pin of the function block, and the data type of FB_Var is BOOL, the device assigned to FB_Var can be a counter or a timer. If FB_Var is the output pin of the function block, and the data type of FB_Var is BOOL, the device assigned to FB_Var can only be a M device, a SM device, or a S device.

Data type		FB_Var (function block)						
		WORD	DWORD	LWORD	FLOAT	COUNTER	TIMER	BOOL
Device (caller)	D device	✓	✓	✓	✓			
	W device	✓	✓	✓	✓			
	SR device	✓	✓	✓	✓			
	V device	✓						
	Z device		✓					
	M device							✓
	SM device							✓
	C device	✓	✓	✓	✓	✓		*
	T device	✓	✓	✓	✓		✓	*
	S device							✓

(2) The relation between the symbol that the caller assigns to FB_Var and the data type of FB_Var is shown below. Please pay attention to * in the table. If FB_Var is the input pin of the function block, and the data type of FB_Var is BOOL, the data type of the symbol assigned to FB_Var can be COUNTER or TIMER. If FB_Var is the output pin of the function block, and the data type of FB_Var is BOOL, the data type of the symbol assigned to FB_Var can only be BOOL.

Data type		FB_Var (function block)						
		WORD	DWORD	LWORD	FLOAT	COUNTER	TIMER	BOOL
P_Var (caller)	WORD	✓						
	DWORD	✓	✓		✓			
	LWORD	✓	✓	✓	✓			
	FLOAT	✓	✓		✓			
	COUNTER	✓				✓		*
	TIMER	✓					✓	*
	BOOL							✓

In addition to the principles for the general data types described above, there are principles for the symbols whose data types are ARRAY.

- **Basic principles for the symbols whose data types are ARRAY**

  (1) If the array type of P_Var is the same as the array type of FB_Var, the size of the array for P_Var must be larger than or equal to the size of the array for FB_VAR.

  (2) If the array type of P_Var is different from the array type of FB_Var, the second basic principle for the general data types must be satisfied. If a single element can call another single element, an array composed of a certain number of elements can call another array composed of the same number of elements. For example, a caller (P_Var) which is an array composed of 11 counters can call another array (FB_Var) composed of 11 word devices.

  (3) If the array type of one symbol is WORD, DWORD, or LWORD, the data type of the other symbol can be WORD, DWORD, LWORD, or ARRAY. Besides, the data sizes of the two symbols have to be the same. For example, if one symbol is an array composed of two word devices, the data type of the other symbol can be DWORD.

- **Description**

  ✓ Example 1: The data type of P_Var (in the caller) is WORD, and the data type of FB_Var (in the function block) is WORD.

  → The second basic principle for the general data types is satisfied. As a result, P_Var and FB_Var are legal symbols.

  ✗ Example 2: The data type of P_Var (in the caller) is WORD, and the data type of FB_Var (in the function block) is DWORD.

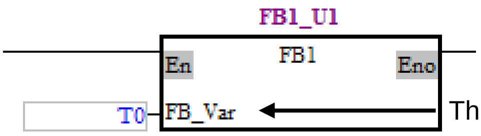
  → The second basic principle for the general data types is not satisfied. As a result, P_Var and FB_Var are illegal symbols.

  ✗ Example 3: The data type of P_Var (in the caller) is TIMER, and the data type of FB_Var, which is the output pin of the function block, is BOOL.

  → The second basic principle for the general data types is not satisfied. As a result, P_Var and FB_Var are illegal symbols.

  ✓ Example 4: The device that the caller assigns to FB_Var is T0, and the data type of FB_Var (in the function block) is BOOL.

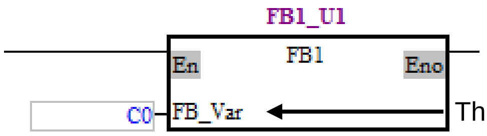
  → The first basic principle for the general data types is satisfied. As a result, P_Var and FB_Var are legal symbols.



The data type of FB_Var is BOOL.

  ✓ Example 5: The device that the caller assigns to FB_Var is C0, and the data type of FB_Var (in the function block) is LWORD.

  → The first basic principle for the general data types is satisfied. As a result, P_Var and FB_Var are legal symbols.



The data type of FB_Var is LWORD.

  ✓ Example 6: The device that the caller assigns to FB_Var is D0, and the data type of FB_Var (in the function block) is LWORD.

➔ The first basic principle for the general data types is satisfied. As a result, P_Var and FB_Var are legal symbols.

✓ Example 7: The device that the caller assigns to FB_Var is S0, and the data type of FB_Var (in the function block) is BOOL.

➔ The first basic principle for the general data types is satisfied. As a result, P_Var and FB_Var are legal symbols.

✓ Example 8: P_VAR is an array composed of four word devices. The data type of FB_Var is LWORD (not ARRAY).

➔ The third basic principle for the symbols whose data types are ARRAY is satisfied. As a result, P_Var and FB_Var are legal symbols.

✓ Example 9: P_VAR is an array composed of four word devices, and FB_Var is an array composed of two double word devices.

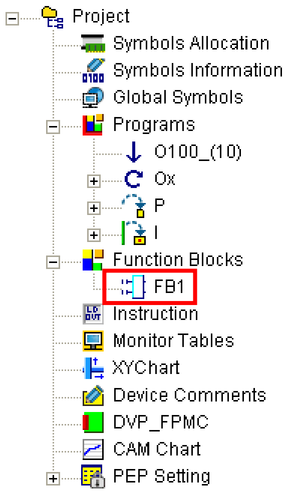
➔ The third basic principle for the symbols whose data types are ARRAY is satisfied. As a result, P_Var and FB_Var are legal symbols.

✕ Example 10: P_VAR is an array composed of thirty-two Boolean devices, and FB_Var is an array composed of two word devices.

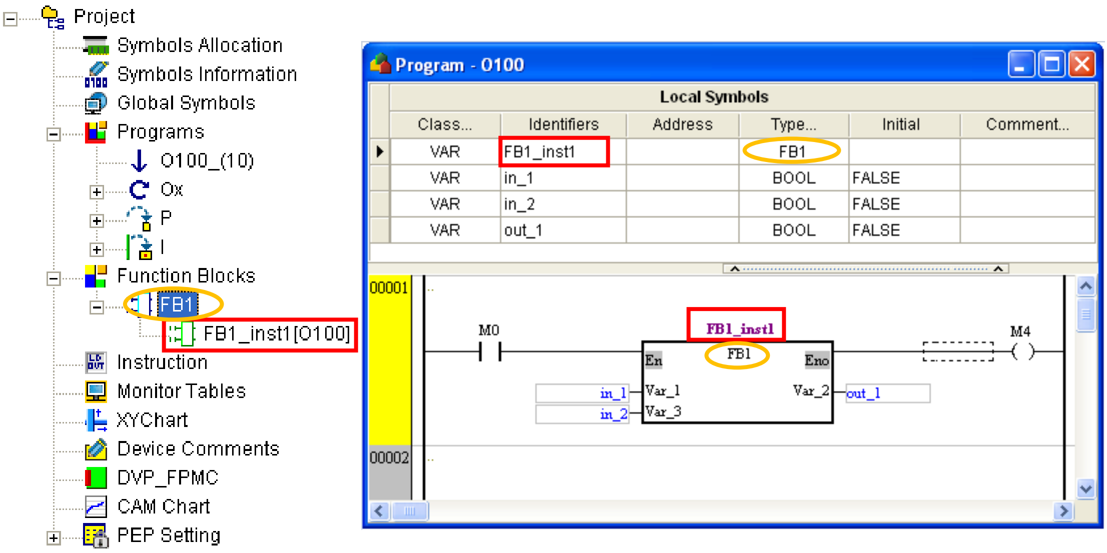
➔ The second basic principle for the symbols whose data types are ARRAY is not satisfied. As a result, P_Var and FB_Var are illegal symbols.

5.2.4 Function Block Definition and Function Block Instance

After users add a POU which is a function block to the system information area in PMSoft, declare local symbols in the POU, and write a program in the POU, they will get a function block definition. If a function block created is not called by any POU which is a program, it will not participate in any operation, and the system will not assign any memory block to the function block and the local symbols in the function block. In other word, if a function block created in a project is not called by any POU which is a program, it is just data saved with the project, and does not appropriate any resource that the motion controller uses.



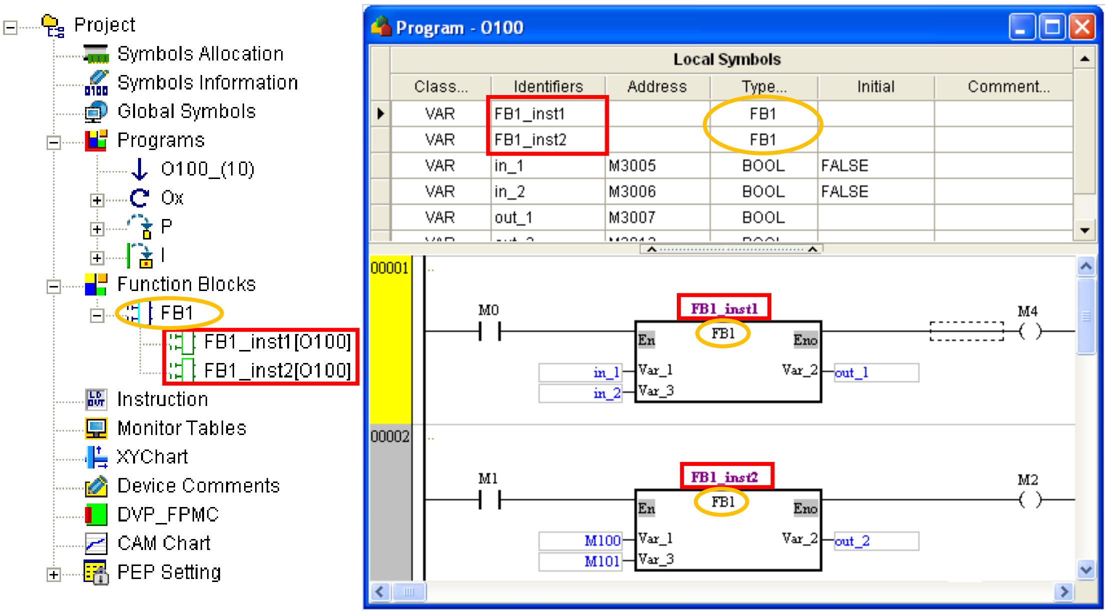
Once a function block is called by a POU which is a program, the function block will participate in an operation, and the system will assign a substantial memory block to the function block and the local symbols in the function block. The object produced is a function block instance. If the data type of a symbol declared in a superior POU (caller) is a function block, the identifier of the symbol is a function block instance. In other words, declaring a symbol whose data type is a function block is equivalent to declaring a function block instance.



Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	FB1_inst1		FB1		
VAR	in_1		BOOL	FALSE	
VAR	in_2		BOOL	FALSE	
VAR	out_1		BOOL	FALSE	

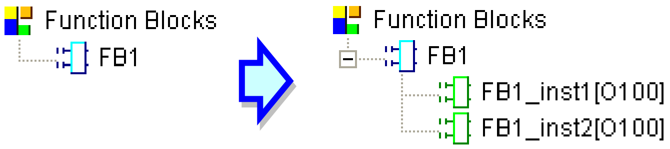
If the same operation is required, and a different memory block is also required, users can make use of the characteristic of a function block instance. After a different symbol is declared for a function block which requires a different memory block, the system will regard the different symbol as a different function block instance. When the program is compiled, a different memory block is assigned to the different function block instance. In the main program O100 in the figure below, the function block definition FB1 has the two function block instances FB1_inst1 and FB1_inst2.



Local Symbols

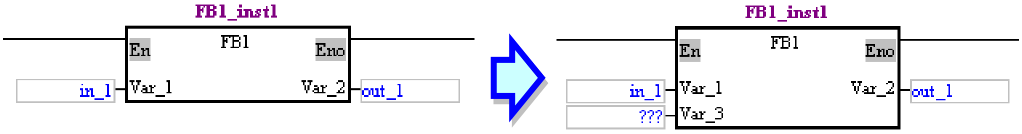
Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	FB1_inst1		FB1		
VAR	FB1_inst2		FB1		
VAR	in_1	M3005	BOOL	FALSE	
VAR	in_2	M3006	BOOL	FALSE	
VAR	out_1	M3007	BOOL		

Only the function block definitions in a project are in the **Function Blocks** section before the program in the project is compiled. After the program in the project is compiled, the function block instances which actually participate in operations will be under the function block definitions. The format of a function block instance is **Instance name[POU name]**.

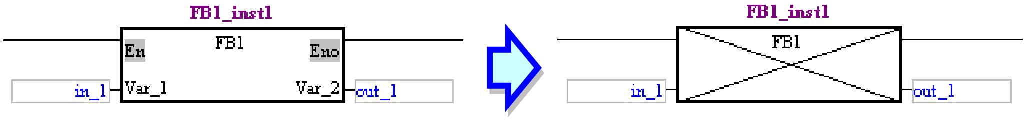


After a symbol whose data type is a function block is declared in a symbol table in a project, a function block instance will be produced even if it is not used in the program in the project.

If a function block is called by a POU, the modification of the definition of the function block will change the appearance of the function block in the superior POU. After users modify the definition of a function block, they have to check whether the input pins and the output pins of the function block are correct.

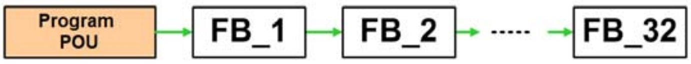


If a POU which is a function block is deleted, the function block representing the POU in the superior POU will be marked with a cross, and users have to delete the function block.

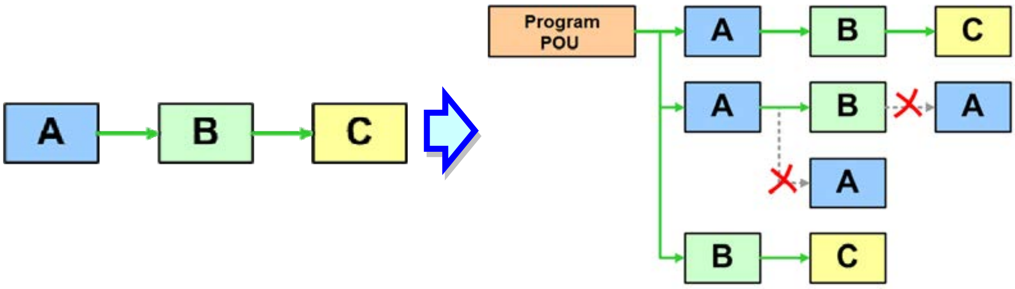


5.2.5 Calling Relation Between Function Blocks

In PMSoft, a function block can call another function block, and 32 layers of function blocks at most can be called. The function block instance called by a POU which is a program is the first layer.

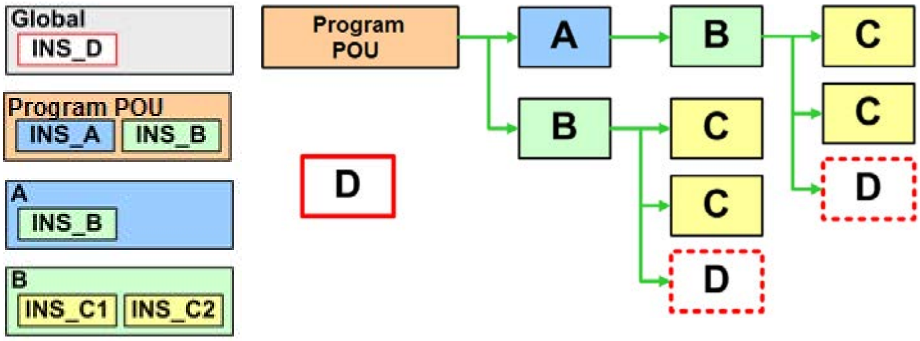


The program in function block A calls function block B, and the program in function block B calls function block C. In other words, function block A is superior to function block B and function block C, and function block B is superior to function block C. In PMSoft, a function block is not allowed to call itself. As a result, if function block A is superior to function block B, function block B and function block C can not call function block A, but a POU which is a program can call function block A, function block B, or function block C.



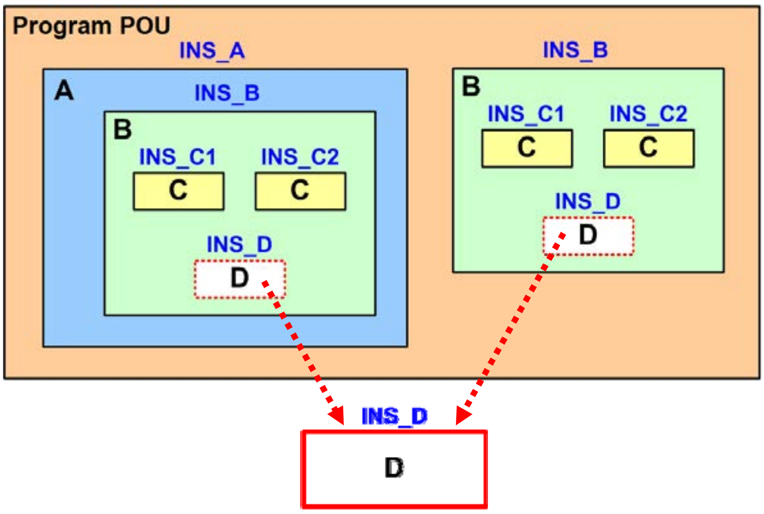
If function block A, function block B, and function C are not called by any POUs which are programs in a project, they will not participate in the execution of the program in the project, and function block instances will not produced after the program is compiled. After function block A is declared in a POU which is a program or in a global symbol table in a project, function block instances will be produced according to the relation among the three function blocks during the compiling of the program.

Please refer to the declaration of the function block instances and the relation among the function blocks below. Function block D is declared in a global symbol table while the other function block instances are declared in the local symbol tables in the superior POUs (callers).

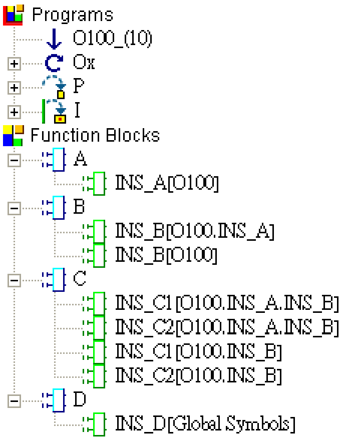


Declaration position		Function block which is declared
Global symbol table		One instance of function block D: INS_D
Local symbol table	POU which is a program	One instance of function block A: INS_A One instance of function block B: INS_B
	Function block A	One instance of function block B: INS_B
	Function block B	Two instances of function block C: INS_C1 and INS_C2

The function block instances produced when the program is compiled according to the relation described above are shown below. Owing to the fact that two instances of function block C are declared in the local symbol table in function block B, the two instances of function block C are produced when another instance of function block B is produced. Besides, owing to the fact that function block D is declared in a global symbol table, the function of an instance of function block D and the data stored in the instance can be used by all the POUs. In other words, the system only produces one instance of function block D. Although function block D is called by two POUs, the same instance of function block D is executed in the two POUs.



The items which will appear in the system information area after the program is compiled are shown in the figure below. The superior POU which is a program and the inferior POUs are in the brackets at the end of a function block instance.

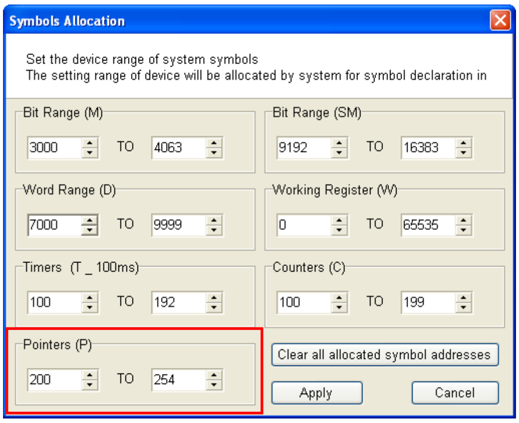


Users are allowed to declare a function block instance in the symbol table in a superior function block. If an inferior function block instance called by a superior function block does not need to be called by another superior POU, the inferior function block instance can be directly declared in the local symbol table in the superior function block which calls the inferior function block instance.

5.2.6 Assigning a Memory Block to a Function Block

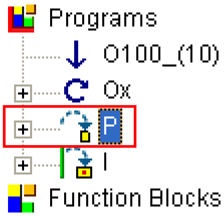
During the process of producing instances of the function blocks in a project, the system assigns memory blocks to the instances, and device addresses to the symbols in the function blocks. Users can assign device address to the symbols of the VAR class in the function blocks. If users assign device addresses to the local symbols in a function block in a project, or global symbols to which the users assign device addresses are used in a function block, the symbols will be regarded as absolute addresses, and the system will not assign device addresses to the symbols after the program in the project is compiled. Even in a different instance of a function block, the device addresses that the users assign to the local symbols participate in an operation. However, it is the system that assigns device addresses to the symbols of the INPUT class, the symbols of the OUTPUT class, and the symbols of the INOUT class. Besides, if a function block instance is declared in a global symbol table in a project, it can be called by all the POUs in the project, and the system will assign an independent memory block to it.

After the system assigns memory blocks to the function block instance in a project, every function block instance will occupy a P device which is in the range set in the **Pointers (P)** section in the **Symbols Allocation** window. The P pointer numbers used in POUs which are programs in a project, and the P subroutine number in the project can not be in the range set in the **Pointers (P)** section in the **Symbols Allocation** window.


Symbols Allocation

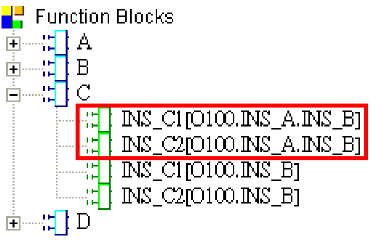
Set the device range of system symbols
The setting range of device will be allocated by system for symbol declaration in

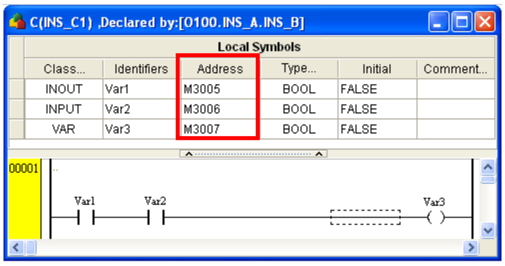
Bit Range (M): 3000 TO 4063
Bit Range (SM): 9192 TO 16383
Word Range (D): 7000 TO 9999
Working Register (W): 0 TO 65535
Timers (T _ 100ms): 100 TO 192
Counters (C): 100 TO 199
Pointers (P): 200 TO 254



The range set in the Pointers (P) section is P200~P254. The P subroutine numbers and the P pointer numbers used in POUs can not be in the range.

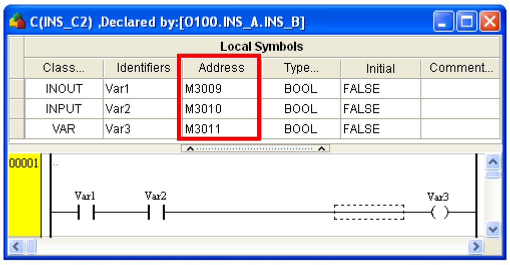
The system assigns devices to the local symbols declared in a function block according to the data types of the local symbols. For example, the system assigns a D device to a local symbol whose data type is WORD, and a M device to a local symbol whose data type is BOOL. As a result, the device that an instance of a function block occupies depends on the definition of the function block. Please refer to the figure below. After the windows for the two function block instances are opened, users can see the devices that the system assigns to the two function block instances.




C(INS_C1) ,Declared by:[O100.INS_A.INS_B]

Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
INOUT	Var1	M3005	BOOL	FALSE	
INPUT	Var2	M3006	BOOL	FALSE	
VAR	Var3	M3007	BOOL	FALSE	


C(INS_C2) ,Declared by:[O100.INS_A.INS_B]

Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
INOUT	Var1	M3009	BOOL	FALSE	
INPUT	Var2	M3010	BOOL	FALSE	
VAR	Var3	M3011	BOOL	FALSE	

5.3 Using a Function Block

5.3.1 Basic Specifications for Function Blocks

The specifications for the function blocks in PMSoft are shown below. When users use function blocks in a project, they have to make sure that the use conforms to the specifications in the table below. Otherwise, an error occurs when the program in the project is compiled or executed.

- **Specifications for function blocks**

Number of function block instances	The number of function block instances depends on the number of P devices available. (One function block instance occupies one P device.)
Size of a function block	The size of a function block depends on the device assigned to the function block.
Number of layers	A function block can call another function block, and 32 layers of function blocks at most can be called.

- **Note**
  - ➢ The use of a function block must conform to the specifications in the table above.
  - ➢ The pins of a function block must be assigned operands. The operands can be device addresses, symbols, or constants.
  - ➢ An inferior function block can not call a superior function block, and a function block can not call itself.
  - ➢ If a function block is exported, the definition of the function block is exported, but the definitions of the inferior function blocks and the global symbols used are not exported.
  - ➢ Users can modify function block definitions, but they can not modify function block instances.
  - ➢ The En pin of a function block must be connected to a contact or a block.
  - ➢ The usage of a function block is the same as that of a P subroutine. A function block can be used by the main program O100, a Ox motion subroutine, or a P subroutine. It can call a Ox motion subroutine, or a P subroutine, but can not call the main program O100. Please refer to DVP-PM Application Manual for more information about the setting of special registers.
  - ➢ A function block used by a Ox motion subroutine supports basic instructions, applied instructions, G-codes, and motion instructions. A function block used by the main program O100 or a P subroutine only supports basic instructions and applied instructions.

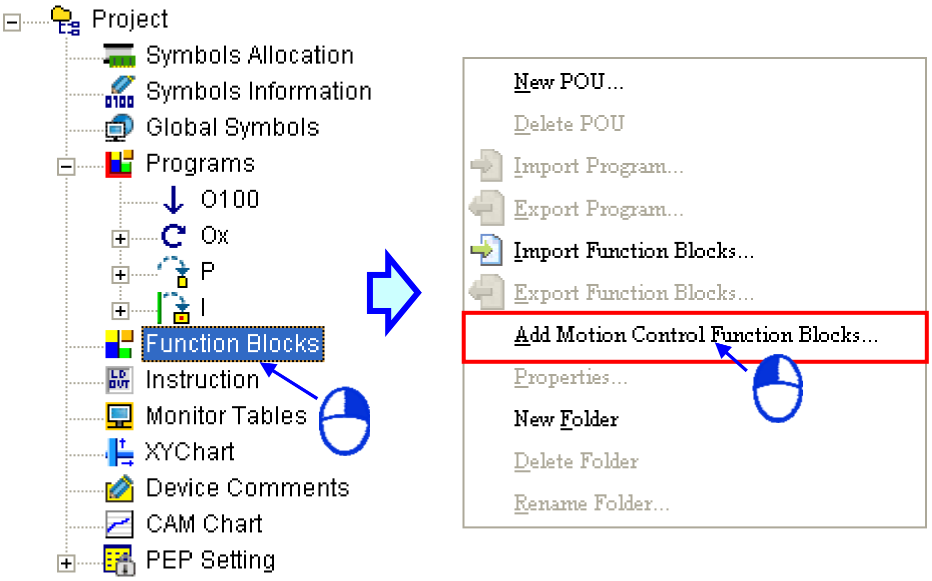
- ➢ If a function block used by a Ox motion subroutine uses a basic instruction or an applied instruction, the instruction used can not be a pulse instruction.
- ➢ If a jump instruction is used in a function block, the jump destination must be a P pointer in the same function block, and can not be outside the function block or in an inferior function block.

***1. Please refer to section 3.2 for more information about creating a POU which is a function block, and refer to section 4.2.2 for more information about declaring a function block instance.**

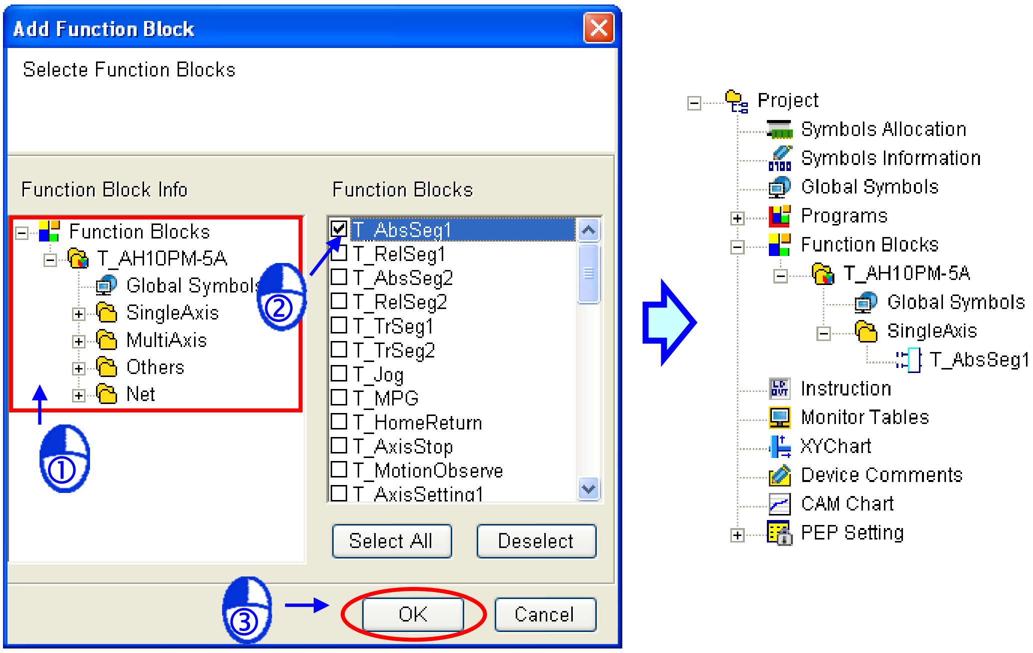
5.3.2 Motion Control Function Blocks

PMSoft provides a series of motion control function blocks for users, including single-axis function blocks, multi-axis function blocks, and Delta new generation high-speed network interfaces (DMCNET function blocks). The usage of the motion control function blocks are described below.

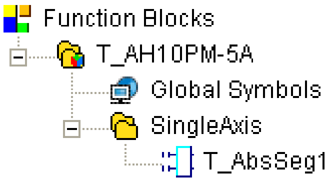
(1) Users have to right-click **Function Blocks** in the system information area, and then click **Add Motion Control Function Blocks...** on the context menu which appears.



(2) In the **Add Function Block window**, the users have to select a function block type in the **Function Block Info** section, and select function block definitions in the **Function Blocks** section. If the users click **Select All**, all the function block definitions in the **Function Blocks** section will be selected. If the users click **Deselect**, all the function block definitions in the **Function Blocks** section will be unselected. Finally, the users have to click **OK**.



(3) The function block definitions selected in the **Add Function Block** window are added to the **Function Blocks** section in the system information area. In the figure below, the **T_AH10PM-5A** folder represents the function block type applicable to AH10PM-5A. The function block definitions which can be added vary with the motion controller used. **Global Symbols** represents the global symbol table applicable to the function block definitions in the **T_AH10PM-5A** folder. The **SingleAxis** subfolder represents the function block type which is added. The **T_AH10PM-5A** folder, the **SingleAxis** subfolder, and **Global Symbols** are created when the function block definition **T_AbsSeg1** is added. The function block definitions and the folders which are created by the users can not be the same as the function block definition and the folders mentioned above. Otherwise, a message will appear, and the users will be asked to modify the function blocks and the folders they create.



(4) The use of the function block definitions which are added is the same as that of general function block definitions. Please refer to section 4.2.2 for more information. The function block definitions which are added can not be modified, and can only be used to produce function block instances.

Chapter 6 Ladder Diagram

Table of Contents

6.1 Introduction of a Ladder Diagram

6.1.1 Ladder Diagram

A ladder diagram is one of the programming languages defined by IEC 61131-3, and is widely used to create a PLC program. A ladder diagram is a programming language that represents a program by a graphical diagram based on the circuit diagrams of relay logic hardware.

The creation of a ladder diagram in PMSoft will be introduced in the following sections. The principle of a ladder diagram will not be described.

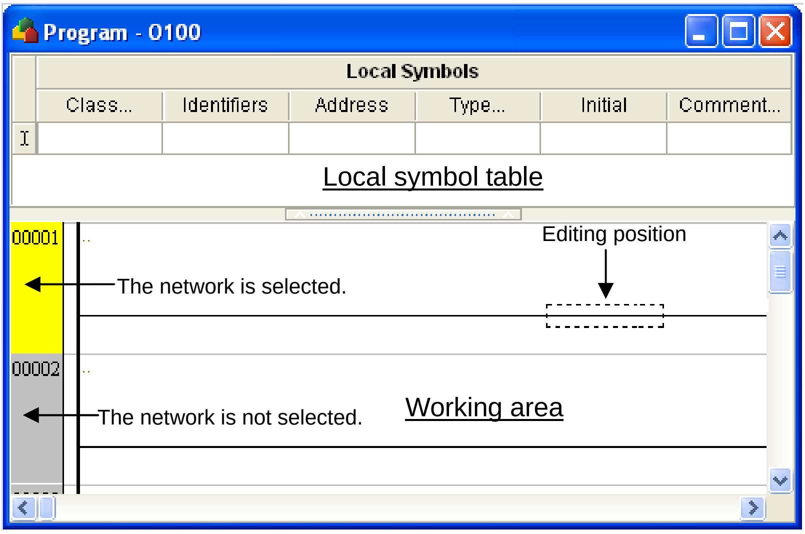
6.1.2 Important Points about Creating a Ladder Diagram

- A ladder diagram is case-insensitive. That is, the words OUT, Out, and out have the same meaning.
- If users want to use constants in a program created by means of a ladder diagram in PMSoft, the constants must be represented in the following ways.
  - Decimal value: 2345
  - Hexadecimal value: 16#5BA0
  - Floating-point number: 4.123

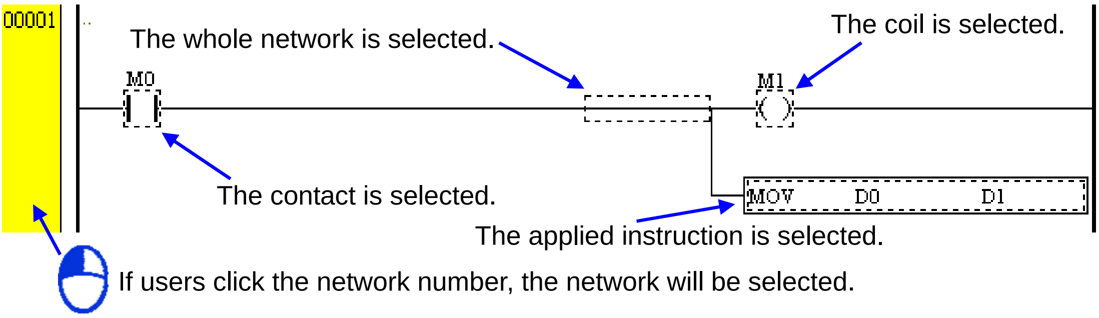
  ***. A ladder diagram still supports the use of K and the use of H.**
- There is no limit on the number of lines which can be created, but users still have to consider whether the size of the program compiled exceeds the capacity of the memory in the motion controller used.
- In a network, the output object of a motion instruction, the output object of a M-code, and the output object of a G-code are allowed to appear independently. However, other instructions in a network must contain output objects and input objects.

6.1.3 Editing Environment

The environment in which a ladder diagram can be edited is shown below. The table at the upper part of the window is a local symbol table, and the area at the lower part of the window is a working area. A ladder diagram consists of networks. The color at the left side of a network indicates the state of the network. A network can be selected or unselected. The network which is selected is in yellow, and the networks which are not selected are in gray.



The black dotted frame in the network selected indicates the position which is being edited presently. It also indicates the object which is being selected presently. The meaning that the black dotted frame in the network selected conveys varies with the position where the black dotted frame appears.



The whole network is selected.

The coil is selected.

The contact is selected.

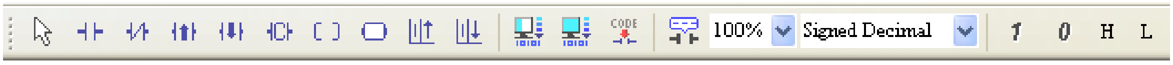
The applied instruction is selected.

If users click the network number, the network will be selected.

6.1.4 Toolbar

After a program editing window in which a ladder diagram can be created is opened, a toolbar will appear in the PMSoft window. The functions of the toolbar are described below.

- PMSoft toolbar

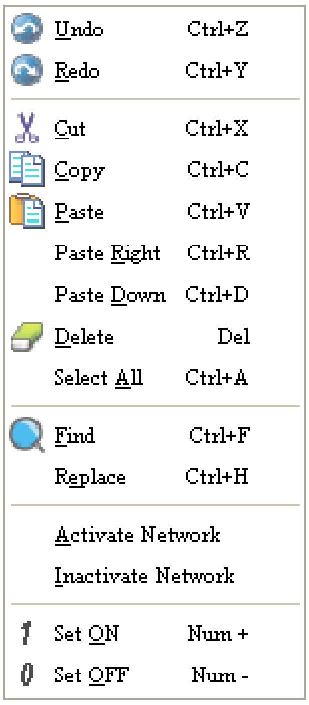


Icon	Keyboard shortcut	Function
↖	Esc	Selection
-| |-	F4	Inserting a normally-open contact
-|/|-	F5	Inserting a normally-closed contact
-|↑|-	F6	Inserting a rising edge-triggered contact
-|↓|-	F7	Inserting a falling edge-triggered contact
-|C|-	F8	Inserting a comparison contact
[ ]	F9	Inserting a coil
▭	F10	Inserting an applied instruction
||↑	F11	Inserting a network above the network selected
||↓	F12	Inserting a network under the network selected
🖥	Ctrl+ F10	Checking the program
🖥	Ctrl + F7	Compiling the program
CODE	None	Transforming an instruction list into a ladder diagram (The function is not applicable to ladder diagrams.)
▭	None	Displaying information
100%	None	Zooming in/Zooming out
Signed Decimal	None	Converting the values monitored
1	Num+	When the online monitoring function is enabled, the contact selected is set to ON.

Icon	Keyboard shortcut	Function
0	Num-	When the online monitoring function is enabled, the contact selected is set to OFF.
H	None	Maximum baud rate
L	None	Minimum baud rate

6.1.5 Context Menu

After users right-click the working area in a program editing window, a context menu will appear. The functions of the items on the context menu are described below.



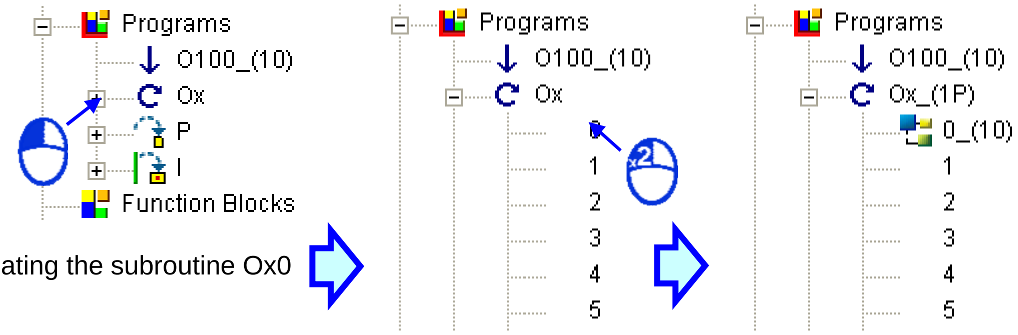
Item	Function
Undo	Undoing the last action (The number of previous actions that can be undone is 20.)
Redo	Redoing an action which has been undone
Cut	Cutting a device, a block, or a network
Copy	Copying a device, a block, or a network
Paste	Pasting the object which has been copied or cut on the present position
Paste Right	Pasting an object at the right side of the position selected (The object will be connected to the position selected in series.)
Paste Down	Pasting an object under the position selected (The object will be connected to the position selected in parallel.)
Delete	Deleting a device, a block, or a network
Select All	Selecting all the networks in the working area
Find	Searching for an item (Please refer to chapter 9 for more information.)
Replace	Replacing the item found with another item (Please refer to chapter 9 for more information.)
Activate Network	Activating the network selected

Item	Function
Inactivate Network	Inactivating the network selected
Set ON	When the online monitoring function is enabled, the contact selected is set to ON.
Set OFF	When the online monitoring function is enabled, the contact selected is set to OFF.

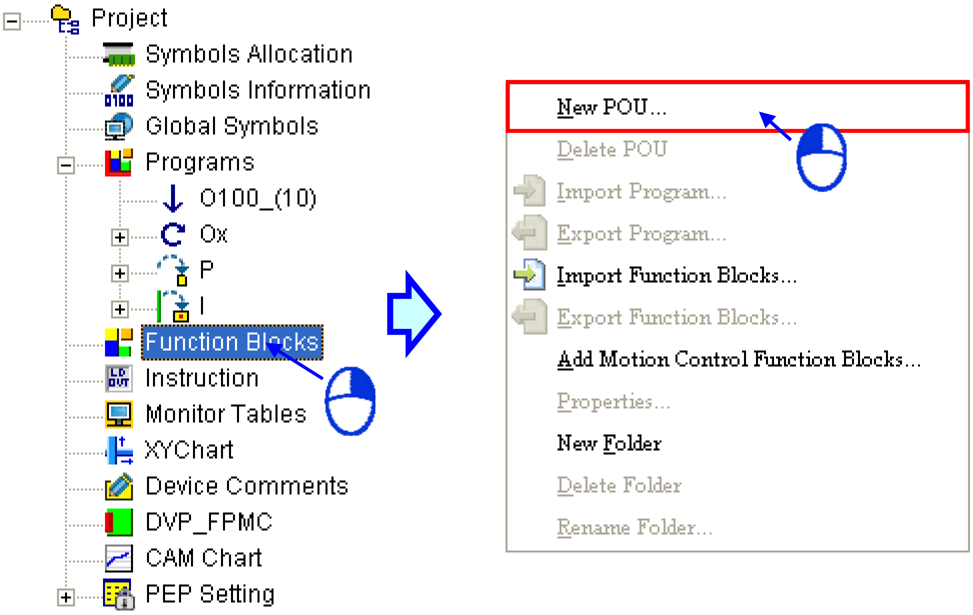
6.2 Creating a Ladder Diagram in PMSoft

6.2.1 Adding a POU which is a Ladder Diagram

(1) If users want to create a POU which is a program, they have to expand a program section in the system information area, and double-click a number. If the users want to create a POU which is a function block, they have to right-click **Function Blocks** in the system information area, and click **New POU…** on the context menu. Please refer to chapter 3 for more information.

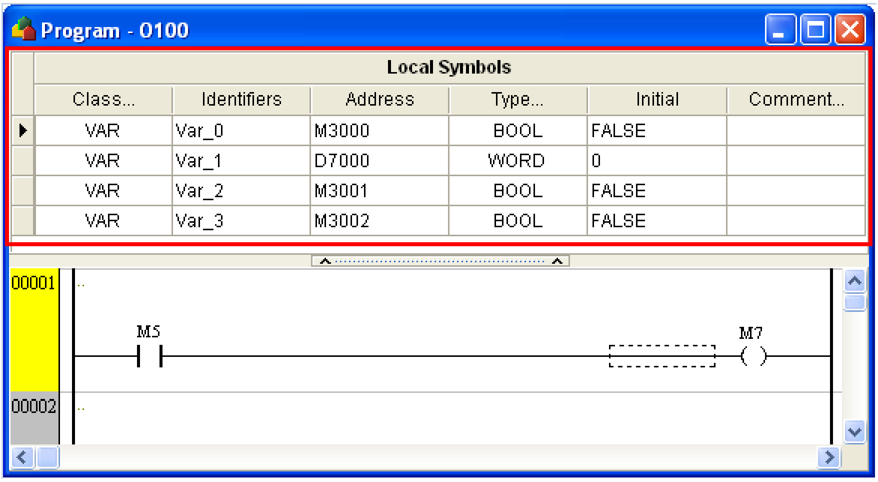


Creating the subroutine Ox0



Creating a POU which is a function block

(2) After a POU is created, PMSoft will automatically open a program editing window. The users can use the PMSoft toolbar in the window, and create local symbols in the local symbol table in the window. Please refer to chapter 4 for more information. There are ten default networks in a program editing window.



Program - 0100

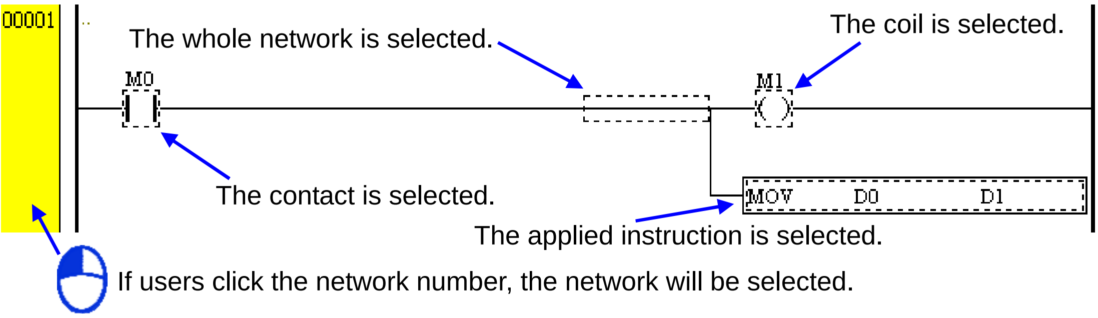
Local Symbols					
Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	Var_0	M3000	BOOL	FALSE	
VAR	Var_1	D7000	WORD	0	
VAR	Var_2	M3001	BOOL	FALSE	
VAR	Var_3	M3002	BOOL	FALSE	

6.2.2 Selecting an Object or a Block

Before an object, a block, or a network is selected, users have to press Esc on the keyboard, or click on the toolbar.

- Selecting an object

  Users have to move the mouse to the object they want to select, and then click the left mouse button. If an object in a network is selected, the color at the left side of the network will be yellow. If no object in a network is selected, the color at the left side of the network will be gray.



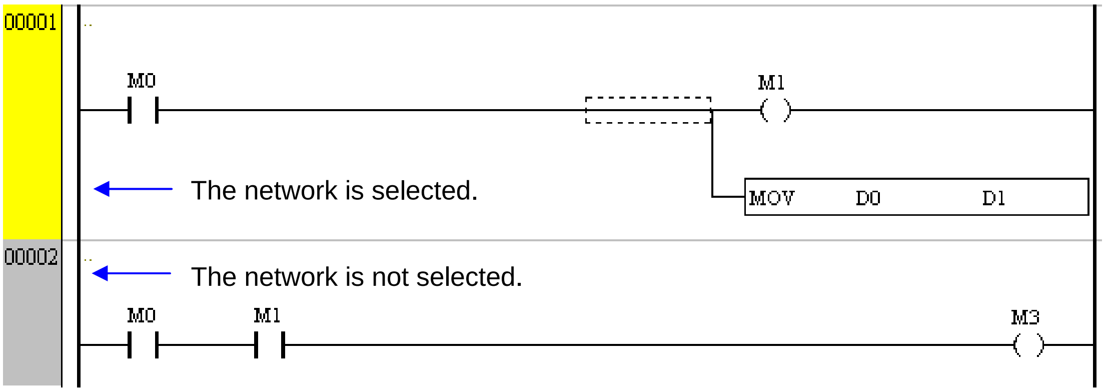
The whole network is selected.

The coil is selected.

The contact is selected.

The applied instruction is selected.

If users click the network number, the network will be selected.

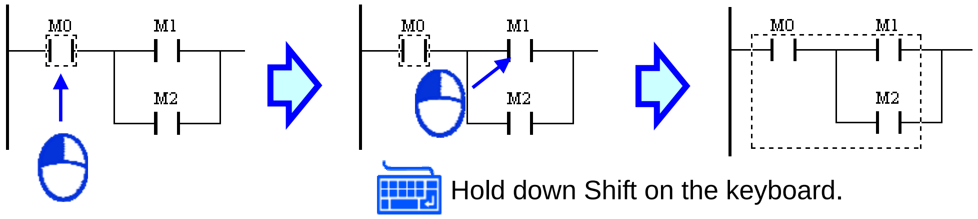


The network is selected.

The network is not selected.

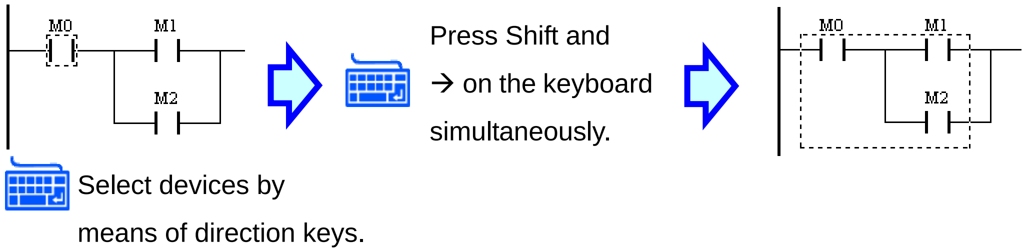
- Selecting a block

  If users click a device, and hold down Shift on the keyboard while they click another device, the devices clicked and the devices between the devices clicked will be selected, and will be in a dotted frame. The users can connect the frame to a position in parallel, connect the frame to a position in series, delete the frame, copy the frame, paste the frame, and cut the frame. The devices selected must be in the same network in a ladder diagram, and must be adjacent to one another. An input device and the output device in a network in a ladder diagram can not be in a dotted frame.



Hold down Shift on the keyboard.

The users can also select devices by means of the direction keys (→, ↑, ↓, and ←) on the keyboard. They can click a device, and hold down Shift on the keyboard while they press the direction keys to enlarge the dotted frame which appears.

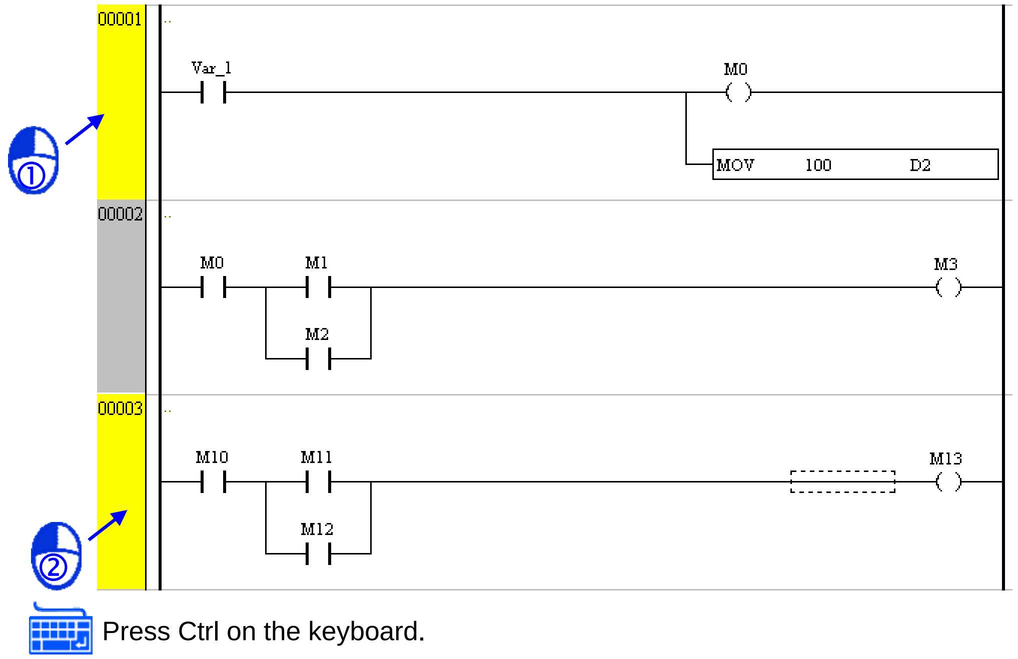


Press Shift and → on the keyboard simultaneously.

Select devices by means of direction keys.

- Selecting multiple networks in a ladder diagram

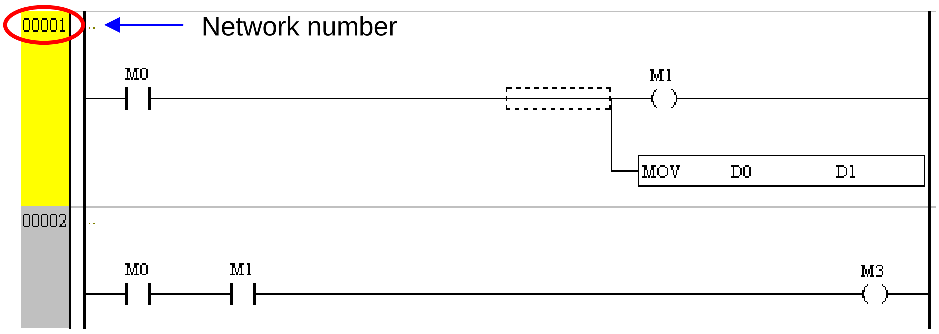
  If users hold down Ctrl on the keyboard while they click networks which are not adjacent to one another, the networks will be selected. If the users click at the start of a selection, scroll to the end of the selection, and hold down Shift on the keyboard while they click where they want the selection to end, they will select a range of networks.



Press Ctrl on the keyboard.

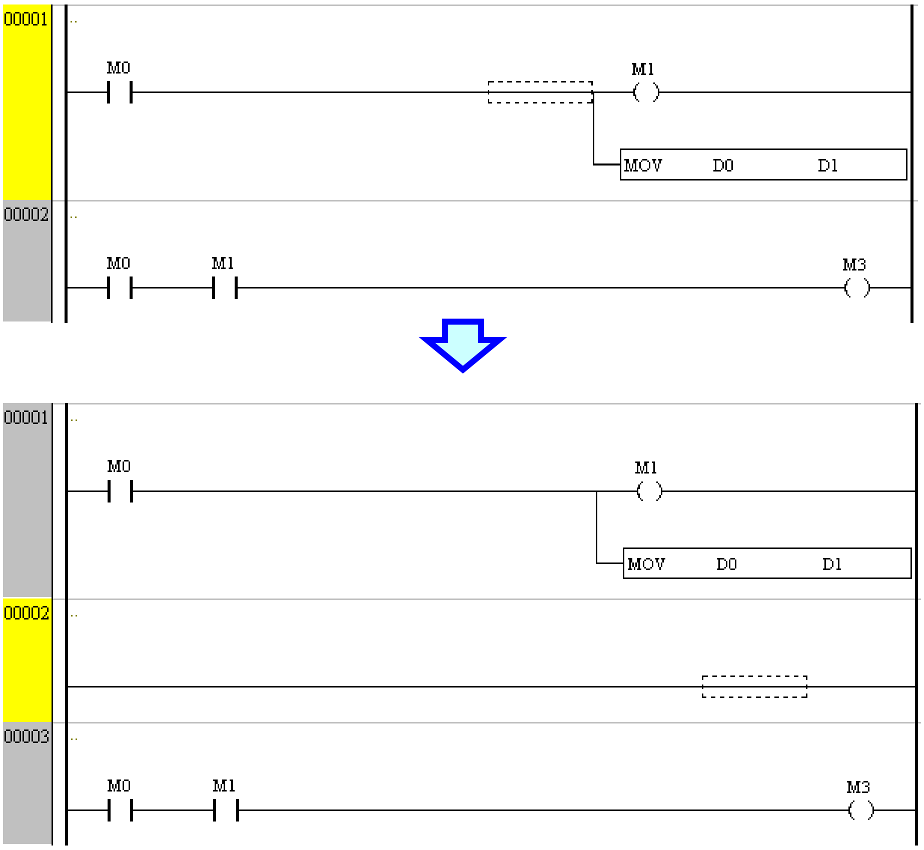
6.2.3 Networks in a Ladder Diagram

A ladder diagram consists of networks. Every network in a ladder diagram is an independent program. 256 contacts at most can be connected in series in PMSoft. There is no limit on the number of contacts or coils which can be connected in parallel, and there is no mark which is used to connect two networks.

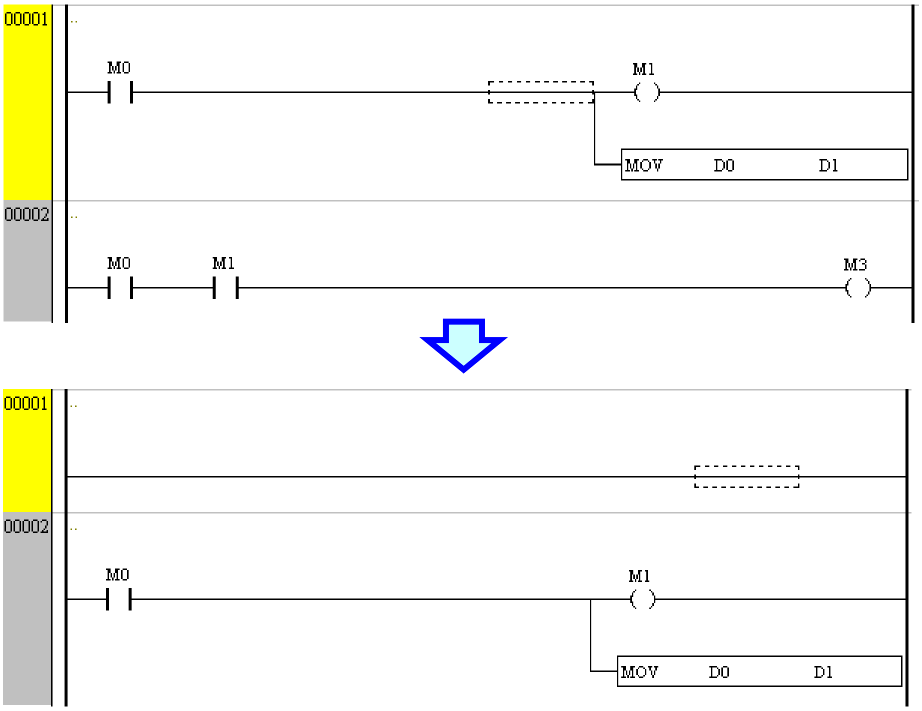

Network number

- **Inserting a network**

(1) If users click on the toolbar, a blank network will be put under the network selected.

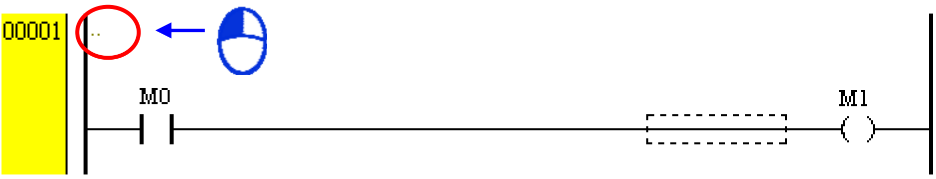


(2) If users click on the toolbar, a blank network will be put above the network selected.

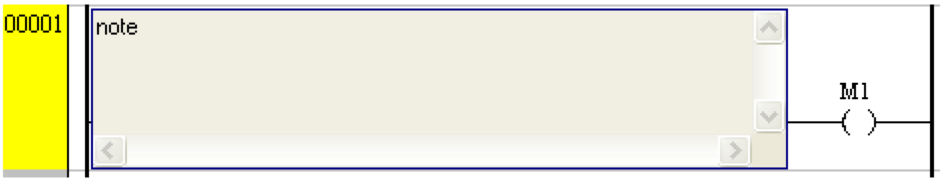


6.2.4 Making a Comment on a Network

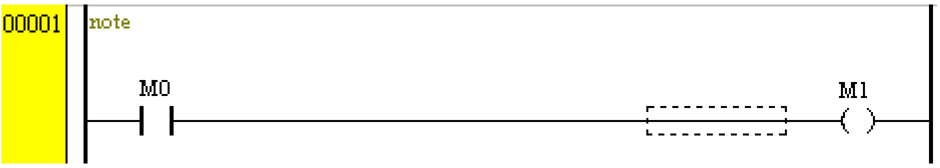
(1) Click **..** in the upper left corner of a network.



(2) Type a comment in the box which appears. If users want to start a new line of text at a specific point, they can press Enter on the keyboard.



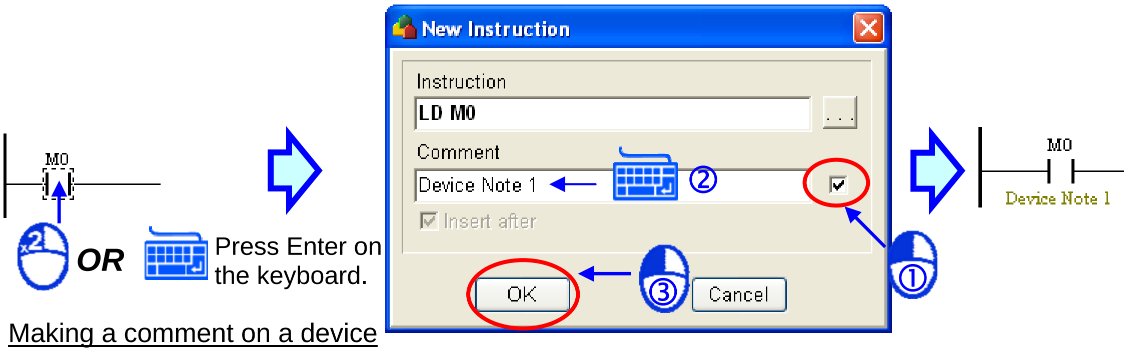
(3) After the users type a comment, they have to click the blank in the network.



6.2.5 Making Comments on Devices and Instructions

Users can make comments on contacts, coils, applied instructions, and comparison contacts. They can only type a line of text in the **Comment** box in the **New Instruction** window. There are two methods of typing comments on devices. Please refer to sections 6.2.8 and 6.2.9 for more information about contacts, coils, applied instructions, and comparison contacts.

- Method 1
  The users have to double-click a device or an instruction, or press Enter on the keyboard after they select a device or an instruction. Then, the users have to select the checkbox at the right side of the **Comment** box in the **New Instrucion** window, and type a comment in the **Comment** box. Finally, the users have to click **OK**. If the users want to modify the comment, they can open the **New Instrucion** window again.



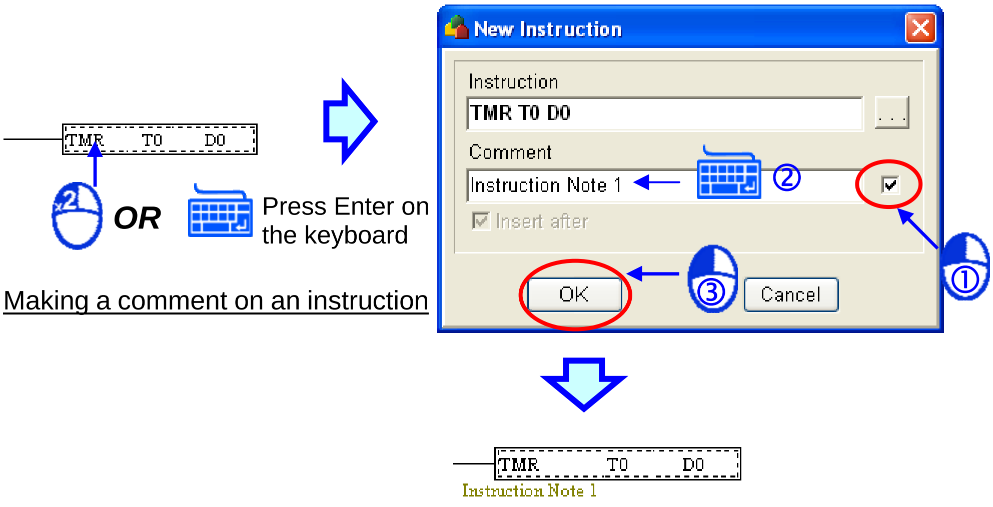
New Instruction

Instruction: LD M0

Comment: Device Note 1 ☑

☑ Insert after

Making a comment on a device



New Instruction

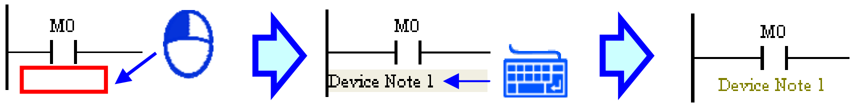
Instruction: TMR T0 D0

Comment: Instruction Note 1 ☑

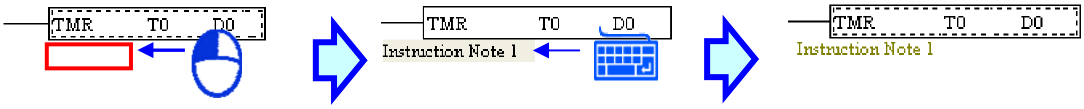
☑ Insert after

Making a comment on an instruction

- Method 2
  The users have to click the position under a device or an instruction in a network. Then, the users have to type a comment. Finally, the users have to press Enter on the keyboard, or click the blank in the network.

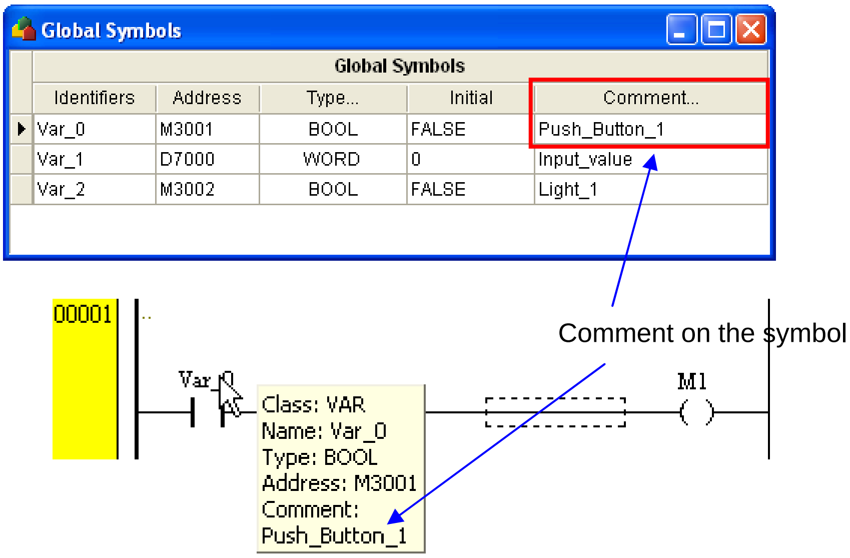


Making a comment on a device



Making a comment on an instruction

If a contact or a coil used is assigned a symbol, the users can only make a comment on the symbol, and they can not make a comment on the contact or the coil. (Please refer to chapter 4 for more information.) Besides, after the mouse cursor stays at the contact or the coil which is assigned a symbol for a while, the comment on the symbol will appear.



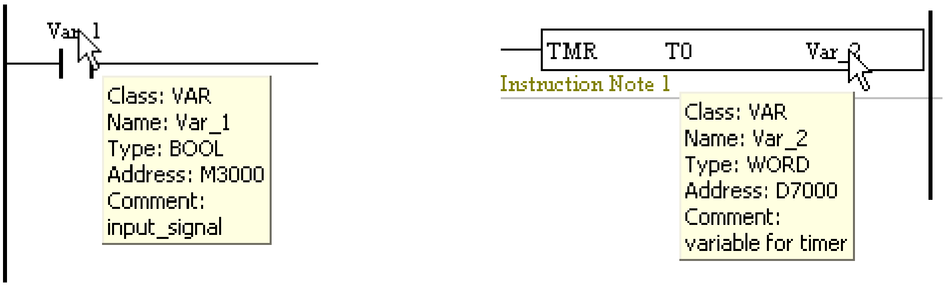
Global Symbols

Identifiers	Address	Type...	Initial	Comment...
Var_0	M3001	BOOL	FALSE	Push_Button_1
Var_1	D7000	WORD	0	Input_value
Var_2	M3002	BOOL	FALSE	Light_1

Class: VAR
Name: Var_0
Type: BOOL
Address: M3001
Comment:
Push_Button_1

6.2.6 Displaying / Hiding Information

If users move the mouse cursor to a device or a symbol (a contact, a coil, an instruction, a function block, or a comparison contact) after they click on the toolbar, the information about the device or the symbol will appear. The information related to a device or a symbol includes a device address and a comment.

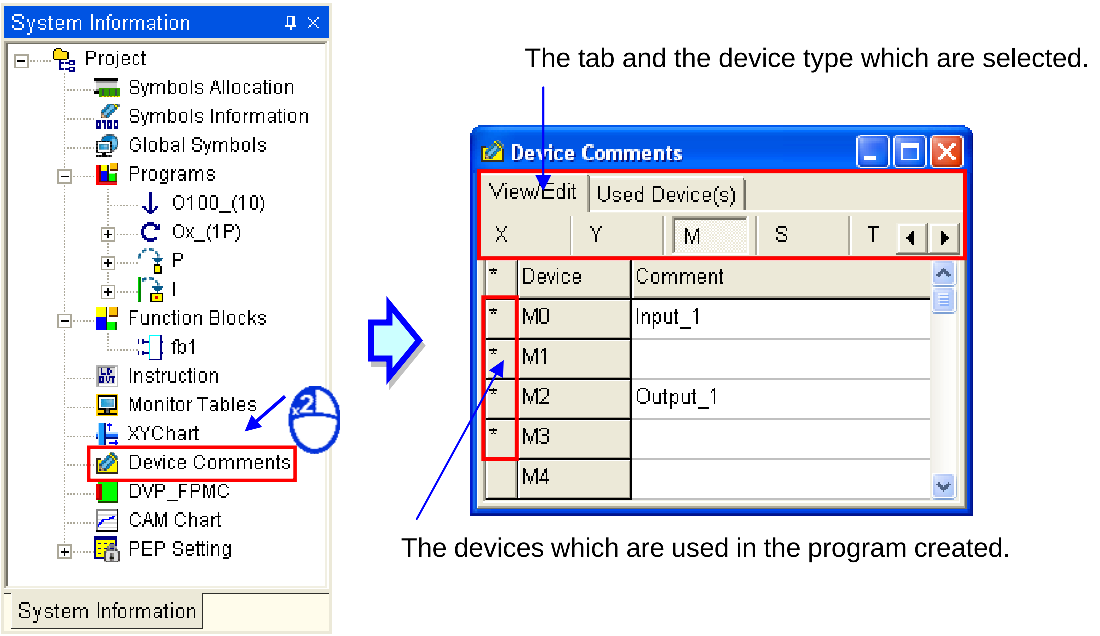


Class: VAR
Name: Var_1
Type: BOOL
Address: M3000
Comment:
input_signal

Class: VAR
Name: Var_2
Type: WORD
Address: D7000
Comment:
variable for timer

6.2.7 Device Comments

After users double-click **Device Comments** in the system information area, the **Device Comments** window will appear. After the users click the View/Edit tab, they can view lists of devices. The devices used in the program created are marked with *. The comments which the users make on devices will appear in the **Comment** cells for the devices. The users can also edit comments in the **Device Comments** window.

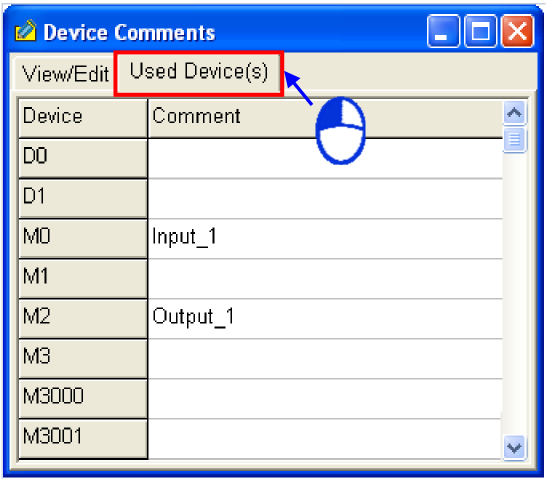


*	Device	Comment
*	M0	Input_1
*	M1	
*	M2	Output_1
*	M3	
	M4	

The tab and the device type which are selected.

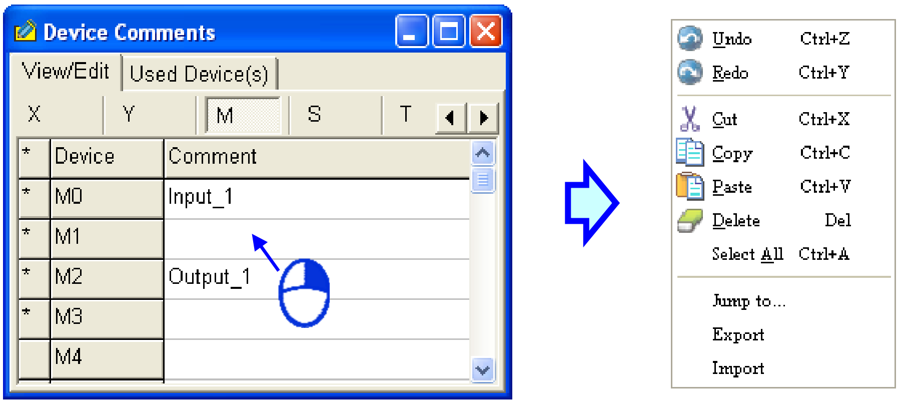
The devices which are used in the program created.

If the users click the Used Device(s) tab, they can view the devices used, and the comments on the devices.



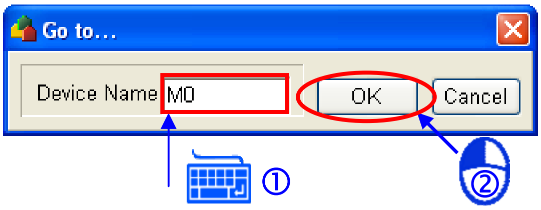
Device	Comment
D0	
D1	
M0	Input_1
M1	
M2	Output_1
M3	
M3000	
M3001	

If the users right-click a **Comment** cell, a context menu will appear. The items on the context menu are described below.



*	Device	Comment
*	M0	Input_1
*	M1	
*	M2	Output_1
*	M3	
	M4	

- Editing functions: **Undo**, **Redo**, **Cut**, **Copy**, **Paste**, **Delete**, and **Select All**
- **Jump to…**: Users can rapidly find a device. The users have to type a device name in the **Device Name** box, and click **OK**.

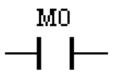


- **Export** and **Import**: If users click **Export**, the comments which the users make on devices will be exported, and will be saved as **DevCmtData.txt** in the **DevCmtData** folder. If the users click Import, **Devcmtdata.txt** in the **DevCmtData** folder will be imported.

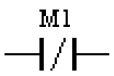
6.2.8 Contacts and Coils

Contacts and a coil which can be used in PMSoft are shown below. There are four types of contacts. They are normally-open contacts, normally-closed contacts, rising edge-triggered contacts, and falling-edge triggered contacts. Normally-open contacts are also called Form A contacts, and normally-closed contacts are also called Form B contacts. OUT coils output states. Besides, the text above a contact or a coil is the device address or the symbol which is assigned to the contact or the coil.

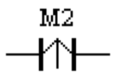
- Normally-open contact



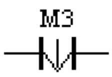
- Normally-closed contact



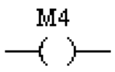
- Rising edge-triggered contact



- Falling edge-triggered contact



- OUT coil

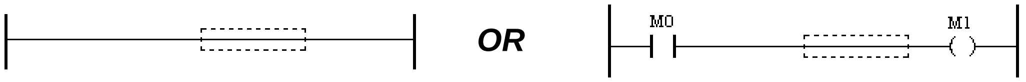


***. The usage of the instruction SET and RESET is the same as the usage of applied instructions. Please refer to section 6.2.9 for more information.**

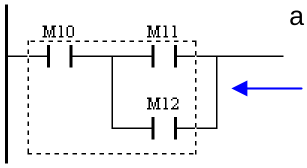
6.2.8.1 Inserting a Contact

There are two methods of inserting a contact. Users can use one of the methods according to their habit.

- Method 1: PMSoft toolbar
  (1) The users can insert a contact into a line, or connect a contact to a contact or a block in series/in parallel. Before the users connect a contact to a block, they have to select the block.

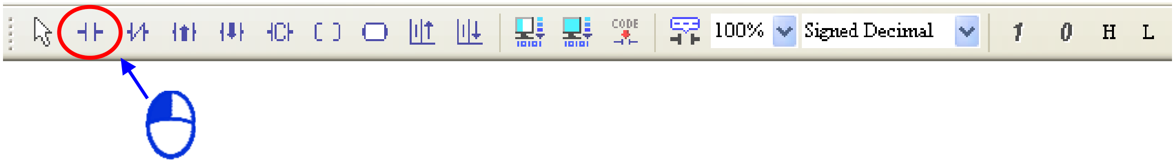


Inserting a contact into a line, or connecting a contact to a contact in series/in parallel

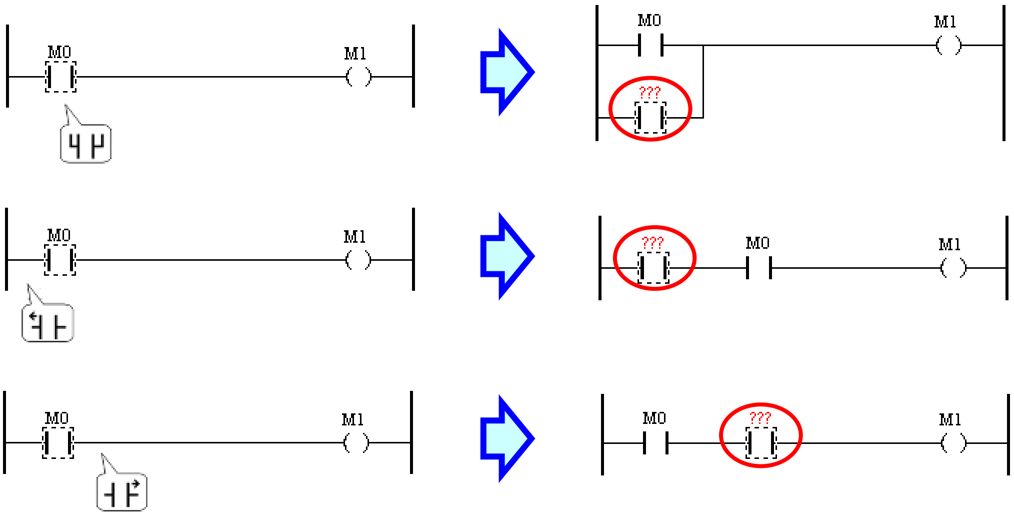


Connecting a contact to a block in series/parallel

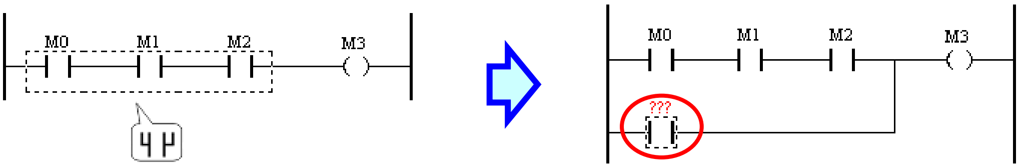
(2) The users have to click ⊣⊢ on the toolbar.



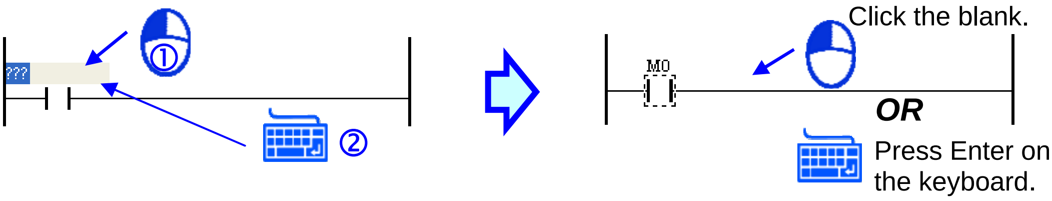
(3) The users have to move the mouse cursor to a position. The mouse cursor becomes a contact when it is at the right side of the position, at the left side of the position, or at the bottom of the position. After the users click the left mouse button, a contact will be inserted. Please refer to the figures below.



(4) If the users want to connect a contact to a block in series/in parallel, they have to select the contact in accordance with the description in section 6.2.2, move the mouse cursor to a position, and click the left mouse button. M0, M1, and M2 in the figure below are selected. A contact is connected to M0, M1, and M2 in parallel.

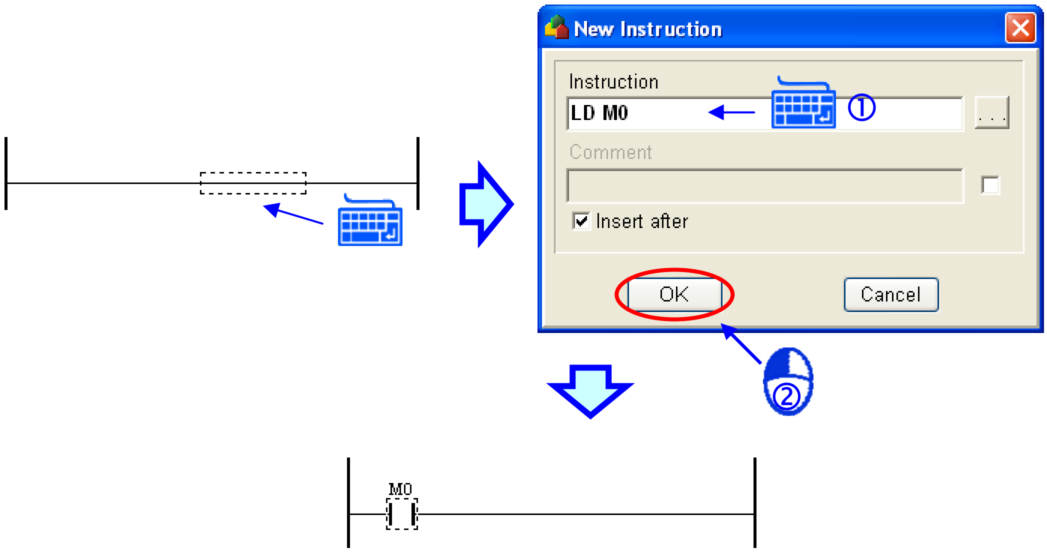


(5) After the users click **???** on the contact added to a network, they can type a device name or a symbol. After the users type a device name or a symbol, they have to press Enter on the keyboard or click the blank in the network. Please refer to chapter 4 for more information about symbols.



- Method 2: Typing an instruction

(1) After the users select a position, they have to press Enter on the keyboard, or type an IL instruction. After the users press Enter on the keyboard, or type an instruction, the **New Instruction** window will appear. After the users type an IL instruction in the **Instruction** box, they have to press Enter on the keyboard, or click **OK**. (The instruction that the users type is case-insensitive. If the users type an incorrect applied instruction, or an incorrect device address, an error message will appear after they click **OK**.)



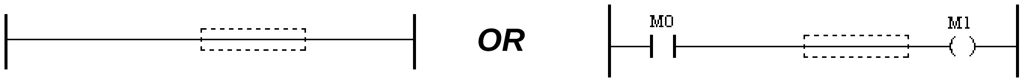
*. Please refer to section 6.2.5 for more information about making comments on devices and instructions.

6.2.8.2 Inserting a Coil

There are two methods of inserting a contact. Users can use one of the methods according to their habit.

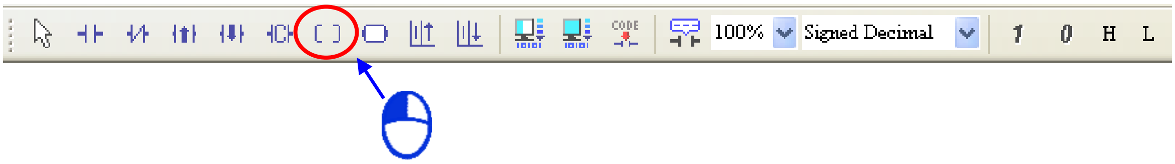
- Method 1: PMSoft toolbar

(1) The users can insert a coil into a line, or connect a coil to a coil in parallel.

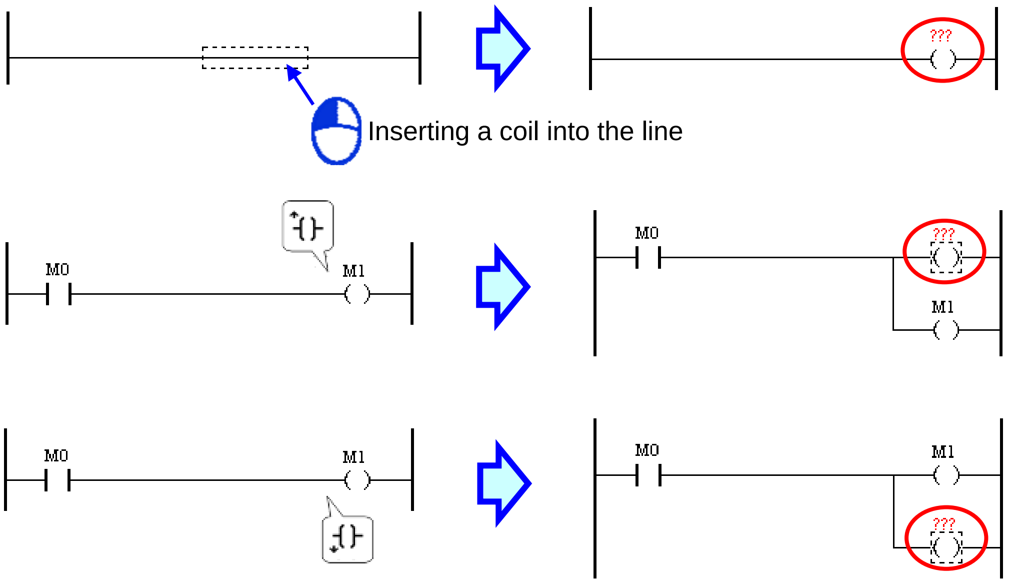


Inserting a coil into a line, or connecting a coil to a coil in parallel

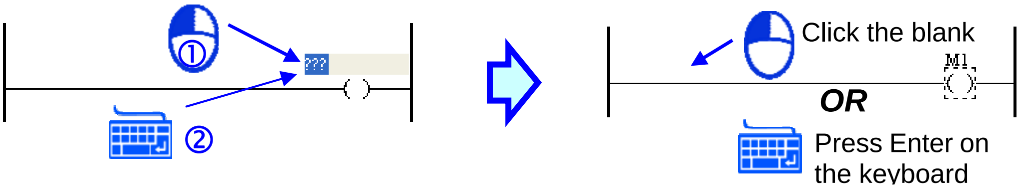
(2) The users have to click [ ] on the toolbar.



(3) The users have to move the mouse cursor to a position. The mouse cursor becomes a coil when it is at the top of the position, or at the bottom of the position. After the users click the left mouse button, a coil will be inserted. Please refer to the figures below.

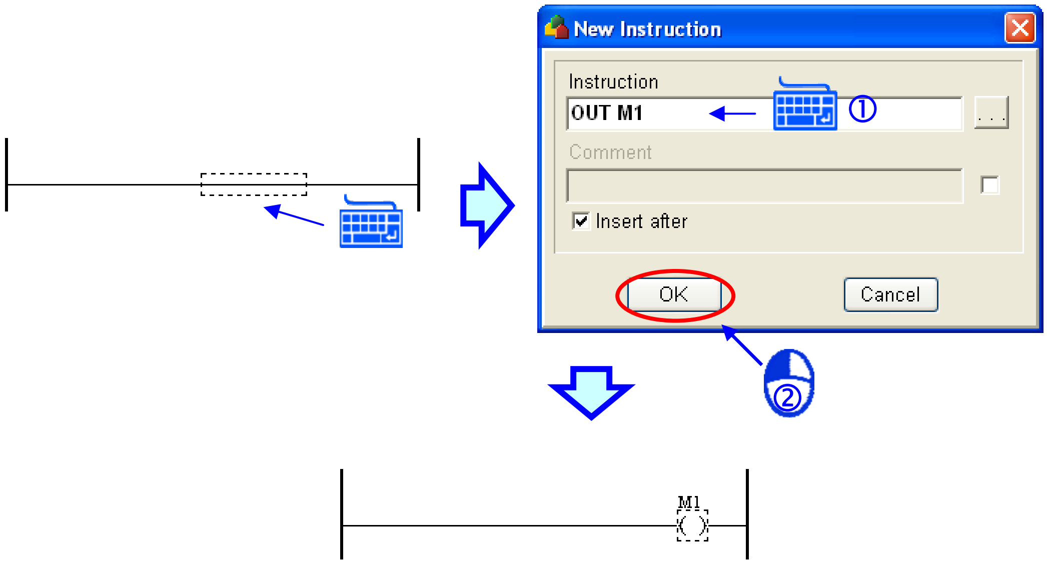


(4) After the users click **???** on the coil added to a network, they can type a device name or a symbol. After the users type a device name or a symbol, they have to press Enter on the keyboard or click the blank in the network. Please refer to chapter 4 for more information about symbols.



- Method 2: Typing an instruction

  After the users select a position, they have to press Enter on the keyboard, or type an IL instruction. After the users press Enter on the keyboard, or type an instruction, the **New Instruction** window will appear. After the users type an IL instruction in the **Instruction** box, they have to press Enter on the keyboard, or click **OK**. (The instruction that the users type is case-insensitive. If the users type an incorrect applied instruction, or an incorrect device address, an error message will appear after they click **OK**.)



New Instruction

Instruction: OUT M1

Comment:

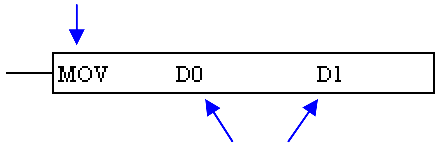
☑ Insert after

*. Please refer to section 6.2.5 for more information about making comments on devices and instructions.

6.2.9 Applied Instructions, Motion Instructions, G-codes, Function Blocks, and Comparison Contacts

The applied instructions, the motion instructions, and G-codes in a ladder diagram are represented by blocks. An instruction name and operands are displayed in a block. The pin at the left side of the block is connected to a logic state. If the logic state is ON, the instruction represented by the block will be executed. Besides, the block can only be added to the end of a network.

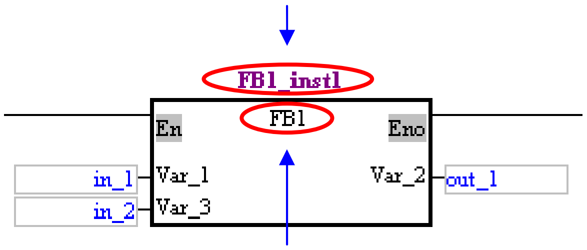
Instruction name



Operands

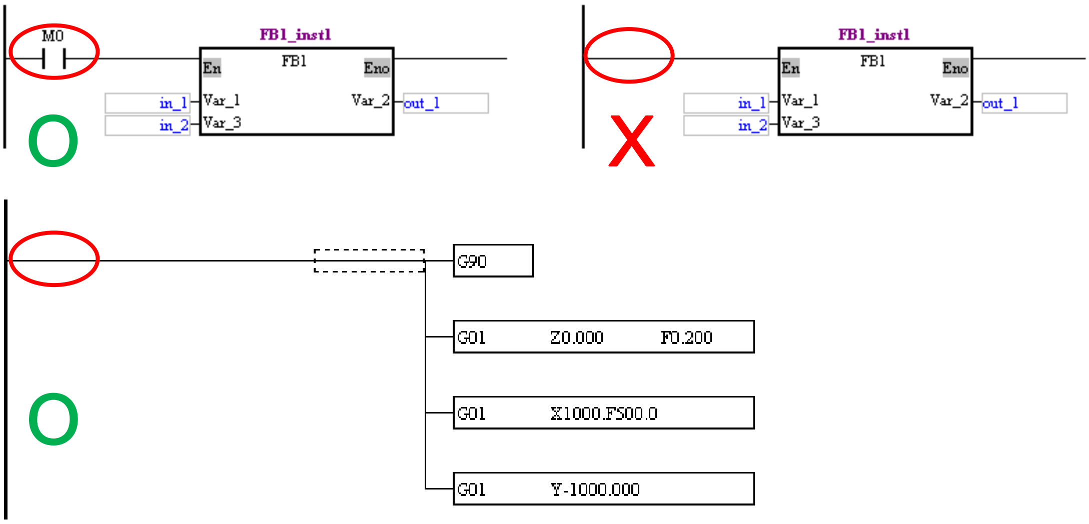
A function block in PMSoft is shown below. The **En** pin is connected to a logic state. If the logic state is ON, the function block will be executed. The Eno pin connects the state of the **En** pin to the device or the contact which follows it. A function block can be put in any position in a ladder diagram, but the **En** pin of the function block must be connected to a device, a function block, or a block. Besides, the name above a function block is the symbol assigned to the function block, i.e. a function block instance. Please refer to chapter 5 for more information about function blocks.

Function block instance



Function block definition

If an applied instruction or a function block is used in a ladder diagram, the pin of the applied instruction or the **En** pin of the function block must be connected to a device or a block, and can not be connected to a busbar directly. However, motion instructions, comparison contacts, and G-codes can be directly connected to the busbars in networks. If the networks are scanned, the motion instructions, comparison contacts, and G-codes will be executed.

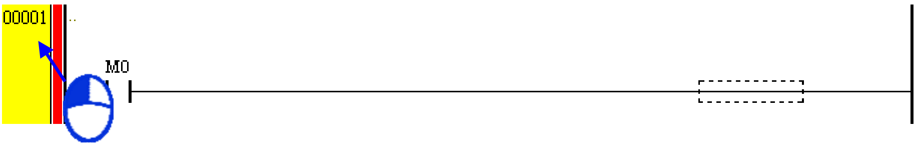


6.2.9.1 Inserting an Applied instruction, a Motion Instruction, or a G-code

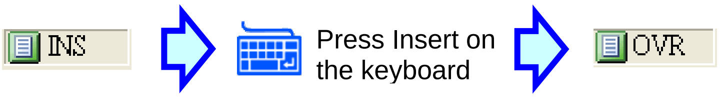
There are three methods of inserting an applied instruction or a motion instruction. Users can use one of the methods according to their habit. Owing to the fact that **Instruction Wizard** does not support G-codes, the users can only use method 2 or method 3 if they want to insert a G-code.

- Method 1: **Instruction Wizard**

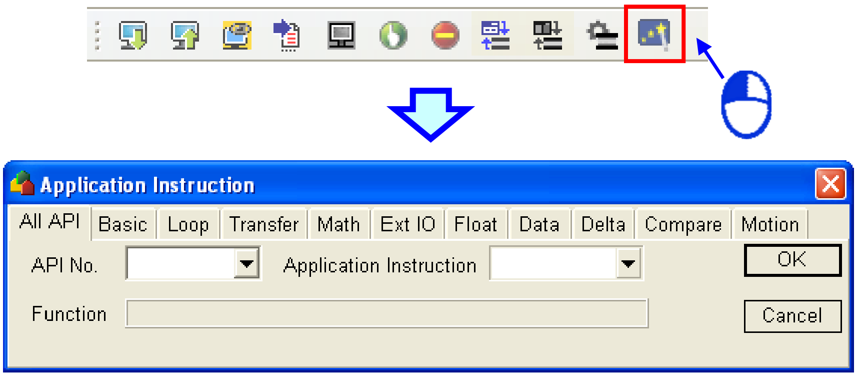
(1) The users have to click a network into which they want to insert an applied instruction or a motion instruction.



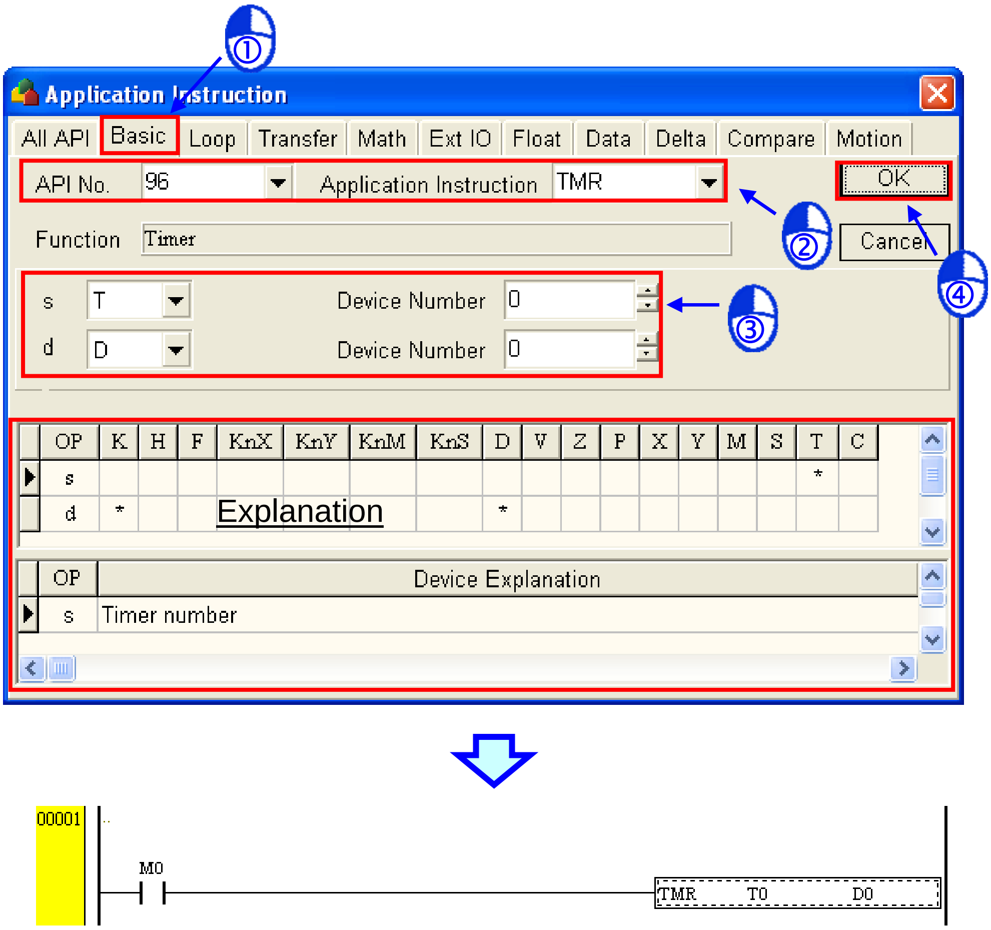
(2) The users have to make sure that the mode displayed in the status bar is the the **OVR** mode. After the users press Insert on the keyboard, the mode displayed in the status bar will be the **OVR** mode.



(3) After the users click on the fast toolbar, the **Application Instruction** window will appear.



(4) The users have to select an instruction type, select an instruction in the **API No.** drop-down list box or the **Application Instruction** drop-down list box, select devices which are supported according to the explanation at the bottom of the window, and click **OK**.



API No. 96

Application Instruction TMR

Function Timer

s T Device Number 0

d D Device Number 0

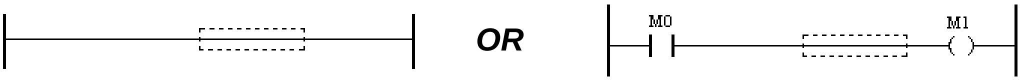
OP	K	H	F	KnX	KnY	KnM	KnS	D	V	Z	P	X	Y	M	S	T	C
s																*	
d	*							*									

OP	Device Explanation
s	Timer number

```
00001
      M0
 ----| |--------------------------------------[TMR    T0    D0 ]
```

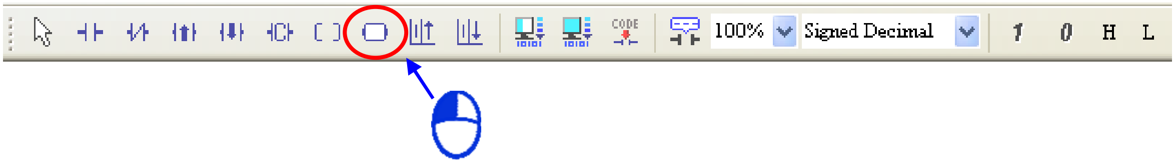
- Method 2: PMSoft toolbar

(1) The users can insert an applied instruction into a line, or connect an instruction to a coil in parallel. Motion instructions and G-codes can not be preceded by contacts.

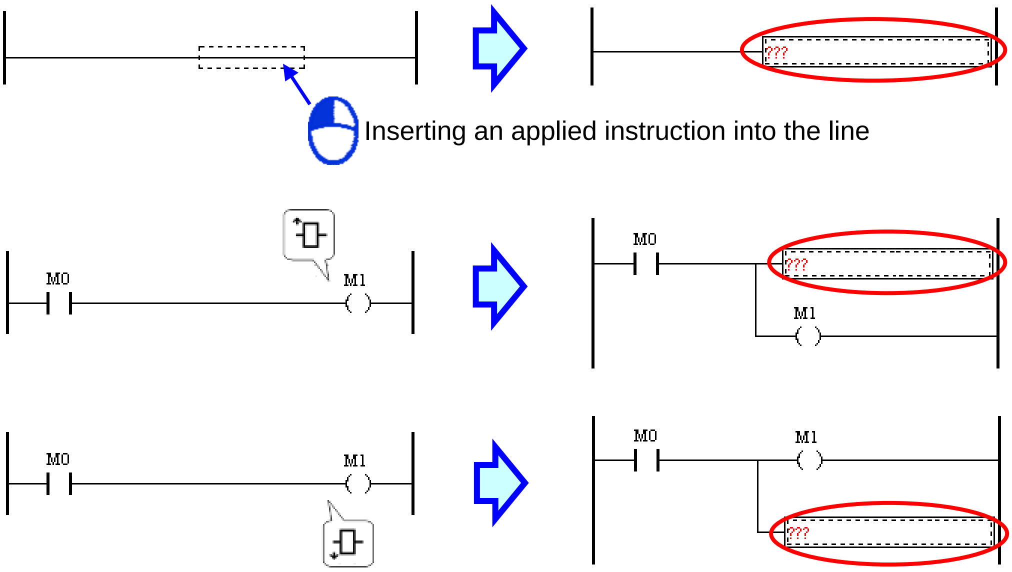


Inserting an applied instruction into a line, or connecting an applied instruction to a coil in parallel

(2) The users have to click ▭ on the toolbar.

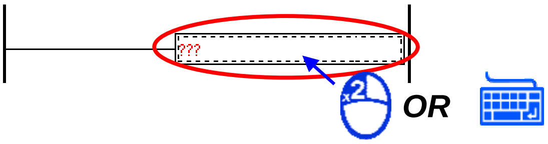


(3) The users have to move the mouse cursor to a position. The mouse cursor becomes an instruction block when it is at the top of the position, or at the bottom of the position. After the users click the left mouse button, an instruction block will be inserted. Please refer to the figures below.



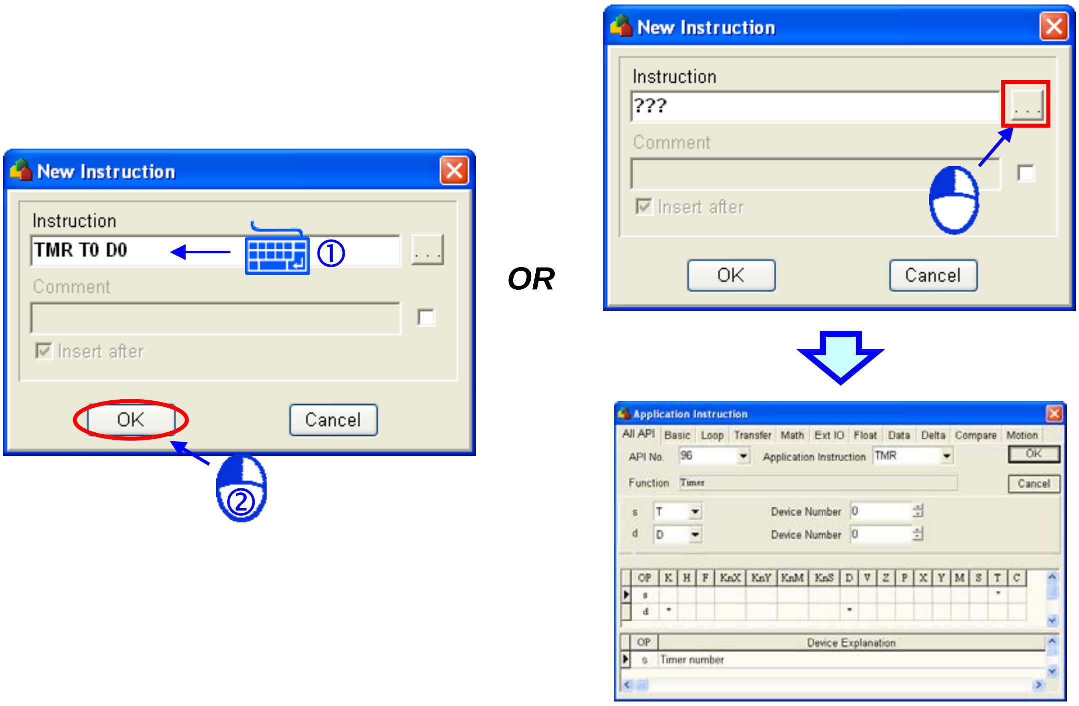
Inserting an applied instruction into the line

(4) The users have to double-click the instruction block inserted, or press Enter on the keyboard after they click the instruction block.



Press Enter on the keyboard

(5) The users have to type an instruction and operands in the **Instruction** box, or click ... . If the users click ..., they have to select an instruction type, select an instruction in the **API No.** drop-down list box or the **Application Instruction** drop-down list box, select devices which are supported according the explanation at the bottom of the window, and click **OK**.



New Instruction

Instruction: TMR T0 D0

Comment
☑ Insert after
OK
Cancel
①
②
OR
New Instruction
Instruction
???
Comment
☑ Insert after
OK
Cancel
Application Instruction
All API | Basic | Loop | Transfer | Math | Ext IO | Float | Data | Delta | Compare | Motion
OK
Cancel
OP | K | H | F | KnX | KnY | KnM | KnS | D | V | Z | P | X | Y | M | S | T | C
OP | Device Explanation

API No. 96, Application Instruction: TMR

Function: Timer

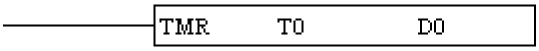
s: T, Device Number: 0

d: D, Device Number: 0

s: Timer number

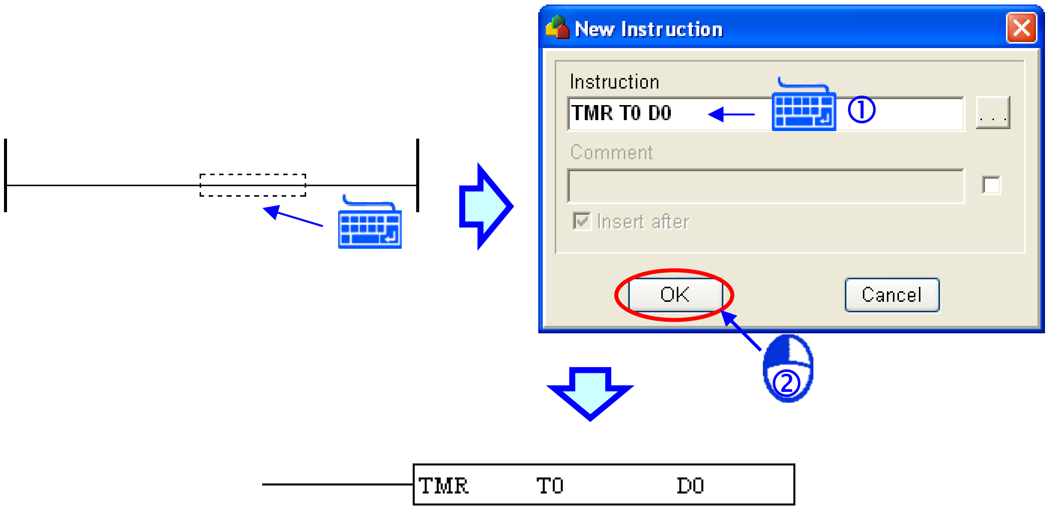
*. Please refer to section 6.2.5 for more information about making comments on devices and instructions.

(6) The result that the users get is like the figure shown below.



- Method 3: Typing an instruction

After the users select a position, they have to press Enter on the keyboard, or type an applied instruction/a motion instruction/a G-code and operands. After the users press Enter on the keyboard, or type an applied instruction/a motion instruction/a G-code and operands, the **New Instruction** window will appear. After the users type an instruction and operands in the **Instruction** box, they have to press Enter on the keyboard, or click **OK**. (The instruction that the users type is case-insensitive. If the users type an incorrect applied instruction, or an incorrect device address, an error message will appear after they click **OK**.)



New Instruction

Instruction: TMR T0 D0

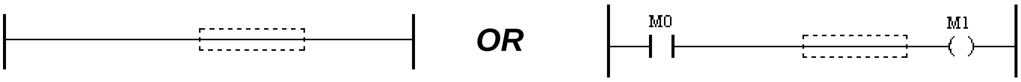
***. Please refer to section 6.2.5 for more information about making comments on devices and instructions.**

6.2.9.2 Inserting a Function Block

There are two methods of inserting a function block. Users can use one of the methods according to their habit.

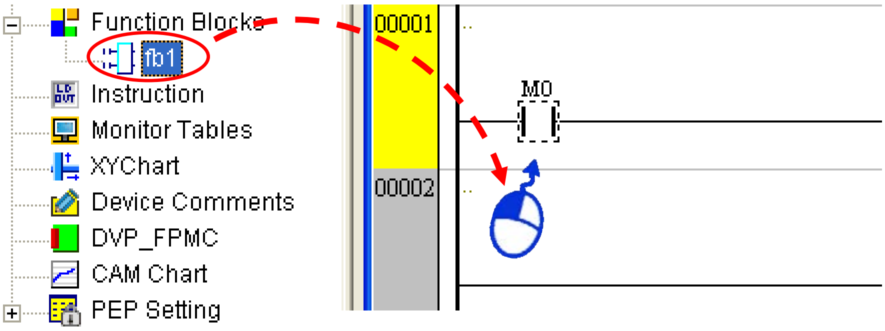
- Method 1: Dragging a function block definition

(1) The users can insert a function block into a line, or connect a function block to a contact in series.

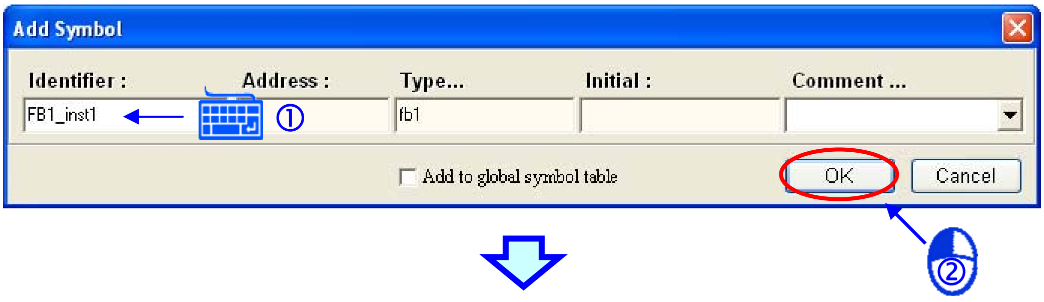


Inserting a function block into a line, or connecting a function block to a contact in series

(2) The users have to press the left mouse button while the mouse cursor hovers over a function block definition. They have to move the mouse cursor to a position while holding the left mouse button down. The **Add Symbol** window will appear after the users release the left mouse button.

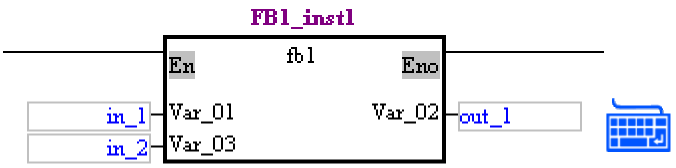


(3) The users have to type a function block instance in the **Add Symbol** window. (The users have to create a symbol whose data type is a function block.) Then, the users have to click **OK**, or press Enter on the keyboard. Finally, the users have to assign devices or symbols to the pins of the function block inserted.


Add Symbol

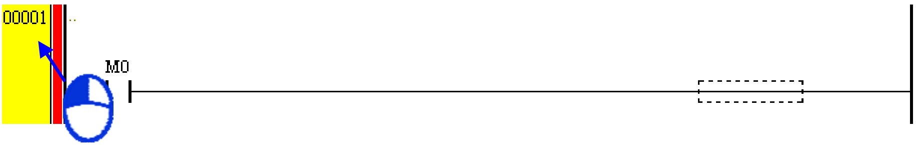
Identifier :	Address :	Type...	Initial :	Comment ...
FB1_inst1		fb1		

□ Add to global symbol table

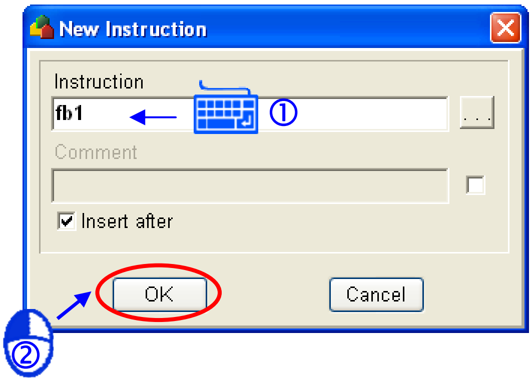


- Method 2: Typing a function block definition

(1) The users have to click a network into which they want to insert a function block.

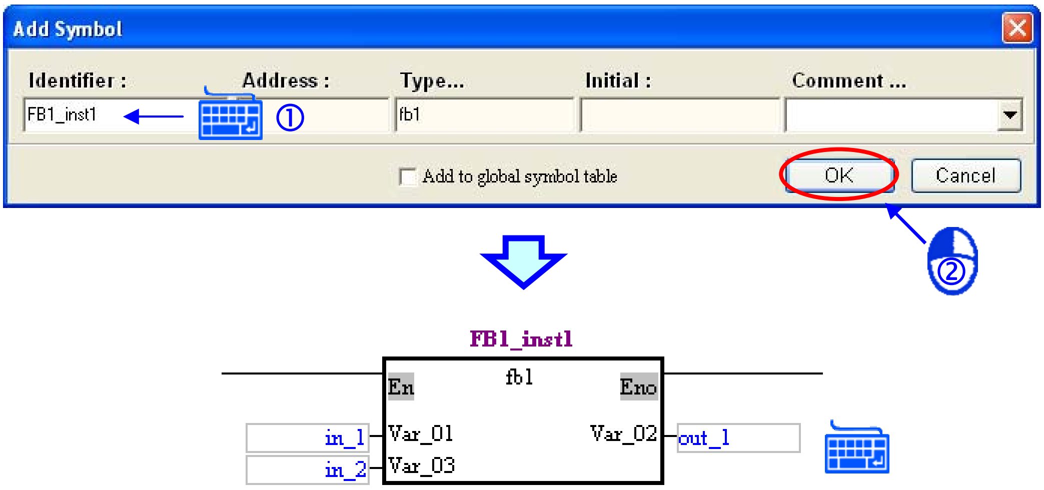


(2) After the users select a position, they have to press Enter on the keyboard, or type a function block definition. After the users press Enter on the keyboard, or type a function block definition, the **New Instruction** window will appear. After the users type a function block definition in the **Instruction** box, they have to press Enter on the keyboard, or click **OK**.



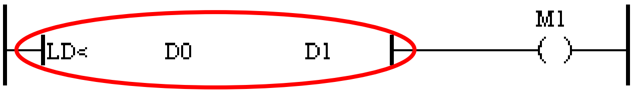
***. Please refer to section 6.2.5 for more information about making comments on devices and instructions.**

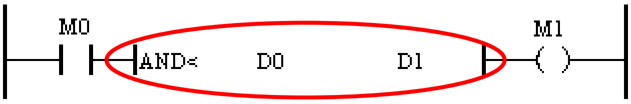
(3) The users have to type a function block instance in the **Add Symbol** window. (The users have to create a symbol whose data type is a function block.) Then, the users have to click **OK**, or press Enter on the keyboard. Finally, the users have to assign devices or symbols to the pins of the function block inserted.

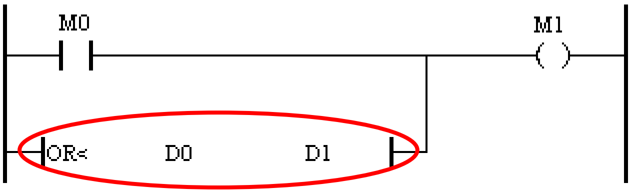


6.2.9.3 Inserting a Comparison Contact

Comparison contacts which can be created in PMSoft are shown below. The operands D0 and D1 are the objects which are compared. The output produced can drive the coil M1, or can be connected to the contact M0 in series/in parallel. There are three types of comparison contacts. They are LD, AND, and OR. Users can insert a comparison contact rapidly.





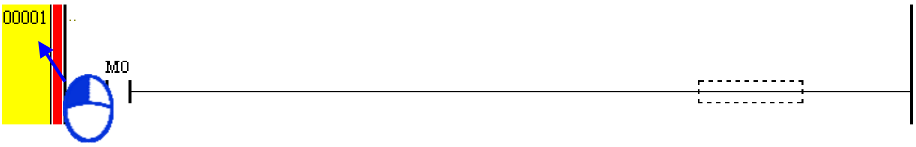


***. If a comparison contact created in a ladder diagram is an incorrect contact, the ladder diagram can still be executed. Besides, after the program is compiled, the incorrect comparison contact in the Instruction window will be modified automatically, but the incorrect comparison contact in the ladder diagram will not be modified. For example, if AND should be created in a ladder diagram, but OR or LD is created instead, the ladder diagram can still be executed.**

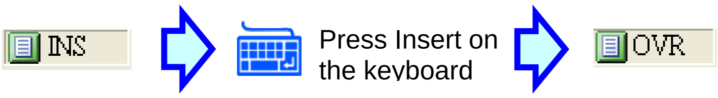
There are three methods of inserting a comparison contact.

- Method 1: **Instruction Wizard**

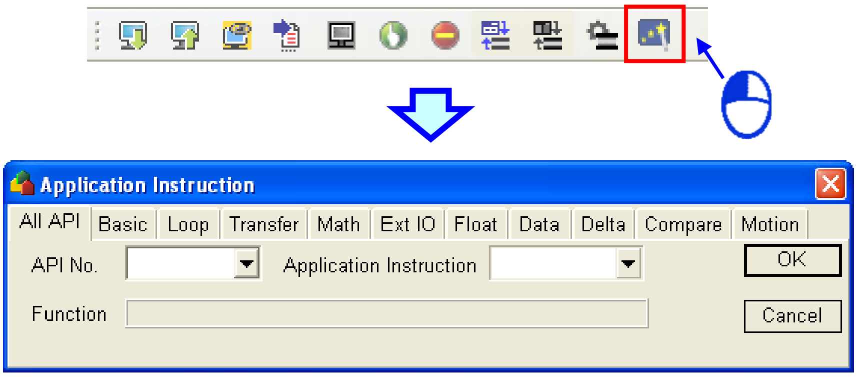
(1) The users have to click a network into which they want to insert a comparison contact.



(2) The users have to make sure that the mode displayed in the status bar is the the **OVR** mode. After the users press Insert on the keyboard, the mode displayed in the status bar will be the **OVR** mode.

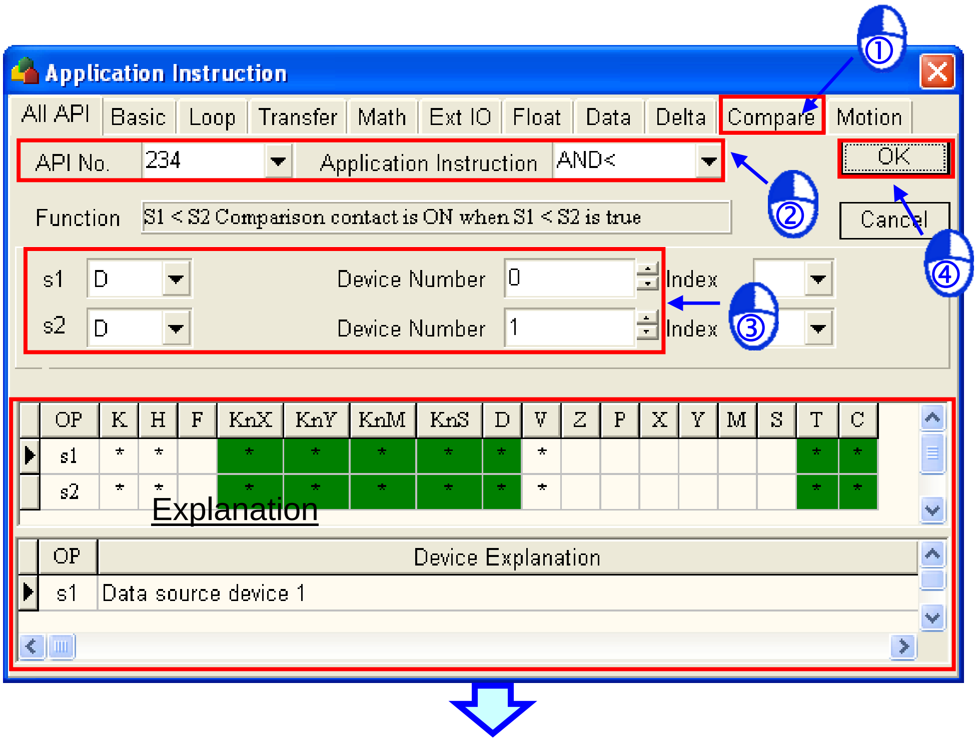


(3) After the users click on the fast toolbar, the **Application Instruction** window will appear.



(4) The users have to click the **Compare** tab, select an instruction in the **API No.** drop-down list

box or the **Application Instruction** drop-down list box, select devices which are supported according the explanation at the bottom of the window, and click **OK**.



API No. 234 Application Instruction AND<

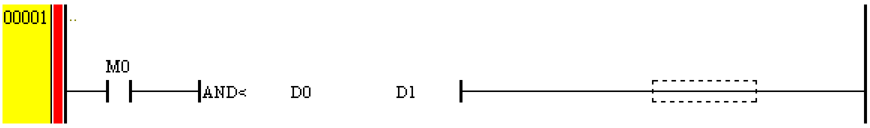
Function S1 < S2 Comparison contact is ON when S1 < S2 is true

s1 D Device Number 0 Index

s2 D Device Number 1 Index

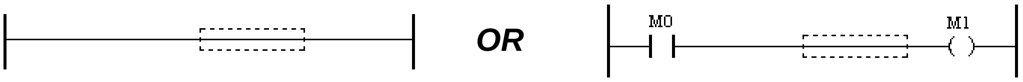
OP	K	H	F	KnX	KnY	KnM	KnS	D	V	Z	P	X	Y	M	S	T	C
s1	*	*		*	*	*	*	*	*							*	*
s2	*	*		*	*	*	*	*	*							*	*

OP	Device Explanation
s1	Data source device 1



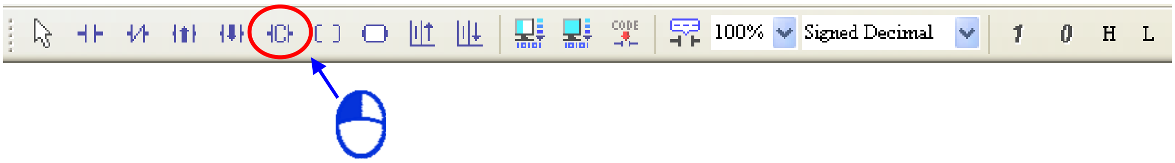
- Method2: PMSoft toolbar

(1) The users can insert a comparison contact into a line, or connect a comparison contact to a contact in series/in parallel.

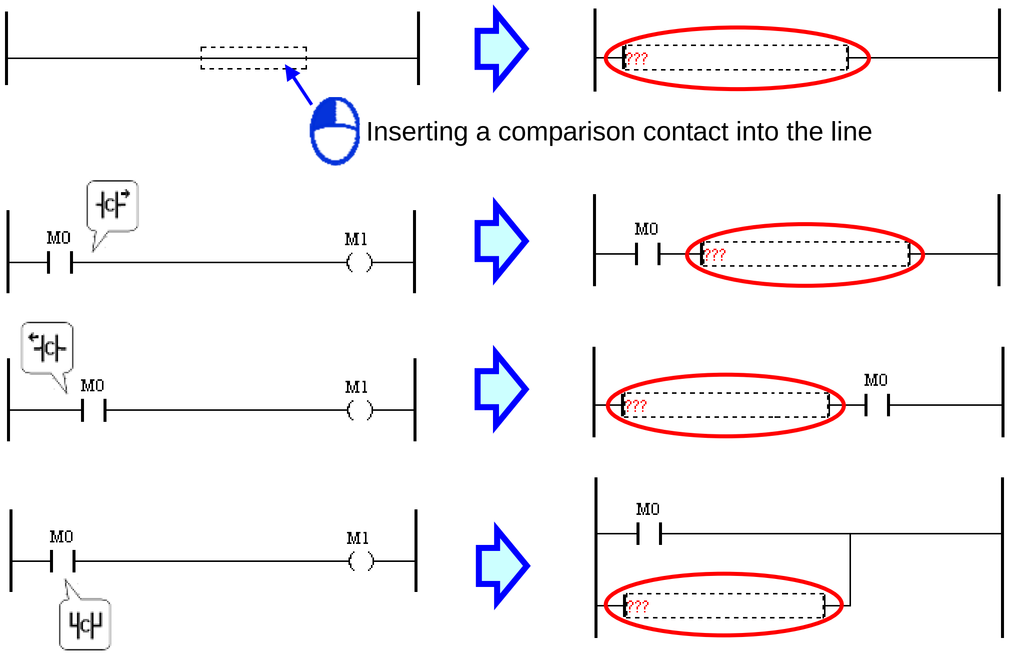


Inserting a comparison contact into a line, or connecting a comparison contact to a contact in series/in parallel

(2) The users have to click on the toolbar.

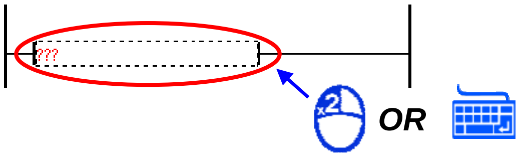


(3) The users have to move the mouse cursor to a position. The mouse cursor becomes a comparison contact when it is at the right side of the position, at the left side of the position, or at the bottom of the position. After the users click the left mouse button, a contact will be inserted. Please refer to the figures below.



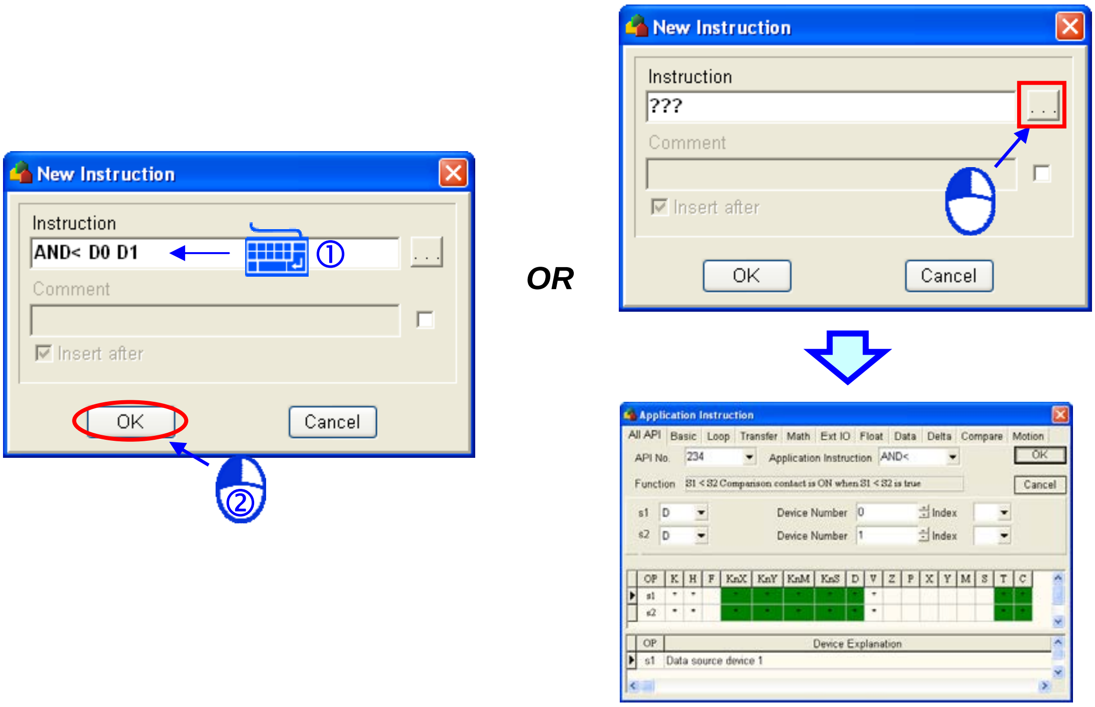
Inserting a comparison contact into the line

(4) The users have to double-click the comparison contact inserted, or press Enter on the keyboard after they click the comparison contact inserted.



OR Press Enter on the keyboard.

(5) The users have to type an instruction and operands in the **Instruction** box, or click ... . If the users click ... , they have to click the **Compare** tab, select an instruction in the **API No.** drop-down list box or the **Application Instruction** drop-down list box, select devices which are supported according the explanation at the bottom of the window, and click **OK**.



New Instruction

Instruction: AND< D0 D1

Comment

☑ Insert after

New Instruction

Instruction: ???

Comment

☑ Insert after

Application Instruction

API No. 234

Application Instruction: AND<

Function: S1 < S2 Comparison contact is ON when S1 < S2 is true

s1: D — Device Number: 0 — Index

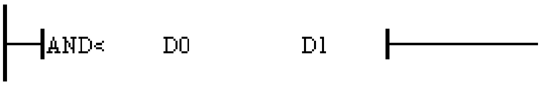
s2: D — Device Number: 1 — Index

OP	K	H	F	KnX	KnY	KnM	KnS	D	V	Z	P	X	Y	M	S	T	C
s1	*	*		*	*	*	*	*	*							*	*
s2	*	*		*	*	*	*	*	*							*	*

OP	Device Explanation
s1	Data source device 1

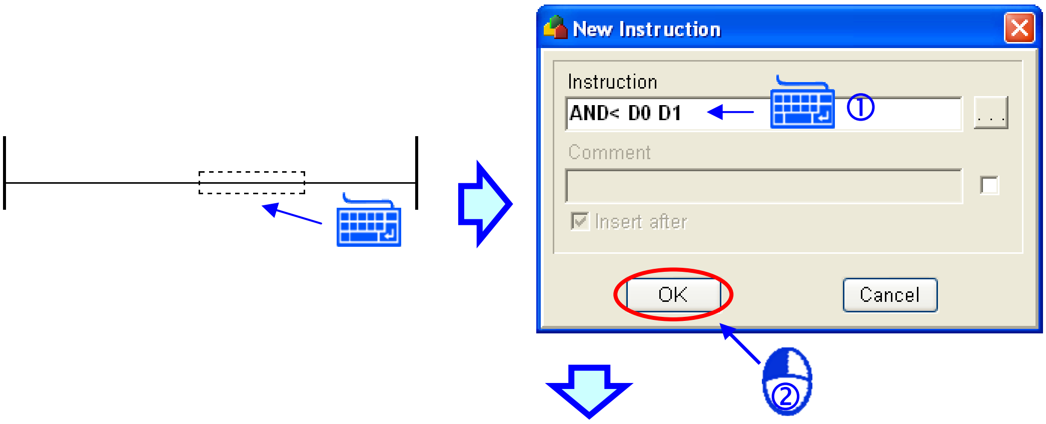
***. Please refer to section 6.2.5 for more information about making comments on devices and instructions.**

(6) The result that the users get is like the figure shown below.



- Method 3: Typing a comparison instruction and operands

After the users select a position, they have to press Enter on the keyboard, or type a comparison instruction and operands. After the users press Enter on the keyboard, or type a comparison instruction and operands, the **New Instruction** window will appear. After the users type a comparison instruction and operands in the **Instruction** box, they have to press Enter on the keyboard, or click **OK**. (The instruction that the users type is case-insensitive. If the users type an incorrect applied instruction, or an incorrect device address, an error message will appear after they click **OK**.)

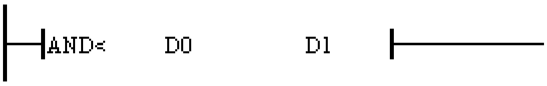


New Instruction

Instruction: AND< D0 D1

Comment

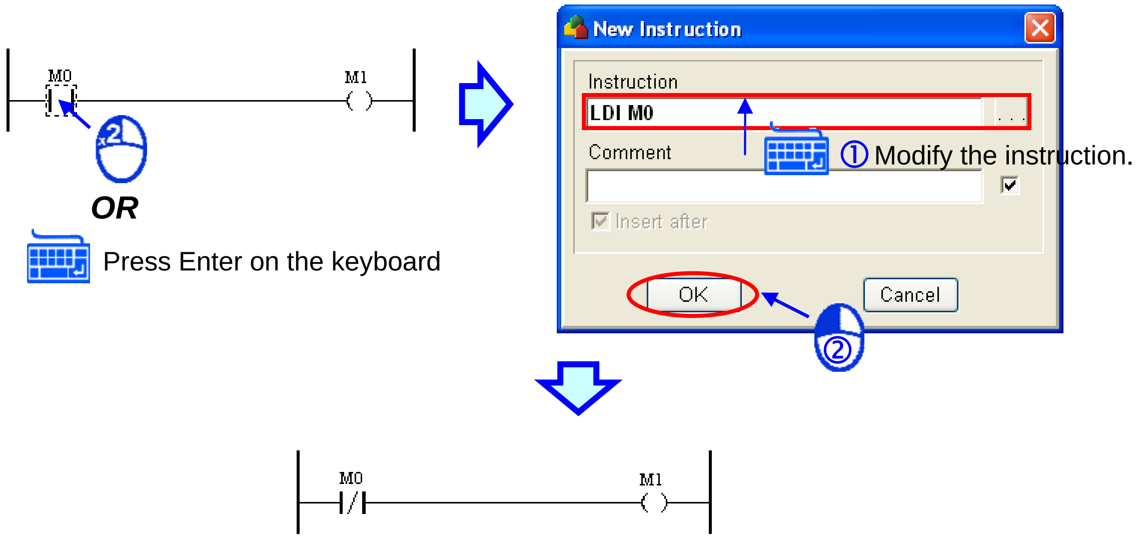
☑ Insert after



***. Please refer to section 6.2.5 for more information about making comments on devices and instructions.**

6.2.10 Modifying Objects

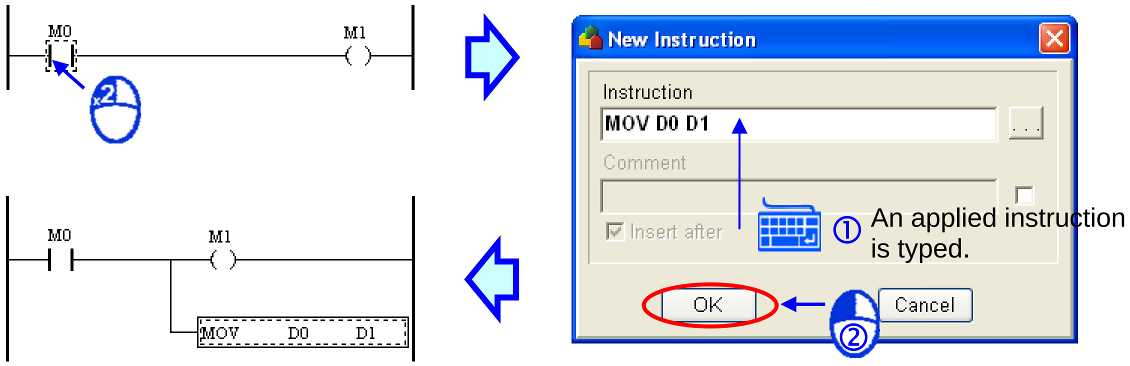
Users can modify the objects in a ladder diagram, including contacts, coils, and instructions. If the users want to change an object in a ladder diagram, they have to double-click the object, or press Enter on the keyboard after they click the object, modify the instruction in the **New Instruction** window, and click **OK**.



Instruction: LDI M0

***. Please refer to section 6.2.5 for more information about making comments on devices and instructions.**

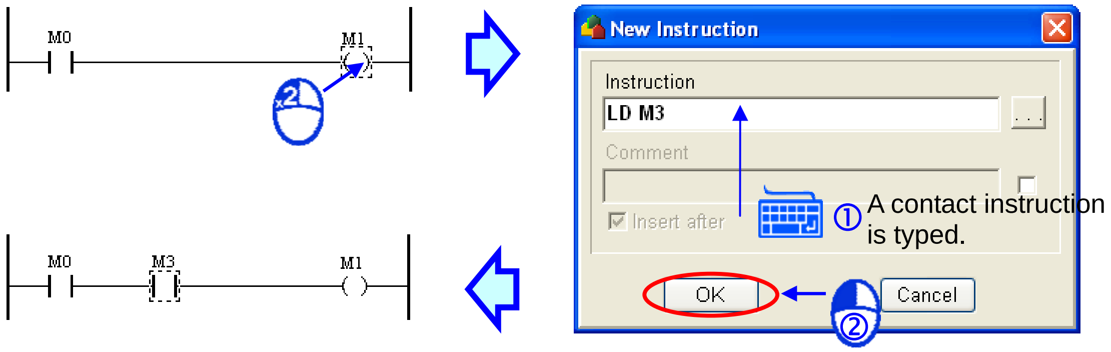
If the users type an applied instruction, a motion instruction or a G-code in the **New Instruction** window after they double-click an input object in a network, the new object created will be put under the output object in the network. If the users type a comparison instruction or a contact instruction in the **New Instruction** window after they double-click the output object in a network, the new object created will be put at the right side of the input objects in the network.



Instruction: MOV D0 D1

① An applied instruction is typed.

OR



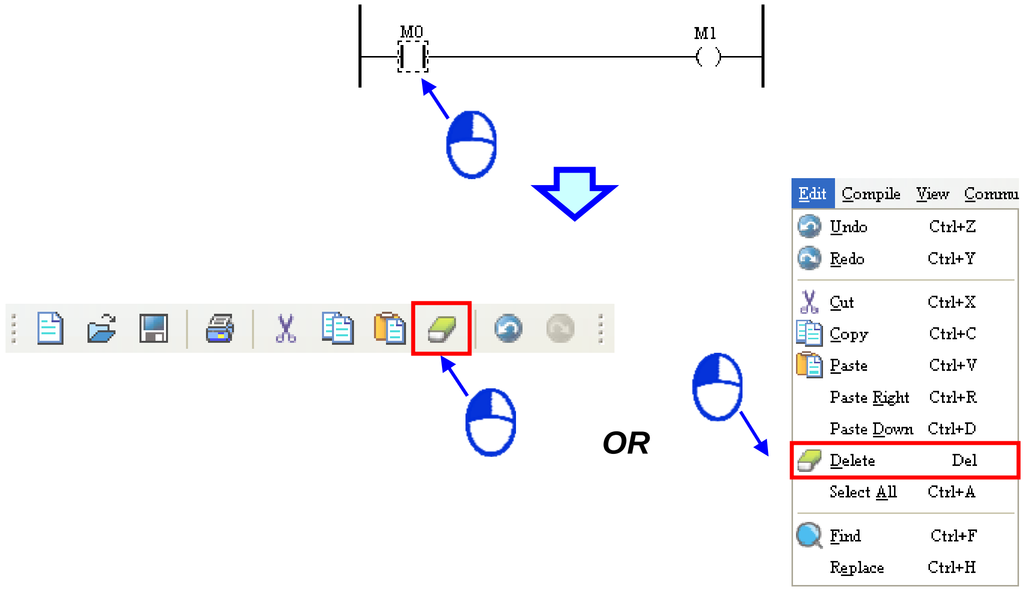
Instruction: LD M3

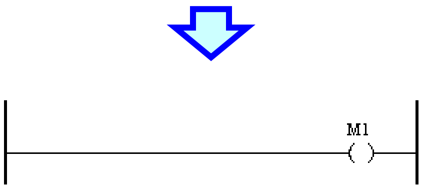
① A contact instruction is typed.

6.2.11 Deleting Objects

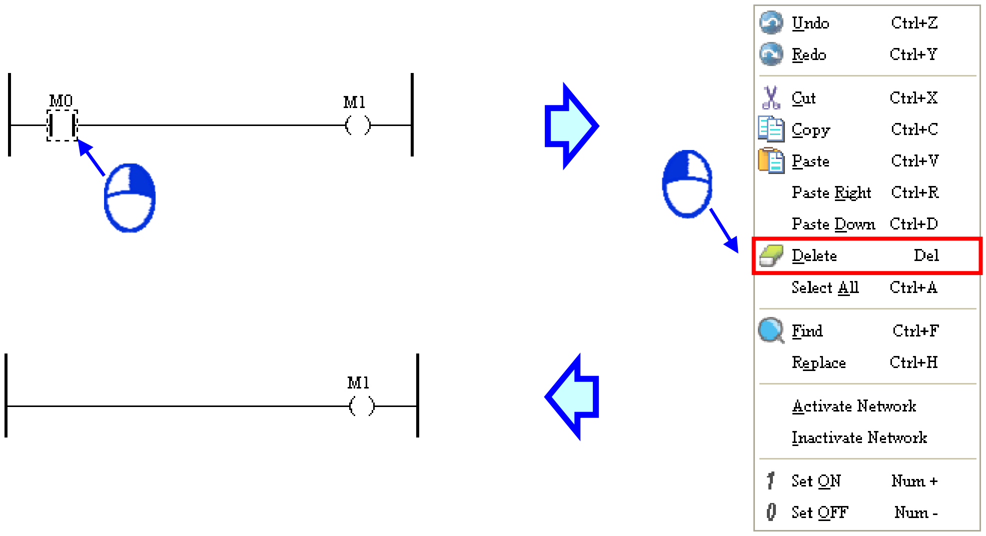
Users can delete the objects in a ladder diagram, including contacts, coils, applied instructions, and blocks. There are three methods of deleting an object in a ladder diagram.

(1) The users have to click an object, and click **Delete** on the **Edit** menu, or 🧽 on the standard toolbar.

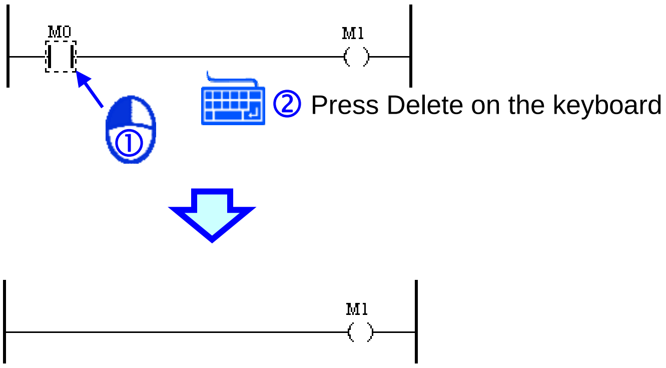




(2) The users have to click an object, right-click the object, and click **Delete** on the context menu.



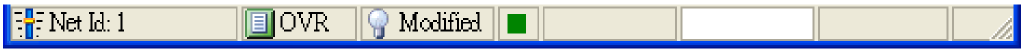
(3) The users have to click an object, and press Delete on the keyboard.



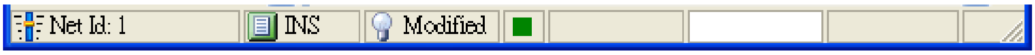
6.2.12 Instruction Editing Mode

In a ladder diagram in PMSoft, users can edit devices, applied instructions, motion instructions, G-codes, and function blocks by means of typing IL instructions. Before the users use the instruction editing mode, they have to make sure that the mode displayed in the status bar is the the **OVR** (repalcement) mode, or the **INS** (insertion) mode. The users can switch between the two modes by pressing Insert on the keyboard.

OVR mode: Users can edit the object selected by means of typing an instruction. The new object created will replace the original object.

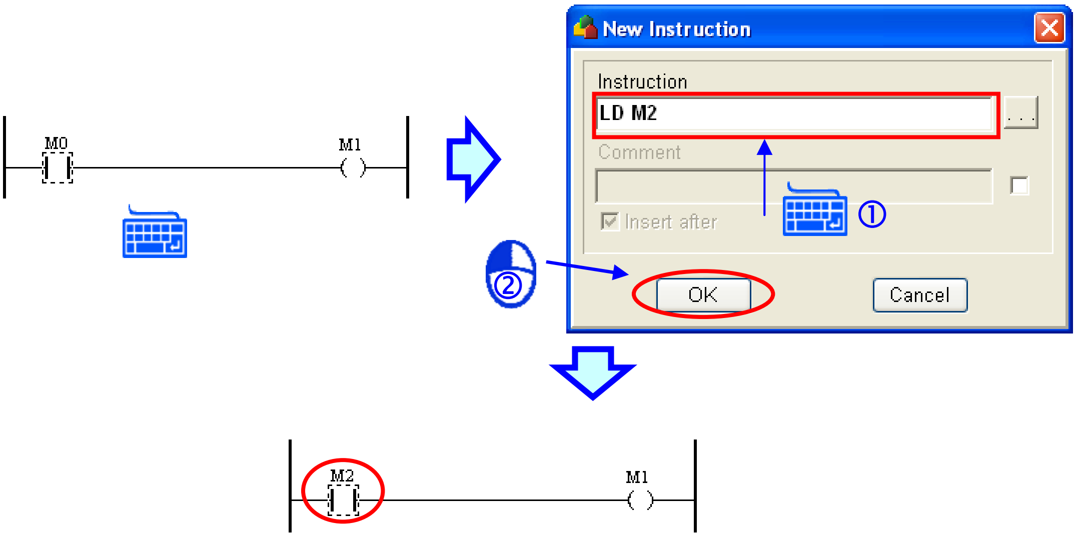


INS mode: Users can edit the object selected by means of typing an instruction. The new object created will be put at the right side of the original object.



The method of using the **OVR** mode/the **INS** mode is described below.

OVR mode: If users click a contact in a network, type an instruction in the **New Instruction** window, and click **OK**, the original contact will be replaced by a new contact.



Instruction: LD M2

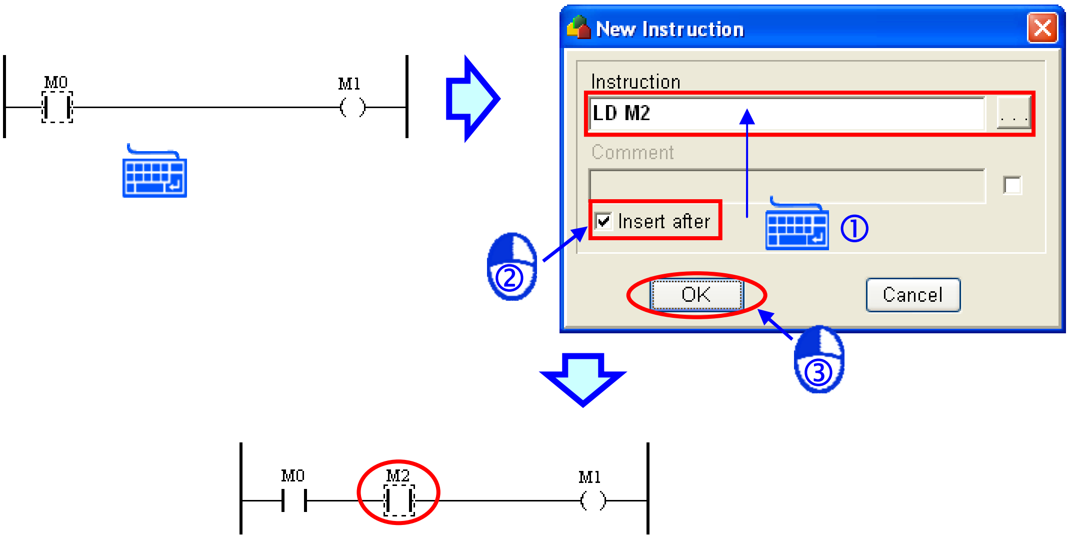
Comment:

☑ Insert after

OK Cancel

*. Please refer to section 6.2.5 for more information about making comments on devices and instructions.

INS mode: If users click a contact in a network, type an instruction in the **New Instruction** window, and select the **Insert after** checkbox, the new object inserted will be put at the right side of the original contact. If the **Insert after** checkbox is not selected, the new object inserted will be put at the left side of the original contact. Likewise, if the users click a coil in a network, type an instruction in the **New Instruction** window, and select the **Insert after** checkbox, the new object inserted will be put under the original coil. If the **Insert after** checkbox is not selected, the new object inserted will be put above the original coil.

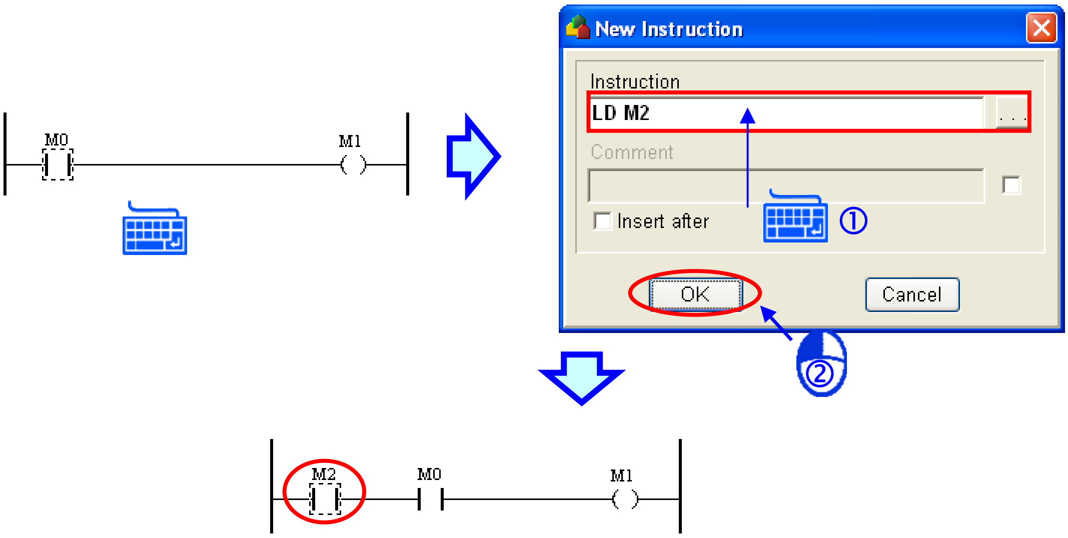


Instruction: LD M2

Comment:

☑ Insert after

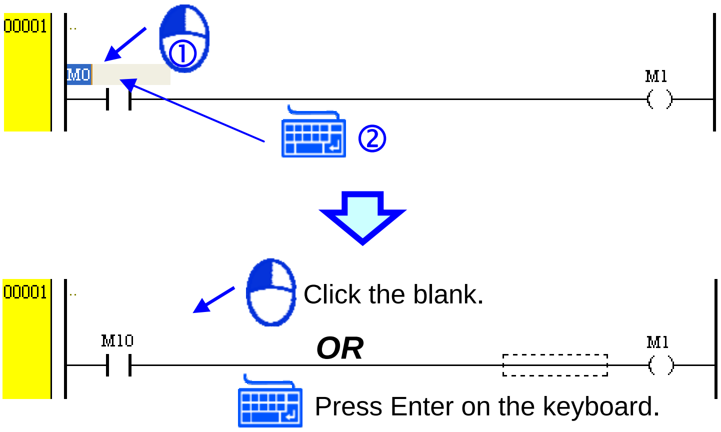
OK Cancel



*. Please refer to section 6.2.5 for more information about making comments on devices and instructions.

6.2.13 Editing Devices or Symbols

If users click a device name or a symbol in a network, they can modify the device name or the symbol. After the users modify the device name or the identifier in the network, they have to click the blank in the network, or press Enter on the keyboard. Please refer to chapter 4 for more information about symbols.



Click the blank.

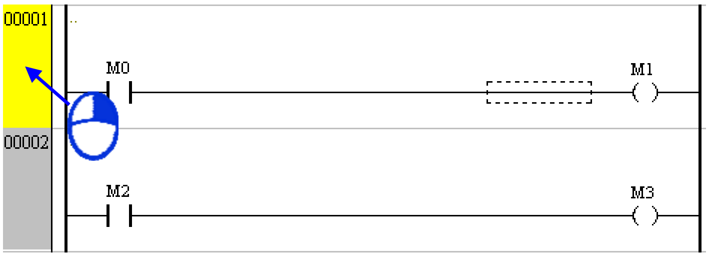
OR

Press Enter on the keyboard.

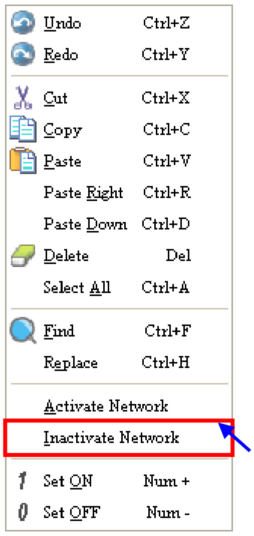
6.2.14 Activating / Inactivating a Network

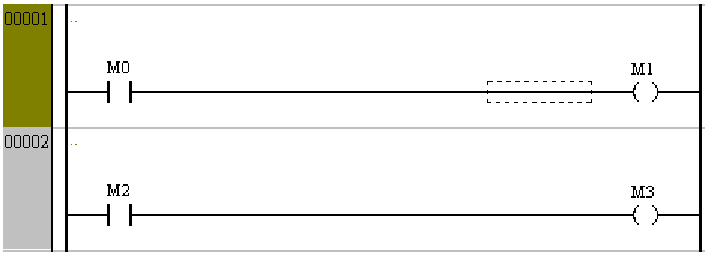
If a network in a ladder diagram is inactivated, the compiling of the ladder diagram will skip the network. Users can temporarily inactivate some parts of a program.

(1) The users have to right-click a network number in a ladder diagram.

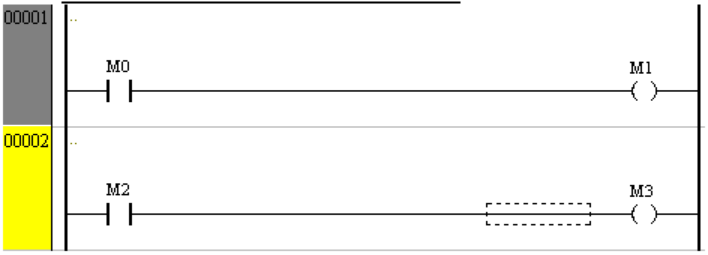


(2) After the users click **Inactivate Network** on the context menu, the network number will be in a dark color.





When the network which is inactivated is selected, the network number is in a dark color.

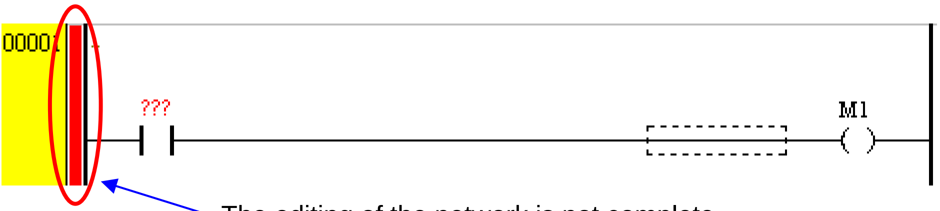


When the network which is inactivated is not selected, the network number is not in a dark color.

(3) If the users want to activate the network, they have to right-click the network, and click **Activate Network** on the context menu.

6.2.15 Error Hint

The completeness of the networks in a ladder diagram in PMSoft is checked automatically. If users do not assign a device to the contact in the figure below, a red bar will appear at the right side of the busbar. The red bar will not disappear until the editing of the network is complete. In a network, the output object of a motion instruction, the output object of a M-code, and the output object of a G-code are allowed to appear independently. However, other instructions in a network must contain output objects and input objects.



The editing of the network is not complete.

MEMO

Chapter 7 Instruction List

Table of Contents

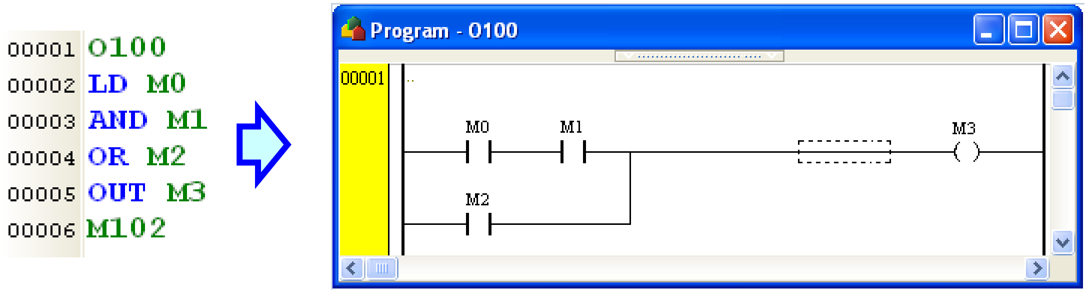
7.1 Introduction of an Instruction List

7.1.1 Instruction List

An instruction list resembles an assembly language. In earlier times, users entered control instructions into a PLC through a programming panel, and the programming language used was an instruction list. Nowadays, although the function of PLC development software is getting stronger, and there are many convenient programming languages available, the high compatibility of an instruction list can not be replaced completely. As a result, an instruction list is still one of the programming languages supported by the IEC 61131-3 standard.

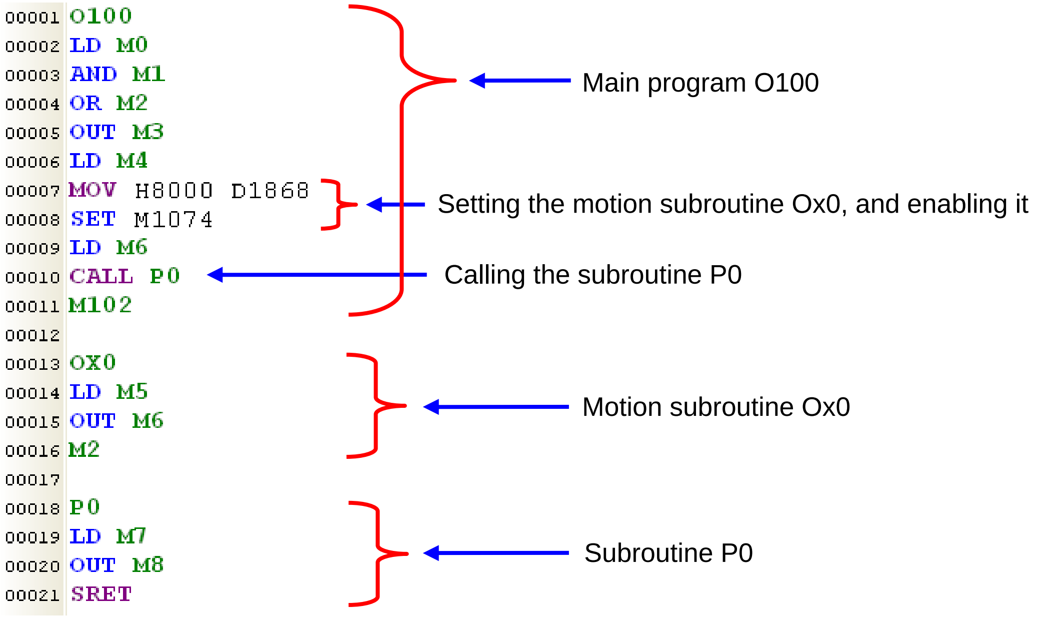
7.1.2 Structure of an Instruction List

An example of an instruction list is shown below. An instruction list is composed of statements. Every statement represents an action.



```
O100
LD  M0
AND  M1
OR  M2
OUT  M3
M102
```

A program is scanned cyclically. The scan of the main program O100 starts from the starting flag O100. After the ending instruction M102 is scanned, the scan of the main program O100 will go back to the starting flag O100. If the main program O100 includes a Ox motion subroutine or a P subroutine, the Ox motion subroutine or the P subroutine will follow the main program O100, and will be called by the main program O100. Please refer to DVP-PM Application Manual for more information.



```
O100
LD  M0
AND  M1
OR  M2
OUT  M3
LD  M4
MOV  H8000  D1868
SET  M1074
LD  M6
CALL  P0
M102

OX0
LD  M5
OUT  M6
M2

P0
LD  M7
OUT  M8
SRET
```

Main program O100

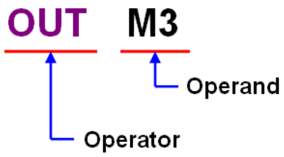
Setting the motion subroutine Ox0, and enabling it

Calling the subroutine P0

Motion subroutine Ox0

Subroutine P0

Every statement on an instruction list is composed of an operator and operands. An operator is a function representing an operation, and an operand is the object of an operation. An operand can be a device, a symbol, or a constant. In the example below, OUT is an operator outputting a state, and M3 is an operand which acts an output object.



IL instructions can be classified into basic instructions and applied instructions. Please refer to DVP-PM Application Manual for more information about the usage of IL instructions.

7.1.3 Important Points about Creating an Instruction List

- As long as the format of an instruction is not destroyed, leaving spaces is allowed.
- An instruction list is case-insensitive. That is, the words OUT, Out, and out have the same meaning.
- If users want to use constants in a program created by means of an instruction list in PMSoft, the constants must be represented in the following ways.
  - Decimal value: 2345
  - Hexadecimal value: 16#5BA0
  - Floating-point number: 4.123

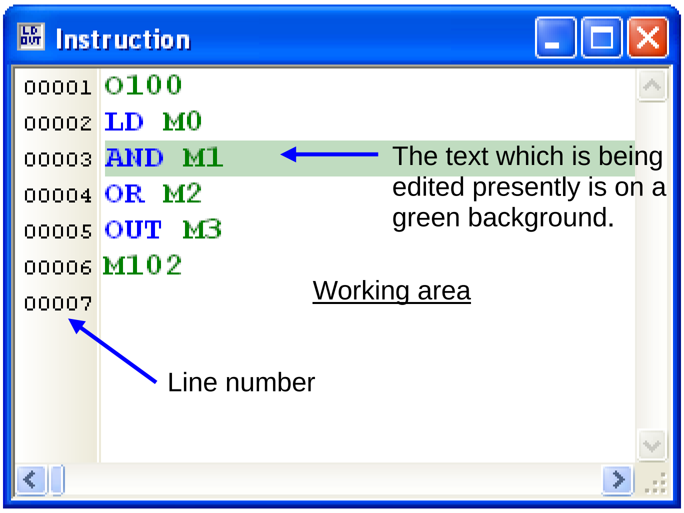
  ***. A instruction list still supports the use of K and the use of H.**
- There is no limit on the number of lines which can be created, but users still have to consider whether the size of the program compiled exceeds the capacity of the memory in the motion controller used.
- The sections of an instruction list in PMSoft can be copied/cut/pasted. Users can copy the text in a file edited with a text editor into an instruction list in PMSoft.

7.2 Editing Environment

7.2.1 Introduction of the Editing Environment

The environment in which an instruction list can be edited is shown below. There are line numbers at the left side of the working area. The text which is being edited presently is on a green background. The use of the environment in which an instruction list can be created is the same as the use of a general text editor. Users type text or modify text in the working area.

- The **Instruction** window is shown below.

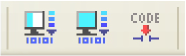

Instruction
The text which is being edited presently is on a green background.
Working area
Line number

```
00001 O100
00002 LD  M0
00003 AND  M1
00004 OR  M2
00005 OUT  M3
00006 M102
00007
```

7.2.2 Toolbar

After a program editing window in which a ladder diagram can be created is opened, a toolbar will appear in the PMSoft window. The functions of the toolbar are described below. After an instruction list in a project is created, users have to click **IL to LD** on the PMSoft toolbar. After the instruction list is transformed into a ladder diagram, the users can click **Check** or **Compile Program** on the PMSoft toolbar. If the users click **Check** or **Compile Program** on the PMSoft toolbar without clicking **IL to LD** on the PMSoft toolbar first, the ladder diagrams (rather than the instruction list) created in the project will be compiled, and the IL code created will overwrite the instruction list in the **Instruction** window.

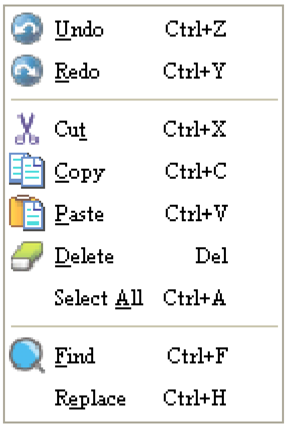
- PMSoft toolbar



Icon	Keyboard shortcut	Function
	Ctrl+F10	Checking the program (The instruction list created must be transformed into a ladder diagram first.)
	Ctrl+F7	Compiling the program (The instruction list created must be transformed into a ladder diagram first.)
CODE	None	Transforming the instruction list into a ladder diagram

7.2.3 Context Menu

After users right-click the working area in a program editing window, a context menu will appear. The functions of the items on the context menu are described below.

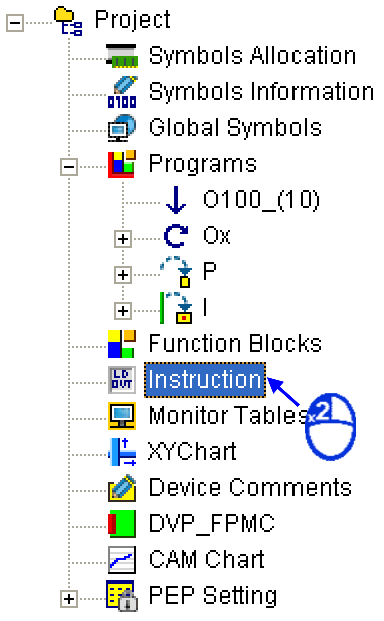


Item	Function
Undo	Undoing the last action (The number of previous actions that can be undone is 20.)
Redo	Redoing an action which has been undone
Cut	Cutting the text block selected
Copy	Copying the text block selected
Paste	Pasting the text block which has been copied or cut on the present position
Delete	Deleting the text block selected
Select All	Selecting all the text in the working area
Find	Searching for an item (Please refer to chapter 9 for more information.)
Replace	Replacing the item found with another item (Please refer to chapter 9 for more information.)

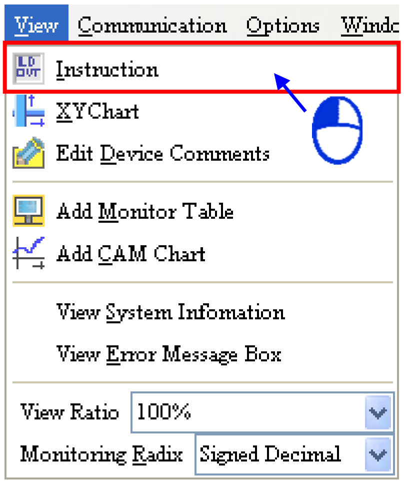
7.3 Creating an Instruction List in PMSoft

7.3.1 Opening the Instruction Window

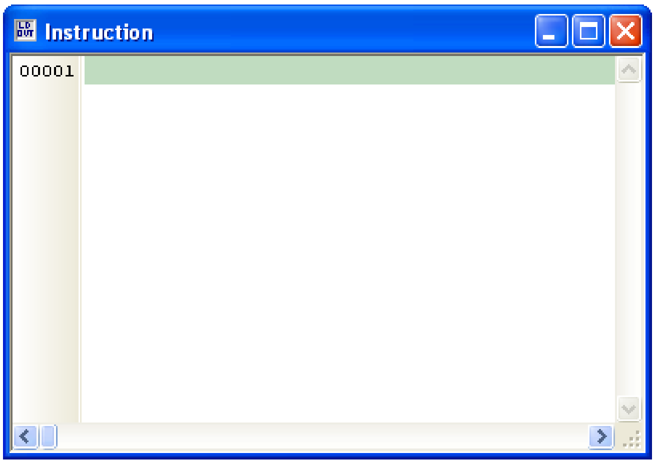
After users double-click **Instruction** in the system information area, or click **Instruction** on the **View** menu, the **Instruction** window will be opened.



OR

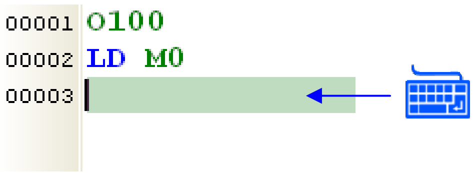


View Ratio 100%
Monitoring Radix Signed Decimal



7.3.2 Editing IL Instructions

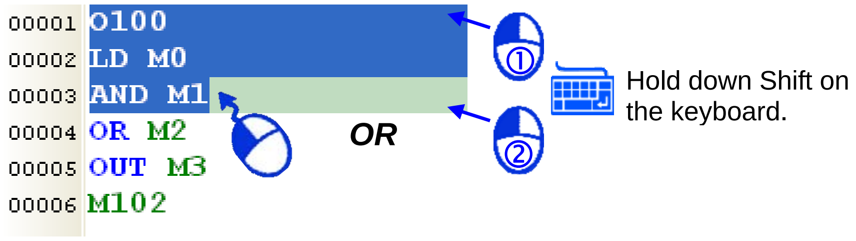
(1) The text which is being edited presently is on a green background. Users have to click a line, and type text. If the users want to add a new line, they can press Enter on the keyboard.



```
00001 O100
00002 LD  M0
00003
```

(2) If users want to select any amount of text, they can click where they want to begin the selection, hold down the left mouse button, and drag the pointer over the text that they want to select. The

users can also click at the start of the selection, scroll to the end of the selection, and hold down Shift on the keyboard while they click where they want the selection to end. The text block selected can be cut, copied, or pasted.



```
00001 O100
00002 LD  M0
00003 AND  M1
00004 OR  M2
00005 OUT  M3
00006 M102
```

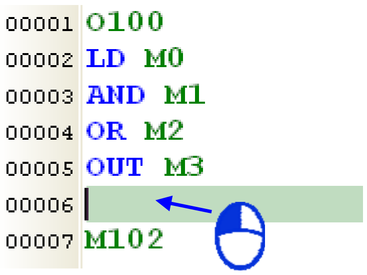
Hold down Shift on the keyboard.

7.3.3 Comparison Contacts, Applied Instructions, Motion Instructions, and G-codes

There are two methods of inserting a comparison contact, an applied instruction, or a motion instruction. Users can use one of the methods according to their habit. Owing to the fact that **Instruction Wizard** does not support G-codes, the users can only use method 2 if they want to insert a G-code.

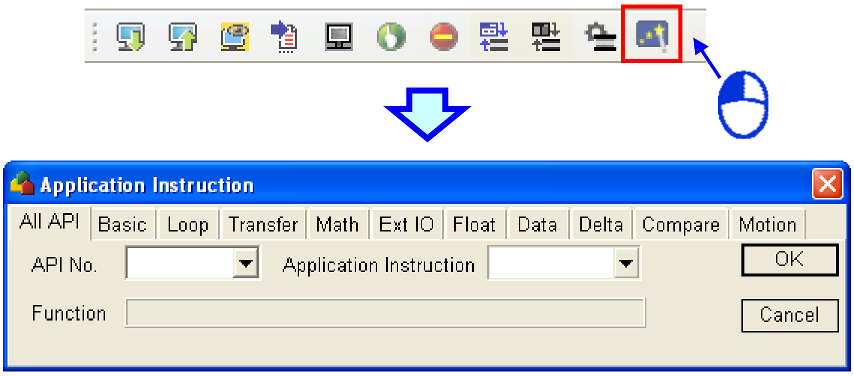
- Method 1: **Instruction Wizard**

(1) The users have to click a line into which they want to insert an instruction in the working area.



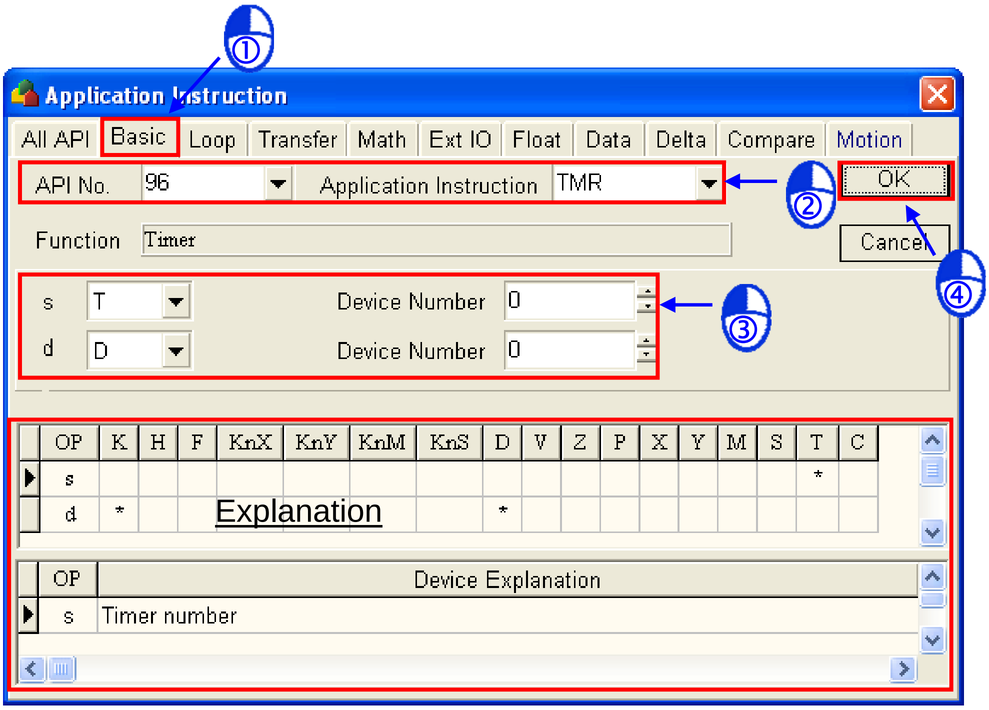
```
00001 O100
00002 LD  M0
00003 AND  M1
00004 OR  M2
00005 OUT  M3
00006 |
00007 M102
```

(2) After the users click on the fast toolbar, the **Application Instruction** window will appear.



Application Instruction

(3) The users have to select an instruction type, select an instruction in the **API No.** drop-down list box or the **Application Instruction** drop-down list box, select devices which are supported according the explanation at the bottom of the window, and click **OK**.



API No. 96 — Application Instruction TMR

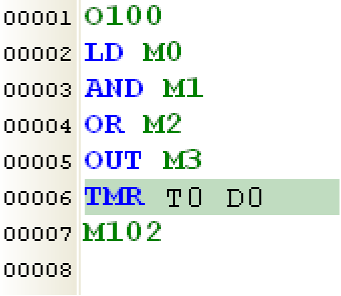
Function: Timer

s: T — Device Number: 0

d: D — Device Number: 0

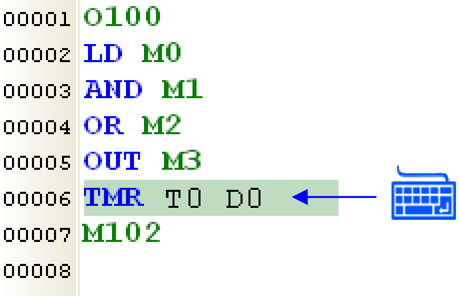
OP	K	H	F	KnX	KnY	KnM	KnS	D	V	Z	P	X	Y	M	S	T	C
s																*	
d	*							*									

OP	Device Explanation
s	Timer number



```
O100
LD  M0
AND  M1
OR  M2
OUT  M3
TMR  T0  D0
M102
```

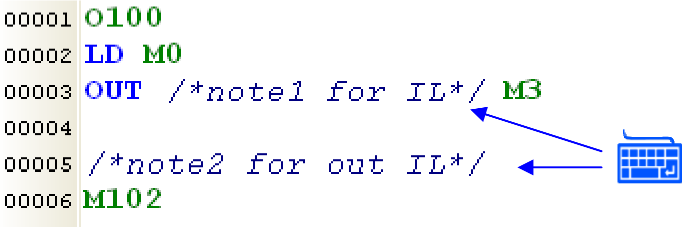
- Method 2: Typing an instruction
  Users have to type an instruction in the working area according to the format of the instruction.



```
O100
LD  M0
AND  M1
OR  M2
OUT  M3
TMR  T0  D0
M102
```

7.3.4 Comments on IL Instructions

The text between /* and */ in an instruction list is regarded as a comment. No matter what instruction or keyword the text includes, the compiling of the instruction list automatically skips the text. Besides, a comment can be added to any position as long as the formats of the instructions are not destroyed. However, the users have to make sure that the program created is readable. The comment shown in the figure below is a legal comment.

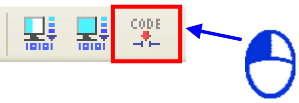


```
00001 O100
00002 LD M0
00003 OUT /*note1 for IL*/ M3
00004
00005 /*note2 for out IL*/
00006 M102
```

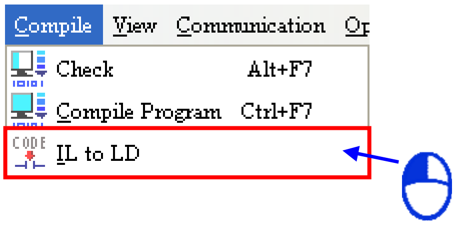
7.4 Transforming an Instruction List into a Ladder Diagram

PMSoft provides a function of transforming an instruction list into a ladder diagram. After an instruction list is created, it can be rapidly transformed into a ladder diagram. However, instruction lists do not support symbols and function blocks. Suppose a program code created is a ladder diagram including symbols and function blocks. Owing to the fact that the instruction list which will be created after the program code is compiled do not support the symbols and the function blocks, the symbols will be replaced by devices assigned by the system after the instruction list is transformed into a ladder diagram, and the function blocks will be regarded as P subroutines.

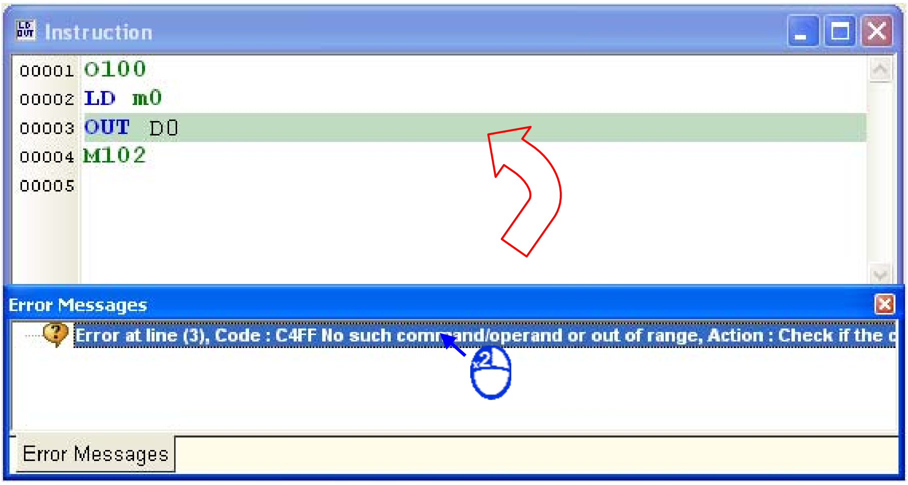
If users click **IL to LD** on the PMSoft toolbar or on the **Compile** menu after they create an instruction list, the instruction list will be transformed into a ladder diagram, and the ladder diagram will be compiled.



OR



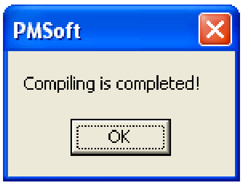
If there is an error in the program code in the **Instruction** window, an error message will appear. The users will be led to the incorrect program after they double-click the error message.



```
00001 O100
00002 LD m0
00003 OUT D0
00004 M102
00005
```

Error at line (3), Code : C4FF No such command/operand or out of range, Action : Check if the c

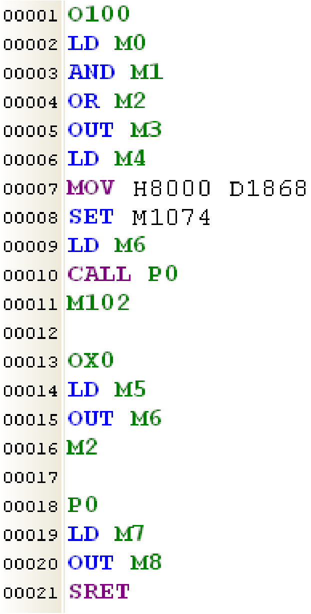
If the users click **IL to LD** on the PMSoft toolbar or on the **Compile** menu after they modify the program, the **PMSoft** window shown below will appear.



PMSoft

Compiling is completed!

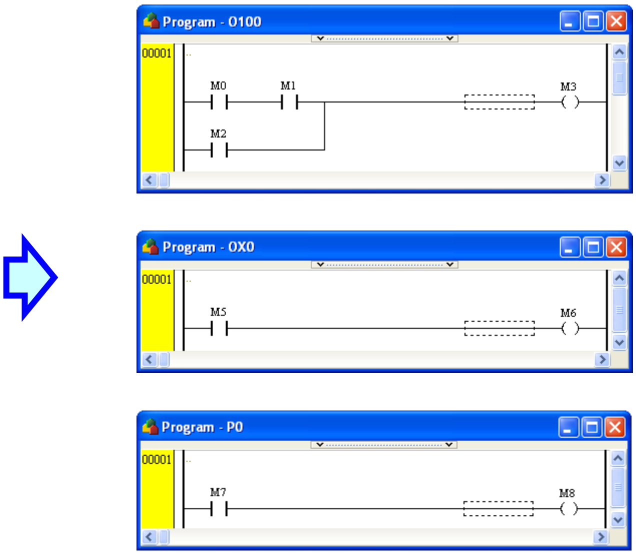
After the users click **OK** in the **PMSoft** window shown above, the system will create the ladder diagram corresponding to the instruction list.



```
O100
LD  M0
AND  M1
OR  M2
OUT  M3
LD  M4
MOV  H8000  D1868
SET  M1074
LD  M6
CALL  P0
M102

OX0
LD  M5
OUT  M6
M2

P0
LD  M7
OUT  M8
SRET
```



MEMO

Chapter 8 G-codes and Electronic Cams

Table of Contents

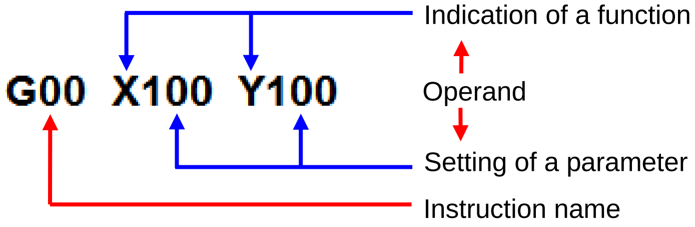
8.1 Introduction of G-Codes

8.1.1 G-Codes

During the control of motion, users can generate G-codes for the complex processing paths of two axes (or more than two axes) by means of computer-aided manufacturing software, and the G-codes can be input to a controller which can drive the axes. The time of developing a motion program can be saved. G-code is a CNC (computer numerical control) programming language widely used in automatic equipment. Delta motion controllers and PMSoft support the majority of G-codes on the market. The minority of G-codes which are not supported are skipped, and are not executed.

8.1.2 Structure of a G-code

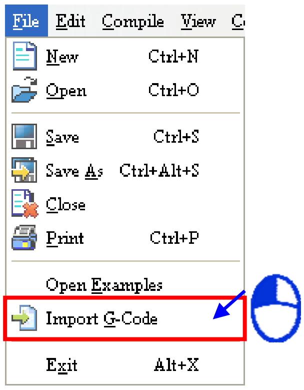
A G-code is composed of an instruction name and operands, and an operand is composed of the indication of a function and the setting of a parameter. In the figure below, the instruction name represents the function which is executed, the indication of functions represents the targets of the operands, and the setting of the parameters represents the values of the operands. The instruction G00 indicates that the x-axis and the y-axis are moved at the maximum speed to the target position (100, 100).

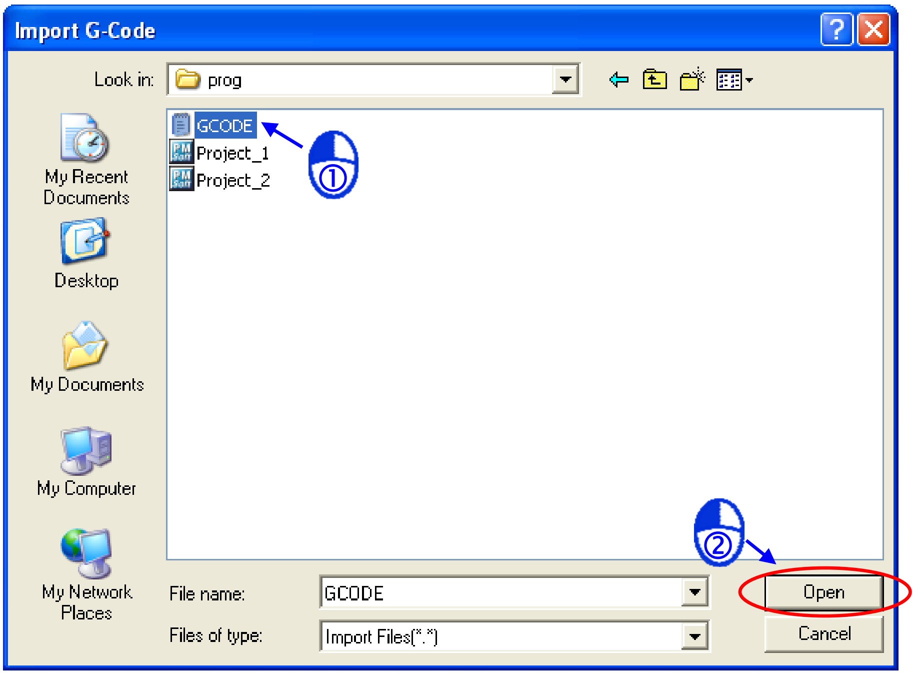


Please refer to DVP-PM Application Manual for more information about G-codes.

8.1.3 Importing G-codes

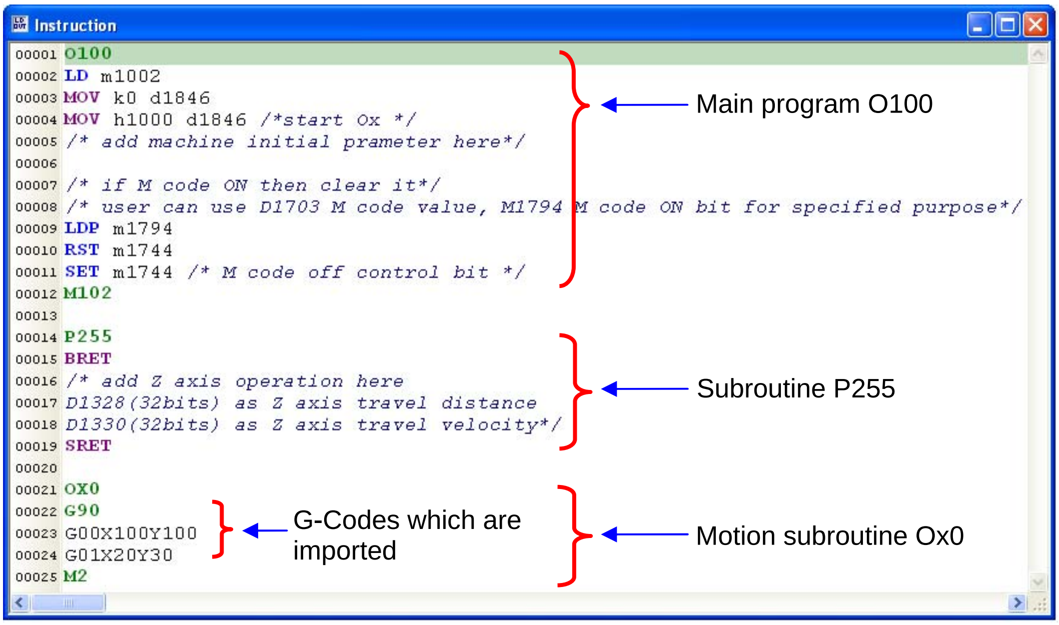
PMSoft allows users to import G-codes into the **Instruction** window. The users have to click **Import G-Code** on the **File** menu, click a .txt file or a .nc file in the **Import G-Code** window, and click **Open**.





After the file selected is imported, the instruction list in the **Instruction** window will be composed of the main program O100, the subroutine P255, and the motion subroutine Ox0. In the figure below, the main program O100 is composed of the program code which enables a Ox motion subroutine and the program code related to flags, the subroutine P255 is a blank program. The users can add program code to the main program O100, the subroutine P255, and the motion subroutine Ox0. Besides, the system removes the comments, the line numbers, and the G-codes which are not supported in the G-codes imported, and then put the G-codes in the motion subroutine Ox0. Please refer to chapter 7 for more information about the **Instruction** window.

***. If the z-axis is used, the subroutine P255 will be used to control the motion of the z-axis.**



```
O100
LD m1002
MOV k0 d1846
MOV h1000 d1846 /*start Ox */
/* add machine initial prameter here*/

/* if M code ON then clear it*/
/* user can use D1703 M code value, M1794 M code ON bit for specified purpose*/
LDP m1794
RST m1744
SET m1744 /* M code off control bit */
M102

P255
BRET
/* add Z axis operation here
D1328(32bits) as Z axis travel distance
D1330(32bits) as Z axis travel velocity*/
SRET

OX0
G90
G00X100Y100
G01X20Y30
M2
```

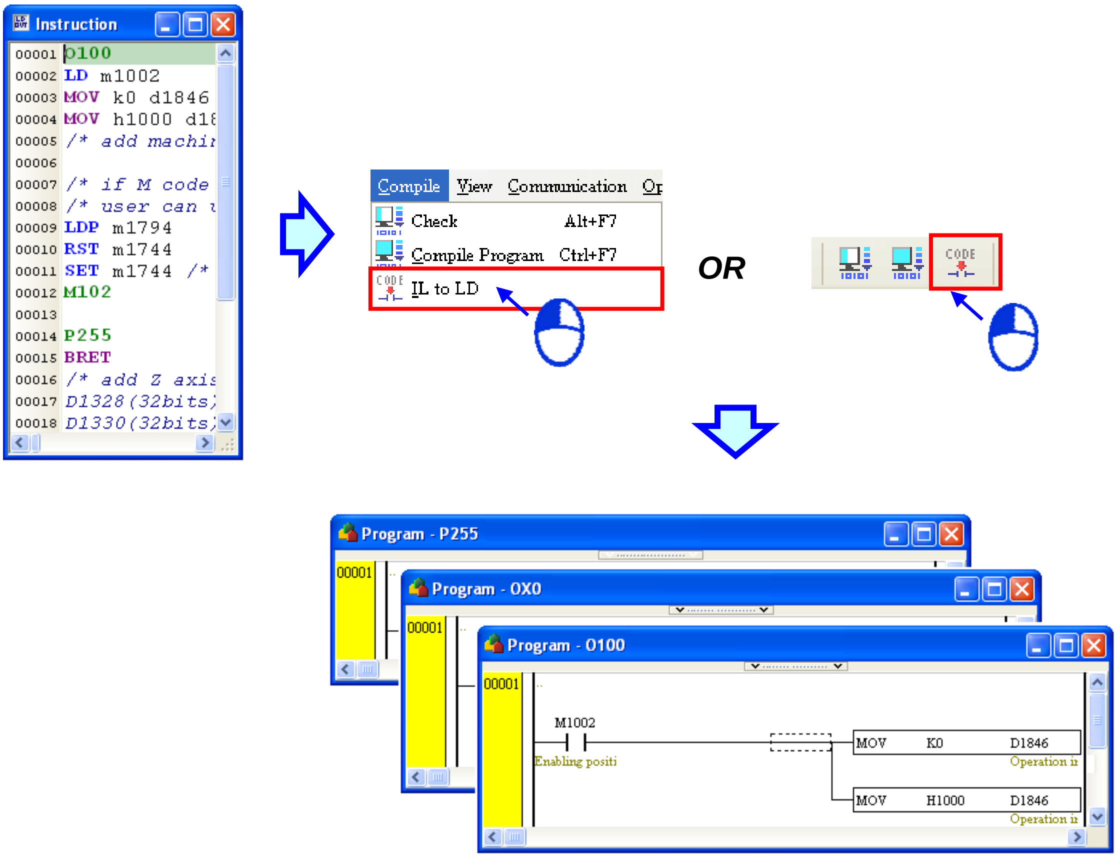
Main program O100

Subroutine P255

G-Codes which are imported

Motion subroutine Ox0

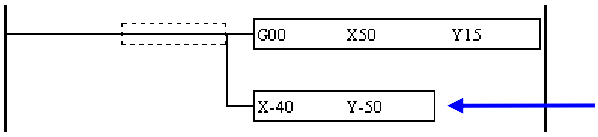
The users can click **IL to LD** on the **Compil**e menu or on the PMSoft toolbar to transform the instruction list into ladder diagrams. Please refer to chapter 6 for more information about editing a ladder diagram.



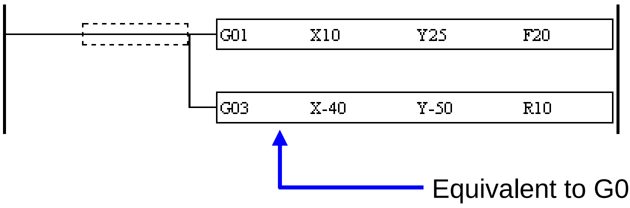
8.1.4 Important Points about G-codes

- As long as the format of an instruction is not destroyed, leaving spaces is allowed. For example, G01X50   Y15F20 is legal, and is equivalent to G01 X50 Y15 F20. G01X50Y15F   20 is illegal because there are spaces between F and 20.

- Users can put several instruction names in a line, e.g. **G91G01**X10Y30F50**G04**X4.5.
- If instructions of the same type are in a line, the last instruction is given preference. For example, **G02G00G01** X10 Y30 F50 is equivalent to **G01** X10 Y30 F50 because G02, G00, and G01 are movement instructions.
- A G-code can be directly connected to a busbar. It does not need to be connected to a condition contact.
- If there is a decimal point in a coordinate or a speed value, the decimal point will be regarded as multiplication by 1000. For example, G01X100Y-125.5F200.0 is equivalent to G01X100Y-125500 F200000.
- G00/G01 can be extended. The speed parameter (F) of G01/G02/G03 can be extended.



Equivalent to G00 X-40 Y-50



Equivalent to G03 X-40 Y-50 R10 F20

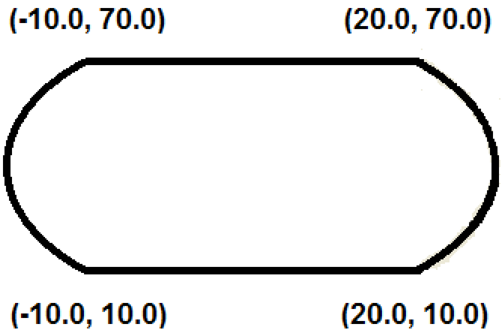
- A G-code can only be put in a Ox motion subroutine, or in the P subroutine called by a Ox motion subroutine. It can not be put in the main program O100, or the P subroutine called by the main program O100.
- Please refer to DVP-PM Application Manual for more information about G-codes.

8.2 Example

8.2.1 Drawing a Path by Means of G-codes

- Program requirement

There are four pairs of absolute coordinates (-10000, 10000), (20000, 10000), (20000, 70000), and (-10000, 70000). They are signed distances from the origin (0, 0). The left path and the right path are arcs. The four pairs of absolute coordinates are expressed as four pairs of decimals for the sake of simplification. The path shown below can be drawn on a XY plane by means of G-codes. The motion controller which is used is DVP-20PM00D



- Programming order
  (1) Writing the main program O100
  (2) Setting and enabling the motion subroutine Ox0 in O100

(3) Writing G-codes in the motion subroutine Ox0

8.2.2 Explanation

- Main program O100

  The special registers which are used are described below. Please refer to DVP-PM Application Manual for more information.

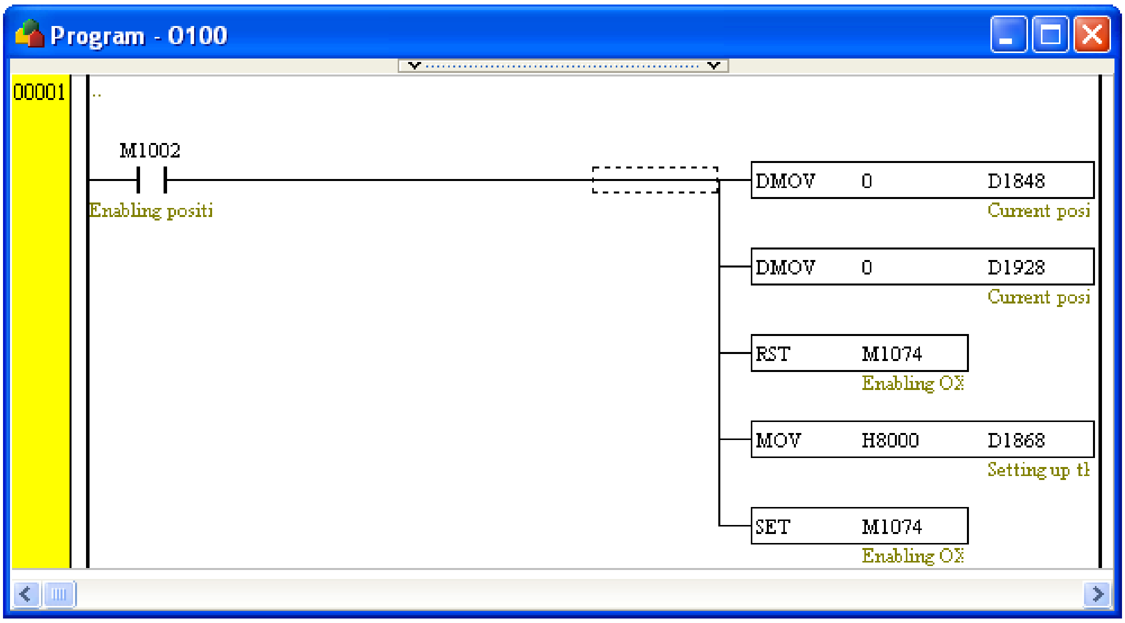
  M1002: M1002 is used for initializing setting. If the motion controller used is set to Auto, M1002 will be ON in the first scan cycle, and will be OFF thereafter.

  D1848 & D1849: The values in D1848 and D1849 represent the current position of the x-axis.

  D1928 & D1929: The value in D1928 and D1929 represent the current position of the y-axis.

  D1868: The value in D1868 is H8000. It indicates that the motion subroutine number used is Ox0.

  M1074: If M1074 is set to ON, the motion subroutine Ox0 will be enabled.



```
00001
M1002
Enabling positi
DMOV    0       D1848
                Current posi
DMOV    0       D1928
                Current posi
RST     M1074
        Enabling OX
MOV     H8000   D1868
                Setting up tl
SET     M1074
        Enabling OX
```

- Motion subroutine Ox0

  The G-codes used are described below. Please refer to DVP-PM Application Manual for more information.

  G90: The coordinates used are absolute coordinates.

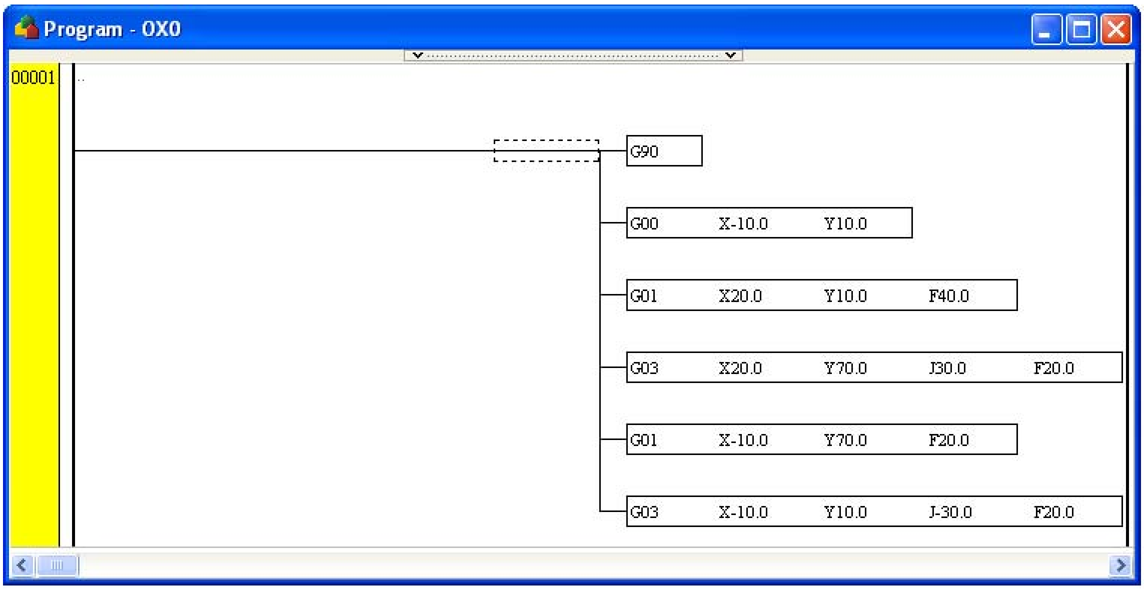
  G00: It moves each axis to the coordinates set at its max speed.

  G01: Linear interpolation

  It moves each axis to the coordinates set at the speed set.

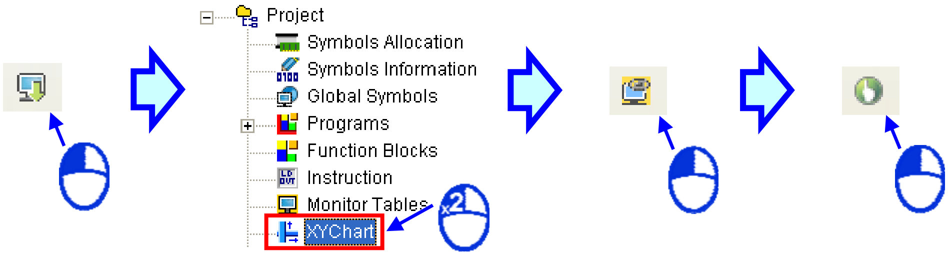
  G03: Circular interpolation

  It moves each axis to the coordinates set along an arc path by setting a center.

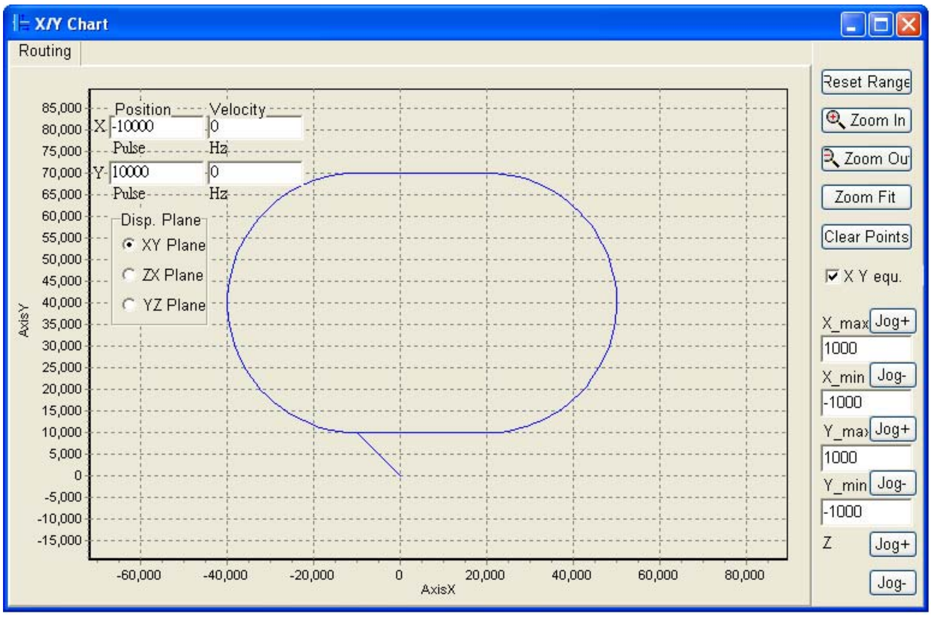


- Execution result

  Please refer to chapter 2 and chapter 11. After a motion controller or a simulator is connected to PMSoft, users have to compile the program created in PMSoft, download the program, double-click **XY Chart** in the system information area, click **Monitoring** on the toolbar, and click **Run O100** on the toolbar. After the path drawn according to the G-codes described above appears in the **X/Y Chart** window, the users can click **Zoom In** or **Zoom Out**. The path starting from the origin (0, 0) goes through the four pairs of coordinates mentioned above.



***. Please refer to section 11.3.6 for more information about the X/Y Chart window.**



X/Y Chart

Routing

Position X: -10000 Pulse, Velocity: 0 Hz

Position Y: 10000 Pulse, Velocity: 0 Hz

Disp. Plane: ● XY Plane ○ ZX Plane ○ YZ Plane

☑ X Y equ.

X_max: 1000

X_min: -1000

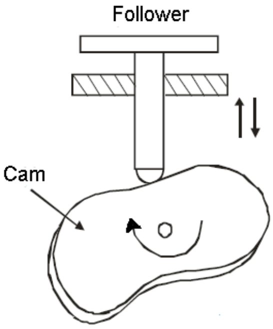
Y_max: 1000

Y_min: -1000

8.3 Introduction of Electronic Cams

8.3.1 Electronic Cams

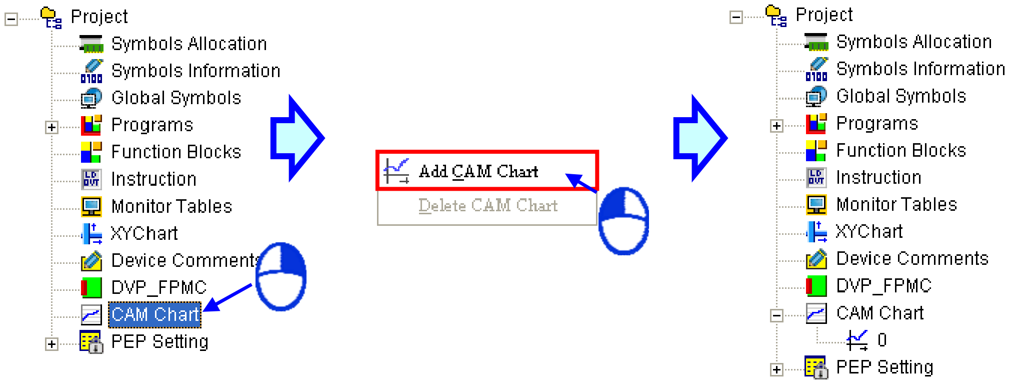
A cam is an input object with an irregular shape. It can pass motion to a follower through direct contact, and make the follower move regularly. A mechanical cam is composed of a cam, a follower, and a support. Please see the figure below. When the cam rotates, the follower moves up and down according to the shape of the cam. If users want to control motion, there must be relation between axes. The relation is established by a cam.



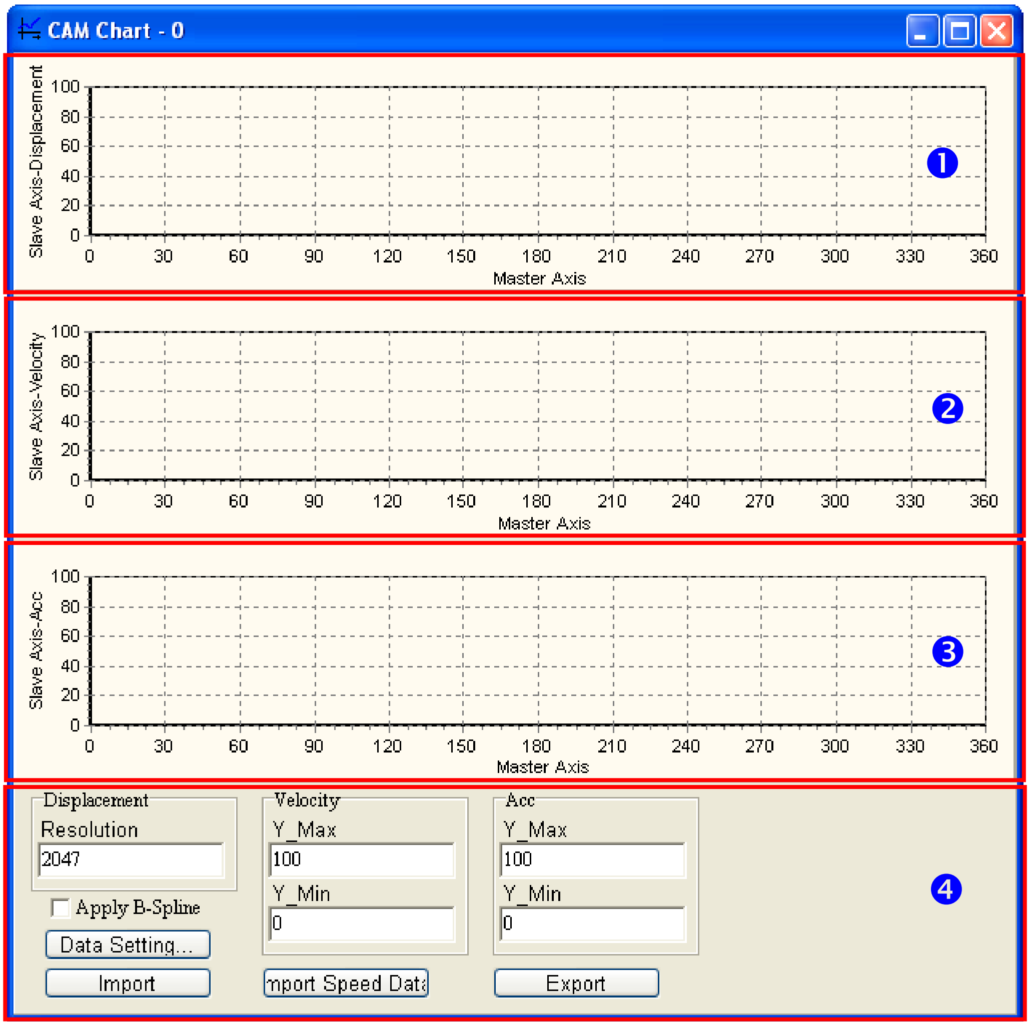
After the relation between an electronic cam and a follower is established by means of creating an electronic cam chart, a motion controller will stimulate the motion of the electronic cam. The electronic cam can be modified by modifying the electronic cam chart. There is no need to change the mechanism of the electronic cam, and the mechanism is not damaged.

8.3.2 Creating a Cam Chart

After users create a project, right-click **CAM Chart** in the system information area, and click **Add CAM Chart** on the context menu, the **CAM Chart-0** window will appear.



The **CAM Chart-0** window is shown below.



Displacement
Resolution
2047

□ Apply B-Spline

Data Setting...

Import

Velocity
Y_Max
100
Y_Min
0

mport Speed Data

Acc
Y_Max
100
Y_Min
0

Export

❶ Displacement: The relation between the master axis and the slave axis is described in terms of displacement.

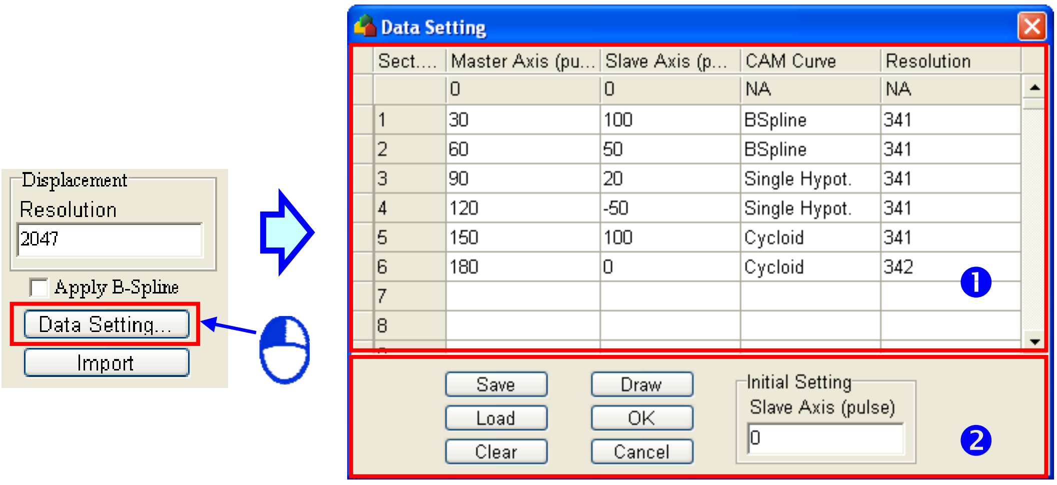
❷ Velocity: The relation between the master axis and the slave axis is described in terms of speed.

❸ Acceleration: The relation between the master axis and the slave axis is described in terms of acceleration.

❹ Data setting area:

- Displacement resolution: Users can set the number of data points required in the electronic cam chart. The number of data points must be in the range of 10 to 2047.
- Velocity: The maximum speed of the slave axis and the minimum speed of the slave axis are shown in this section. They are calculated by the system according to the data related to displacement. Users can change the maximum speed of the slave axis and the minimum speed of the slave axis by themselves.
- Acceleration: The maximum acceleration of the slave axis and the minimum acceleration of the slave axis are shown in this section. They are calculated by the system according to the data related to displacement. Users can change the maximum acceleration of the slave axis and the minimum acceleration of the slave axis by themselves.
- Data setting: The description of the relation between the master axis and the slave axis in terms of displacement is shown in the **Data Setting** window. The displacement resolution set in the data setting area will be brought into the **Data Setting** window after the **Data Setting** window is opened. If users click the **Apply B-spline** checkbox in the data setting area, **B-spline** will be automatically selected in the **Data Setting** window.
- Import: Importing the description of the relation between the master axis and the slave axis in terms of displacement
- Export: Exporting the description of the relation between the master axis and the slave axis in terms of displacement
- Importing speed data: Importing the description of the relation between the master axis and the slave axis in terms of speed

After the users click **Data Setting…** in the **CAM Chart-0** window, the **Data Setting** window will appear. The **Data Setting** window is composed of sections. The users can set a section of a cam curve in every section. A complete cam curve is composed of several sections. The users can set 360 sections at most. An electronic cam cycle is composed of the sections created by the users.



Sect....	Master Axis (pu...	Slave Axis (p...	CAM Curve	Resolution
	0	0	NA	NA
1	30	100	BSpline	341
2	60	50	BSpline	341
3	90	20	Single Hypot.	341
4	120	-50	Single Hypot.	341
5	150	100	Cycloid	341
6	180	0	Cycloid	342
7				
8				

❶ Users can define the relation between the master axis and the slave axis in every section.

- Master axis: Users can set the displacement of the master axis. A pulse is a unit of the measurement for displacement. The values that the users type in the **Master Axis (pulse)** column must be greater than 0, and must be in numerical order.
- Slave axis: Users can set the displacement of the master axis. A pulse is a unit of the measurement for displacement. The values that the users type in the **Slave Axis (pulse)** column can be positive values or negative values.
- Cam curve: The functions which can be selected are **Const Speed**, **Const Acc.**, **Single Hypot.**, **Cycloid**, and **B-Spline**. If users click the **Apply B-spline** checkbox in the **CAM Chart-0** window, **B-spline** will be automatically selected in the **Data**

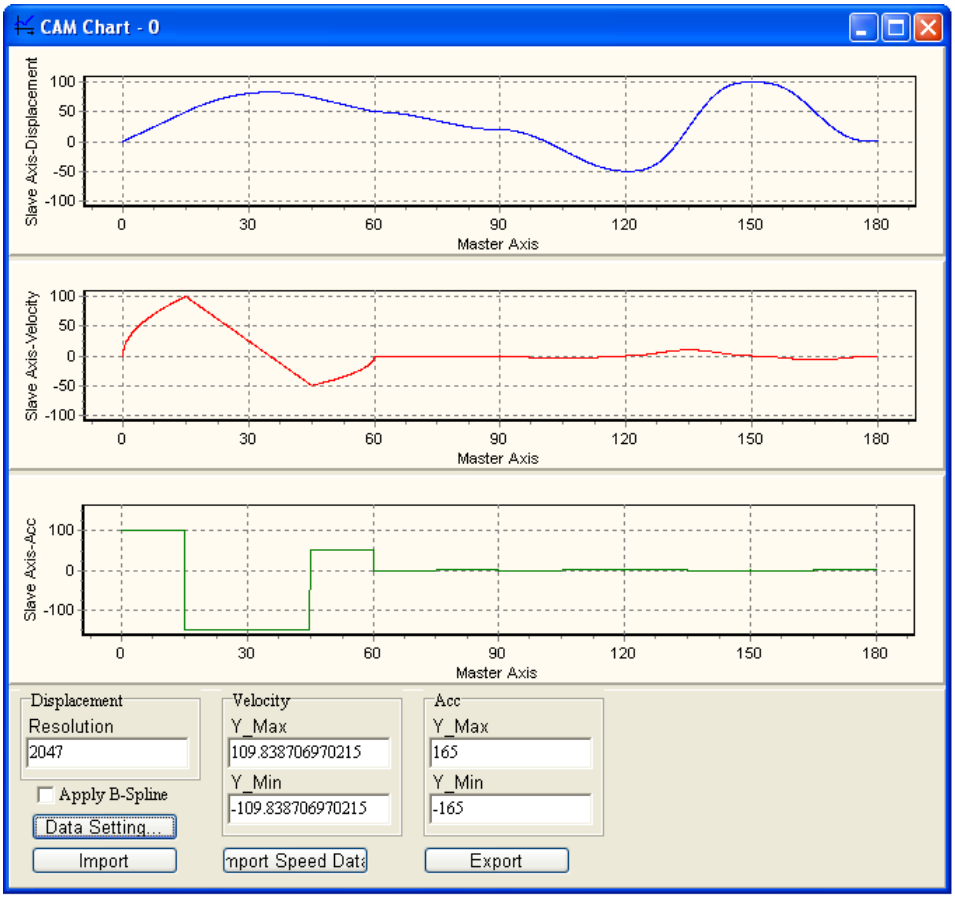
Setting window.

- Resolution: Users can set the number of data points used in a section. The number of data points must be in the range of 10 to 2047. If the users do not set resolutions for sections, the number of data points left will be equally distributed to the sections. The users have to set resolutions according to equipment's requirements. The higher the resolutions set are, the more smoothly the equipment used operates. Besides, the size of the electronic cam data gotten is big if the resolutions set are high.

❷ After sections of a cam curve are created, users can click **Save**, **Load**, **Clear**, **Draw**, **OK**, **Cancel**, or set the initial position of the slave axis.

- **Save**: Saving the data set in sections
- **Load**: Loading the data which was saved
- **Clear**: Clearing all the data in sections
- **Draw**: Compiling the data set in sections, and drawing the electronic cam data gotten on the electronic cam chart created
- **OK**: Compiling the data set in sections, drawing the electronic cam data gotten on the electronic cam chart created, and closing the **Data Setting** window
- **Cancel**: Closing the **Data Setting** window.
- **Initial Setting**: Setting the initial position of the slave axis

The electronic cam chart created is shown below.



Displacement
Resolution: 2047

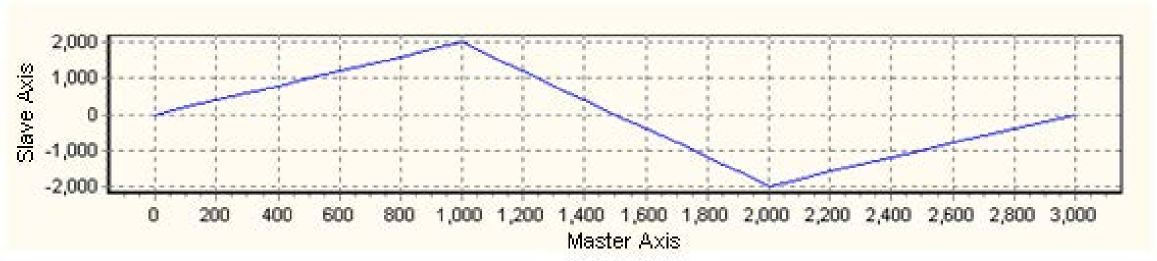
□ Apply B-Spline

Velocity
Y_Max: 109.838706970215
Y_Min: -109.838706970215

Acc
Y_Max: 165
Y_Min: -165

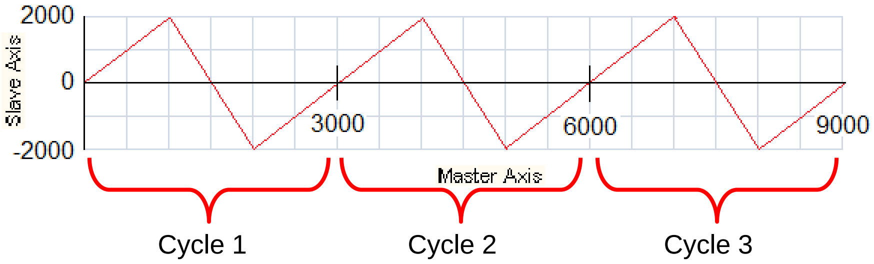
8.3.3 Explanation of an Electronic Cam

A cam chart is shown below. The horizontal axis represents the master axis used, and the vertical axis represents the slave axis used. Every value in the chart represents the number of pulses sent. The range 0~3000 on the horizontal axis is an electronic cam cycle.



If the cam chart is used cyclically, the description of the relation between the master axis and the master axis in terms of pulse output will be as shown below. The electronic cam created is used by a motion controller repeatedly. After the master axis completes an electronic cam cycle, the number of pulses that the master axis sends will continue to increase. However, after the slave axis completes an electronic cam cycle, it will repeat the electronic cam cycle.

If a motion controller is connected to a master servo motor and a slave servo motor, the master servo motor will regarded as a drive shaft rotating in a direction, and the slave servo motor will be regarded as a processing shaft which rotates back and forth. This characteristic can be applied to flyingsaws and flyingcuts.



8.3.4 Important Points about an Electronic Cam

- The source of the signals sent to a master axis can be a servo encoder, the feedback on the pulse output sent by the master axis, or a virtual axis. Please refer to DVP-PM Application Manual for more information about wiring a motion controller.
- An electronic cam can operates cyclically or non-cyclically. If an electronic cam operates non-cyclically, the relation between the master axis of the electronic cam and the slave axis of the electronic cam will be maintained for only one cycle. If an electronic cam operates cyclically, the relation between the master axis of the electronic cam and the slave axis of the electronic cam will be repeated.
- Users can create 12 cam charts at most in a PMSoft project. Only three of the twelve cam charts created in a project for a DVP series motion controller can be used, and the twelve cam charts created in a project for an AH500 series motion controller can be used.

8.4 Example

8.4.1 Drawing a Path by Means of an Electronic Cam

- Program requirement

  Users have to create the cyclic electronic cam relation shown in figure 1. The master axis is the y-axis of the motion controller used. The output terminals (FP1+, FP1-, RP1+, and RP1-) of the y-axis are connected to the input terminals (A0+, A0-, B0+, and B0-) of the pulse generator which are connected to the output terminals of the x-axis of the motion controller, as shown in figure 2. If M0 is ON, the signals sent to the input terminals of the pulse generator will undergo the transformation related to the electronic cam, and the transformation results will be sent to the output terminals of the x-axis. The x-axis is the slave axis. The motion controller used is

DVP-20PM00D.

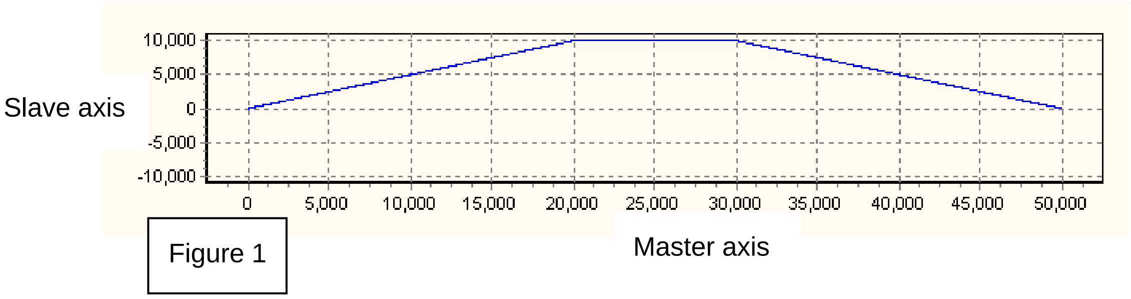


Figure 1

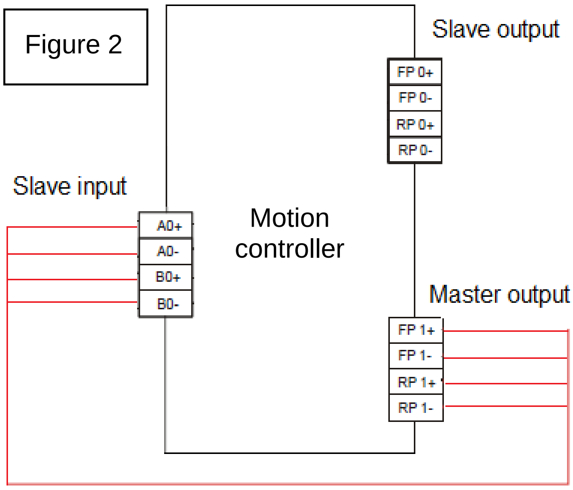
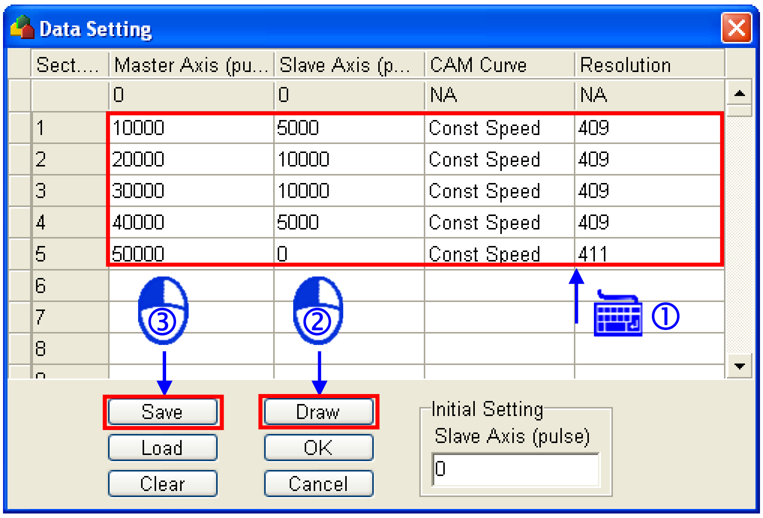


Figure 2

- Programming order
  (1) Creating a cam chart
  (2) Setting the parameters in the main program O100

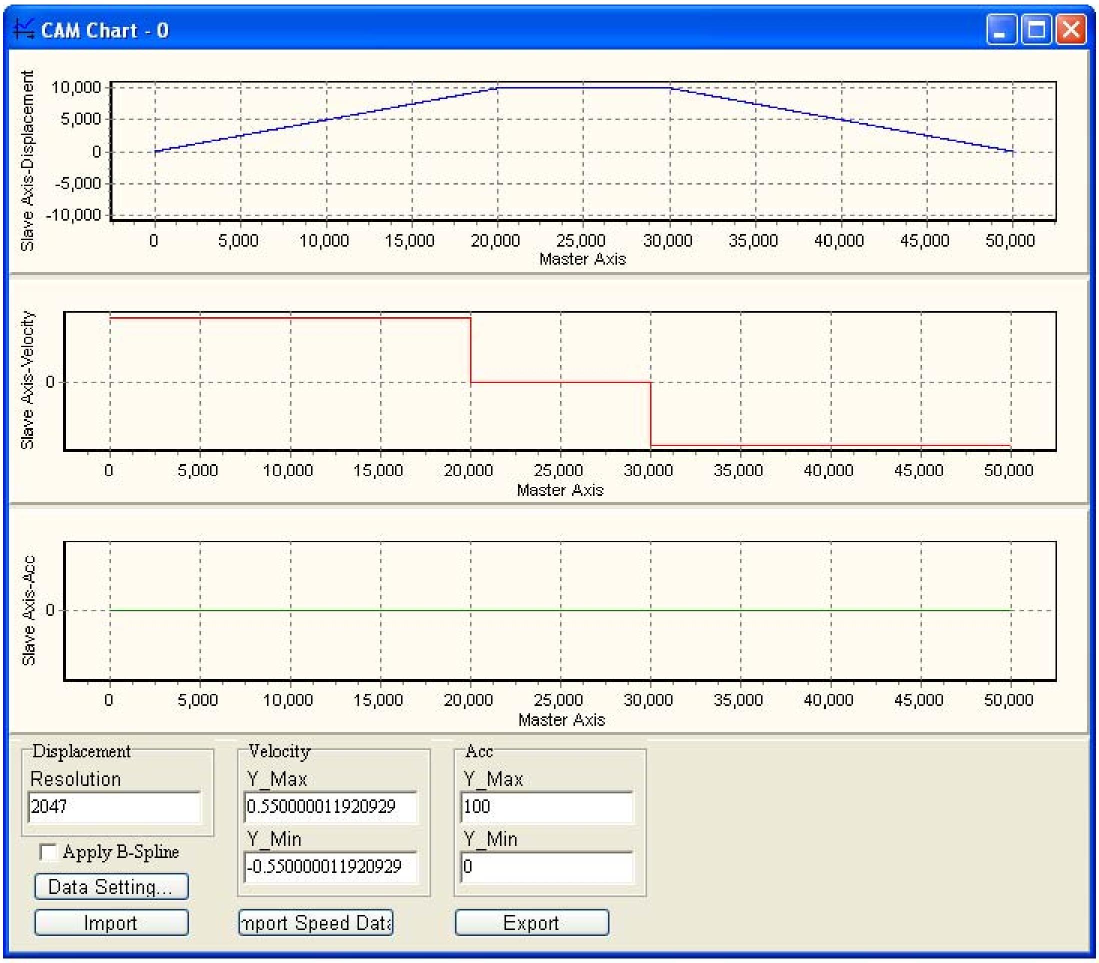
8.4.2 Explanation

- Cam chart
  Users have to set sections in the **Data Setting** window. They can set resolutions according to their need. After the users click **Draw**, click **Save**, and close the **Data Setting** window, a cam chart will be created.



Sect....	Master Axis (pu...	Slave Axis (p...	CAM Curve	Resolution
	0	0	NA	NA
1	10000	5000	Const Speed	409
2	20000	10000	Const Speed	409
3	30000	10000	Const Speed	409
4	40000	5000	Const Speed	409
5	50000	0	Const Speed	411
6				
7				
8				

Initial Setting
Slave Axis (pulse): 0



Displacement
Resolution: 2047

□ Apply B-Spline

Velocity
Y_Max: 0.550000011920929
Y_Min: -0.550000011920929

Acc
Y_Max: 100
Y_Min: 0

- Main program O100

  The special registers which are used are described below. Please refer to DVP-PM Application Manual for more information.

  M1002: M1002 is used for initializing setting. If the motion controller used is set to Auto, M1002 will be ON in the first scan cycle, and will be OFF thereafter.

  D1848 & D1849: The values in D1848 and D1849 represent the current position of the x-axis.

  D1928 & D1929: The value in D1928 and D1929 represent the current position of the y-axis.

D1860 & D1861: The values in D1860 and D1861 represent the frequency of pulses sent to the input terminals of the pulse generator connected to the output terminals of the x-axis.

D1862 & D1863: The values in D1862 and D1863 represent the number of pulses sent to the input terminals of the pulse generator connected to the output terminals of the x-axis.

D1864: The value in D1864 is H200. It indicates that the signals received by the input terminals of the pulse generator connected to the output terminals of the x-axis are A/B-phase signals.

D1896: The value in D1896 is H30. It indicates that the unit used is a motor unit, and the signals sent by the output terminals of the y-axis are A/B phase signals.

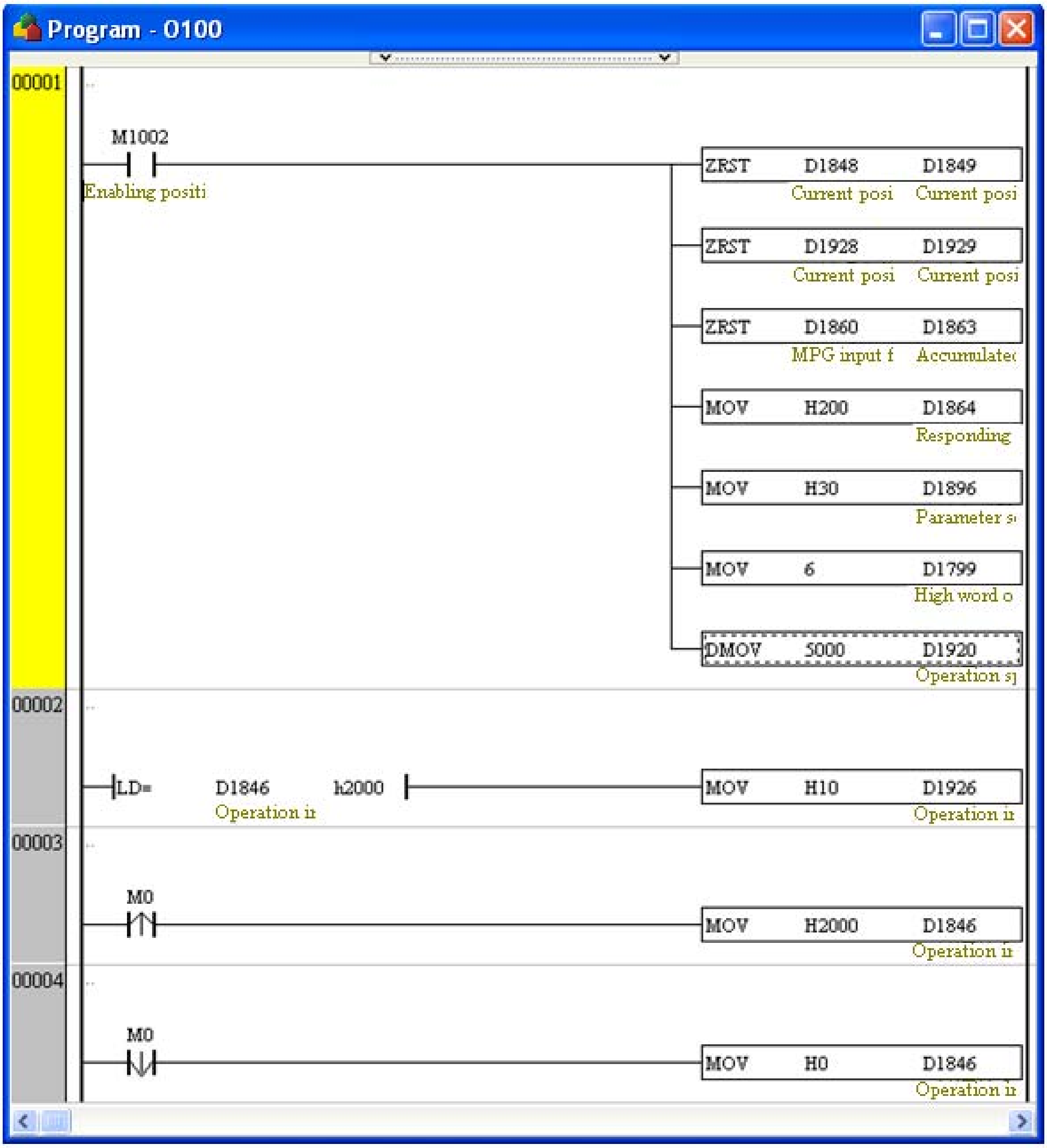
D1799: The input terminals MPGA and MPGB are normally-open contacts.

D1920: The value in D1920 is 5000. It represents the speed at which the y-axis rotates.

D1926: The value in D1926 is H10. It indicates that the rotation of the y-axis is enabled.

D1846: The value in D1846 is H2000. It indicates that cam 0 is a cyclic cam.

D1868: The value in D1868 represents the cam chart number selected. The default value in D1868 is 0. The default value does not need to be changed.

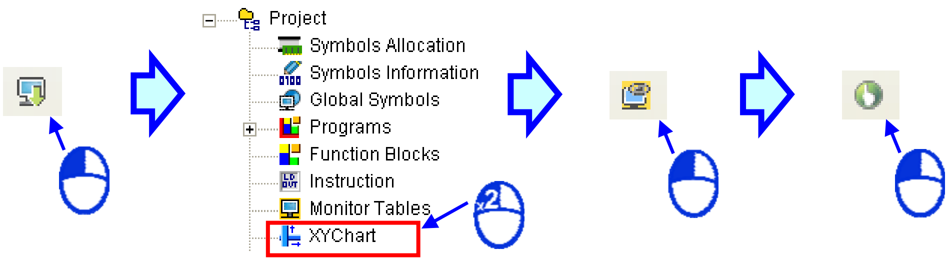


Program - 0100

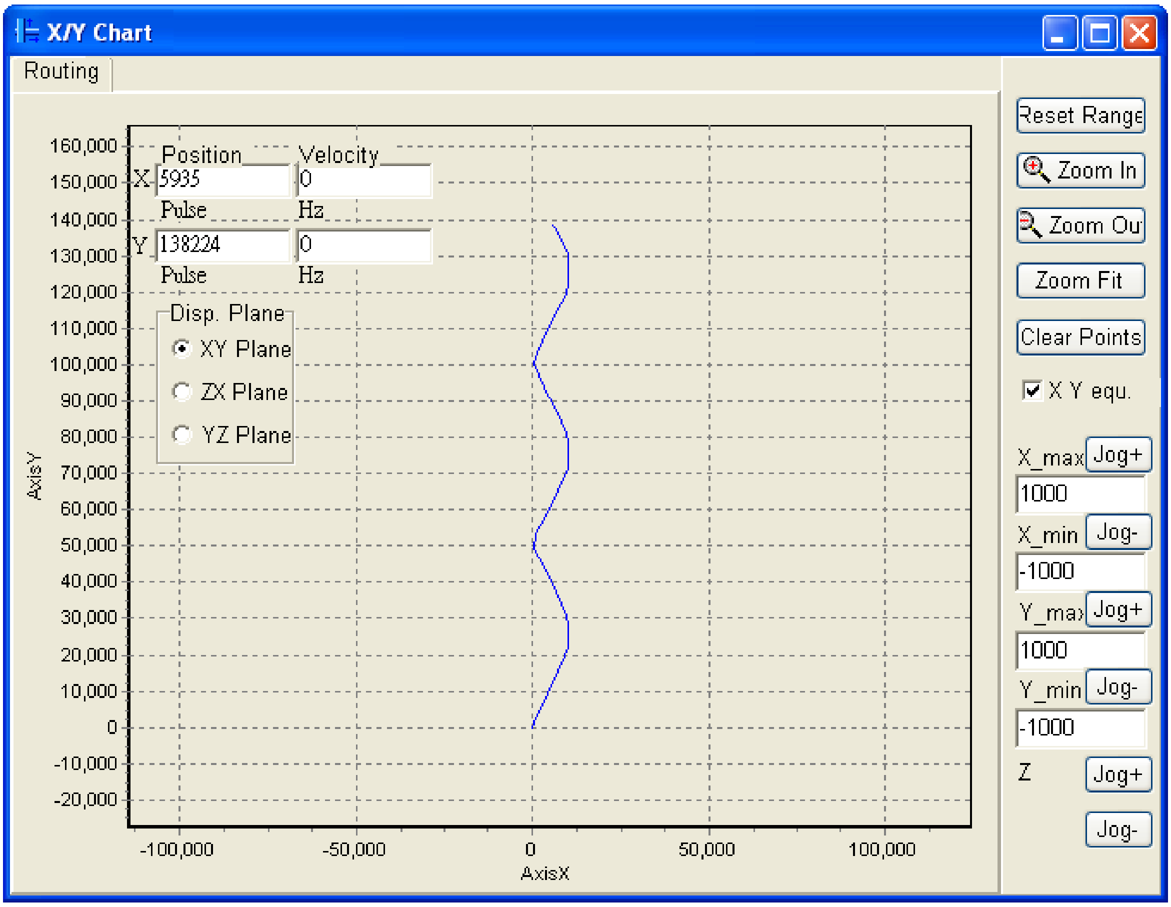
```
00001
        M1002
      ──┤ ├──────────────┬─ ZRST   D1848        D1849
      Enabling positi    │         Current posi Current posi
                         ├─ ZRST   D1928        D1929
                         │         Current posi Current posi
                         ├─ ZRST   D1860        D1863
                         │         MPG input f  Accumulatec
                         ├─ MOV    H200         D1864
                         │                      Responding
                         ├─ MOV    H30          D1896
                         │                      Parameter s
                         ├─ MOV    6            D1799
                         │                      High word o
                         └─ DMOV   5000         D1920
                                                Operation sj
00002
      ──┤LD=   D1846        h2000 ├────────── MOV    H10          D1926
               Operation ir                                      Operation ir
00003
        M0
      ──┤↑├────────────────────────────────── MOV    H2000        D1846
                                                                 Operation ir
00004
        M0
      ──┤↓├────────────────────────────────── MOV    H0           D1846
                                                                 Operation ir
```

- Execution result

  Please refer to chapter 2 and chapter 11. After the users connect the terminals, they have to compile the program created in PMSoft, download the program, double-click **XY Chart** in the system information area, click **Monitoring** on the toolbar, and click **Run O100** on the toolbar. After the path drawn according to the electronic cam relation described above appears in the **X/Y Chart** window, the users can click **Zoom In** or **Zoom Out**. Owing to the fact that the electronic cam used is a cyclic electronic cam, the electronic cam cycle created is repeated. The path drawn is shown below.



*. Please refer to section 11.3.6 for more information about the X/Y Chart window.



	Position	Velocity
X	5935	0
	Pulse	Hz
Y	138224	0
	Pulse	Hz

Disp. Plane
● XY Plane
○ ZX Plane
○ YZ Plane

☑ X Y equ.

X_max 1000
X_min -1000
Y_ma 1000
Y_min -1000

MEMO

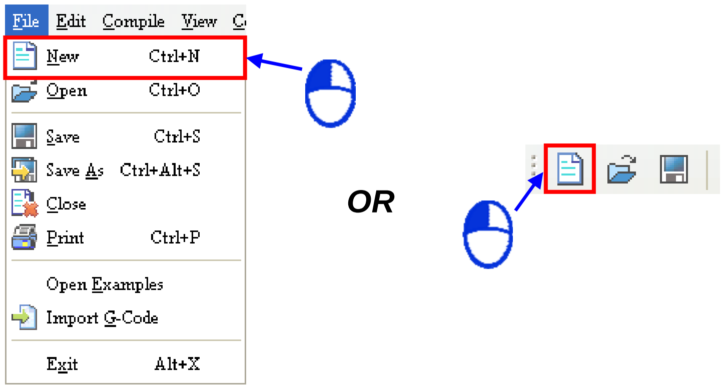
Chapter 9 Project Management

Table of Contents

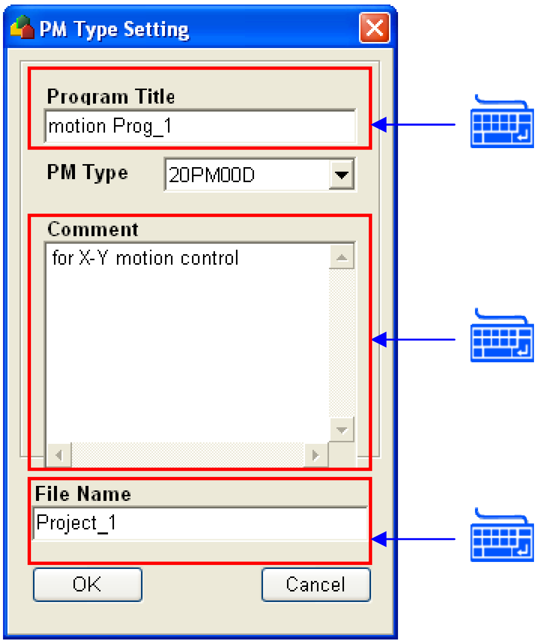
9.1 File Operation

9.1.1 Creating a New Project

(1) Users have to click **New** on the **File** menu, or 🗋 on the toolbar.



(2) The users have to type a program title, a file name, and a comment in the **PM Type Setting** window.



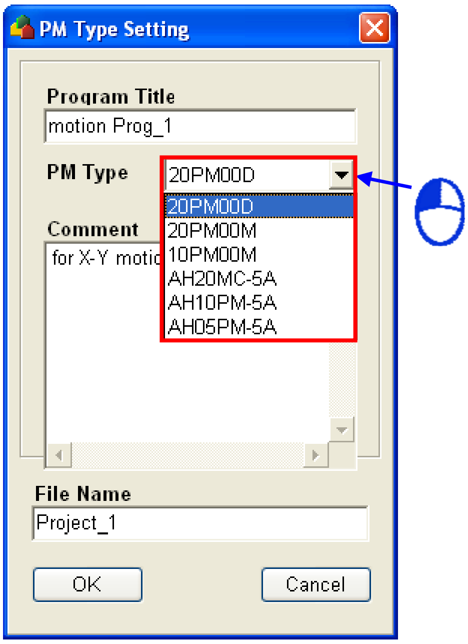
Program Title: motion Prog_1

PM Type: 20PM00D

Comment: for X-Y motion control

File Name: Project_1

(3) The users have to select a model in the **PM Type** drop-down list box.



PM Type Setting

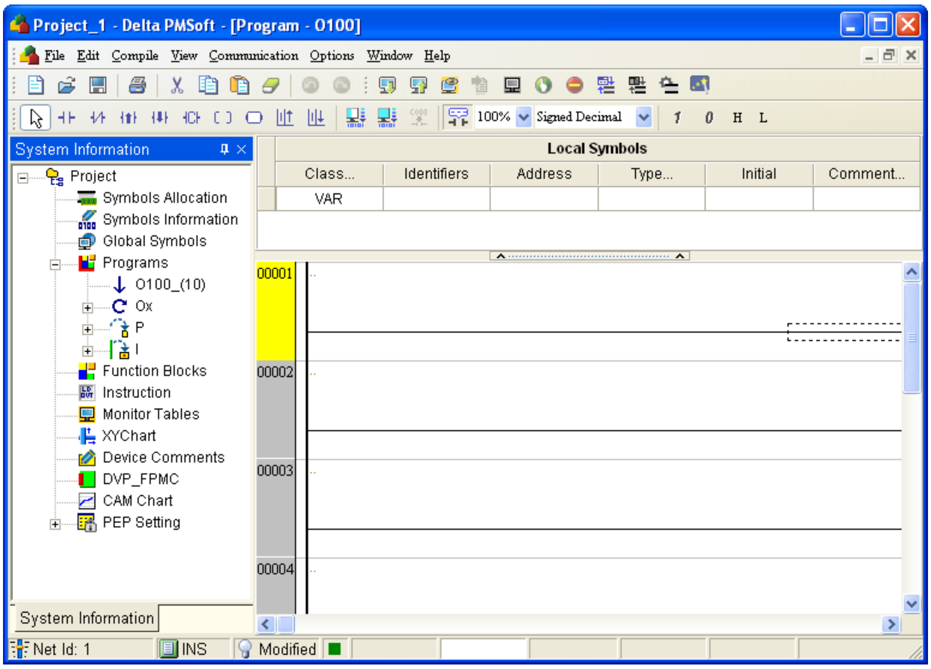
Program Title: motion Prog_1

PM Type: 20PM00D

Comment: for X-Y motio

File Name: Project_1

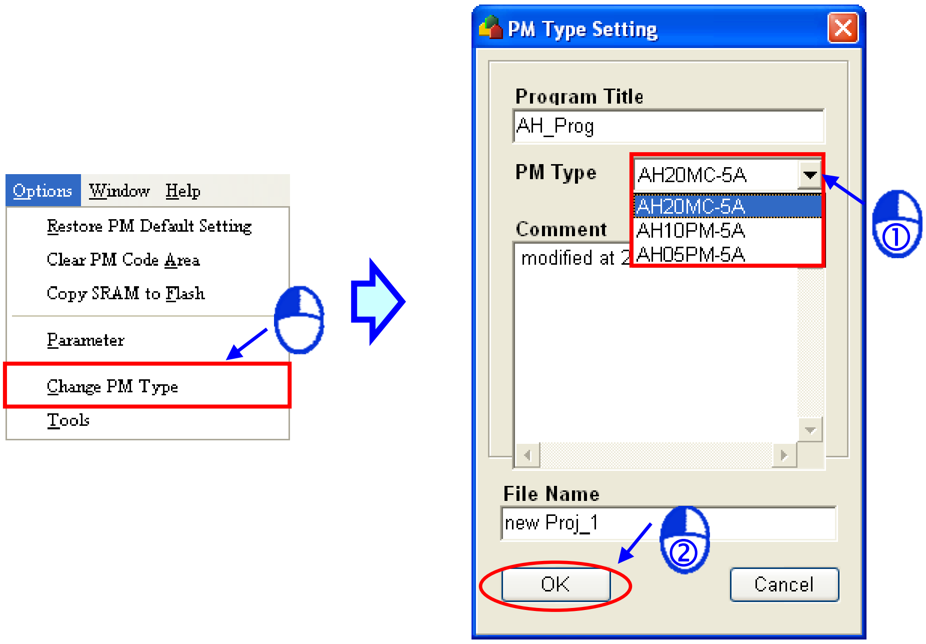
(4) After the users click **OK** in the **PM Type Setting** window, a project window will be opened.



Project_1 - Delta PMSoft - [Program - O100]

9.1.2 Changing the Model and the Project information

After users create a project, they can click **Change PM Type** on the **Option** menu to view or modify the information in the **PM Type Setting** window. The users can only change the motion controller selected in the **PM Type** drop-down list box to another motion controller of the same series. For example, AH20MC-5A can only be changed to AH10PM-5A or AH05PM-5A, and can not be changed to a DVP series motion controller.



Program Title: AH_Prog

PM Type: AH20MC-5A

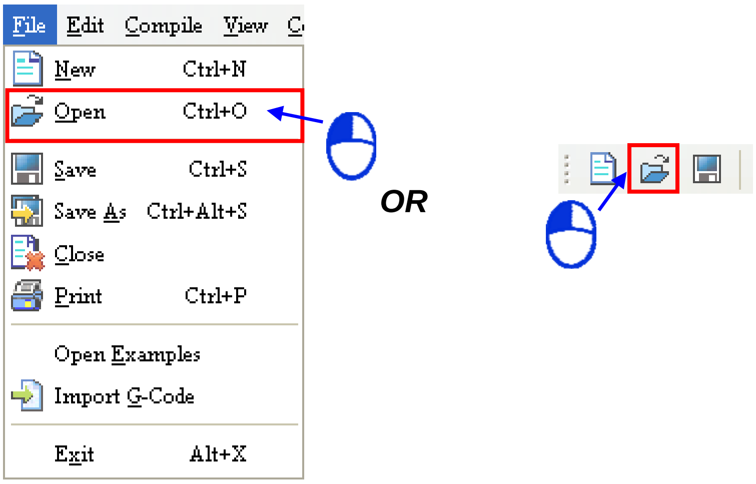
Comment: modified at 2

File Name: new Proj_1

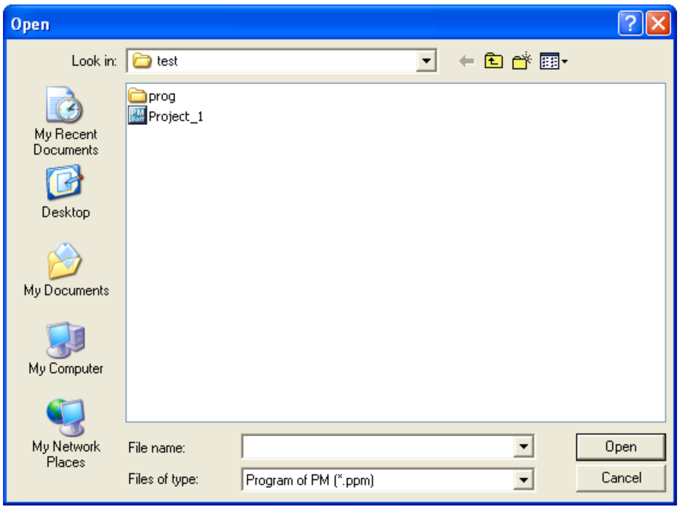
9.1.3 Opening a Project

- Method 1

(1) Users have to click **Open** on the **File** menu, or on the toolbar.



(2) The users have to click a .ppm file in the **Open** window, and click **Open**.



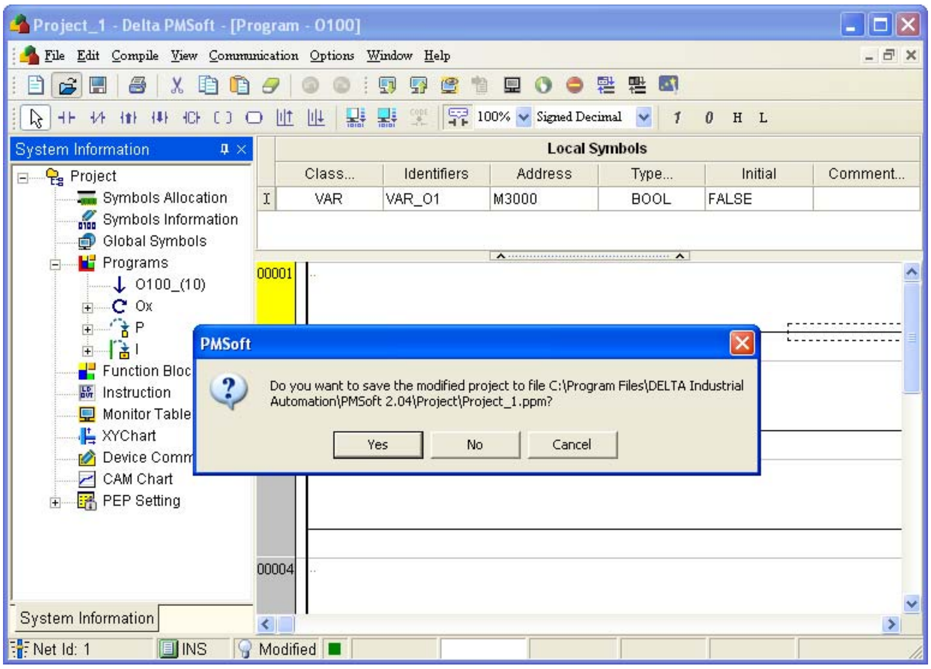
Open

Look in: test

prog
Project_1

File name:
Files of type: Program of PM (*.ppm)

(3) After the users click **Open** in the **Open** window, the system will close the project which is being edited presently. If the project has not been saved, the system will ask the users whether they want to save the project.



Project_1 - Delta PMSoft - [Program - 0100]

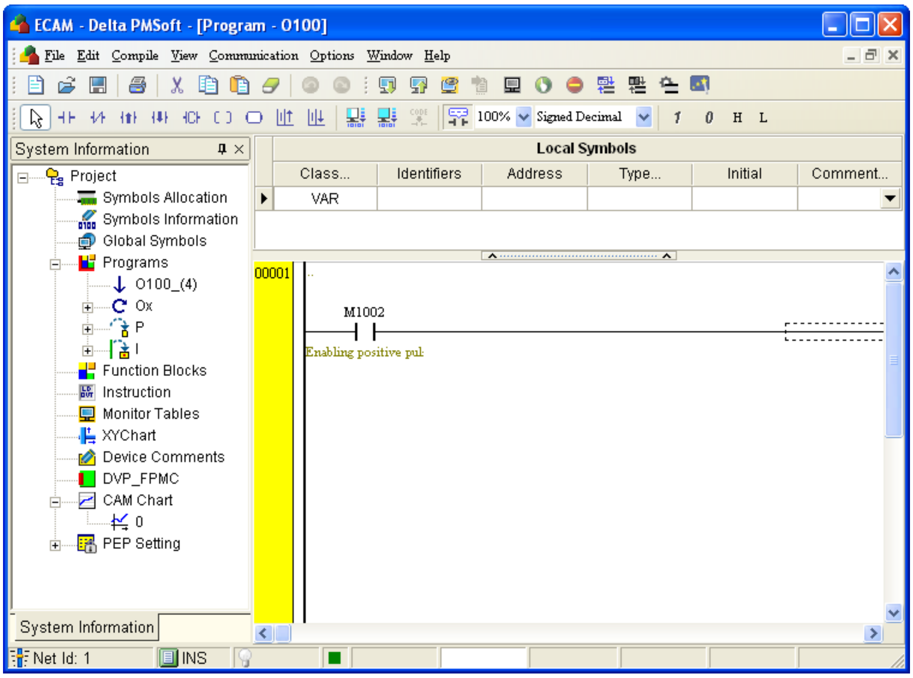
Local Symbols

	Class...	Identifiers	Address	Type...	Initial	Comment...
I	VAR	VAR_01	M3000	BOOL	FALSE	

PMSoft

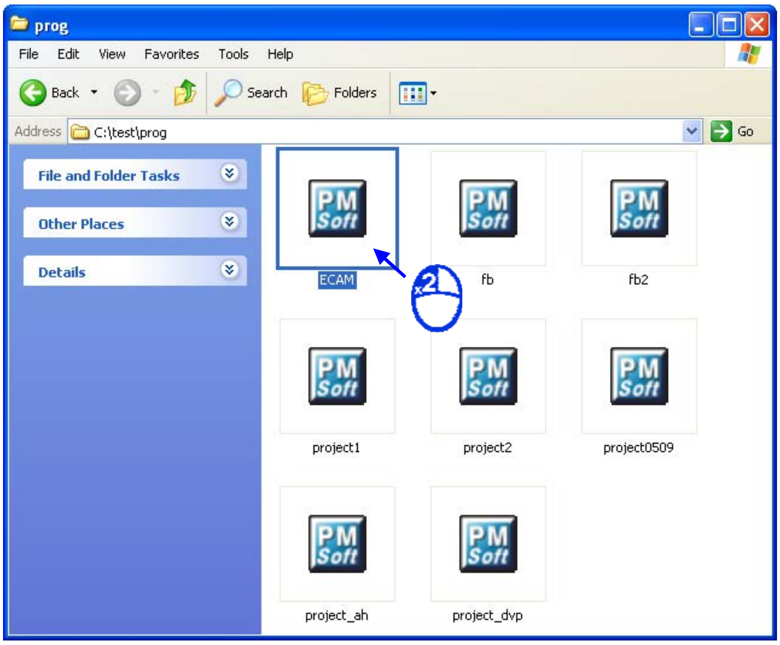
Do you want to save the modified project to file C:\Program Files\DELTA Industrial Automation\PMSoft 2.04\Project\Project_1.ppm?

(4) After the users click **Yes** or **No** in the PMSoft window, the .ppm file clicked in the **Open** window will be opened.

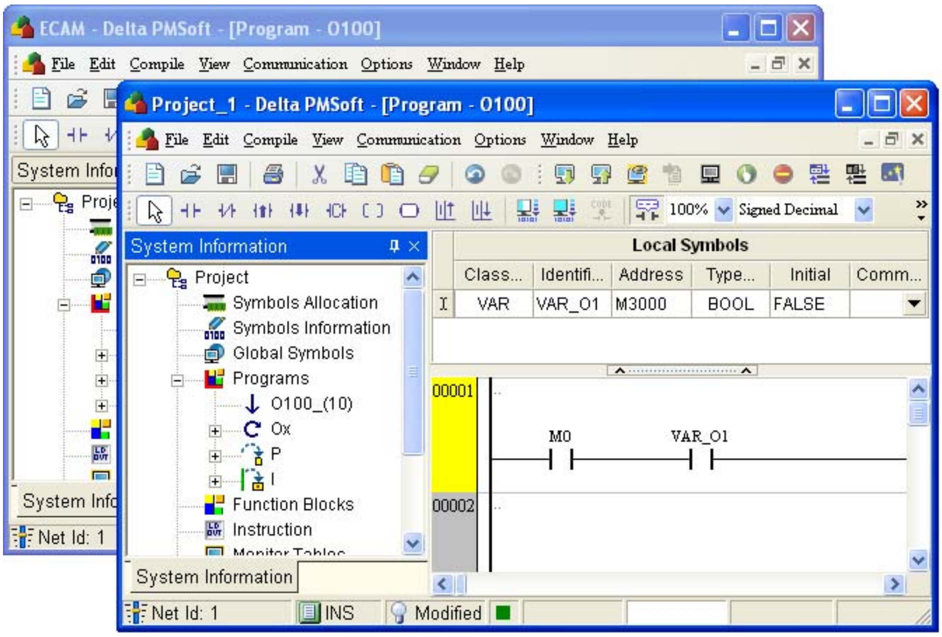


- Method 2

(1) Users have to open a window, and double-click a .ppm file in the window.

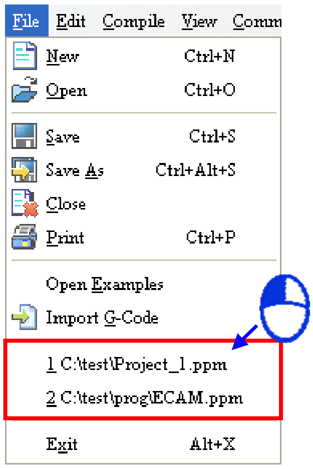


(2) After the users double-click a .ppm file, the system will open another PMSoft window. The users can copy/cut and paste parts of the program in a PMSoft window into another PMSoft window.

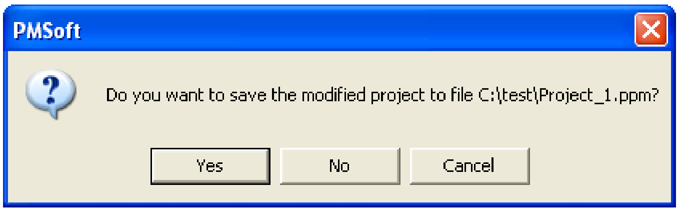

ECAM - Delta PMSoft - [Program - 0100]
File Edit Compile View Communication Options Window Help
Project_1 - Delta PMSoft - [Program - 0100]
File Edit Compile View Communication Options Window Help
100% Signed Decimal
System Information
Project
Symbols Allocation
Symbols Information
Global Symbols
Programs
0100_(10)
Ox
P
I
Function Blocks
Instruction
Monitor Tables
Local Symbols
Class... Identifi... Address Type... Initial Comm...
VAR VAR_01 M3000 BOOL FALSE
00001
M0
VAR_01
00002
Net Id: 1
INS
Modified

9.1.4 Opening a Recently-edited Project

(1) After users click the **File** menu, they will find the projects which have been edited recently. The users have to click a project.



(2) If the project which is being edited presently has not been saved, the system will ask the users whether they want to save the project. After the users click **Yes** or **No** in the PMSoft window, the project they click on the **File** menu will be opened.

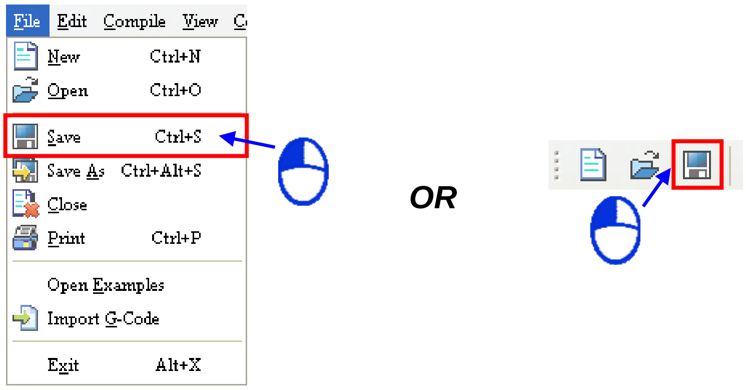

PMSoft

Do you want to save the modified project to file C:\test\Project_1.ppm?

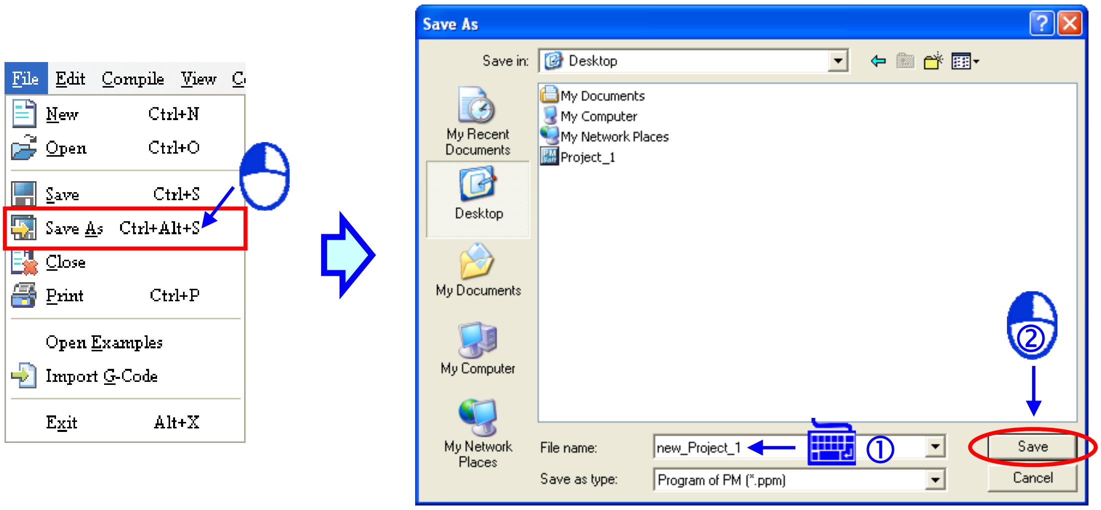
9.1.5 Saving / Saving as

(1) After users click **Save** on the **File** menu, or on the toolbar, the project which is being

edited presently will be saved. If the project is saved for the first time, the **Save As** window will appear. The users can type a file name, and set a path in the **Save As** window. If the users do not set a file name and a path in the **Save As** window, the project will be saved as a .ppm file with a default file name in a folder in a disk. If the users save the project again, the project will overwite the project saved last time.



(2) If the users want to save the project with another file name, or in another folder, they have to click **Save As** on the **File** menu, set a path and a file name in the **Save As** window, and click **Save**.



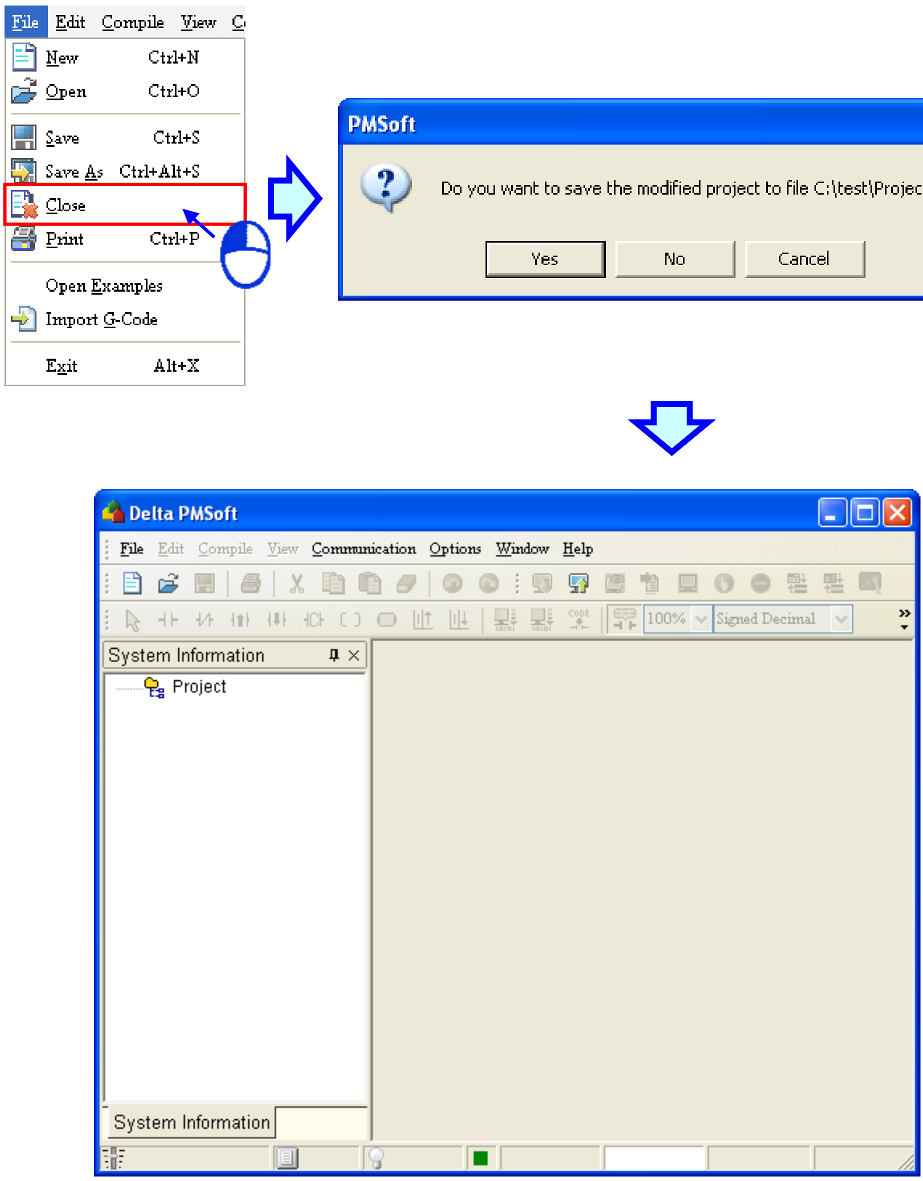
Save in: Desktop

File name: new_Project_1

Save as type: Program of PM (*.ppm)

9.1.6 Closing a Project

After users click **Close** on the **File** menu, the project which is being edited presently will be closed. If the project has not been saved, the system will ask the users whether they want to save the project. After the users click **Yes** or **No** in the PMSoft window, the project will be closed.



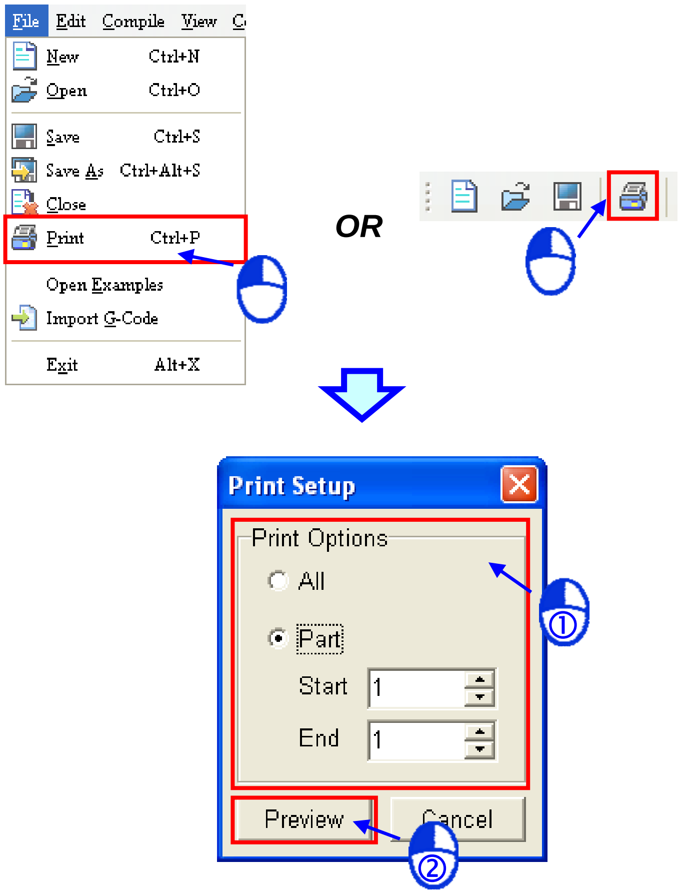
Do you want to save the modified project to file C:\test\Project_1.ppm?

9.1.7 Printing

After users write a program, they can print related data. PMSoft can print ladder diagrams and instruction lists.

- Printing a ladder diagram

  Users have to open a window where a ladder diagram is created, and click **Print** on the **File** menu, or 🖨 on the toolbar.



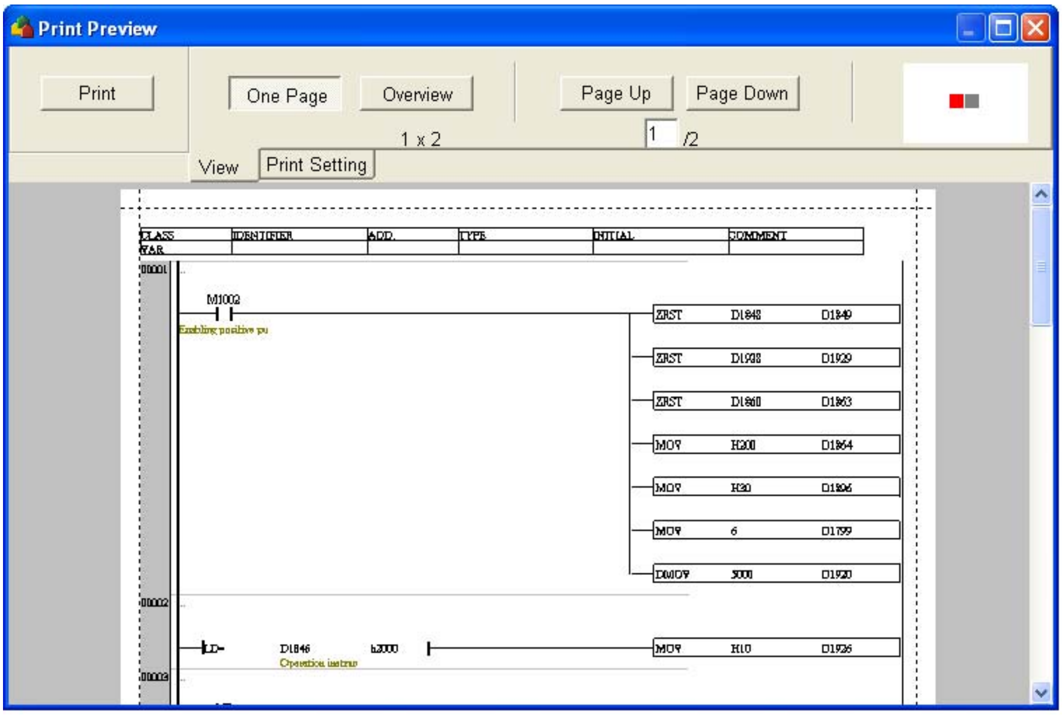
○ All

● Part

Start 1

End 1

The users have to select the **All** option button, or the **Part** option button in the **Print Options** section in the **Print Setup** window. If the users select the **All** option button, all the networks in the ladder diagram will be previewed. If the users select the **Part** option button, they have to select a network number in the **Start** box, and a network number in the **End** box. After the users select the **All** option button or the **Part** option button, they have to click **Preview**.



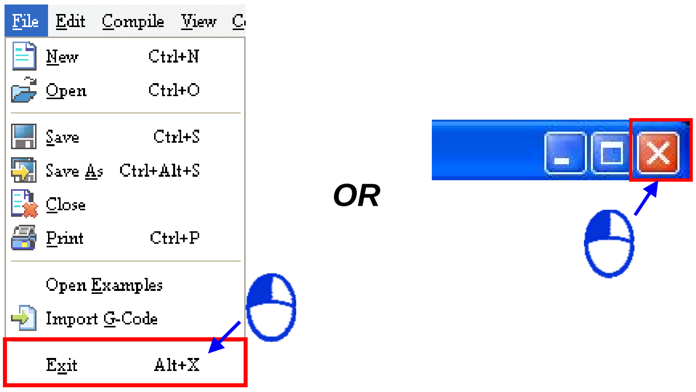
After the users click **Preview**, they can switch among the pages, zoom in/out on the pages, setting a printer, and etc.

- Printing an instruction list

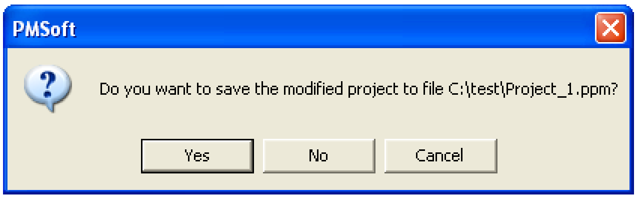
  Users have to open the **Instructio**n window. After the users click **Print** on the **File** menu, or 🖨 on the toolbar, the **Print Preview** window will be opened. There is no **Pint Setting** tab in the **Print Preview** window.

9.1.8 Exiting PMSoft

(1) If users want to close a PMSoft window, they have to click **Exit** on the **File** menu, or ☒ in the upper right corner of the window.



(2) If the project which is being edited presently has not been saved, the system will ask the users whether they want to save the project.

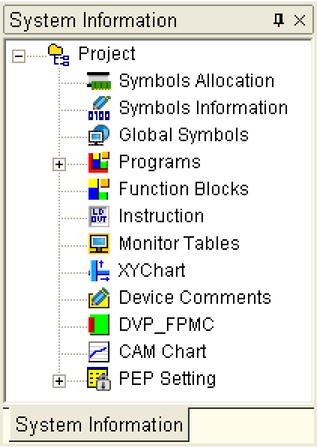

PMSoft

Do you want to save the modified project to file C:\test\Project_1.ppm?

9.2 Functions Related to Project Management

9.2.1 System Information Area

The system information area in PMSoft adopts an interface which uses a hierarchical tree structure. Users can manage projects by means of this interface. The system information area is described below.



System Information

- Project
  - Symbols Allocation
  - Symbols Information
  - Global Symbols
  - Programs
  - Function Blocks
  - Instruction
  - Monitor Tables
  - XYChart
  - Device Comments
  - DVP_FPMC
  - CAM Chart
  - PEP Setting

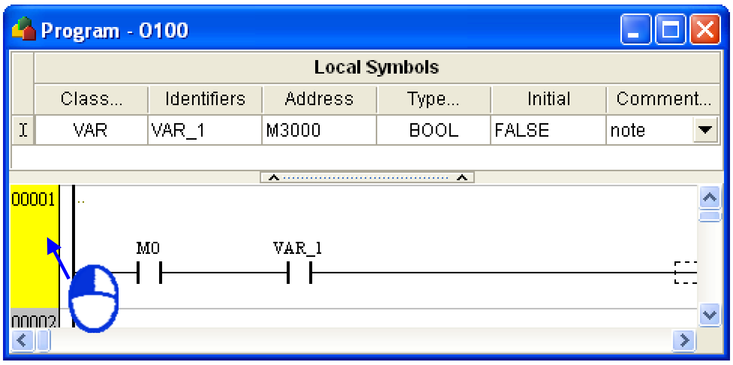
System Information

- Information about symbols and devices: **Symbols Allocation**, **Symbols Information**, **Global Symbols**, and **Device Comments**
- Programming objects: **Programs**, **Function Blocks**, and **CAM Chart**
- **Instruction**: The instruction list which is created is shown in the **Instruction** window. The instruction list can be transformed into a ladder diagram.
- Objects which are monitored: **Monitor Tables** and **XY Chart**
- **DVP_FPMC**: Setting the function card DVP-FPMC
- **PEP Setting**: Setting a password for a program

9.2.2 Searching for / Replacing an Object in a Ladder Diagram

- **Searching for an object**

  (1) After users open a window, they have to click an editing position.

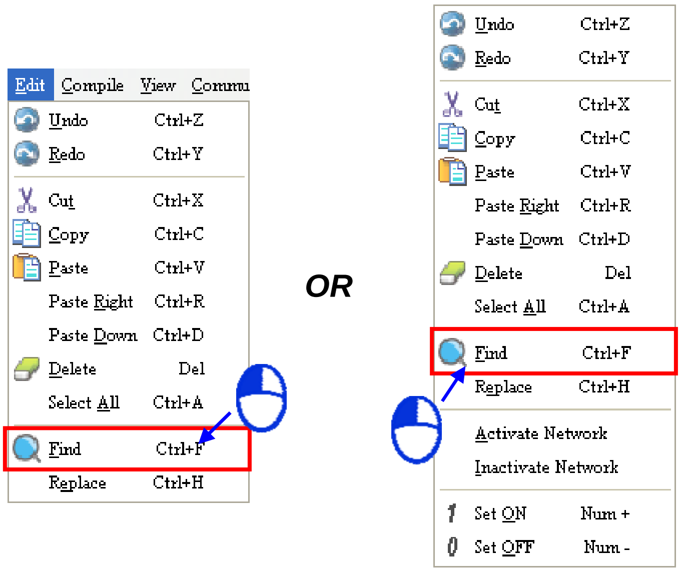


Program - 0100

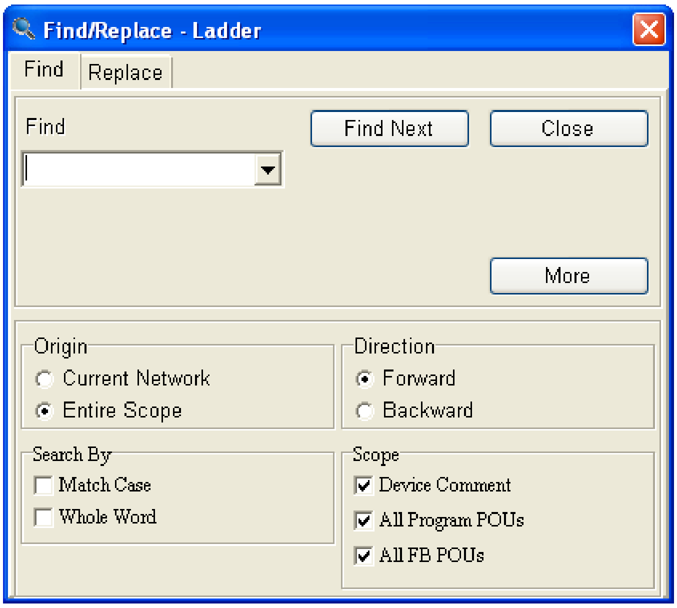
Local Symbols

	Class...	Identifiers	Address	Type...	Initial	Comment...
I	VAR	VAR_1	M3000	BOOL	FALSE	note

(2) The users have to click **Find** on the **Edit** menu. The users can right-click the working area in the window, and then click **Find** on the context menu.



(3) The users have to type an object in the **Find** box in the **Find/Replace-Ladder** window. They can do a search on the object typed in the **Find** box by clicking **Find Next**. If the users click **More**, they can set search conditions.


Find/Replace - Ladder

Find | Replace

Find

Find Next | Close | More

Origin: ○ Current Network ● Entire Scope

Direction: ● Forward ○ Backward

Search By: □ Match Case □ Whole Word

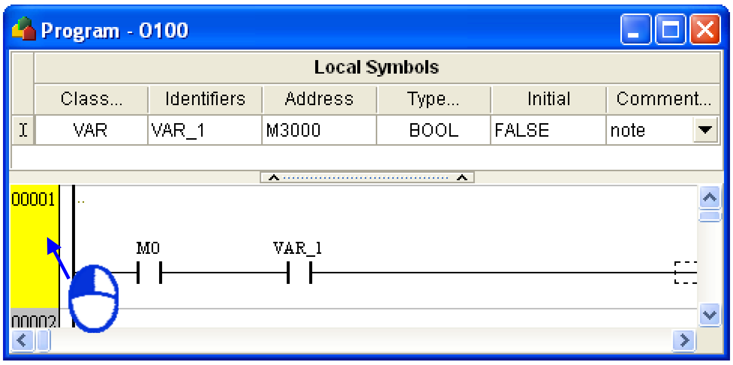
Scope: ☑ Device Comment ☑ All Program POUs ☑ All FB POUs

- **Find**: Users have to type the text which will be searched for, or select text which has been searched for after they click ▼ in the box.
- **Origin**: If the **Current Network** option button is clicked, the search starts at the position selected. If the **Entire Scope** option button is clicked, the search starts at the beginning of the program.
- **Direction**: Users can search down or search up.
- **Search By**: If the **Match Case** checkbox is selected, PMSoft searches only for words that match the case of the word that users type in the **Find** box. If the **Whole Word** checkbox is selected, the **Whole Word** checkbox instructs PMSoft to find complete words only, and not to find words that only contain what users type.
- **Scope**: Users can select ranges which will be searched.

*** Users can switch between the Find tab and the Replace tab in the Find/Replace-Ladder window.**

- **Replacing an object**

(1) After users open a window, they have to click an editing position.

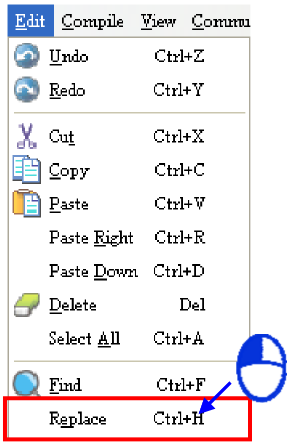


Program - 0100

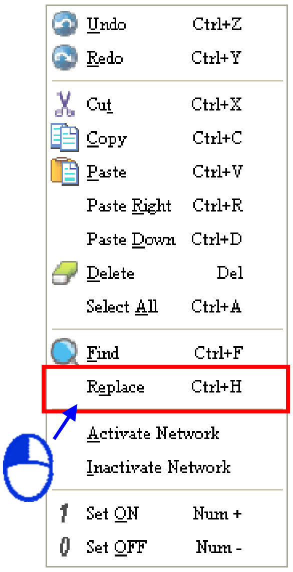
Local Symbols

	Class...	Identifiers	Address	Type...	Initial	Comment...
I	VAR	VAR_1	M3000	BOOL	FALSE	note

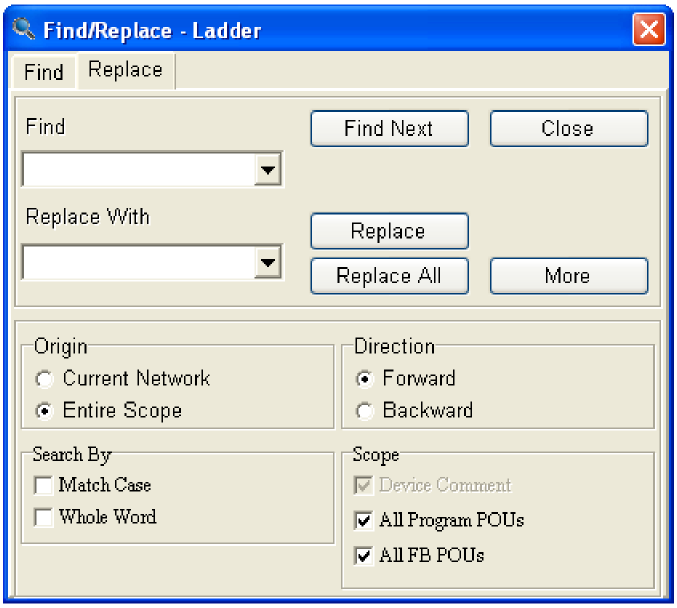
(2) The users have to click **Replace** on the **Edit** menu. The users can right-click the working area in the window, and then click **Replace** on the context menu.



OR



(3) The users have to type an object in the **Find** box, and an object in the **Replace With** box in the **Find/Replace-Ladder** window. They can do a search on the object typed in the **Find** box by clicking **Find Next**. If the users click **Replace**, an occurrence of the object typed in the **Find** box will be replaced by the replacement object. If users click **Replace All**, all the occurrences of the object typed in the **Find** box will be replaced. If the users click **More**, they can set replacement conditions.



Find:

Replace With:

Origin
○ Current Network
● Entire Scope

Direction
● Forward
○ Backward

Search By
□ Match Case
□ Whole Word

Scope
☑ Device Comment
☑ All Program POUs
☑ All FB POUs

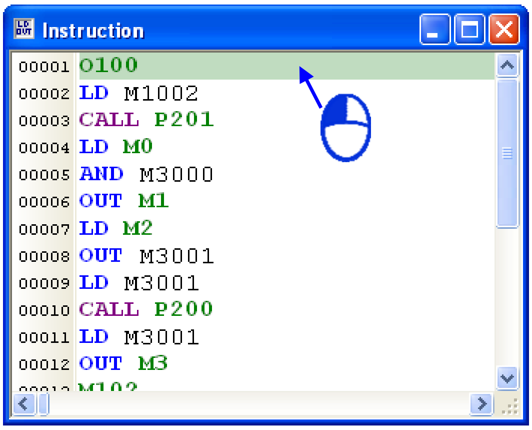
- **Find**: Users have to type the text which will be searched for, or select text which has been searched for after they click ▾ in the box.
- **Replace With**: Users have to type replacement text, or select replacement text which has been typed after they click ▾ in the box.
- **Origin**: If the **Current Network** option button is clicked, the search starts at the position selected. If the **Entire Scope** option button is clicked, the search starts at the beginning of the program.
- **Direction**: Users can search down or search up.
- **Search By**: If the **Match Case** checkbox is selected, PMSoft searches only for words that match the case of the word that users type in the **Find** box. If the **Whole Word** checkbox is selected, the **Whole Word** checkbox instructs PMSoft to find complete words only, and not to find words that only contain what users type.
- **Scope**: Users can select ranges which will be searched.

*** Users can switch between the Find tab and the Replace tab in the Find/Replace-Ladder window.**

9.2.3 Searching for / Replacing an Object on an Instruction List

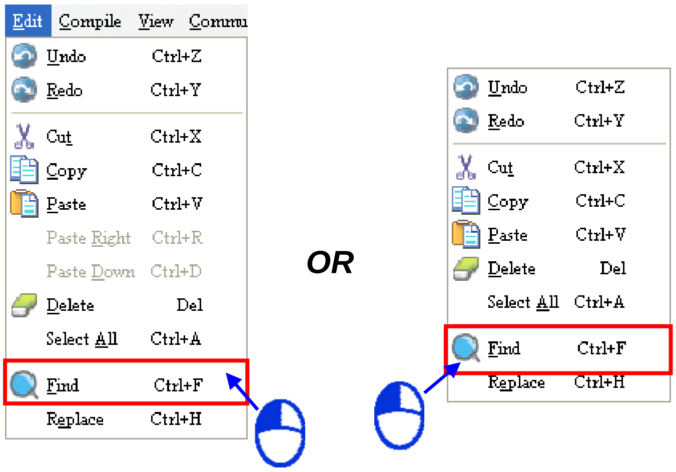
- **Searching for an object**

(1) After users open a window, they have to click an editing position, or select an amount of text.

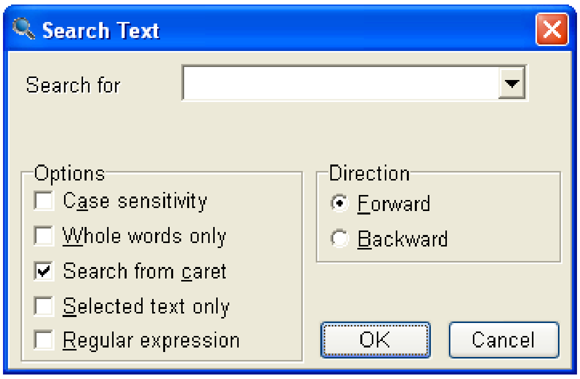


```
O100
LD  M1002
CALL  P201
LD  M0
AND  M3000
OUT  M1
LD  M2
OUT  M3001
LD  M3001
CALL  P200
LD  M3001
OUT  M3
M102
```

(2) The users have to click **Find** on the **Edit** menu. The users can right-click the working area in the window, and then click **Find** on the context menu.



(3) The users have to set search conditions in the **Search Text** window. After they click **OK**, the **Search Text** window will be closed automatically, and the search for the text typed in the **Search for** box will be carried out.


Search Text

Search for

Options
- □ Case sensitivity
- □ Whole words only
- ☑ Search from caret
- □ Selected text only
- □ Regular expression

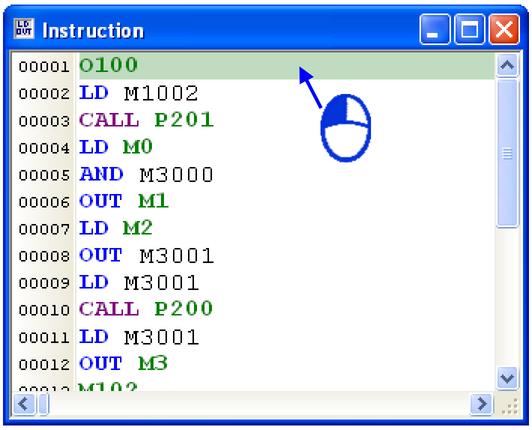
Direction
- ● Forward
- ○ Backward

- ➢ **Search for**: Users have to type the text which will be searched for, or select text which has been searched for after they click ▾ in the box.
- ➢ **Case sensitivity**: PMSoft **searches** only for words that match the case of the word that users type in the **Search for** box.
- ➢ **Whole words only**: The **Whole words only** checkbox instructs PMSoft to find complete words only, and not to find words that only contain what users type.
- ➢ **Search from caret**: If this checkbox is selected, the search will start at the text cursor. If this checkbox is not selected, the system will search the whole program.
- ➢ **Selected text only**: If this checkbox is selected, the search will start at the text selected.
- ➢ **Regular expression**: If this checkbox is selected, regular expressions can be used in the **Search for** box.
- ➢ **Direction**: Users can search down or search up.

*** A regular expression is a type of standard syntax. Please refer to related technical documents for more information.**

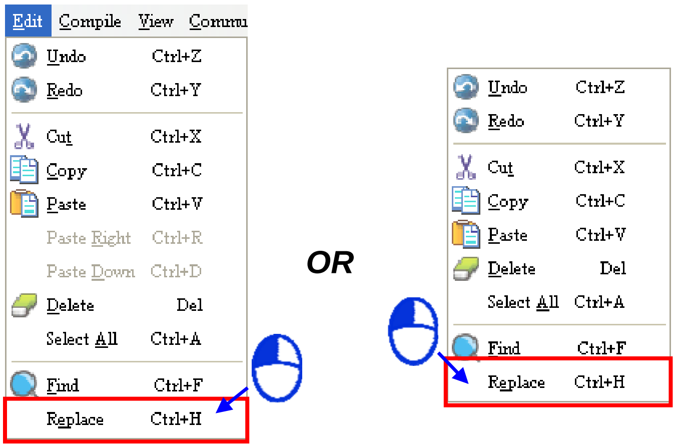
- **Replacing an object**

(1) After users open a window, they have to click an editing position, or select an amount of text.

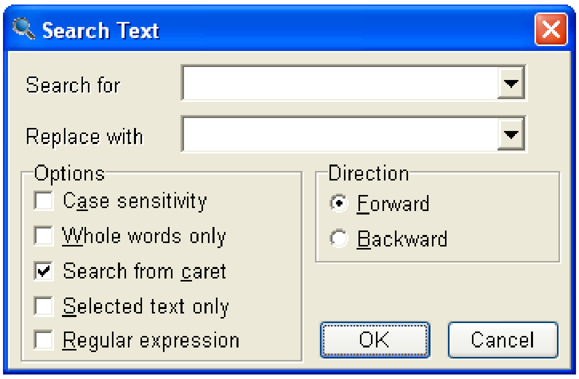


```
O100
LD M1002
CALL P201
LD M0
AND M3000
OUT M1
LD M2
OUT M3001
LD M3001
CALL P200
LD M3001
OUT M3
```

(2) The users have to click **Replace** on the **Edit** menu. The users can right-click the working area in the window, and then click **Replace** on the context menu.



(3) The users have to set search conditions in the **Search Text** window.



Search for

Replace with

Options
- ☐ Case sensitivity
- ☐ Whole words only
- ☑ Search from caret
- ☐ Selected text only
- ☐ Regular expression

Direction
- ● Forward
- ○ Backward

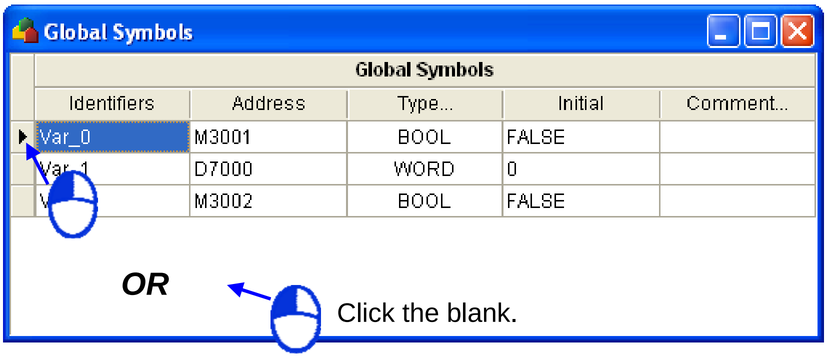
- **Search for**: Users have to type the text which will be searched for, or select text which has been searched for after they click ▾ in the box.
- **Replace With**: Users have to type replacement text, or select replacement text which has been typed after they click ▾ in the box.
- **Case sensitivity**: PMSoft searches only for words that match the case of the word that users type in the **Search for** box.
- **Whole words only**: The **Whole words only** checkbox instructs PMSoft to find complete words only, and not to find words that only contain what users type.

- ➢ **Search from caret**: If this checkbox is selected, the search will start at the text cursor. If this checkbox is not selected, the system will search the whole program.
- ➢ **Selected text only**: If this checkbox is selected, the search will start at the text selected.
- ➢ **Regular expression**: If this checkbox is selected, regular expressions can be used in the **Search for** box.
- ➢ **Direction**: Users can search down or search up.

*** A regular expression is a type of standard syntax. Please refer to related technical documents for more information.**

9.2.4 Searching for an Object in a Symbol Table

(1) After users open a window, they have to click the left side of the symbol table, or the blank in the symbol table. The users can search for an object in a global symbol table or an object in a local symbol table. Besides, the users can not replace an object in a symbol table



Global Symbols

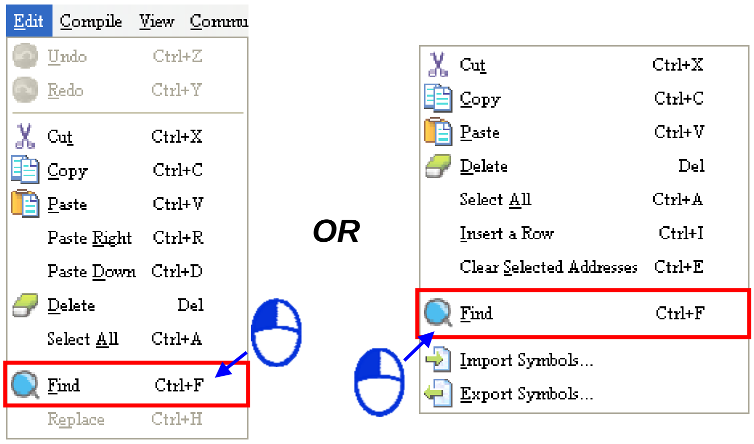
Identifiers	Address	Type...	Initial	Comment...
Var_0	M3001	BOOL	FALSE	
Var_1	D7000	WORD	0	
V	M3002	BOOL	FALSE	

OR

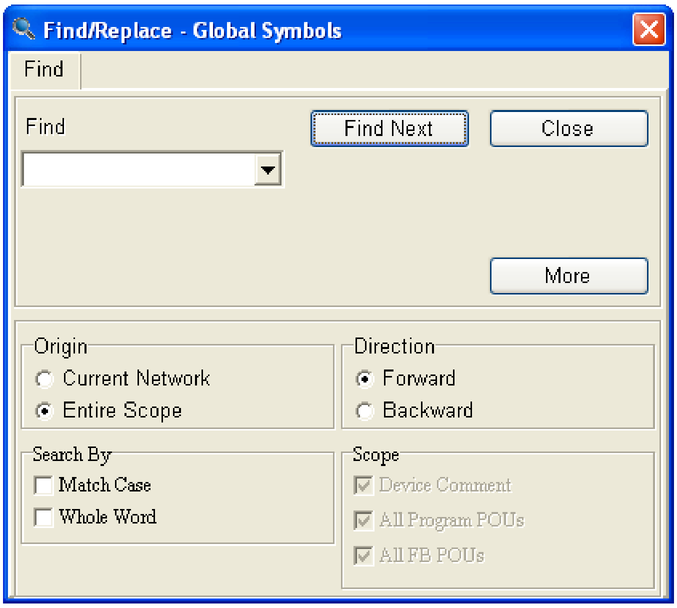
Click the blank.

(2) The users have to click **Find** on the **Edit** menu. The users can right-click the symbol table in the window, and then click **Find** on the context menu.

*** If the users click a cell in the symbol table in the previous step, the context menu which will appear after they right-click the cell will be a context menu in Windows 2000/NT/Me/XP/Vista/7.**



(3) The users have to type an object in the **Find** box in the **Find/Replace** window. They can do a search on the object typed in the **Find** box by clicking **Find Next**. If the users click **More**, they can set search conditions.



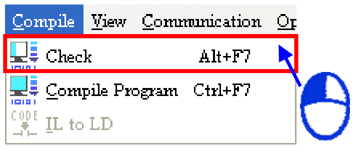
- **Find**: Users have to type the text which will be searched for, or select text which has been searched for after they click ▼ in the box.
- **Origin**: If the **Current Network** option button is clicked, the search starts at the position selected. If the **Entire Scope** option button is clicked, the search starts at the beginning of the program.
- **Direction**: Users can search down or search up.
- **Search By**: If the **Match Case** checkbox is selected, PMSoft searches only for words that match the case of the word that users type in the **Find** box. If the **Whole Word** checkbox is selected, the **Whole Word** checkbox instructs PMSoft to find complete words only, and not to find words that only contain what users type.

9.2.5 Checking and Compiling the Program in a Project

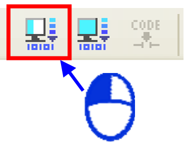
After users write a program, they have to check and compile the program. In PMSoft, checking a program and compiling a program are two independent functions. The former checks the syntax of a program. The latter will create an execution code after a program is checked. In principle, there is no regulation on the order in which the two functions are executed. The users can compile a program without checking the program. They can use the functions according to their habits. The checking/compiling of a ladder diagram, and the compiling of an instruction list are described below. Please refer to section 7.4 for more information.

- **Checking a program**

(1) Method 1: Click **Check** on the **Compile** menu.



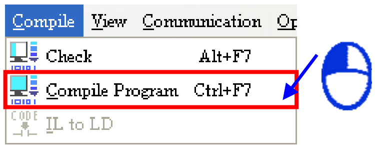
(2) Method 2: Click on the toolbar.



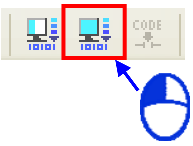
(3) Method 3: Press Alt+F7 on the keyboard.

- **Compiling a program**

(1) Method 1: Click **Compile Program** on the **Compile** menu.

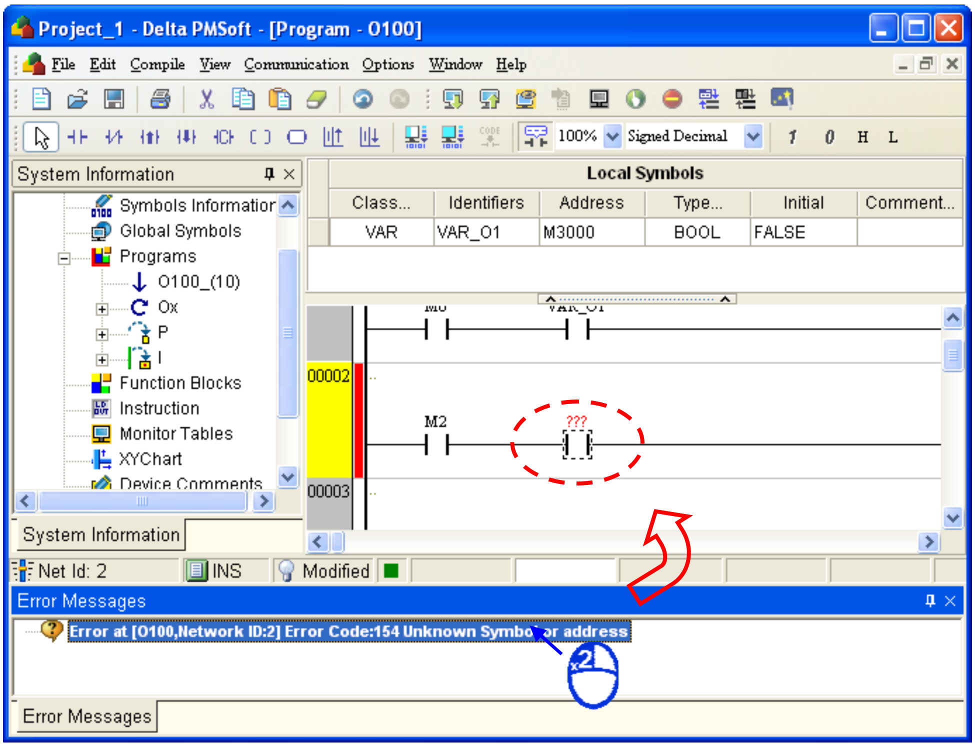


(2) Method 2: Click on the context menu.



(3) Method 3: Press Ctrl+F7 on the keyboard.

After the users check or compile a program, results will be displayed in the area under the PMSoft window. If an error occurs, the message related to the error will be displayed in this area. After the users double-click the message related to an error in this area, they will be lead to the position where the error occurs. The users can remove the error. After the users remove the error, they can check or compiled the program again.

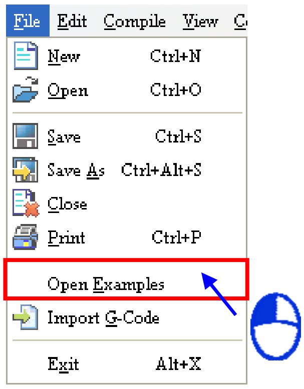


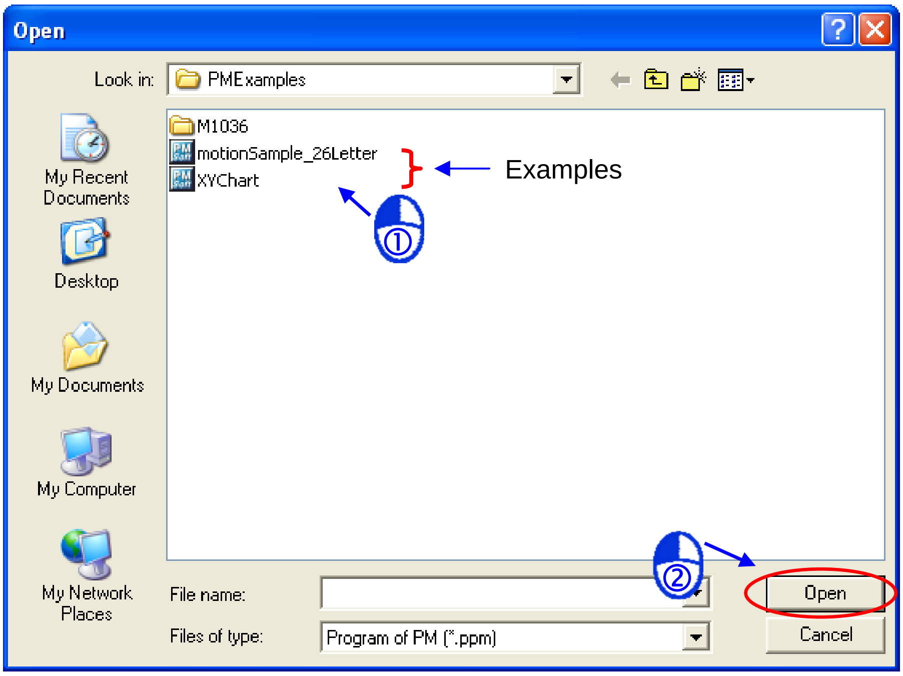
Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	VAR_O1	M3000	BOOL	FALSE	

Error at [O100,Network ID:2] Error Code:154 Unknown Symbol or address

9.3 Examples

PMSoft provides two examples for users. After the users click **Open Examples** on the **File** menu, they have to click an example in the **Open** window. There are two files in the **PMExamples** folder. They are motionSample_26Letter.ppm and XYChart.ppm.


File
Edit
Compile
View
New Ctrl+N
Open Ctrl+O
Save Ctrl+S
Save As Ctrl+Alt+S
Close
Print Ctrl+P
Open Examples
Import G-Code
Exit Alt+X


Open
Look in:
PMExamples
My Recent Documents
Desktop
My Documents
My Computer
My Network Places
M1036
motionSample_26Letter
XYChart
Examples
①
②
File name:
Files of type:
Program of PM (*.ppm)
Open
Cancel

MEMO

Chapter 10 Managing Passwords

Table of Contents

10.1 Managing Passwords in PMSoft

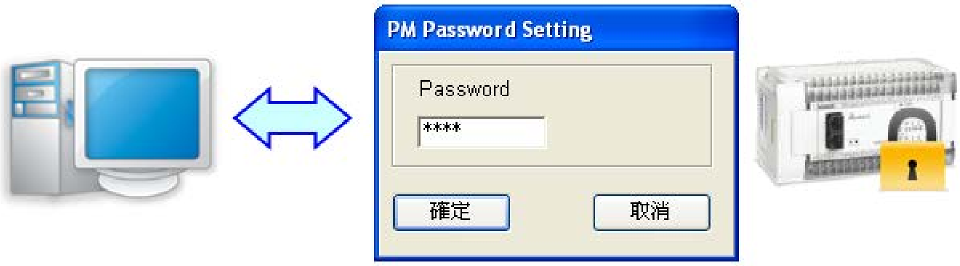
A Delta motion controller provides two protection mechanisms for internal data. A PM password and a PEP (Program Encryption Protection) password are the two protection mechanisms. The two passwords are independent from each other. Users can set or remove a PM password or a PEP password in PMSoft. When data is downloaded to a motion controller, or data is uploaded from a motion controller, a PM password protects the data in the motion controller. A PEP password protects POUs which are programs in a project.

A PM password and a PEP password are different from a POU password. A POU password protects a POU which is a function block in a project. It is used to protect the project.

***. Please refer to section 3.2 for more information about setting a POU password.**

10.2 PM Password

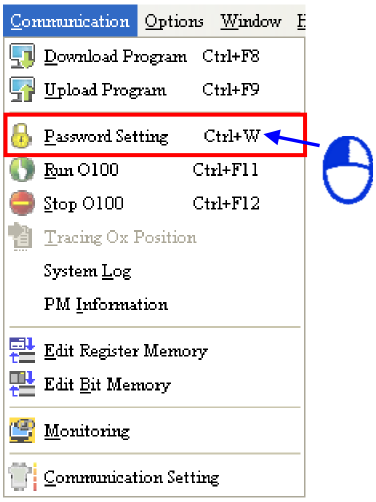
A PM password is set in a motion controller. It is used to protect the data in the motion controller. If users want to download a project from a computer to a motion controller, or upload data from a motion controller to a computer, they have to type a PM password.



10.2.1 Setting and Removing a PM Password

Before users set a PM password for a motion controller, or remove a PM password from a motion controller, they have to make sure that PMSoft is connected to the motion controller normally.

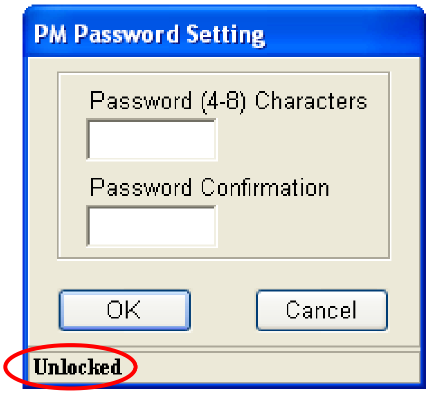
(1) The users have to click **Password Setting** on the **Communication** menu.



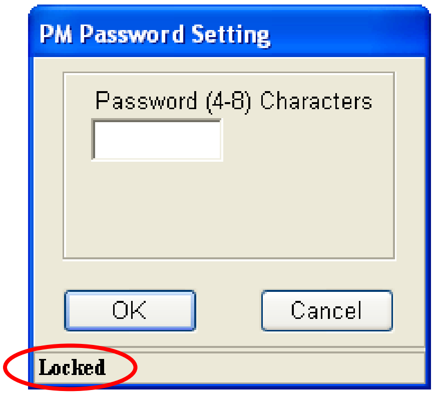
***. Users can set a PM password when they download a program. Please refer to section 10.2.2 for more information.**

(2) If **Unlocked** appears at the bottom of the **PM Password Setting** window, the motion controller is protected by a PM password. If **Locked** appears at the bottom of the **PM Password Setting** window, the motion controller is not protected by a PM password.

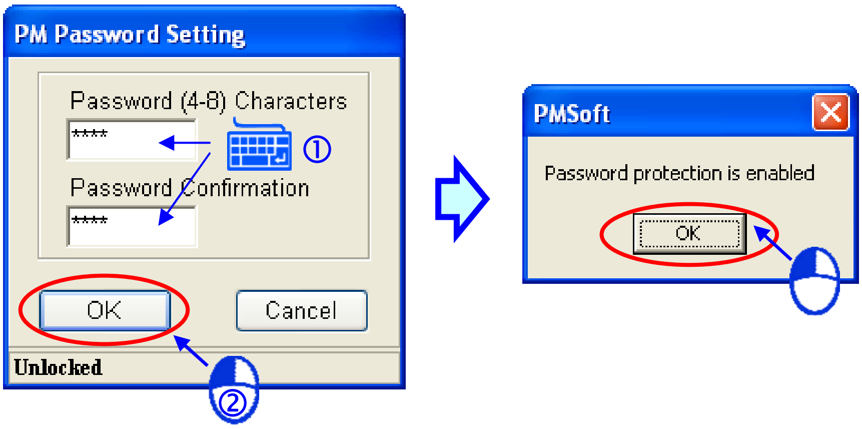
- The motion controller is not protected by a PM password



- The motion controller is protected by a PM password.

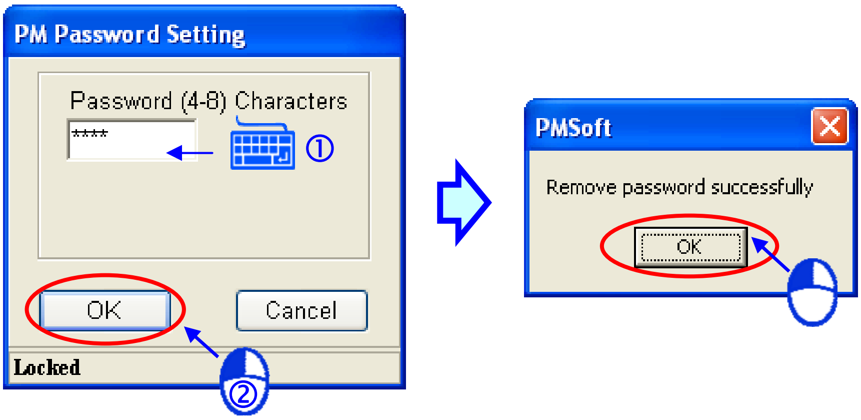


- **Setting a PM password**
  If **Unlocked** appears at the bottom of the **PM Password Setting** window, the users have to type a password twice in the **PM Password Setting** window, click OK in the **PM Password Setting** window, and click **OK** in the **PMSoft** window.



Password protection is enabled

- **Removing a PM password**
  The users have to type a correct PM password in the **PM Password Setting** window, click **OK** in the **PM Password Setting** window, and click **OK** in the **PMSoft** window.

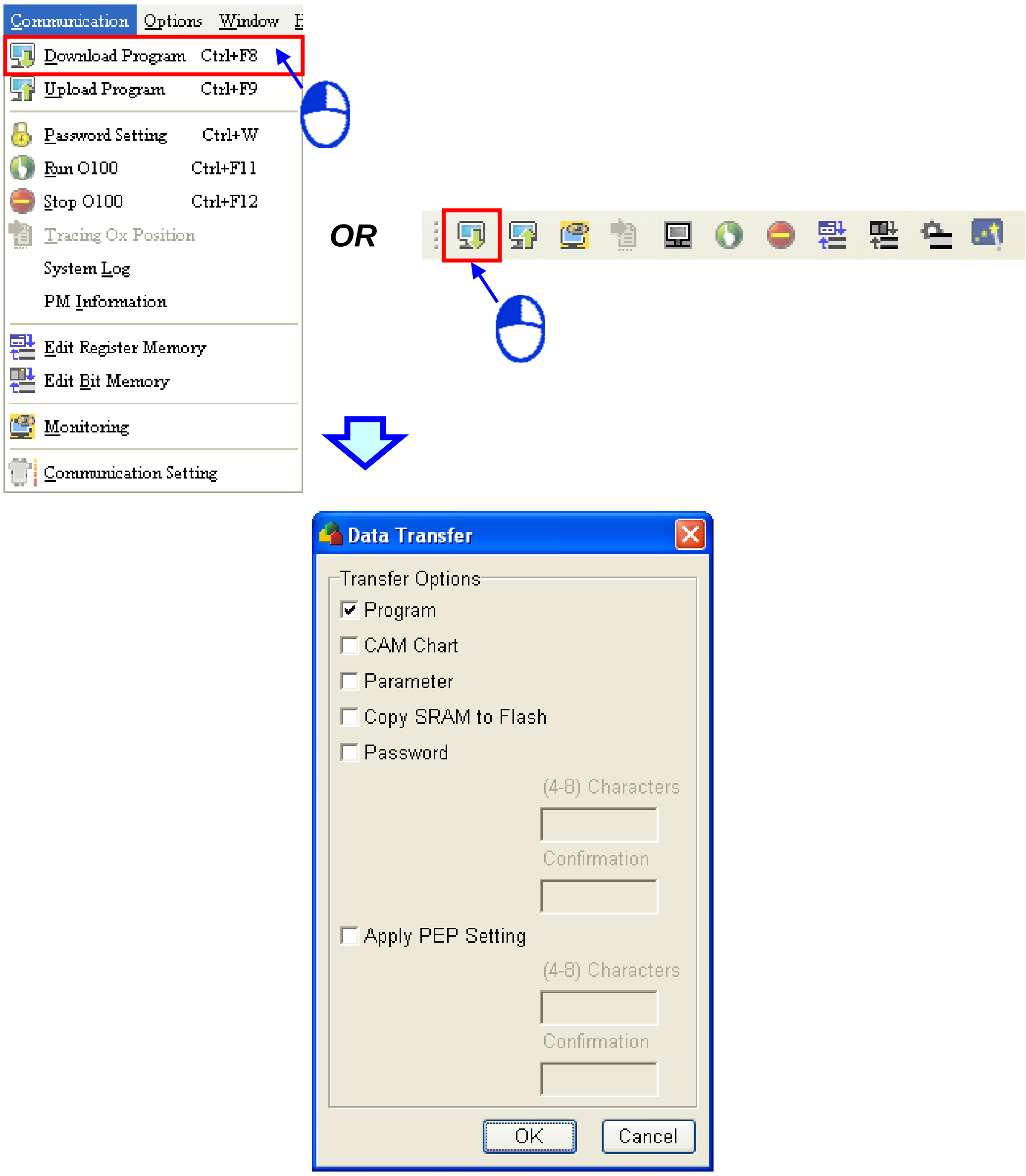


Remove password successfully

10.2.2 Downloading a Program

Before users download a program, they have to make sure that PMSoft is connected to a motion controller normally.

(1) After the users click **Download Program** on the **Communication** menu, or on the toolbar, the **Data Transfer** window will appear.



Data Transfer

Transfer Options

☑ Program

□ CAM Chart

□ Parameter

□ Copy SRAM to Flash

□ Password

(4-8) Characters

Confirmation

□ Apply PEP Setting

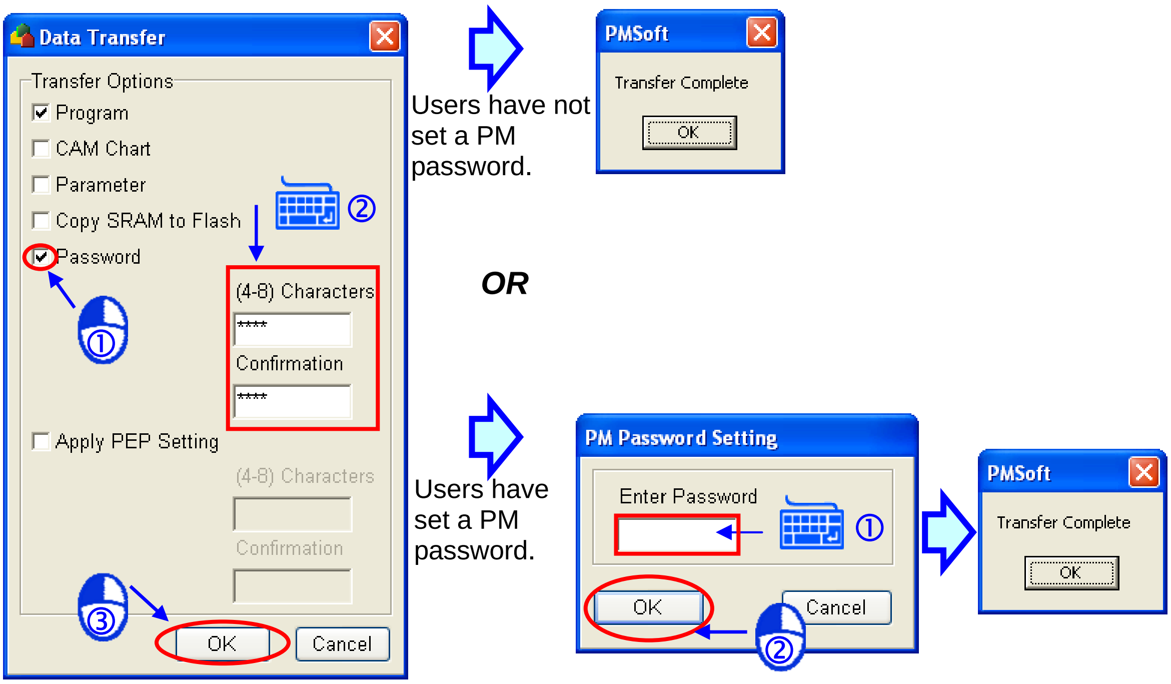
(4-8) Characters

Confirmation

OK Cancel

(2) Selecting/Unselecting the **Password** checkbox

- Selecting the **Password** checkbox, setting a password, and clicking **OK**
  - If the users have not set a PM password for the motion controller, the program and the password set in the **Data Transfer** window will be downloaded to the motion controller.
  - If the users have set a PM password, they have to type the PM password in the **PM Password Setting** window. After the users click **OK** in the **PM Password Setting** window, the program and the password set in the **Data Transfer** window will be downloaded to the motion controller.



Data Transfer

Transfer Options

☑ Program

□ CAM Chart

□ Parameter

□ Copy SRAM to Flash

☑ Password

(4-8) Characters: ****

Confirmation: ****

□ Apply PEP Setting

(4-8) Characters

Confirmation

Users have not set a PM password.

PMSoft

Transfer Complete

OR

Users have set a PM password.

PM Password Setting

Enter Password

PMSoft

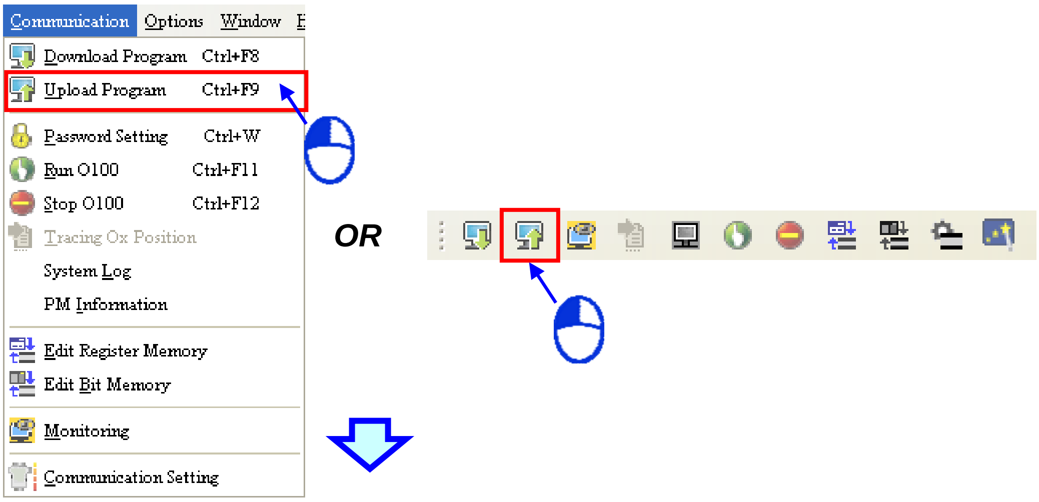
Transfer Complete

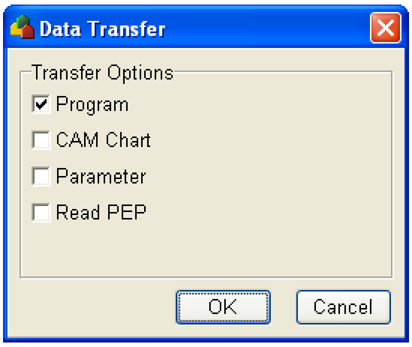
- Unselecting the **Password** checkbox, and clicking **OK**
  - If the users have not set a PM password for the motion controller, the program will be downloaded to the motion controller.
  - If the users have set a PM password, they have to type the PM password in the **PM Password Setting** window. After the users click **OK** in the **PM Password Setting** window, the program will be downloaded to the motion controller.

10.2.3 Uploading a Program

Before users upload a program, they have to make sure that PMSoft is connected to a motion controller normally.

(1) After the users click **Upload Program** on the **Communication** menu, or on the toolbar, the **Data Transfer** window will appear.





Data Transfer

Transfer Options

☑ Program

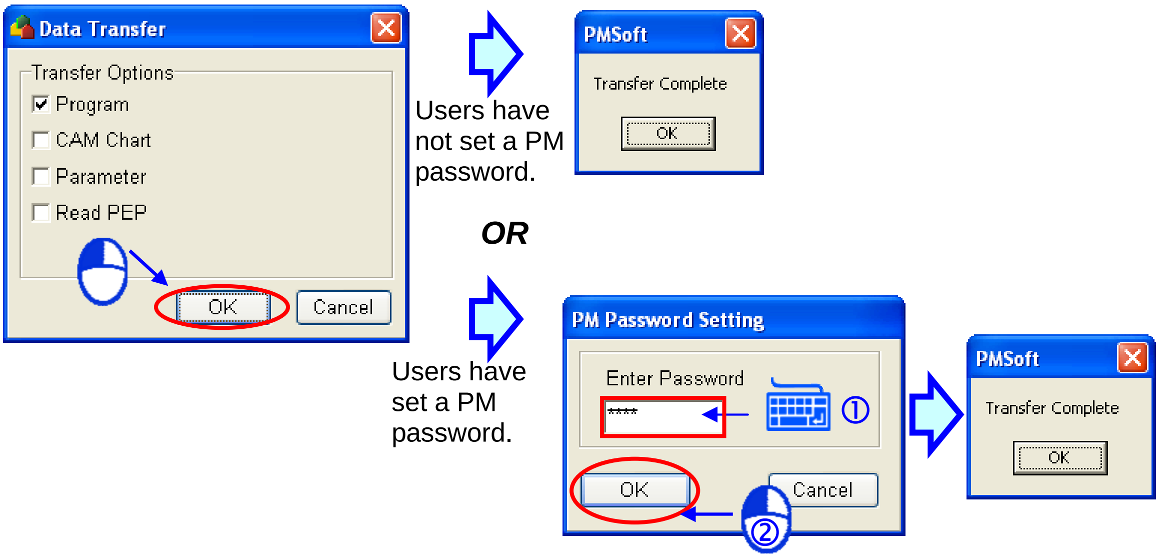
□ CAM Chart

□ Parameter

□ Read PEP

(2) Clicking **OK**

- If the users have not set a PM password for the motion controller, the program will be uploaded to PMSoft.
- If the users have set a PM password, they have to type the PM password in the **PM Password Setting** window. After the users click **OK** in the **PM Password Setting** window, the program will be uploaded to PMSoft.



Data Transfer

Transfer Options

☑ Program

□ CAM Chart

□ Parameter

□ Read PEP

Users have not set a PM password.

PMSoft

Transfer Complete

OR

Users have set a PM password.

PM Password Setting

Enter Password: ****

PMSoft

Transfer Complete

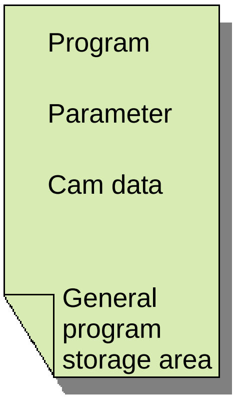
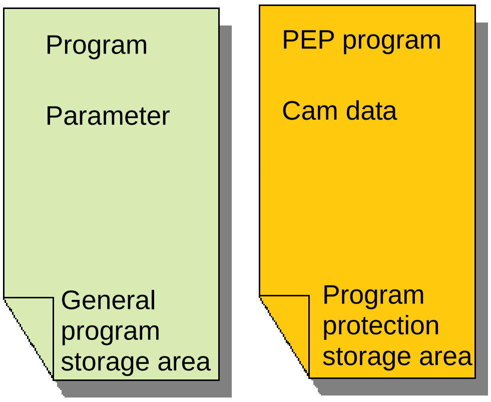
10.3 PEP Password

Users can protect POUs which are programs by means of setting a PEP password. They can decide whether to protect the main program O100, the Ox motion subroutines Ox0~Ox99, and the P subroutines P0~P255. Before the users download the POUs which are programs in a project to a motion controller, they can set a PEP password. After the PEP password is downloaded to the motion controller, the POUs which the users decide to protect will be protected by the PEP password. If the users want to upload the POUs which are protected by the PEP password, they have to type the PEP password.

10.3.1 Storage Areas in a Motion Controller

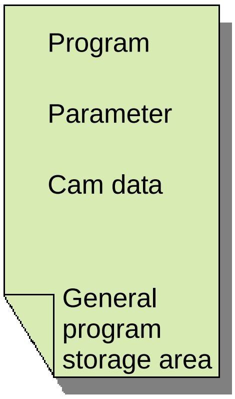
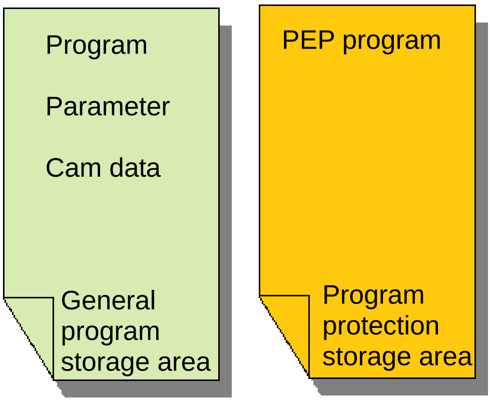
The memory in a Delta motion controller is divided into two parts. One part is a general program storage area, and the other part is a program protection storage area. The data stored in the general program storage area is the data which is not protected by a PEP password, and the data stored in the program protection storage area is the data protected by a PEP password. There are two types of Delta motion controllers. They are DVP series motion controllers and AH500 series motion controllers. The division of the memory in a DVP series motion controller is different from the division of the memory in an AH500 series motion controller.

The storage areas in a DVP series motion controller are shown below. If users set a PEP password, the general program storage area will not be protected by the PEP password, the POUs which are not protected by the PEP password and the values of the parameters will be stored in the general program storage area, and the POUs which are protected by the PEP password and the cam data will be stored in the program protection storage area. If the users want to upload data from the motion controller, or download data to the motion controller, they have to type the PEP password.

DVP series motion controller	
Users have not set a PEP password.	Users have set a PEP password.
	

Program
Parameter
Cam data
General program storage area
Program
Parameter
General program storage area
PEP program
Cam data
Program protection storage area

If users set a PEP password for an AH500 series motion controller, the cam data will be stored in the general program storage area, and the POUs which are protected by the PEP password will be stored in the program protection storage area.

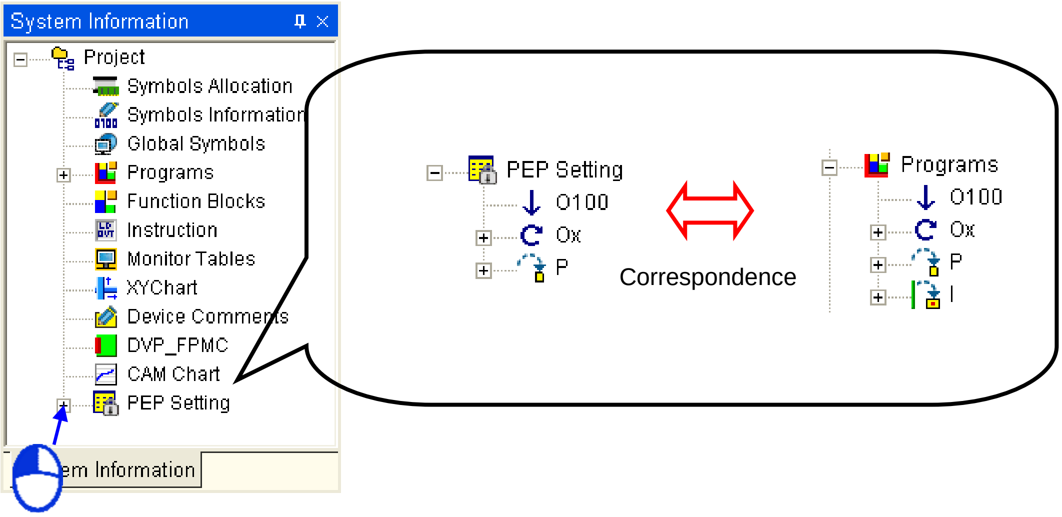
AH500 series motion controller	
Users have not set a PEP password.	Users have set a PEP password.
	

Program
Parameter
Cam data
General program storage area
Program
Parameter
Cam data
General program storage area
PEP program
Program protection storage area

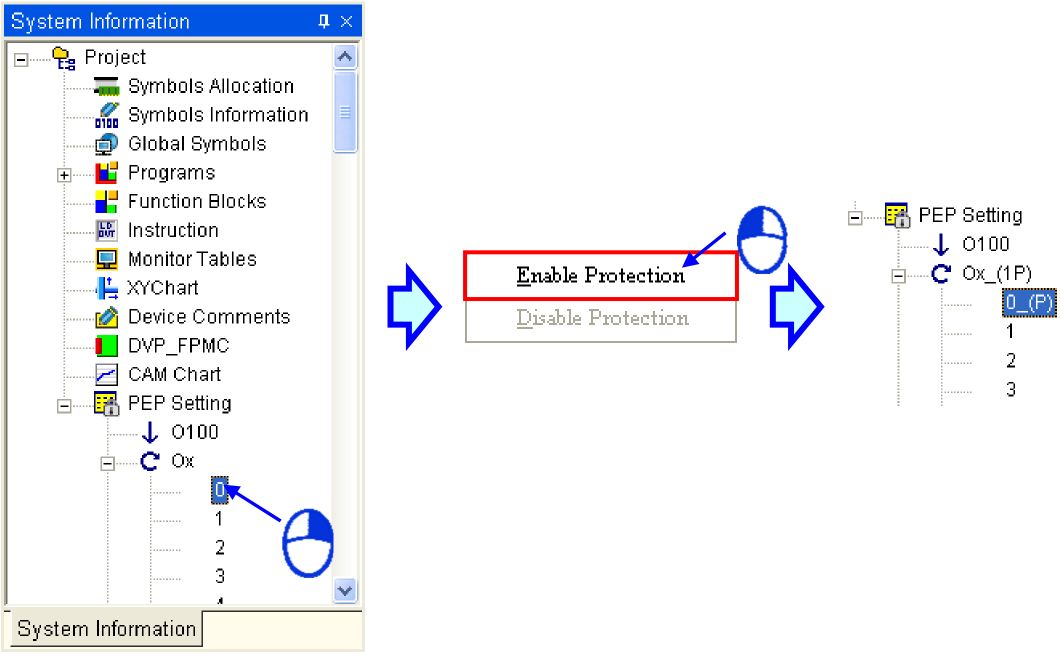
***. If there is cam data in an AH500 series motion controller, and a PEP password has been set for the AH500 series motion controller, users are asked to type the PEP password when they download cam data to the AH500 series motion controller.**

10.3.2 Enabling the Protection of POUs

(1) Users have to expand the **PEP Setting** section in the system information area. The POU numbers in the **PEP Setting** section correspond to the POU numbers in the **Programs** section. However, the I interrupt subroutines in a project can not be protected by a PEP password.



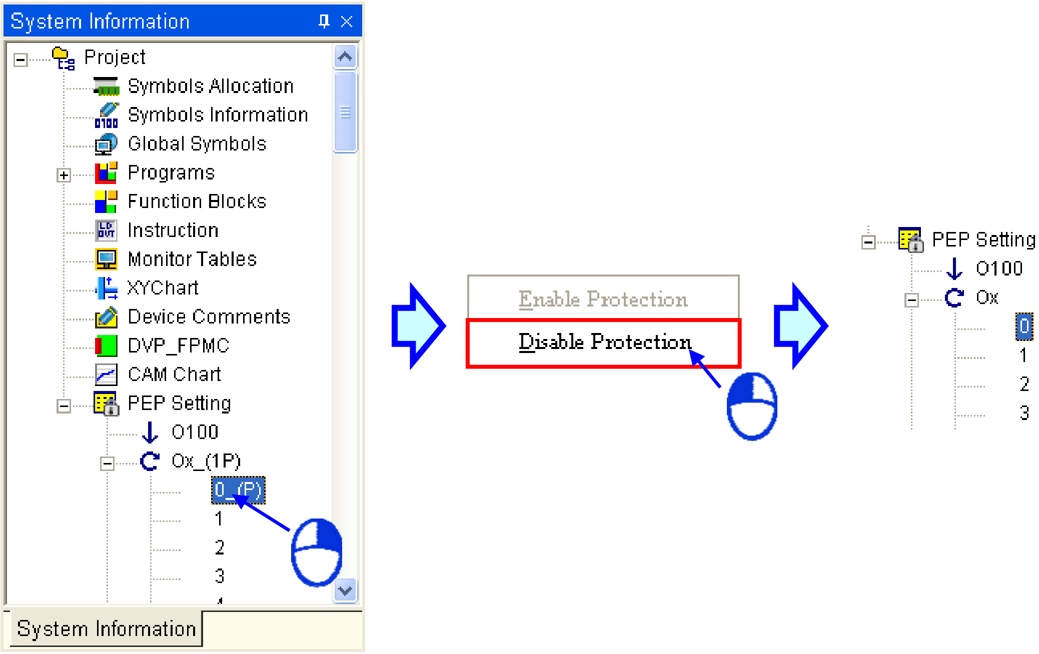
(2) After the users right-click a POU number under a program type, and click **Enable Protection** on the context menu, the POU number will be marked with "_(P)" (the POU represented by the POU number will be protected by a PEP password), and the program type will be marked with "_(nP)". ("n" in "_(nP)" indicates the number of POUs which are protected by a PEP password.) In the example below, only Ox0 is protected, and therefore Ox is marked with "_(1P)".



***. In this section, the users only specify a POU which will be protected by a PEP password in the project. After the users download the POUs created in the project, and set a PEP password, the POU specified will be protected by the PEP password.**

10.3.3 Disabling the Protection of POUs

(1) After users right-click a POU number which represents a POU protected by a PEP password under a program type, and click **Disable Protection** on the context menu, "_(P)" at the right side of the POU number will be removed (the POU represented by the POU number will not be protected by a PEP password), and "n" in "_(nP)" at the right side of the program type will decrease by 1 (the number of POUs protected by a PEP password will decrease by 1). If "n" in "_(nP)" at the right side of a program type becomes 0, "_(nP)" will be removed.

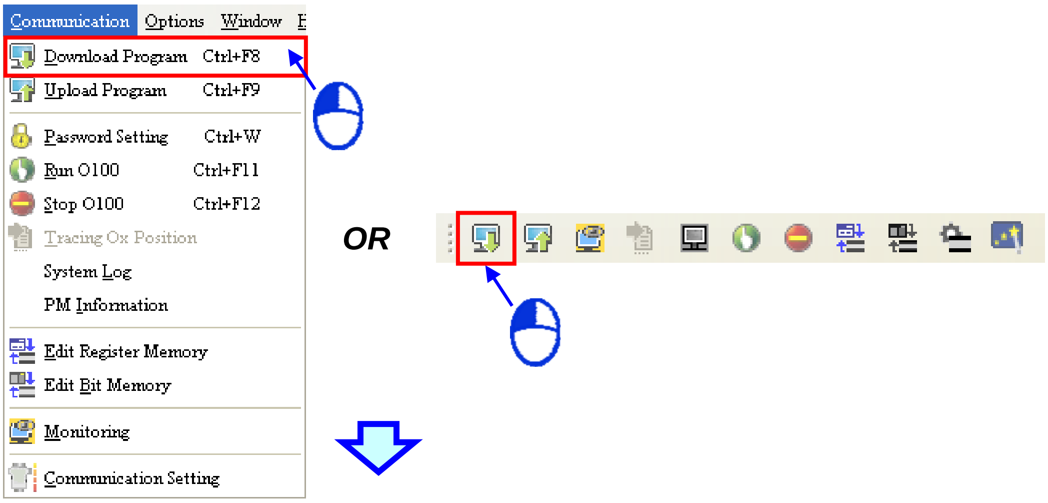


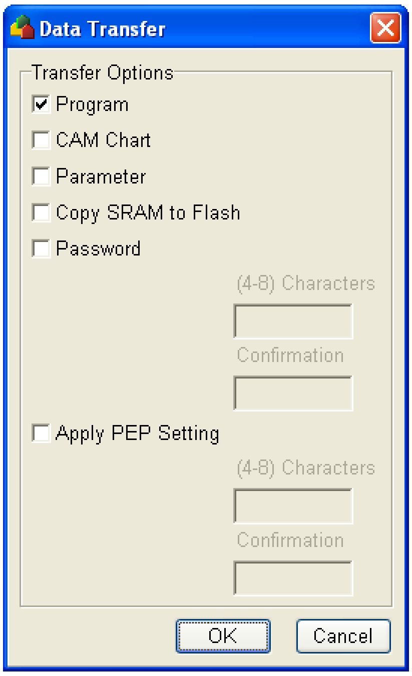
*. In this section, the users only disable the protection of a POU. Please refer to section 10.3.6 for more information about removing a PEP password.

10.3.4 Downloading a Program

Before users download the program in a project, they have to make sure that PMSoft is connected to a motion controller normally.

(1) After the users click **Download Program** on the **Communication** menu, or on the toolbar, the **Data Transfer** window will appear.





Data Transfer

Transfer Options

☑ Program

□ CAM Chart

□ Parameter

□ Copy SRAM to Flash

□ Password

(4-8) Characters

Confirmation

□ Apply PEP Setting

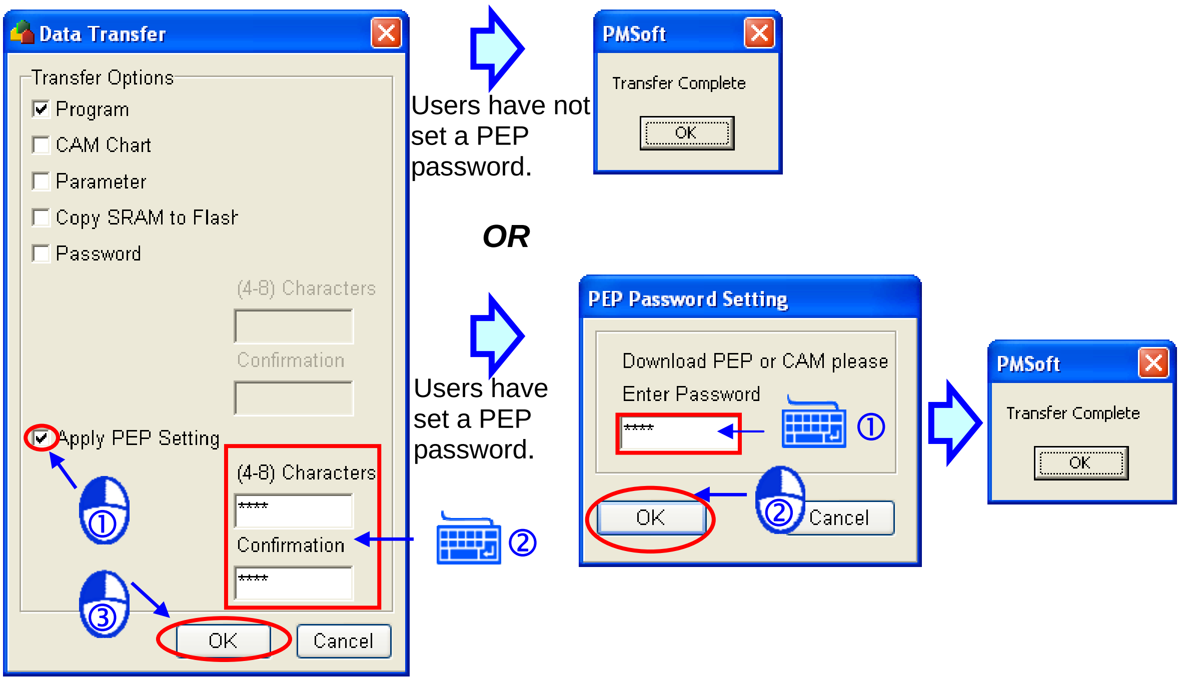
(4-8) Characters

Confirmation

OK Cancel

(2) Selecting/Unselecting the **Apply PEP Setting** checkbox

- Selecting the **Apply PEP Setting** checkbox, setting a password, and clicking **OK**
  - ➢ If the users have not set a PEP password for the motion controller, the program and the password set in the **Data Transfer** window will be downloaded to the motion controller, and the data downloaded will replace the data in the motion controller.
  - ➢ If the users have set a PEP password, they have to type the PEP password in the **PEP Password Setting** window. After the users click **OK** in the **PEP Password Setting** window, the program and the password set in the **Data Transfer** window will be downloaded to the motion controller, and the data downloaded will replace the data in the motion controller (the POUs which are not protected by the PEP password will be stored in the general program storage area, and the POUs which are protected by the PEP password will be stored in the program protection storage area).



Data Transfer

Transfer Options

☑ Program

□ CAM Chart

□ Parameter

□ Copy SRAM to Flash

□ Password

(4-8) Characters

Confirmation

☑ Apply PEP Setting

(4-8) Characters

****

Confirmation

****

OK Cancel

Users have not set a PEP password.

PMSoft

Transfer Complete

OK

OR

Users have set a PEP password.

PEP Password Setting

Download PEP or CAM please Enter Password

****

OK Cancel

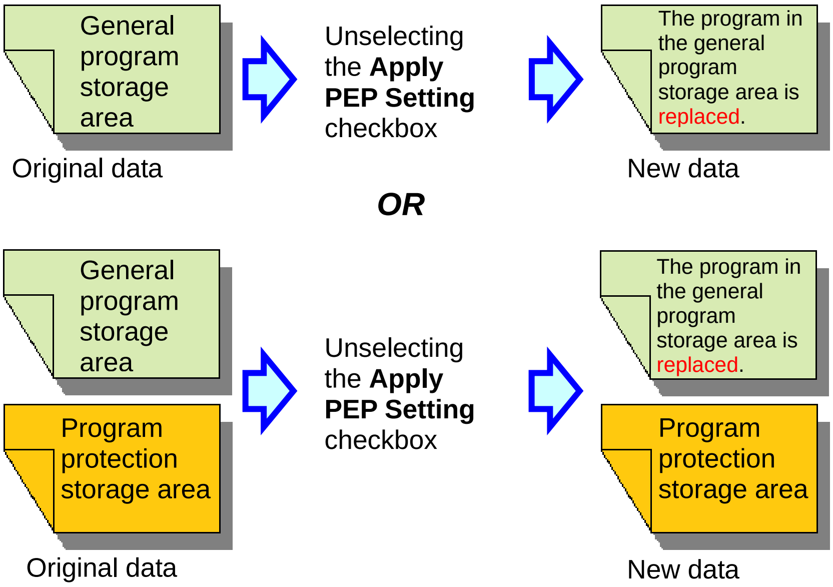
PMSoft

Transfer Complete

OK

  - If the motion controller used is a DVP series motion controller, the cam chart in the project will be downloaded. The **CAM Chart** checkbox in the **Data Transfer** window can not be unselected. The data which can be protected by a PEP password varies with the motion controller used. The storage area where cam data is stored varies with the motion controller used. Please refer to section 10.3.1 for more information.

- Unselecting the **Apply PEP Setting** checkbox, and clicking **OK**
  - If the users have not set a PEP password for the motion controller, the program in the general program storage area will be replaced by the program downloaded.
  - If the users have set a PEP password, the program in the program protection area will be retained, and the program in the general program storage area will be replaced by the program downloaded. The original data in the motion controller and the new data in the motion controller are shown below.

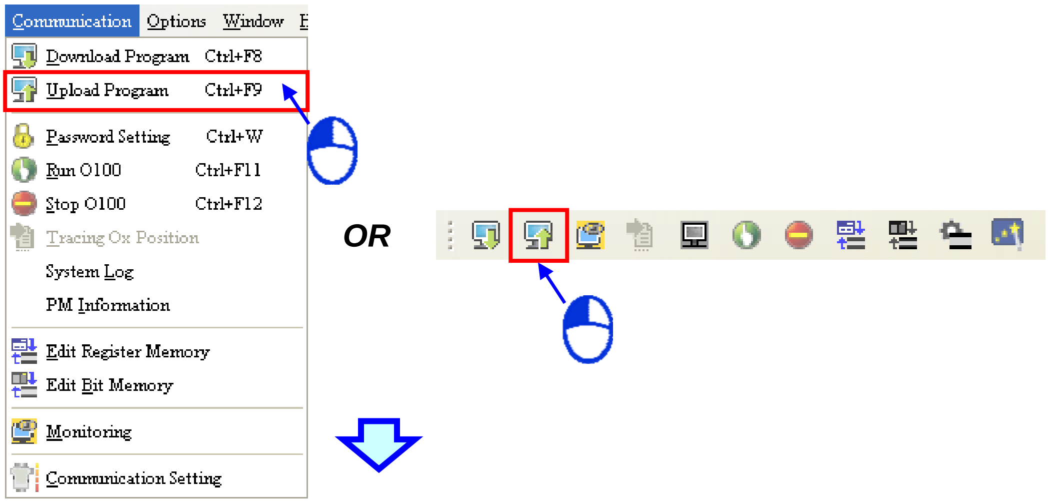

General program storage area
Original data
Unselecting the Apply PEP Setting checkbox
The program in the general program storage area is replaced.
New data
OR
General program storage area
Program protection storage area
Original data
Unselecting the Apply PEP Setting checkbox
The program in the general program storage area is replaced.
Program protection storage area
New data

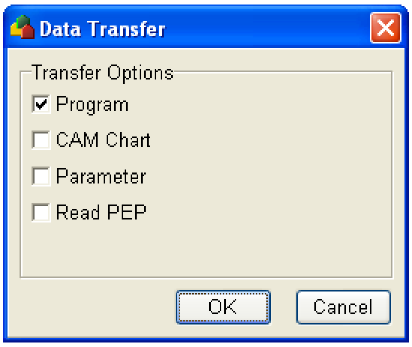
  - If the motion controller used is a DVP series motion controller, and the users have set a PEP password, the users are asked to type the PEP password when they download the cam data in the project. If the **Apply PEP Setting** checkbox is unselected, the PEP password which has been set is removed when the cam data in the project is downloaded. If the **Apply PEP Setting** checkbox is selected, the password set in the **Data Transfer** window will replace the PEP password which has been set.

10.3.5 Uploading a Program

Before users upload a program, they have to make sure that PMSoft is connected to a motion controller normally.

(1) After the users click **Upload Program** on the **Communication** menu, or on the toolbar, the **Data Transfer** window will appear.





Data Transfer

Transfer Options

☑ Program

□ CAM Chart

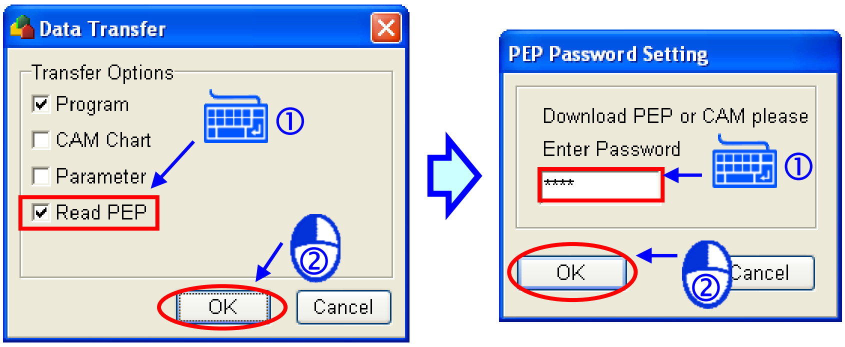
□ Parameter

□ Read PEP

OK Cancel

(2) Selecting/Unselecting the **Read PEP** checkbox

- Selecting the **Read PEP** checkbox, and clicking **OK**
  - ➢ If the users have not set a PEP password for the motion controller, the data in the general program storage area will be uploaded to PMSoft.
  - ➢ If the users have set a PEP password for the motion controller, they have to type the PEP password in the **PEP Password Setting** window. After the users click **OK** in the **PEP Password Setting** window, the data in the general program storage area and the data in the program protection storage area will be uploaded to PMSoft.



Data Transfer

Transfer Options

☑ Program

□ CAM Chart

□ Parameter

☑ Read PEP

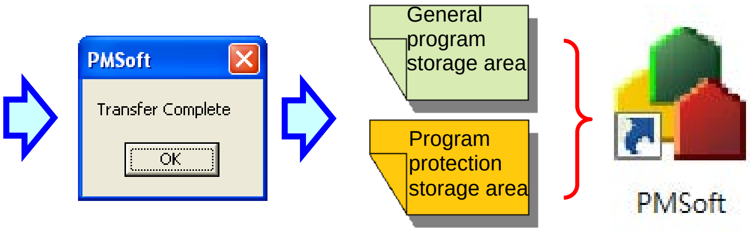
OK Cancel

PEP Password Setting

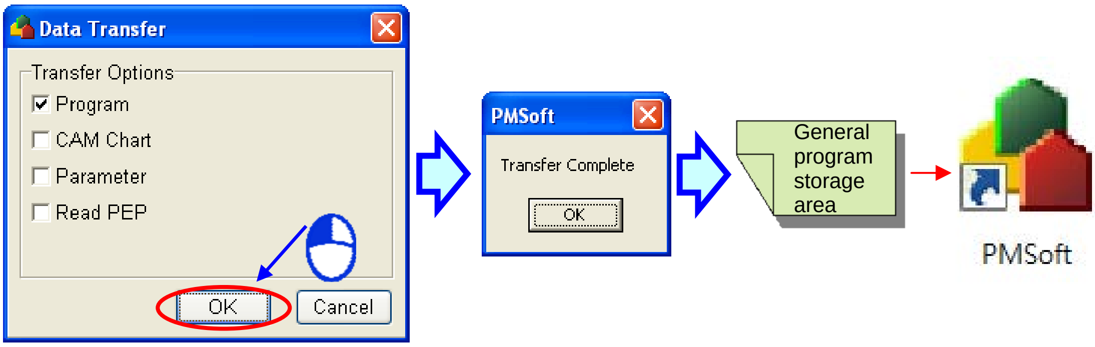
Download PEP or CAM please Enter Password

****

OK Cancel



- Unselecting the **Read PEP** checkbox, and clicking **OK**
  - ➢ The data in the general program storage area will be uploaded to PMSoft, whether the users have set a PEP password for the motion controller.



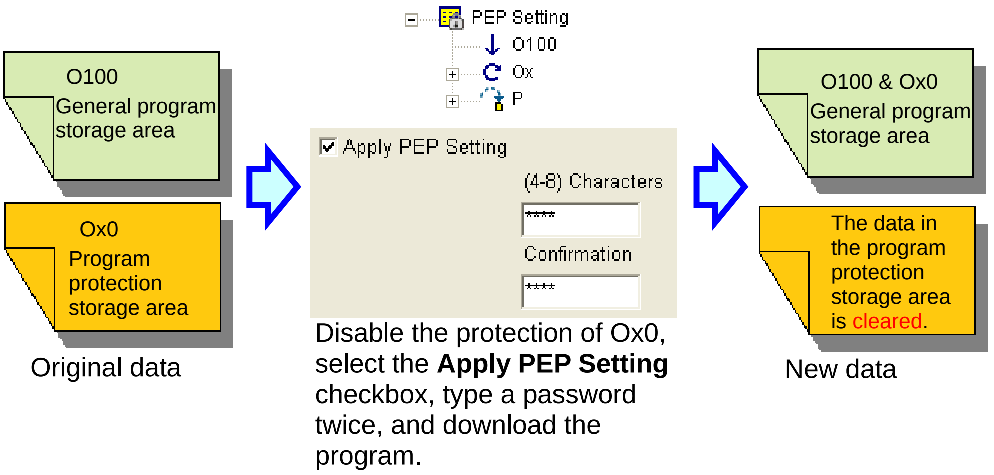
  - ➢ If the motion controller used is a DVP series motion controller, and the users have set a PEP password, the users are asked to type the PEP password when they upload the cam data in the motion controller. If the **Read PEP** checkbox is unselected, the POUs which are protected by the PEP password will not be uploaded, and the POUs which are not protected by the PEP password and the cam data will be uploaded. If the **Read PEP** checkbox is selected, all the data will be uploaded.

10.3.6 Important Points about Disabling the Protection of POUs

In a motion controller, the POUs which are protected by a PEP password are stored in the general program storage area, and the POUs which are not protected by a PEP password are stored in the program protection storage area. Once the POUs which are protected by a PEP password in a project are downloaded to a motion controller, users can not disable the protection of the POUs stored in the program protection storage area even if the users disable the protection of the POUs in the project and download the POUs. If the users want to disable the protection of the POUs stored in the program protection storage area, they have to disable the protection of the POUs in the project, and select the **Apply PEP Setting** checkbox in the **Data Transfer** window. The user can disable the protection of the POUs stored in the program protection storage area by returning the motion controller to the factory setting.

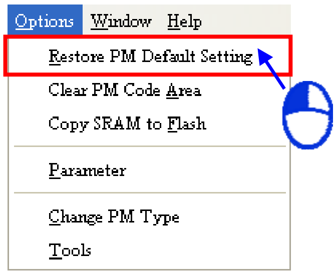
- Disabling the protection of POUs in a project, and selecting the **Apply PEP Setting** checkbox in the **Data Transfer** window

  The POUs in the program protection storage area will be cleared only if the **Apply PEP Setting** checkbox in the **Data Transfer** window is selected. After users disable the protection of POUs in a project, select the **Apply PEP Setting** checkbox in the **Data Transfer** window, and download the program in the project to a motion controller, the data in the program protection storage area will be cleared, the POUs which are protected by a PEP password in the project will be stored in the program protection storage area, the POUs in the general program storage area will be cleared, and the POUs which are not protected by a PEP password in the project will be stored in the general storage area. This method can be used if a POU which is protected by a PEP password has to be added. In the figure below, O100 is stored in the general program storage area, and Ox0 is stored in the program protection storage area. After the protection of Ox0 is disabled, and the **Apply PEP Setting** checkbox in the **Data Transfer** window is selected, O100 and Ox0 will be stored in the general program storage area, and the data in the program protection area will be cleared.


O100
General program storage area
Ox0
Program protection storage area
Original data
PEP Setting
O100
Ox
P
☑ Apply PEP Setting
(4-8) Characters
****
Confirmation
****
O100 & Ox0
General program storage area
The data in the program protection storage area is cleared.
New data

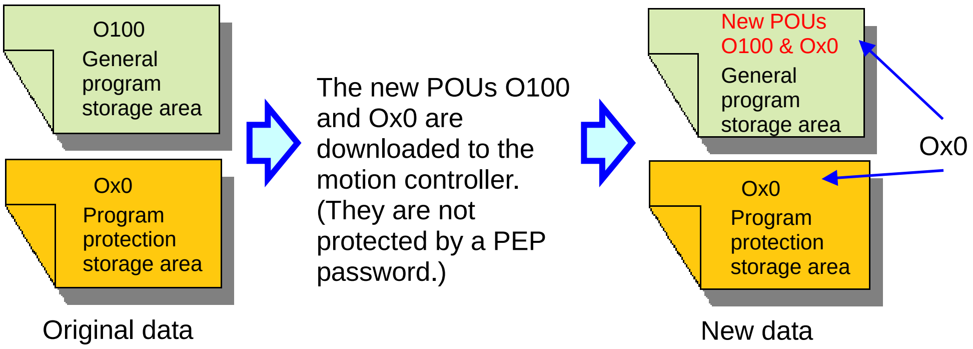
Disable the protection of Ox0, select the **Apply PEP Setting** checkbox, type a password twice, and download the program.

- Returning a motion controller to its factory setting
  If users want to delete the data and the setting in a motion controller, they have to click **Restore PM Default Settin**g on the **Options** menu

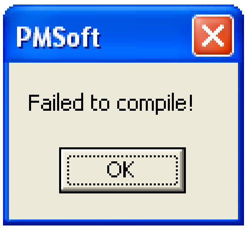


If the users do not use one of the methods described above to clear the POUs in the program protection storage area, and they download the same POUs again, the motion controller may not operate normally.

In the figure below, O100 is stored in the general program storage area, and Ox0 is stored in the program protection storage area. After the data in the general program storage area is cleared, the new main program O100, and the new motion subroutine Ox0 (which is not protected by a PEP password) will be downloaded to the general program storage area, and the original motion subroutine Ox0 in the program protection storage area will be retained. If the motion controller is started, it may not operate normally.


O100
General program storage area
Ox0
Program protection storage area
Original data
The new POUs O100 and Ox0 are downloaded to the motion controller. (They are not protected by a PEP password.)
New POUs O100 & Ox0
General program storage area
Ox0
Ox0
Program protection storage area
New data

If the program is uploaded to PMSoft, the error message shown below will appear. As a result, it is suggested that users should return a motion controller to its factory setting before they download a new program to the motion controller.



Failed to compile!

***. If the motion controller used is a DVP series motion controller, and users do not select the Apply PEP Setting checkbox, they are asked to type a PEP password when they download the cam data in a project.**

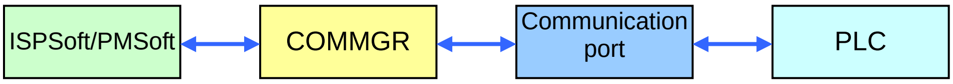
MEMO

Chapter 11 Online Functions

Table of Contents

11.1 Online Functions

After users connect a computer to a motion controller in accordance with chapter 2, they can execute the online functions of the motion controller through PMSoft. The online functions of a motion controller are the uploading of a program, the downloading of a program, the reading of data from registers, the writing data to registers, the starting of the motion controller, the stopping of the motion controller, the monitoring of the motion controller. They are described in this chapter. The functions that an AH500 series motion controller supports are different from the functions that a DVP series motion controller supports. The difference will be described in the following sections.

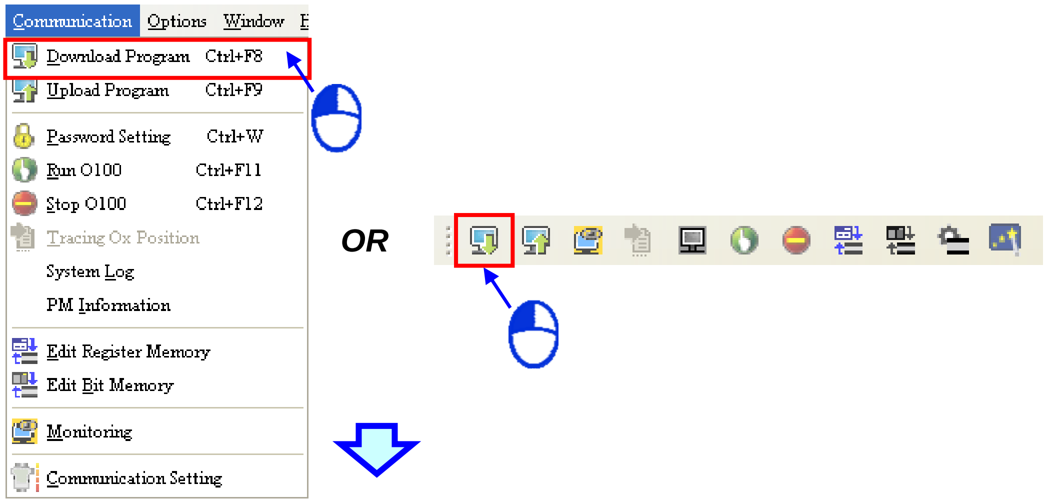


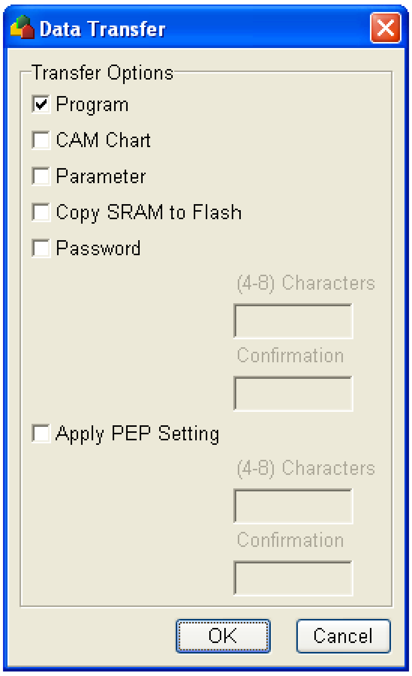
11.2 Online Operation

11.2.1 Downloading / Uploading a Project

- **Downloading data**

  After users click **Download Program** on the **Communication** menu, or on the toolbar, the **Data Transfer** window will appear. The users have to select checkboxes in the **Data Transfer** window, and click **OK**.





Data Transfer

Transfer Options

☑ Program

□ CAM Chart

□ Parameter

□ Copy SRAM to Flash

□ Password

(4-8) Characters

Confirmation

□ Apply PEP Setting

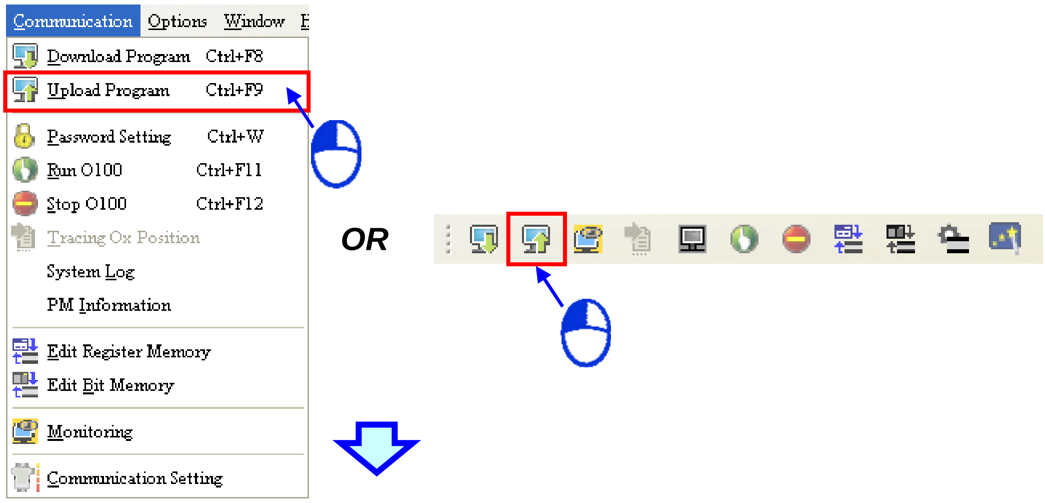
(4-8) Characters

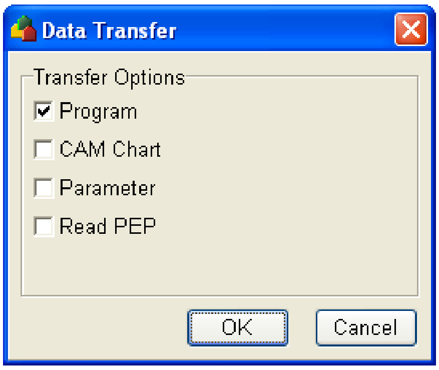
Confirmation

- **Program**: Execution code
- **CAM Chart**: The storage area in which a cam chart is stored depends on the model used. Please refer to section 10.3.1 for more information.
- **Parameter**: The parameters which are related to motion control are downloaded. Please refer to chapter 2 for more information.
- **Copy SRAM to Flash**: The data in the SRAM in a motion controller is copied into the flash memory in the motion controller when a project is downloaded.
- **Password**: Please refer to chapter 10 for more information about setting a PM **password**.
- **Apply PEP Setting**: Please refer to chapter 10 for more information about setting a PEP password.

- **Uploading a project**

After users click **Upload Program** on the **Communication** menu, or on the toolbar, the **Data Transfer** window will appear. The users have to select checkboxes in the **Data Transfer** window, and click **OK**.





Data Transfer

Transfer Options

☑ Program

□ CAM Chart

□ Parameter

□ Read PEP

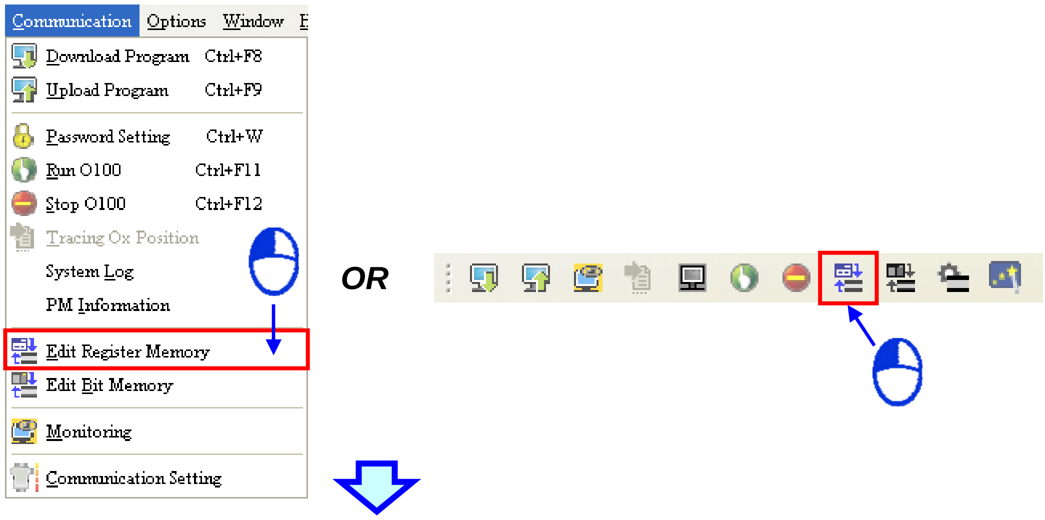
OK Cancel

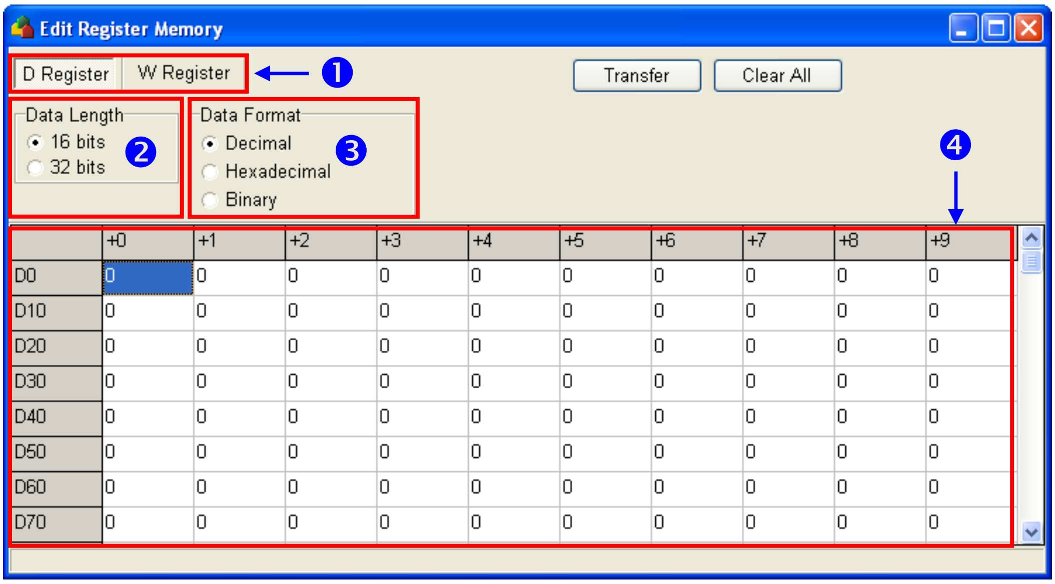
- **Program**: Execution code
- **CAM Chart**: The cam chart in a motion controller is uploaded. The storage area from which a cam chart is uploaded depends on the model used. Please refer to section 10.3.1 for more information.
- **Parameter**: The parameters which are related to motion control are uploaded. Please refer to chapter 2 for more information.
- **Read PEP**: The POUs which are protected by a PEP password in the program protection storage area in a motion controller is read. Please refer to chapter 10 for more information.

11.2.2 Editing Registers

User can change or read the values in the D registers and the W registers in a motion controller. There are D registers and W registers in an AH500 series motion controller. There are D registers in a DVP series motion controller.

(1) The users have to click **Edit Register Memory** on the **Communication** menu, or on the toolbar.





Edit Register Memory

D Register | W Register

Transfer | Clear All

Data Length
● 16 bits
○ 32 bits

Data Format
● Decimal
○ Hexadecimal
○ Binary

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9
D0	0	0	0	0	0	0	0	0	0	0
D10	0	0	0	0	0	0	0	0	0	0
D20	0	0	0	0	0	0	0	0	0	0
D30	0	0	0	0	0	0	0	0	0	0
D40	0	0	0	0	0	0	0	0	0	0
D50	0	0	0	0	0	0	0	0	0	0
D60	0	0	0	0	0	0	0	0	0	0
D70	0	0	0	0	0	0	0	0	0	0

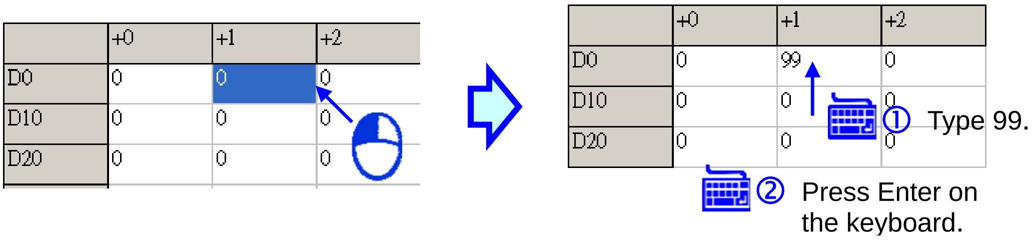
❶ The users can switch between the **D Register** tab and the **W Register** tab.

❷ The users can select the **16 bits** option button or the **32 bits** option button. If they select the **32 bits** option button, a value will occupy two consecutive registers.

❸ The users can select the **Decimal** option button, the **Hexadecimal** option button, or the **Binary** option button. If a value that the users type does not match the option button selected, an error message will appear.

❹ The users can change the values in the registers. There are ten registers in a row. For example, D0~D9 are in the first row.

(2) The users have to click a tab, select an option button in the **Data Length** section, and select an option button in the **Data Format** section. After the users click a cell, they have to type a value in the cell, and press Enter on the keyboard. If the users want to change the values in the cells to 0, they have to click **Clear All**.



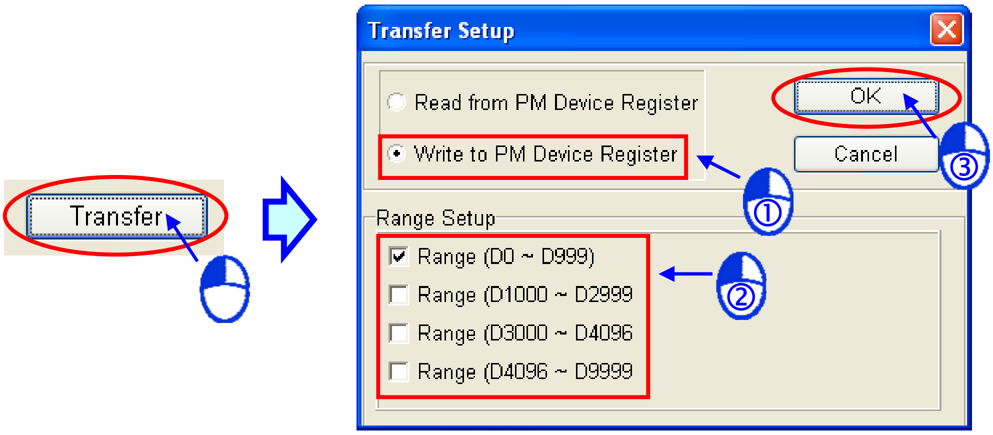
	+0	+1	+2
D0	0	0	0
D10	0	0	0
D20	0	0	0

	+0	+1	+2
D0	0	99	0
D10	0	0	0
D20	0	0	0

① Type 99.

② Press Enter on the keyboard.

(3) After the users type values in cells, they have to click **Transfer**. In the **Transfer Setup** window, the users have to select the **Write to PM Device Register** option button, and select a range in the **Range Setup** section. After the users click **OK** in the **Transfer Setup** window, the values in cells will be written into to the motion controller connected to PMSoft.



Transfer Setup

○ Read from PM Device Register

● Write to PM Device Register

OK | Cancel

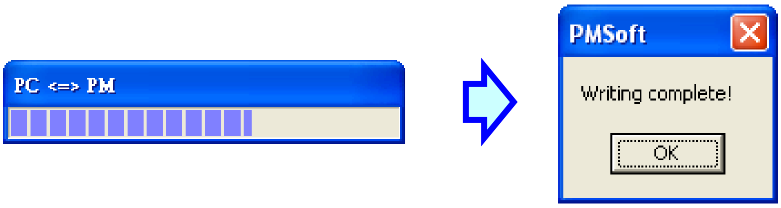
Range Setup

☑ Range (D0 ~ D999)

☐ Range (D1000 ~ D2999

☐ Range (D3000 ~ D4096

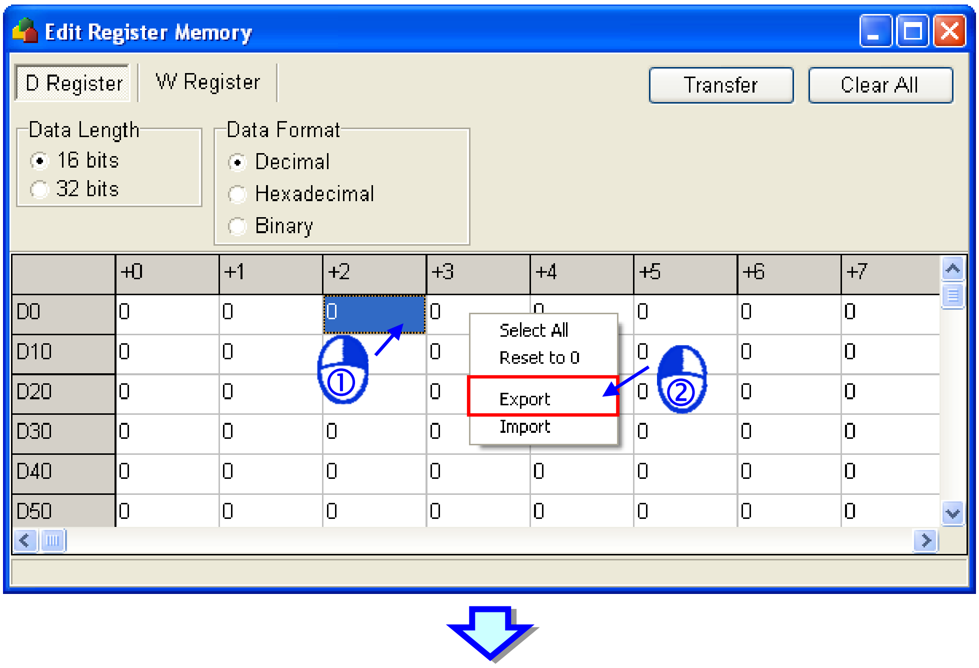
☐ Range (D4096 ~ D9999



Writing complete!

(4) If the users want to read the values in registers in the motion controller, they have to select the **Read from PM Device Register** option button in the **Transfer Setup** window, select a range in the **Range Setup** section, and click **OK**.

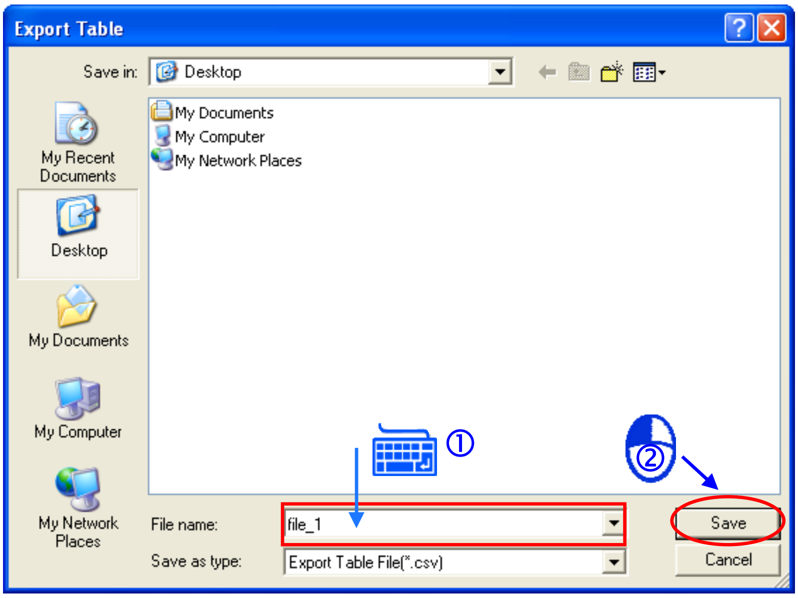
(5) If the users want to save the values in a table in the computer, they have to right-click the table, click **Export** on the context menu, set a path and a file name in the **Export Table** window, and click **Save** in the **Export Table** window.



Data Length: ● 16 bits ○ 32 bits

Data Format: ● Decimal ○ Hexadecimal ○ Binary

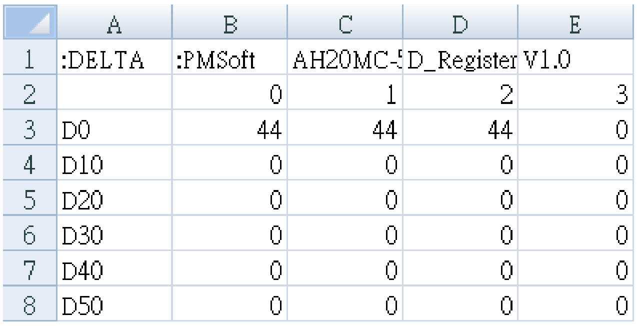
	+0	+1	+2	+3	+4	+5	+6	+7
D0	0	0	0	0	0	0	0	0
D10	0	0		0		0	0	0
D20	0	0		0		0	0	0
D30	0	0	0	0		0	0	0
D40	0	0	0	0	0	0	0	0
D50	0	0	0	0	0	0	0	0



File name: file_1

Save as type: Export Table File(*.csv)

(6) The values exported are saved as a .csv file. The .csv file can be edited through Microsoft Excel. Besides, the users can type values in the .csv file, and then import the .csv file.



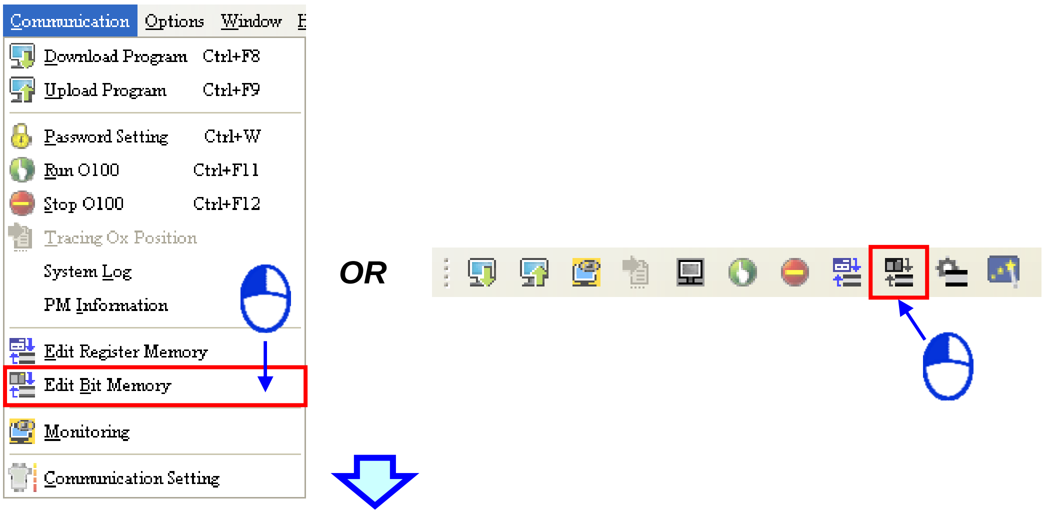
	A	B	C	D	E
1	:DELTA	:PMSoft	AH20MC-5	D_Register	V1.0
2		0	1	2	3
3	D0	44	44	44	0
4	D10	0	0	0	0
5	D20	0	0	0	0
6	D30	0	0	0	0
7	D40	0	0	0	0
8	D50	0	0	0	0

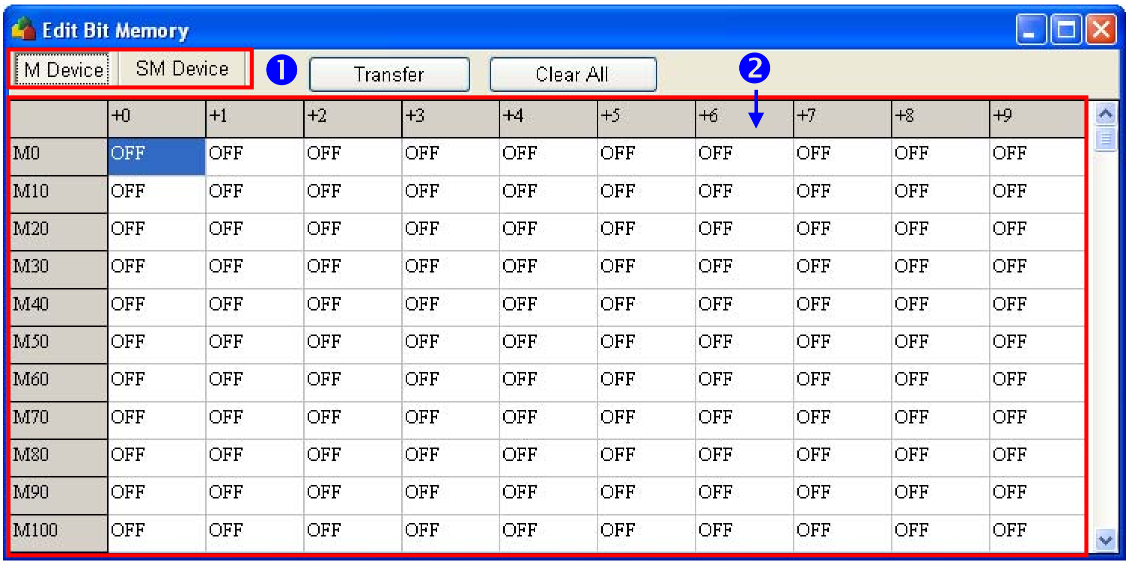
(7) If the users want to import the .csv file, they have to right-click the table, click **Import** on the context menu, click the .csv file in the **Import Table** window, and click **Open** in the **Import Table** window.

11.2.3 Editing the States of Devices

Users can change or read the states of the M devices and the SM devices in a motion controller. There are M devices and SM devices in an AH500 series motion controller. There are M devices in a DVP series motion controller.

(1) The users have to click **Edit Bit Memory** on the **Communication** menu, or on the toolbar.



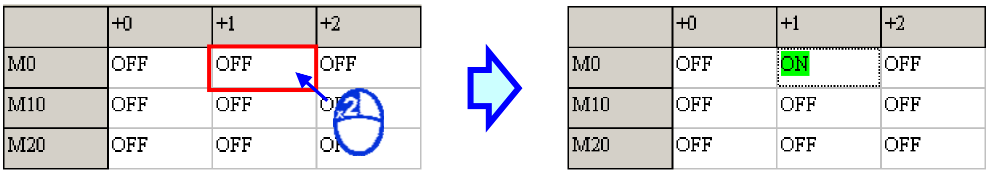


	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9
M0	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M10	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M20	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M30	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M40	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M50	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M60	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M70	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M80	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M90	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M100	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF

❶ The users can switch between the **M Device** tab and the **SM Device** tab.

❷ The users can change the states of the devices. There are ten devices in a row. For example, M0~M9 are in the first row.

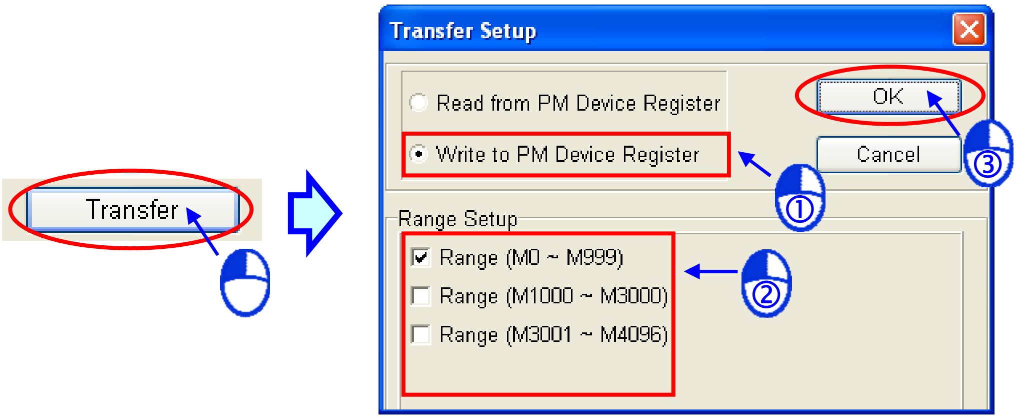
(2) The users have to click a tab, and double-click a cell. If the users want to change the states in the cells to OFF, they have to click **Clear All**.



	+0	+1	+2
M0	OFF	OFF	OFF
M10	OFF	OFF	OFF
M20	OFF	OFF	OFF

	+0	+1	+2
M0	OFF	ON	OFF
M10	OFF	OFF	OFF
M20	OFF	OFF	OFF

(3) After the users change the states in cells, they have to click **Transfer**. In the **Transfer Setup** window, the users have to select the **Write to PM Device Register** option button, and select a range in the **Range Setup** section. After the users click **OK** in the **Transfer Setup** window, the states in cells will be written into to the motion controller connected to PMSoft.



Transfer

Transfer Setup

○ Read from PM Device Register

● Write to PM Device Register

OK

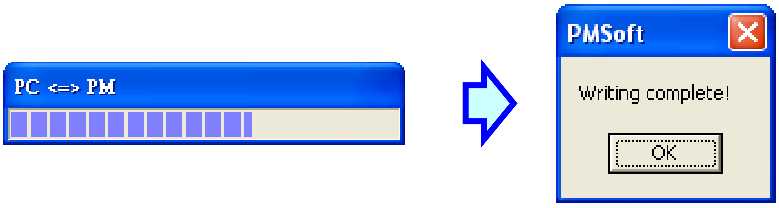
Cancel

Range Setup

☑ Range (M0 ~ M999)

□ Range (M1000 ~ M3000)

□ Range (M3001 ~ M4096)



PC <=> PM

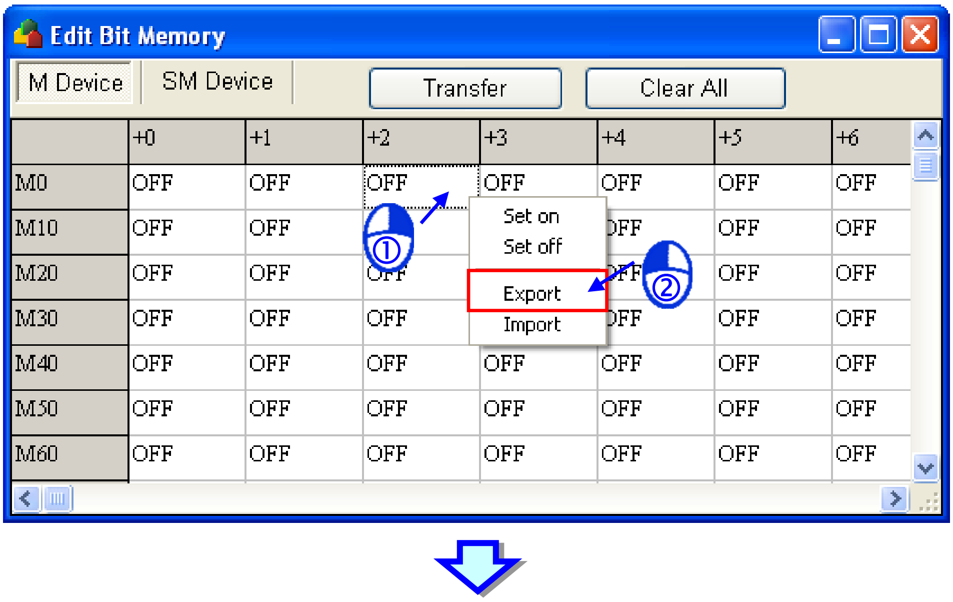
PMSoft

Writing complete!

OK

(4) If the users want to read the states of registers in the motion controller, they have to select the **Read from PM Device Register** option button in the **Transfer Setup** window, select a range in the **Range Setup** section, and click **OK**.

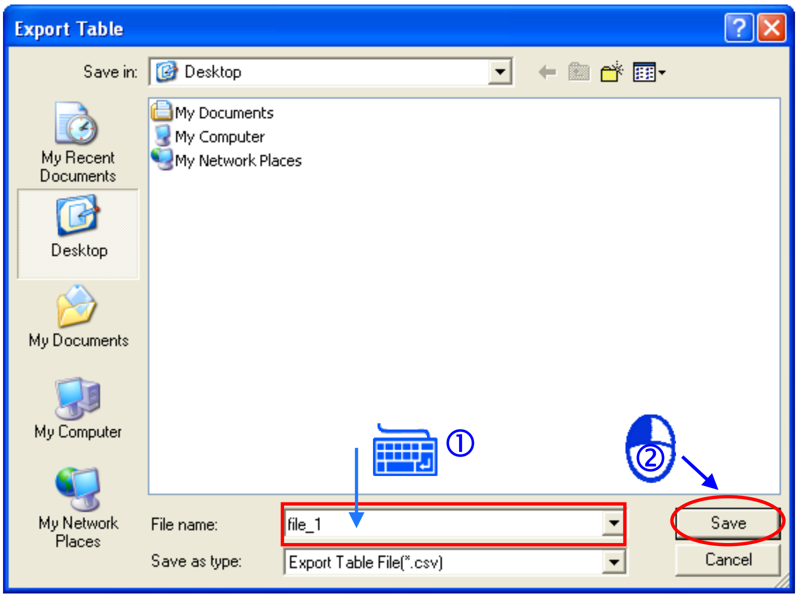
(5) If the users want to save the states in a table in the computer, they have to right-click the table, click **Export** on the context menu, set a path and a file name in the **Export Table** window, and click **Save** in the **Export Table** window.



Edit Bit Memory

M Device | SM Device | Transfer | Clear All

	+0	+1	+2	+3	+4	+5	+6
M0	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M10	OFF	OFF			OFF	OFF	OFF
M20	OFF	OFF	OFF		OFF	OFF	OFF
M30	OFF	OFF	OFF		OFF	OFF	OFF
M40	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M50	OFF	OFF	OFF	OFF	OFF	OFF	OFF
M60	OFF	OFF	OFF	OFF	OFF	OFF	OFF



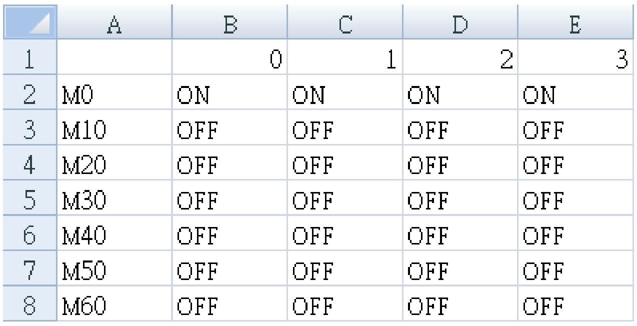
Export Table

Save in: Desktop

File name: file_1

Save as type: Export Table File(*.csv)

(6) The states exported are saved as a .csv file. The .csv file can be edited through Microsoft Excel. Besides, the users can type states in the .csv file, and then import the .csv file.



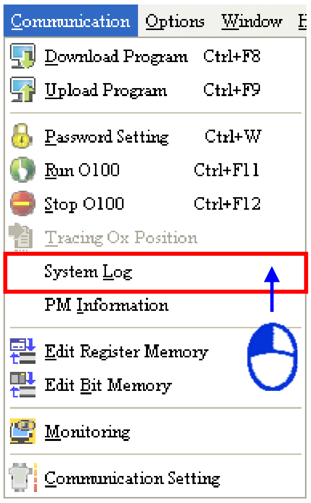
	A	B	C	D	E
1		0	1	2	3
2	M0	ON	ON	ON	ON
3	M10	OFF	OFF	OFF	OFF
4	M20	OFF	OFF	OFF	OFF
5	M30	OFF	OFF	OFF	OFF
6	M40	OFF	OFF	OFF	OFF
7	M50	OFF	OFF	OFF	OFF
8	M60	OFF	OFF	OFF	OFF

(7) If the users want to import the .csv file, they have to right-click the table, click **Import** on the context menu, click the .csv file in the **Import Table** window, and click **Open** in the **Import Table** window.

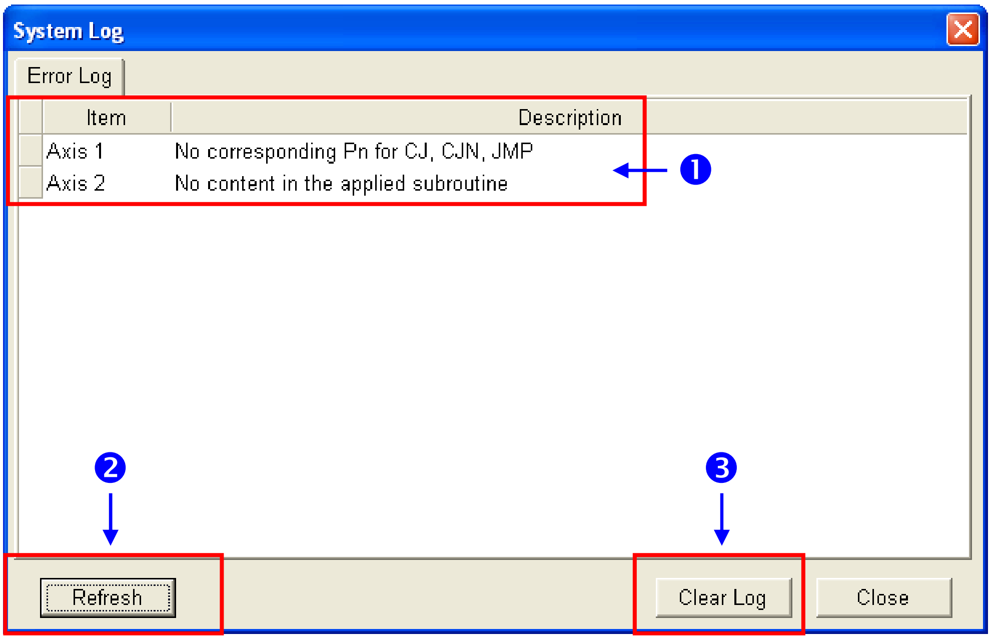
11.2.4 System Log

If an error occurs in the control of an axis of a motion controller after the motion controller is connected to PMSoft and executes a program, the information about the error will be recorded. Users can view the information in the **System Log** window.

(1) The users have to click **System Log** on the **Communication** menu



(2) The users can view the error messages in the **System Log** window, and eliminate the errors that occur. After the users click **Close**, the **System Log** window will be closed.



Item	Description
Axis 1	No corresponding Pn for CJ, CJN, JMP
Axis 2	No content in the applied subroutine

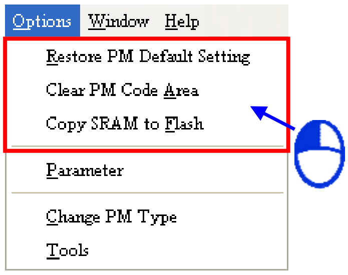
❶ The axes where errors occur, and the information about the errors are shown here.

❷ Refreshing the information in the **System Log** window

❸ Clearing the information in the **System Log** window

11.2.5 Operate the Memory in a Motion Controller

Before users operate the memory in a motion controller, they have to make sure that the motion controller is connected to a computer. The users have to click the **Options** menu, and click **Restore PM Default Setting**, **Clear PM Code Area**, or **Copy SRAM to Flash**.



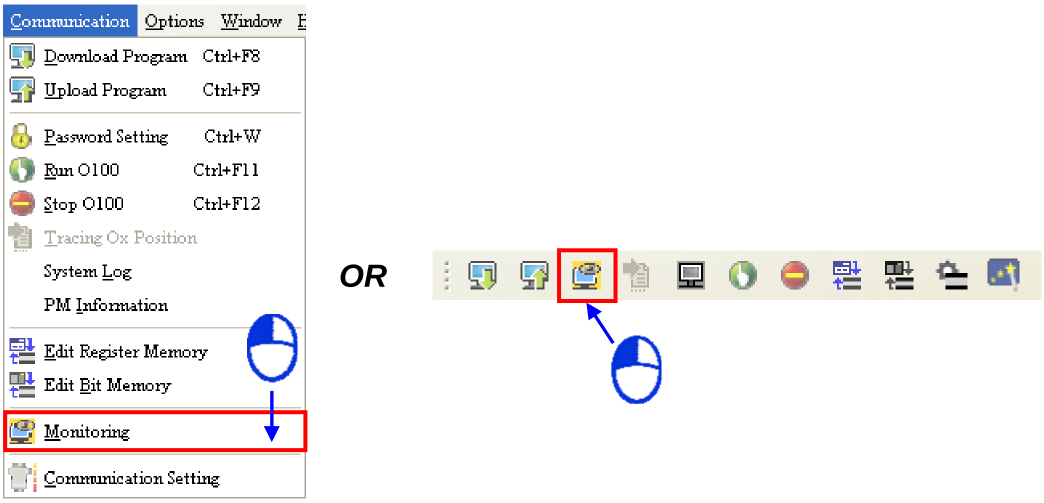
- **Restore PM Default Setting**: All the data in the motion controller is cleared, including the program, the values of the parameters, and the passwords. The motion controller is returned to its factory setting.
- **Clear PM Code Area**: **The** program in the motion controller is cleared, i.e. the POUs in the general program storage area and the program protection storage area are cleared. The values of the parameters, the cam data (DVP series motion controller), the PM password, and the PEP password are not cleared.
- **Copy SRAM to Flash**: The data in the SRAM is copied into the flash memory.

11.3 Monitoring a Motion Controller

After a program is downloaded to a motion controller, users can monitor the state of the motion controller by means of PMSoft. In other words, the state of the motion controller can be accessed, and displayed in PMSoft. The users can change the values in the devices in the motion controller by means of PMSoft.

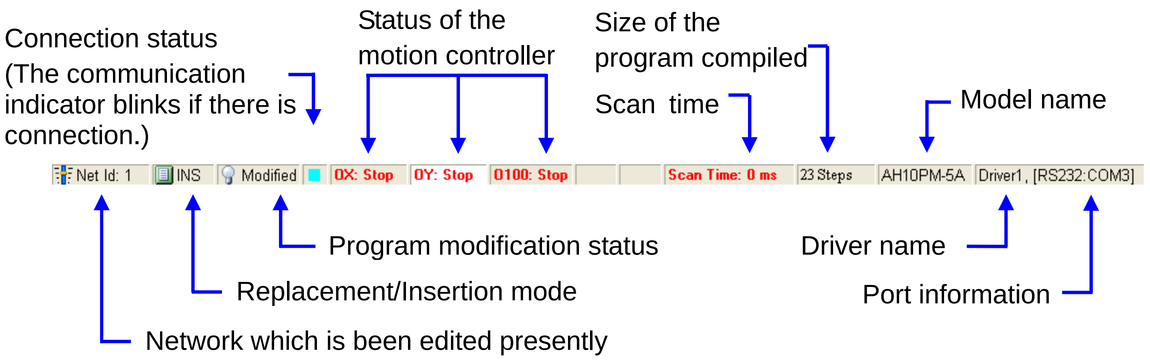
⚠Before users change the state of a motion controller, or change the value in a device, they have to make sure that the operation does affect the system, equipment or staff.

(1) The users have to click **Monitoring** on the **Communication** menu, or on the toolbar.



The users can monitor the ladder diagrams, devices, the motion instructions, the XY chart in the motion controller by means of PMSoft.

(2) When the state of the motion controller is monitored, the information about the motion controller is displayed in the status bar, and the communication indicator blinks.

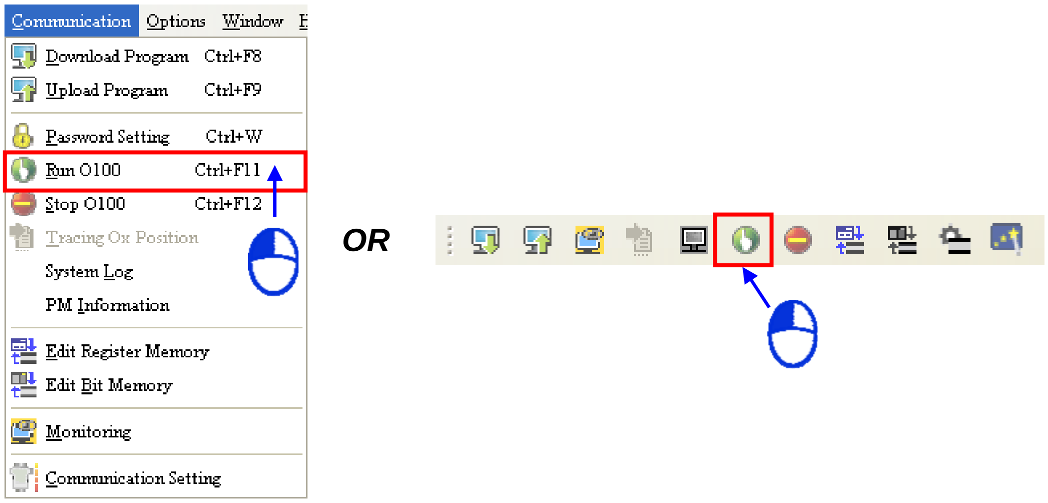

Connection status
(The communication indicator blinks if there is connection.)
Status of the motion controller
Size of the program compiled
Scan time
Model name
Net Id: 1
INS
Modified
0X: Stop
0Y: Stop
O100: Stop
Scan Time: 0 ms
23 Steps
AH10PM-5A
Driver1, [RS232:COM3]
Program modification status
Driver name
Replacement/Insertion mode
Port information
Network which is been edited presently

(3) If the users do not want to monitor the state of the motion controller, they have to click **Monitoring** on the **Communication** menu, or on the toolbar again.

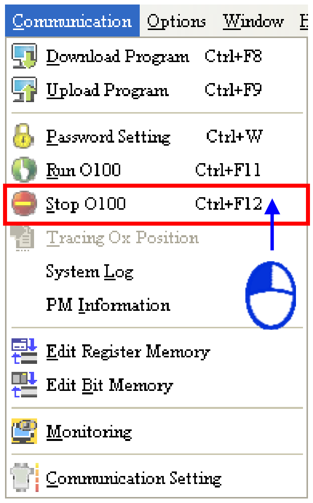
11.3.1 Running / Stopping a Motion Controller

When the state of a motion controller is monitored, the main program O100 in the motion controller can be enable/disabled by means of PMSoft.

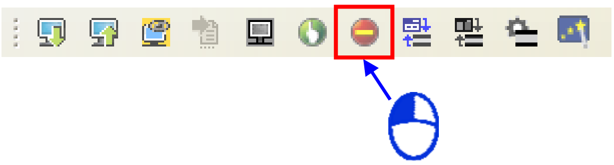
(1) When the state of a motion controller is monitored, users can click **Run O100** on the **Communication** menu, or on the toolbar to enable the main program O100 in the motion controller.



(2) After the users click **Stop O100** on the **Communication** menu, or on the toolbar, the main program O100 will be diabled.



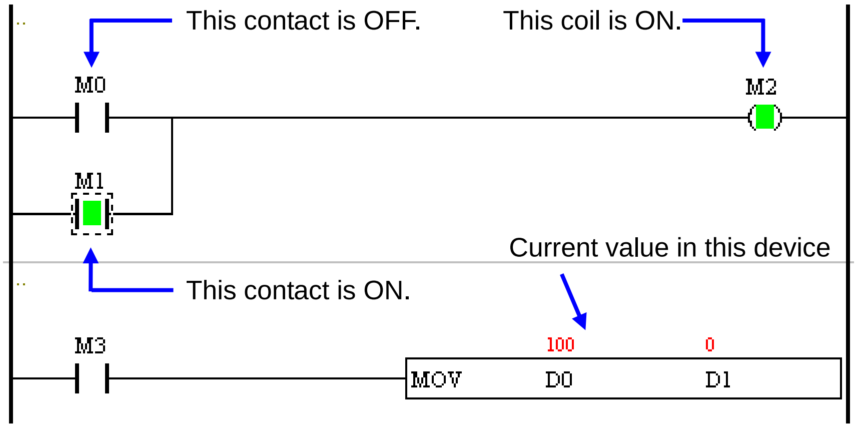
OR



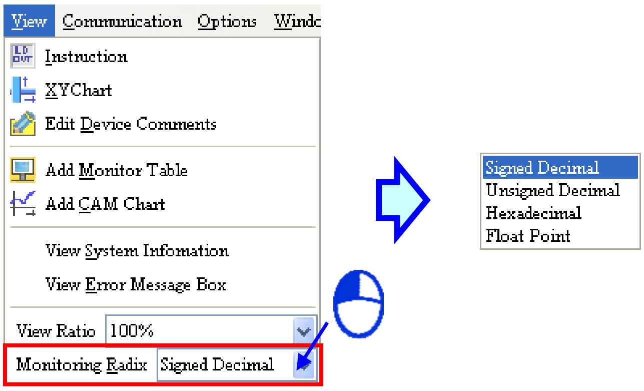
11.3.2 Monitoring a Ladder Diagram

Users have to connect a motion controller to a computer, and download a program to the motion controller. When the state of the motion controller is monitored, the users can monitor a ladder diagram.

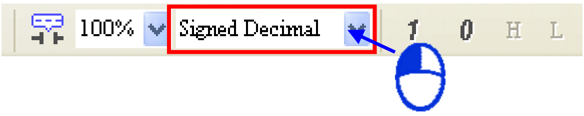
(1) The users have to open a program editing window in which a ladder diagram is created.



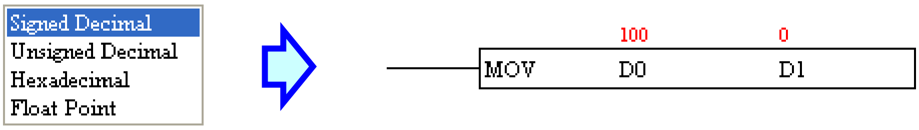
The users can select **Signed Decimal**, **Unsigned Decimal**, **Hexadecimal**, or **Float Point** in the **Monitoring Radix** drop-down list box on the **View** menu, or in Signed Decimal on the PMSoft toolbar.



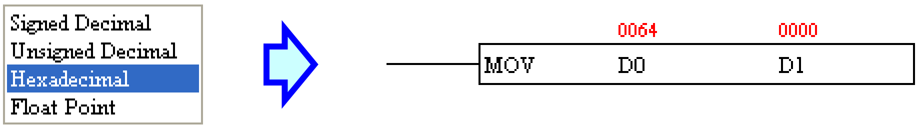
OR



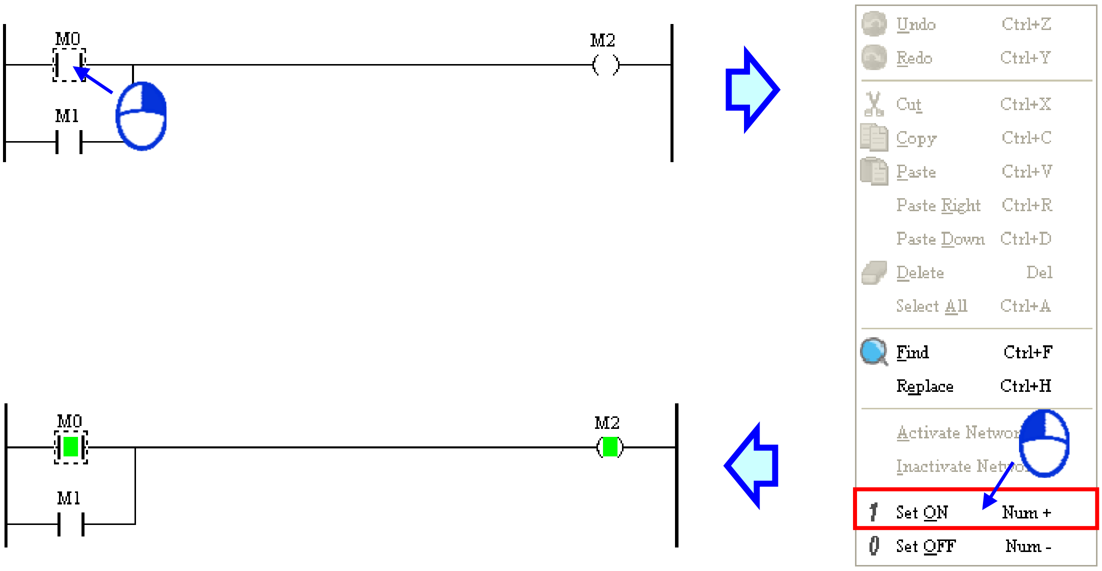
- <u>**Signed Decimal** is selected.</u>



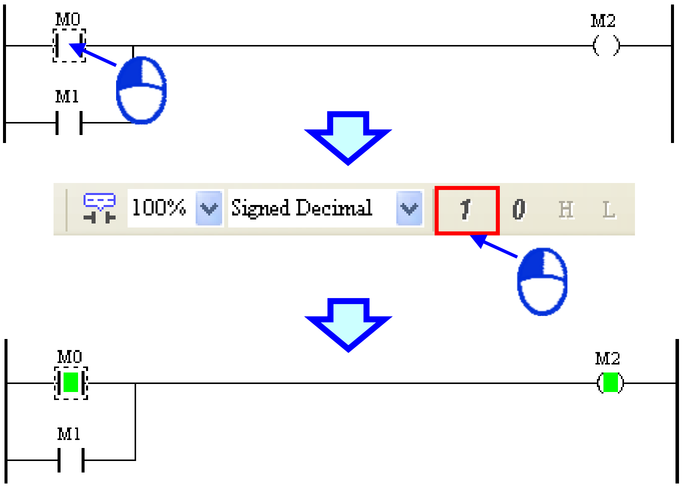
- <u>**Hexadecimal** is selected.</u>



(2) If the users want to control a contact, they have to select the contact, right-click the contact, and click **Set ON** or **Set OFF** on the context menu, or click 1 or 0 on the toolbar.

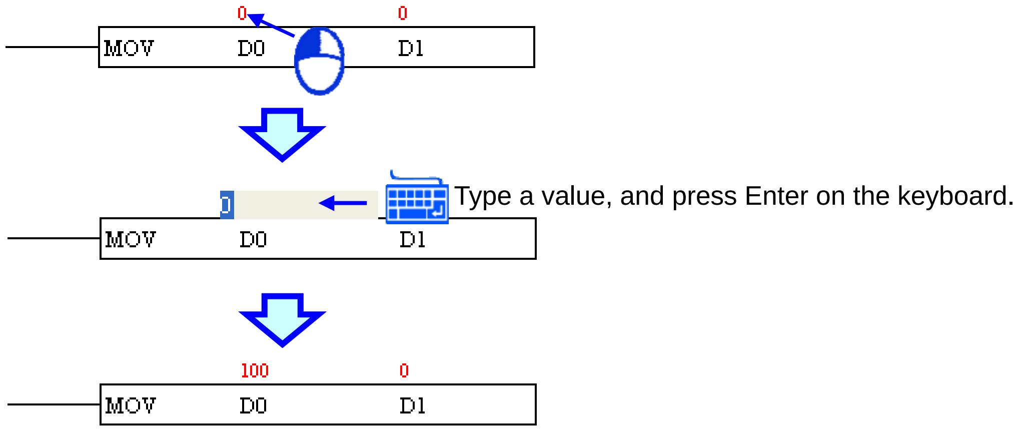


OR



- **Set ON**: The contact selected is set to ON.
- **Set OFF**: The contact selected is set to OFF.

(3) If the users want to change the value in a device, they have to click the value, type a value, and press Enter on the keyboard.



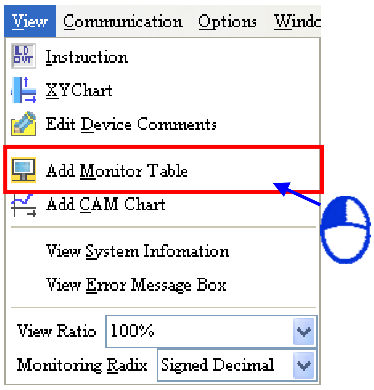
Type a value, and press Enter on the keyboard.

11.3.3 Creating a Device Monitoring Table

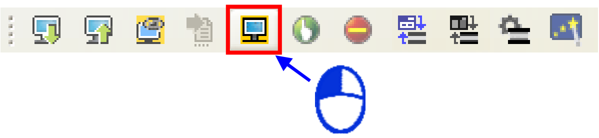
Users can create a device monitoring table offline. **Values** will be displayed in the **Value** column only if a motion controller is monitored.

(1) There are three methods of creating a device monitoring table.

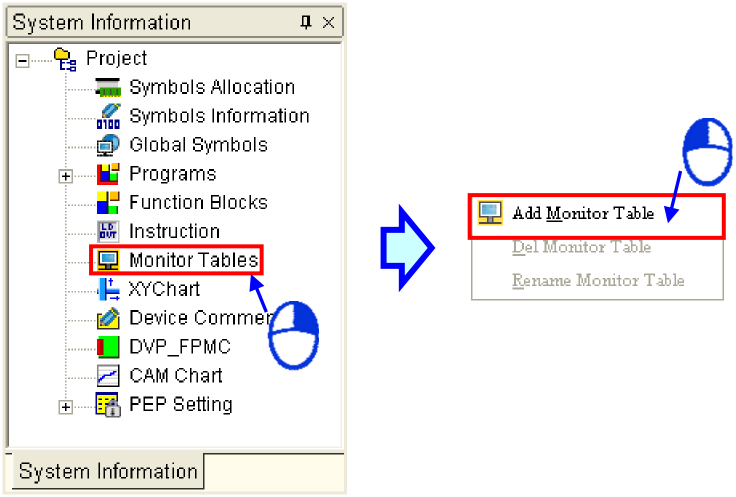
- Method 1: The users have to click **Add Monitor Table** on the **View** menu.



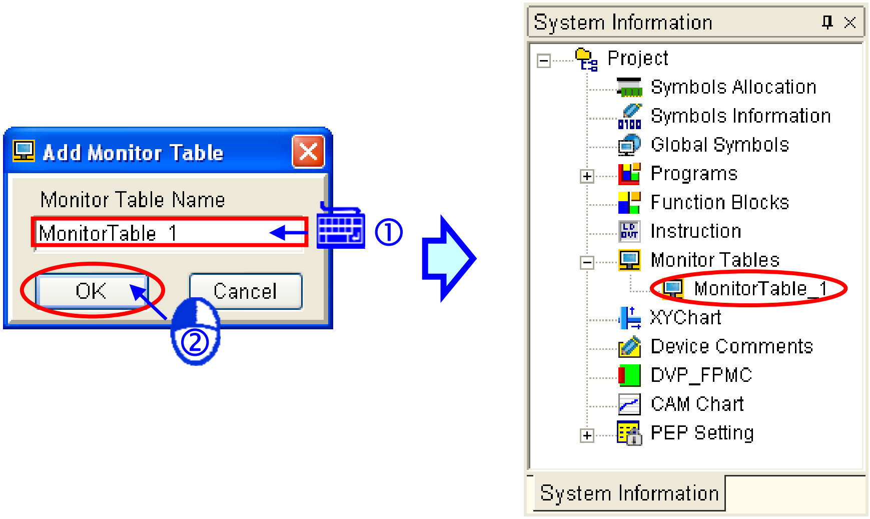
- Method 2: The users have to click 🖥 on the toolbar.



- Method 3: The users have to click **Monitor Tables** in the system information area, right-click **Monitor Tables**, and click **Add Monitor Table** on the context menu.

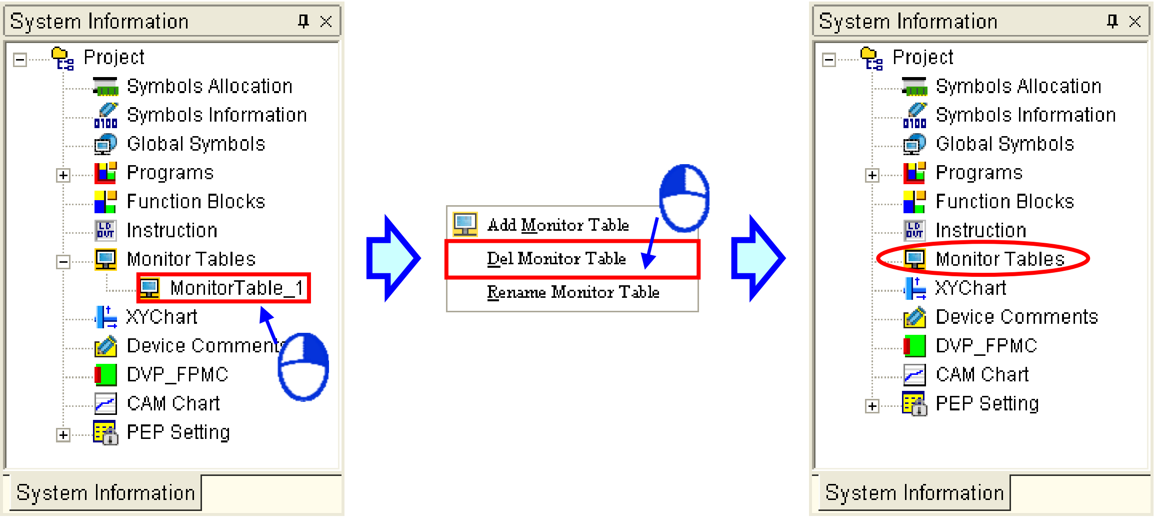


(2) The users have to type a name in the **Add Monitor Table** window. After the users click **OK** in the **Add Monitor Table** window, a device monitoring table will be added to **Monitor Tables** in the system information area.

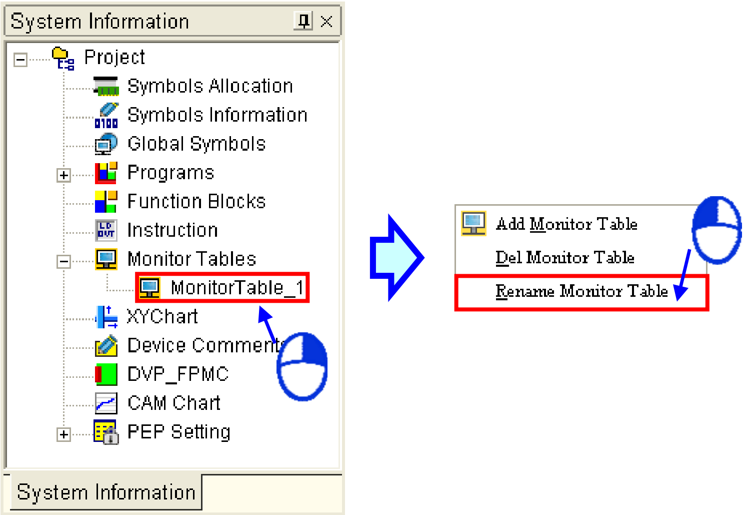

Monitor Table Name

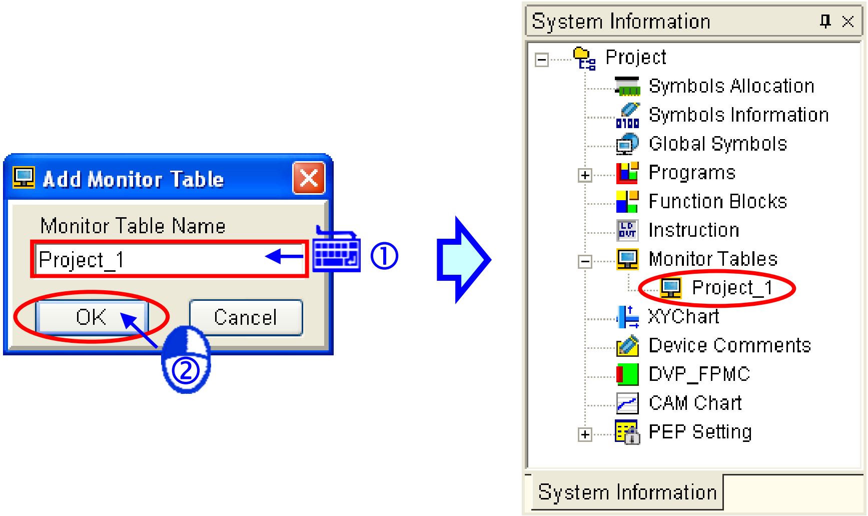
MonitorTable_1

(3) If the users want to delete a device monitoring table, they have to click the device monitoring table, right-click the device monitoring table, and click **Del Monitor Table** on the context menu.

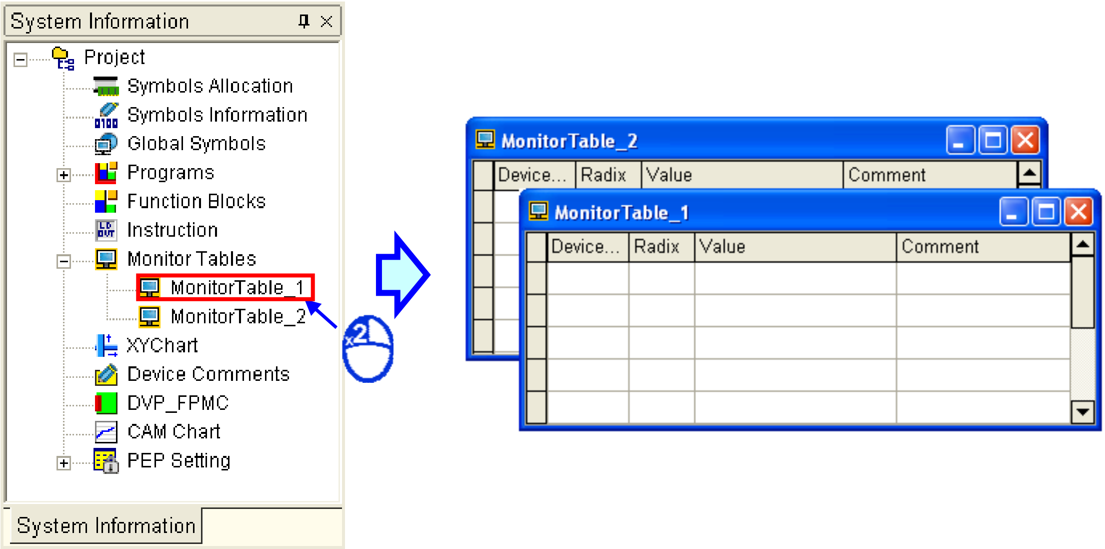


(4) If the users want to rename a device monitoring table, they have to click the device monitoring table, right-click the device monitoring table, click **Rename Monitor Table** on the context menu, type a name in the **Add Monitor Table** window, and click **OK** in the **Add Monitor Table** window.



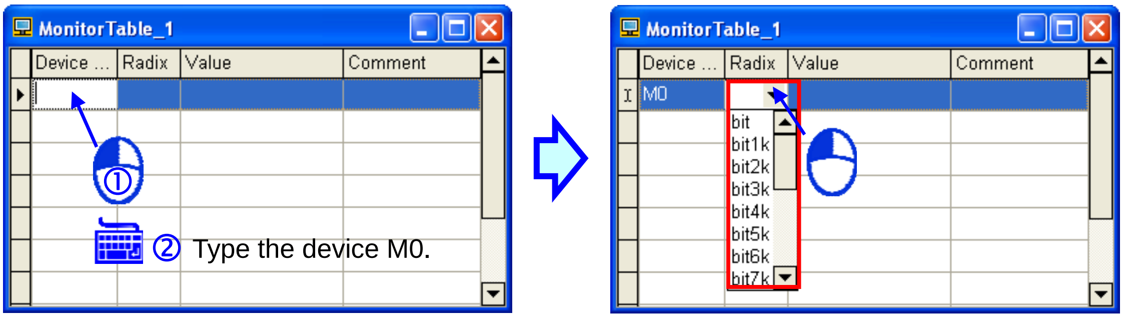


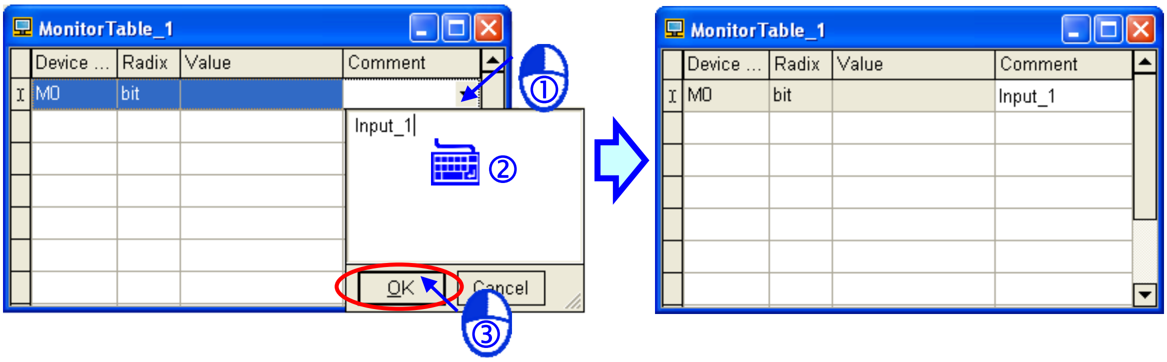
(5) After the users double-click a device monitoring table in the system information area, a window in which the device monitoring table is created will be opened. The users can open several windows in which device monitoring tables area created.



(6) The users have to add devices or symbols to a device monitoring table.

- If the users want to add a device to a device monitoring table, they have to click the **Device No.** cell in a row, type a device in the **Device No.** cell, and select an item in the **Radix** drop-down list cell in the row. After the users type a comment in a **Comment** cell, they have to click **OK**.





- **Device No.**: Device which is monitored
- **Radix**:

  bit: A bit device is monitored. If the item selected in the **Radix** drop-down list cell in a row is **bit**, the value in the **Value** cell in the row will be 0 or 1.

bitnk: Several bit devices are monitored. If the item selected in the **Radix** drop-down list cell in a row is **bit1k**, **bit2k**, **bit3k**, or **bit4k**, the value in the **Value** cell in the row will be a 16-bit value. If the item selected in the **Radix** drop-down list cell in a row is **bit1k**, and the device in the **Device No.** cell in the row is M5, the four bit devices M5~M8 will be monitored. If the item selected in the **Radix** drop-down list cell in a row is **bit5k**, **bit6k**, **bit7k**, or **bit8k**, the value in the **Value** cell in the row will be a 32-bit value. If the item selected in the **Radix** drop-down list cell in a row is **bit6k**, and the device in the **Device No.** cell in the row is M0, the twenty-four bit devices M0~M23 will be monitored.

b16: A 16-bit binary value represents the value in a device which is monitored. If the item selected in the **Radix** drop-down list cell in a row is **b16**, the value in the **Value** cell in the row will be a 16-bit value.

b32: A 32-bit binary value represents the combination of the value in a device which is monitored and the value of the following device. If the item selected in the **Radix** drop-down list cell in a row is **b32**, the value in the **Value** cell in the row will be a 32-bit value. For example, if the item selected in the **Radix** drop-down list cell in a row is **b32**, and the device in the **Device No.** cell in the row is T10, the value in the **Value** cell in the row will be the combination of the value in T11 and the value in T10.

d16u: A 16-bit unsigned real number represents the value in a device which is monitored. It must be in the range of 0 to 65,535.

d16s: A 16-bit signed real number represents the value in a device which is monitored. It must be in the range of -32,768 to 32,767.

d32u: A 32-bit unsigned real number represents the value in a device which is monitored. It must be in the range of 0 to 4,294,967,295.

d32s: A 32-bit signed real number represents the value in a device which is monitored. It must be in the range of -2147,483,648 to 2,147,483,647.

h16: A 16-bit hexadecimal value represents the value in a device which is monitored. It must be in the range of 0 to FFFF.

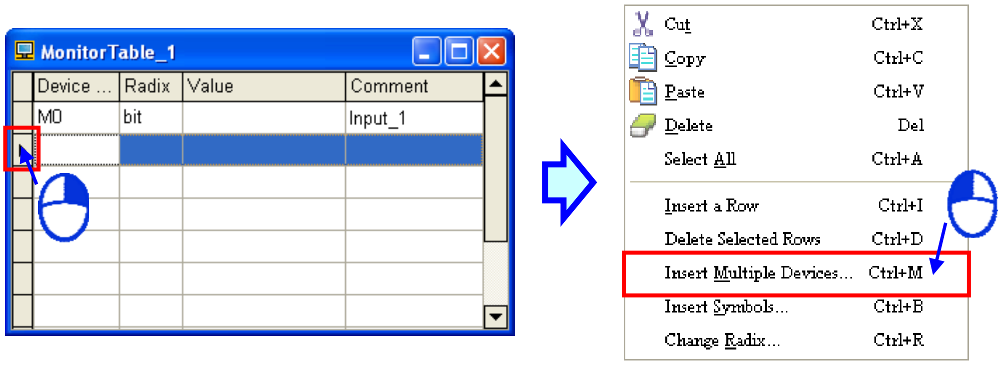
h32: A 32-bit hexadecimal value represents the value in a device which is monitored. It must be in the range of 0 to FFFFFFFF.

float: A floating-point number represents the value in a device which is monitored. The IEEE 754 standard is used.

➢ **Value**: When a motion controller is monitored, the value in a device can be displayed or set.

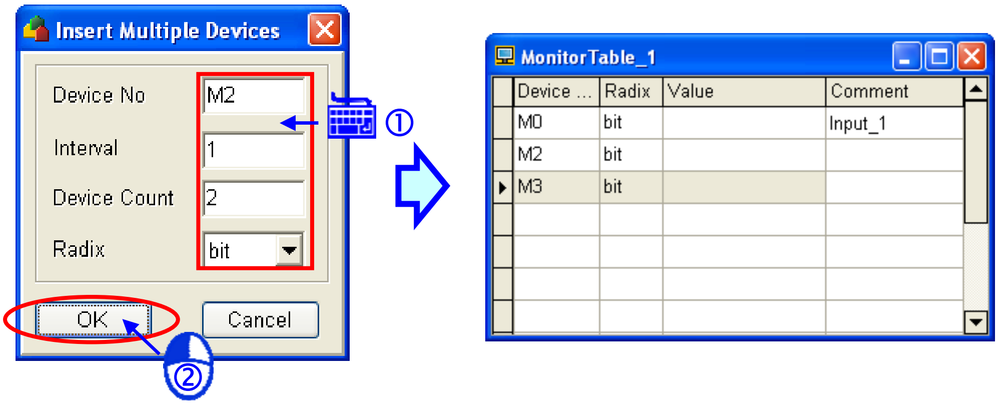
➢ **Comment**: Users can make a comment on a device.

- If the users want to add a device to a device monitoring table, they have to right-click the blank in the device monitoring table, click **Insert Multiple Devices…** on the context menu, type a device number in the **Device No** box, type an interval between two devices in the **Interval** box, and type the number of devices that they want to add in the **Device Count** box, select a data format in the **Radix** drop-down list box, and click **OK** in the **Insert Multiple Devices** window.



MonitorTable_1

Device ...	Radix	Value	Comment
M0	bit		Input_1



Insert Multiple Devices

Device No: M2

Interval: 1

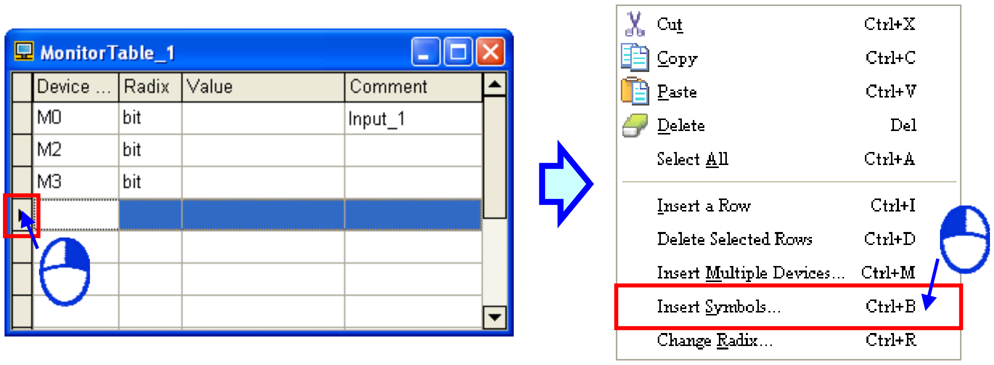
Device Count: 2

Radix: bit

MonitorTable_1

Device ...	Radix	Value	Comment
M0	bit		Input_1
M2	bit		
M3	bit		

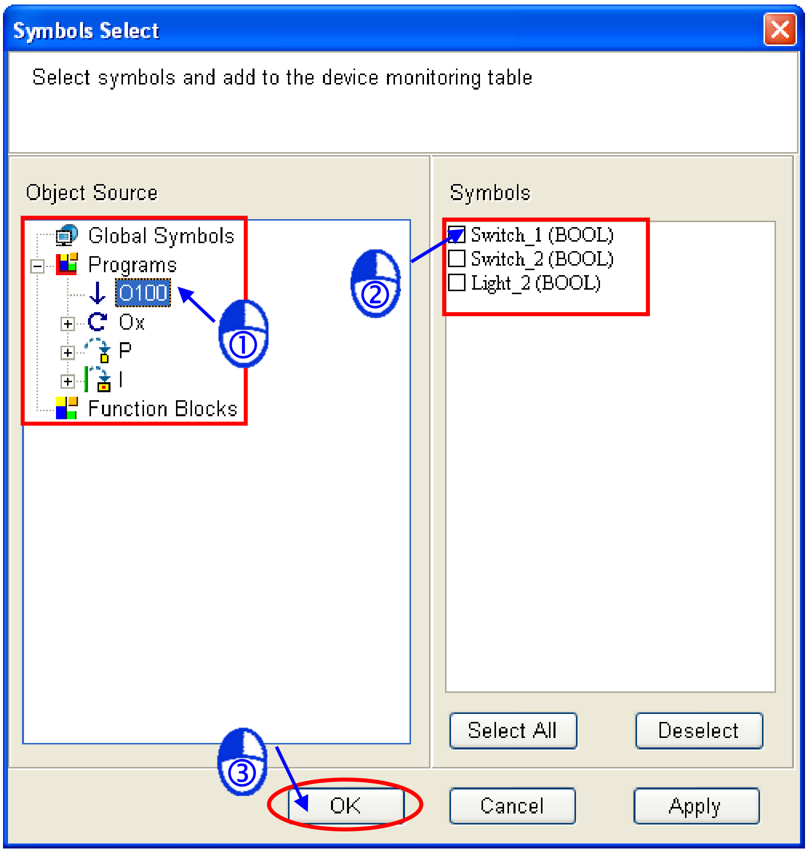
- If the users right-click a device monitoring table, click **Insert Symbols…** on the context menu, click a POU in the **Object Source** section, select symbols in the **Symbols** section, and click **OK** in the **Symbols Select** window, the symbols selected will be added to the device monitoring table. If the users click **Select All** in the **Symbols Select** window, all the symbols in the **Symbols** section will be selected. If the users click **Deselect** in the **Symbols Select** window, all the symbols in the **Symbols** section will not be selected. If the users right-click a device monitoring table, click **Insert Symbols…** on the context menu, click a POU in the **Object Source** section, select symbols in the **Symbols** section, and click **Apply** in the **Symbols Select** window, the symbols selected will be added to the device monitoring table. After the users click **Apply**, they can click another POU in the **Object Source** section, and select symbols in **Symbols** section.



MonitorTable_1

Device ...	Radix	Value	Comment
M0	bit		Input_1
M2	bit		
M3	bit		

After the users add a symbol in a POU to a device monitoring table, the symbol and the POU will be shown in a **Device No.** cell. O100.Switch_1 in the figure below indicates that Switch_1 is a symbol declared in the main program O100. The users can also add a symbol to a device monitoring table by means of the keyboard they use.



Symbols Select

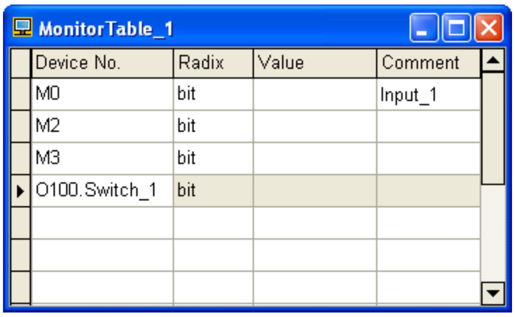
Select symbols and add to the device monitoring table

Object Source

- Global Symbols
- Programs
  - O100
  - Ox
  - P
  - I
- Function Blocks

Symbols

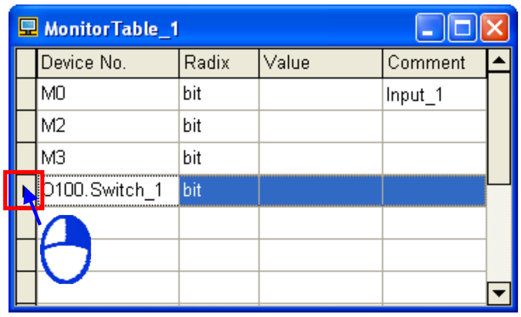
- ☑ Switch_1 (BOOL)
- □ Switch_2 (BOOL)
- □ Light_2 (BOOL)



MonitorTable_1

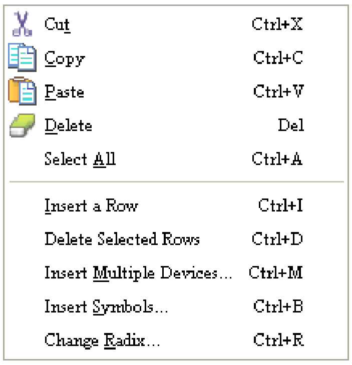
Device No.	Radix	Value	Comment
M0	bit		Input_1
M2	bit		
M3	bit		
O100.Switch_1	bit		

- After the users right-click the blank in a device monitoring table, they can click **Cut**, **Copy**, or **Paste** on the context menu. The users can cut or copy data in a device monitoring table, and paste it into another device monitoring table.



MonitorTable_1

Device No.	Radix	Value	Comment
M0	bit		Input_1
M2	bit		
M3	bit		
O100.Switch_1	bit		

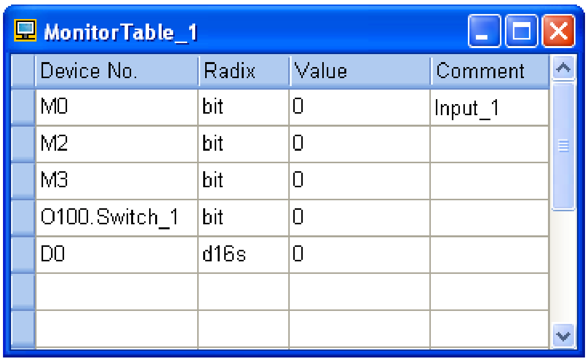


Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Delete	Del
Select All	Ctrl+A
Insert a Row	Ctrl+I
Delete Selected Rows	Ctrl+D
Insert Multiple Devices...	Ctrl+M
Insert Symbols...	Ctrl+B
Change Radix...	Ctrl+R

11.3.4 Monitoring the Devices in a Device Monitoring Table

Users have to connect a motion controller to PMSoft, and download a program to the motion controller. When the state of the motion controller is monitored, the users can monitor the devices in a device monitoring table.

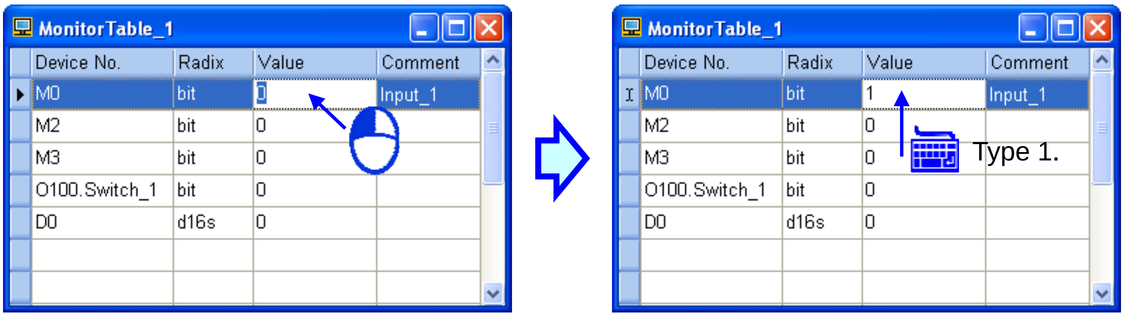
(1) The users have to open a window in which a device monitoring table is created when the motion controller is monitored. The first row and the first column in the table are filled with light blue. The users can create a device monitoring table online or offline.



MonitorTable_1

Device No.	Radix	Value	Comment
M0	bit	0	Input_1
M2	bit	0	
M3	bit	0	
O100.Switch_1	bit	0	
D0	d16s	0	

(2) The users have to click the **Value** cell for a device, and type a value. If the motion controller operates, the values in the devices related to the device will be changed.

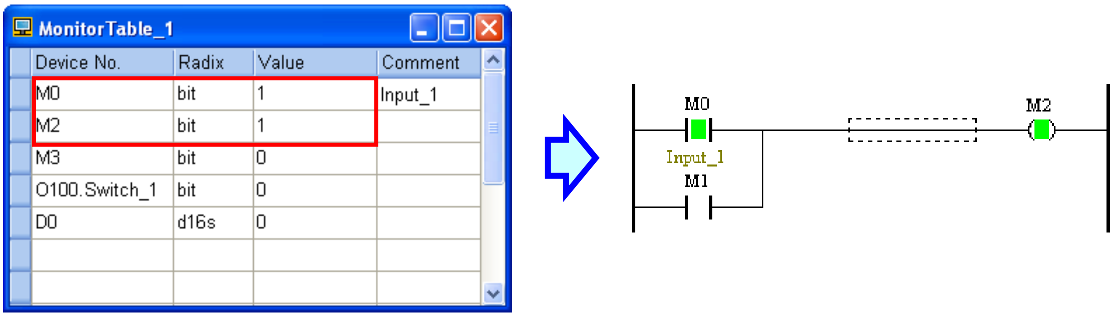


MonitorTable_1

Device No.	Radix	Value	Comment
M0	bit	0	Input_1
M2	bit	0	
M3	bit	0	
O100.Switch_1	bit	0	
D0	d16s	0	

MonitorTable_1

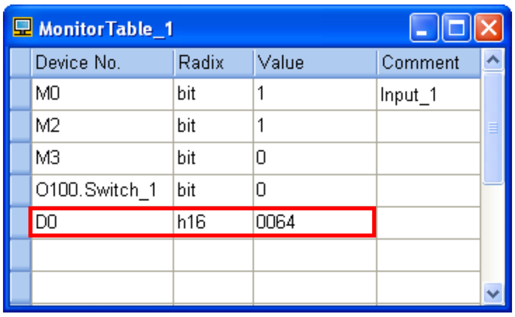
Device No.	Radix	Value	Comment
M0	bit	1	Input_1
M2	bit	0	
M3	bit	0	
O100.Switch_1	bit	0	
D0	d16s	0	



MonitorTable_1

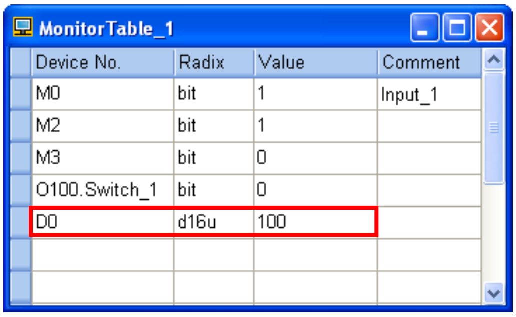
Device No.	Radix	Value	Comment
M0	bit	1	Input_1
M2	bit	1	
M3	bit	0	
O100.Switch_1	bit	0	
D0	d16s	0	

(3) The users can also change the data format selected in a **Radix** drop-down list box. Please refer to section 11.3.3 for more information about the data formats which can be selected in a **Radix** drop-down list box.



MonitorTable_1

Device No.	Radix	Value	Comment
M0	bit	1	Input_1
M2	bit	1	
M3	bit	0	
O100.Switch_1	bit	0	
D0	h16	0064	



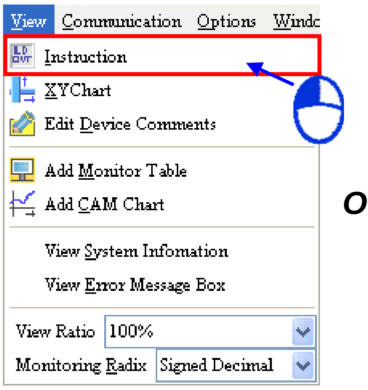
MonitorTable_1

Device No.	Radix	Value	Comment
M0	bit	1	Input_1
M2	bit	1	
M3	bit	0	
O100.Switch_1	bit	0	
D0	d16u	100	

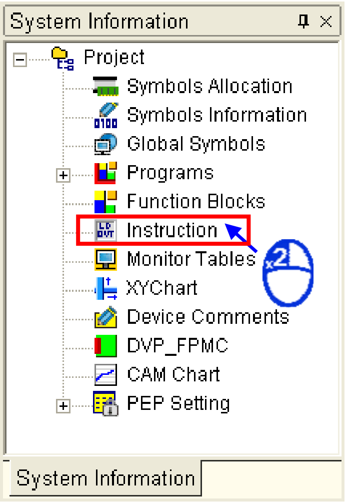
11.3.5 Monitoring the Motion Instruction which is being Executed

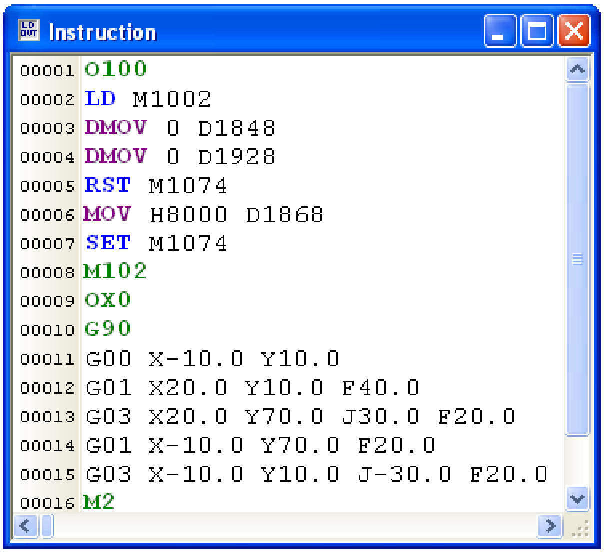
Only the motion instructions executed by a DVP series motion controller can be monitored. When the state of a motion controller is monitored, users can monitor the instruction which is being executed presently in a Ox motion subroutine. The users can make sure of the motion instruction which is being executed presently when the motion controller operates. A motion controller has to be connected to a computer, and a program has to be downloaded to the motion controller. When the state of the motion controller is monitored, the motion instruction which is being executed presently can be monitored.

(1) The users have to click **Instruction** on the **View** menu, or double-click **Instruction** in the system information area.



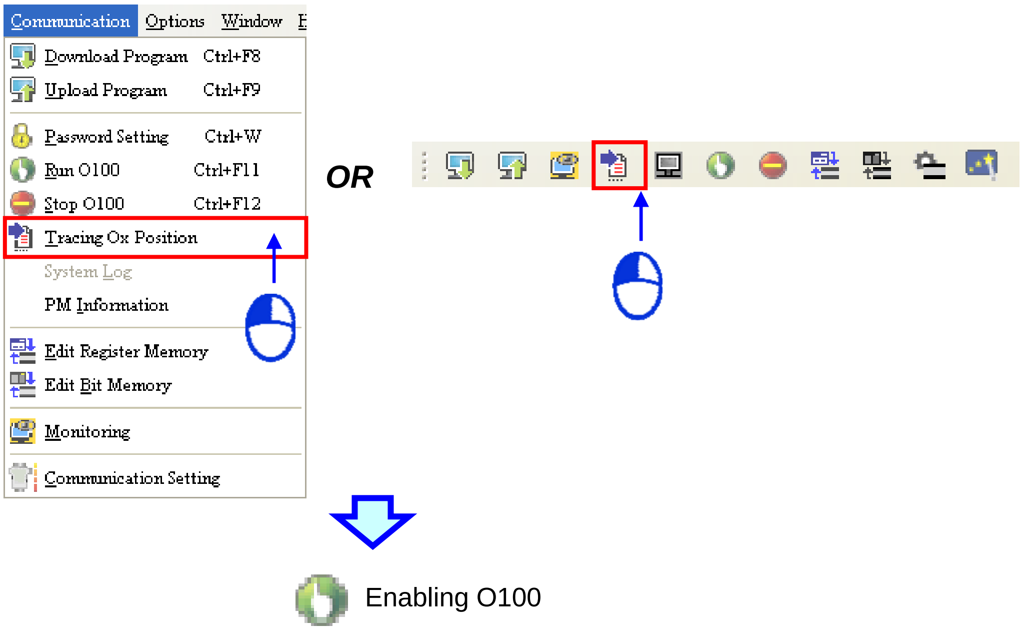
OR





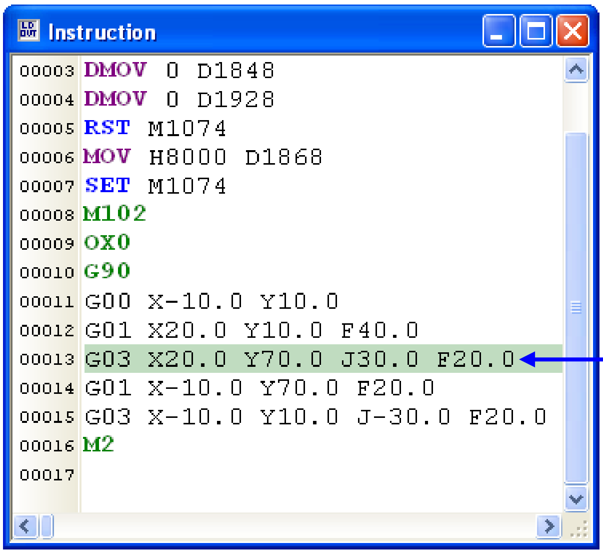
```
O100
LD  M1002
DMOV  0  D1848
DMOV  0  D1928
RST  M1074
MOV  H8000  D1868
SET  M1074
M102
OX0
G90
G00  X-10.0  Y10.0
G01  X20.0  Y10.0  F40.0
G03  X20.0  Y70.0  J30.0  F20.0
G01  X-10.0  Y70.0  F20.0
G03  X-10.0  Y10.0  J-30.0  F20.0
M2
```

(2) The users have to click **Tracing Ox Position** on the **Communication** menu, or click the toolbar icon. They can click **Tracing Ox Position** on the **Communication** menu, or the toolbar icon, only when the **Instruciton** window is opened. Besdies, the users have to enable the main program O100 in the motion controller.



Enabling O100

(3) The motion instruction which is being executed presently is on a green background. The users can monitor the motion instruction which is being executed presently.



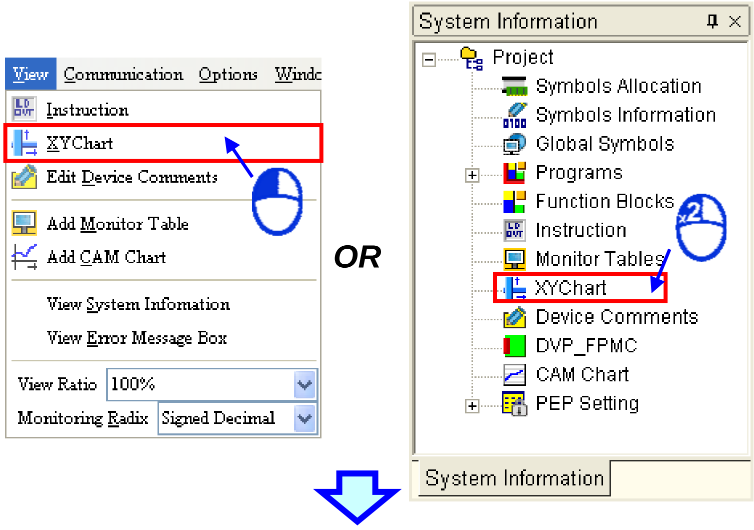
```
00003 DMOV  0 D1848
00004 DMOV  0 D1928
00005 RST  M1074
00006 MOV  H8000 D1868
00007 SET  M1074
00008 M102
00009 OX0
00010 G90
00011 G00 X-10.0 Y10.0
00012 G01 X20.0 Y10.0 F40.0
00013 G03 X20.0 Y70.0 J30.0 F20.0
00014 G01 X-10.0 Y70.0 F20.0
00015 G03 X-10.0 Y10.0 J-30.0 F20.0
00016 M2
00017
```

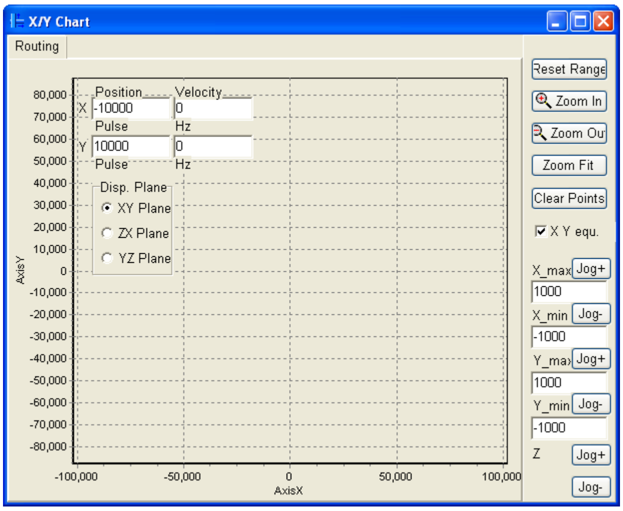
The motion instruction is being executed.

11.3.6 Monitoring a XY Chart

When the control of the x-axis/y-axis/z-axis of a motion controller is monitored, a motion path can be drawn on a XY chart. Users can check whether the motion of the x-axis/y-axis/z-axis is correct by monitoring the XY chart. A motion controller has to be connected a computer, and a program has to be downloaded to the motion controller. When the state of the motion controller is monitored, a XY chart can be monitored.

(1) The users have to click **XY Chart** on the **View** menu, or double-click **XY Chart** in the system information area.





Position: X -10000 Pulse; Y 10000 Pulse

Velocity: X 0 Hz; Y 0 Hz

Disp. Plane: ● XY Plane ○ ZX Plane ○ YZ Plane

☑ X Y equ.

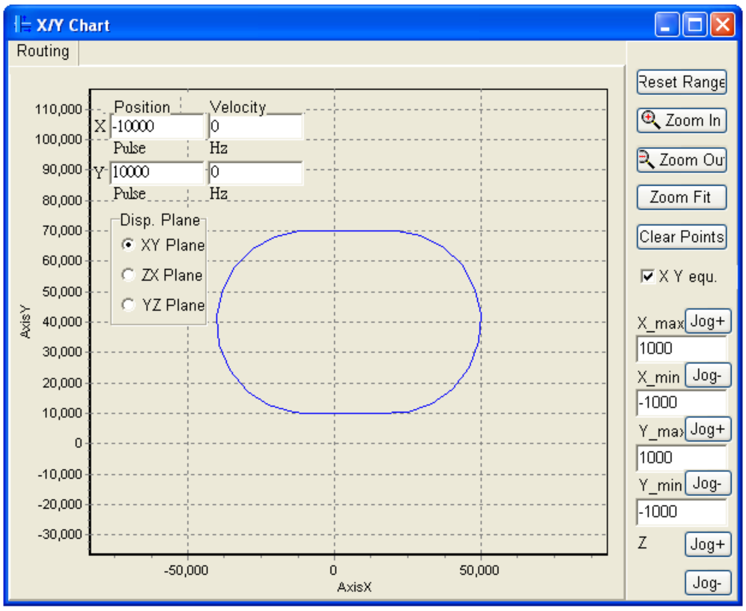
X_max 1000

X_min -1000

Y_max 1000

Y_min -1000

(2) After the motion controller begins to operate, the motion paths of the two axes of the motion controller will be drawn in the **X/Y Chart** window according to the motion instructions executed. If the motion controller has three axes, the users can view the motion paths of the three axes by switching among the planes defined by the three axes.



Position: X -10000 Pulse; Y 10000 Pulse

Velocity: X 0 Hz; Y 0 Hz

Disp. Plane: ● XY Plane ○ ZX Plane ○ YZ Plane

☑ X Y equ.

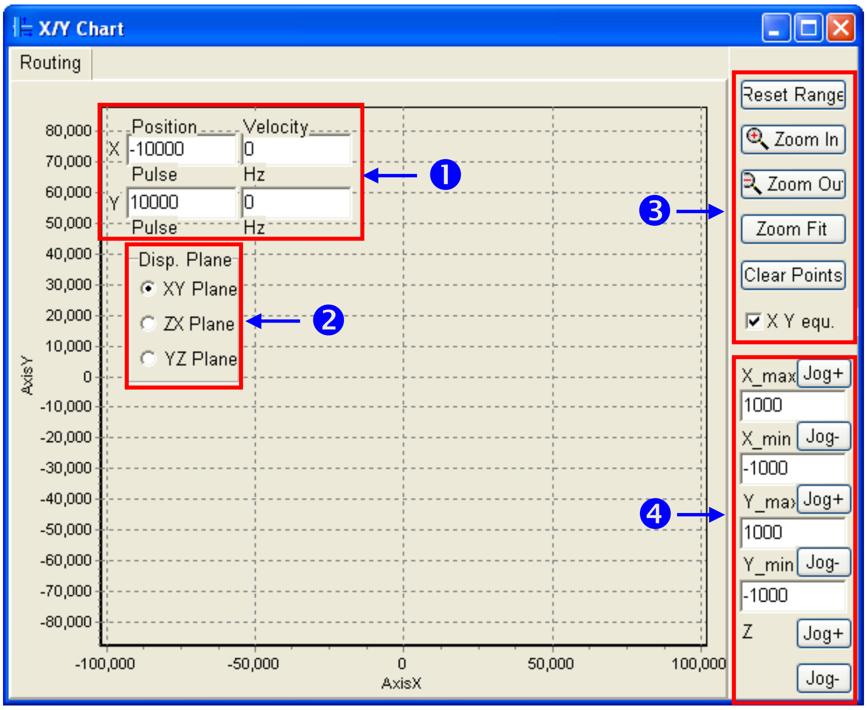
X_max 1000

X_min -1000

Y_max 1000

Y_min -1000

(3) The **X/Y Chart** window is described below.



❶ The current coordinates and the current speeds are shown here.

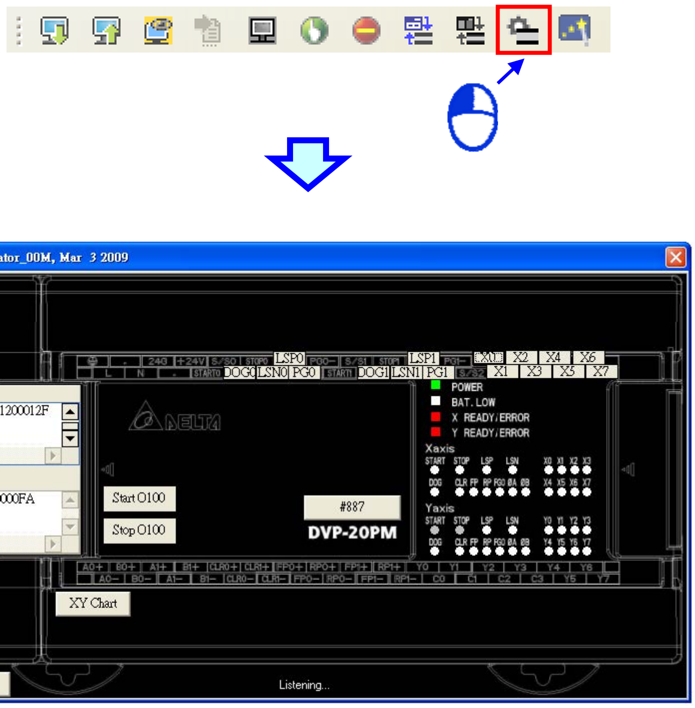
❷ The users can switch among the planes defined by the x-axis, the y-axis, and the z-axis.

❸ The users can adjust the range in which the plane selected is displayed, and clear the path on the plane selected.

❹ The users can click the buttons at the right sides of the axes, and set boundary values for the x-axis and the y-axis. If the users set boundary values, and click **Reset Range**, the boundaries of the chart shown in the **X/Y Chart** window will change. The users can not set boundary values for the z-axis.

11.4 Simulator

The simulators which can be connected to PMSoft are DVP series simulators. They can simulate the operation of motion controllers offline. PMSoft will be connected to a simulator after on the toolbar is clicked, whether PMSoft has been connected to a motion controller or not. If PMSoft has been connected to a motion controller, it will not connect to the motion controller after on the toolbar is clicked. The operation of a simulator is the same as the operation of a motion controller.

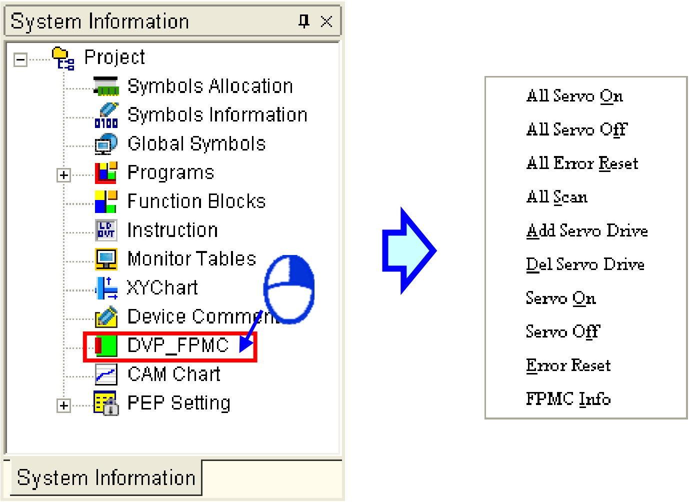


If users want to stop the operation of the simulator which is used, they have to close the **PMsimulator** window, or click on the toolbar. If PMSoft is connected to a motion controller before it is connected to a simulator, it will be connected to the motion controller after the operation of the simulator is stopped.

11.5 DVP-FPMC

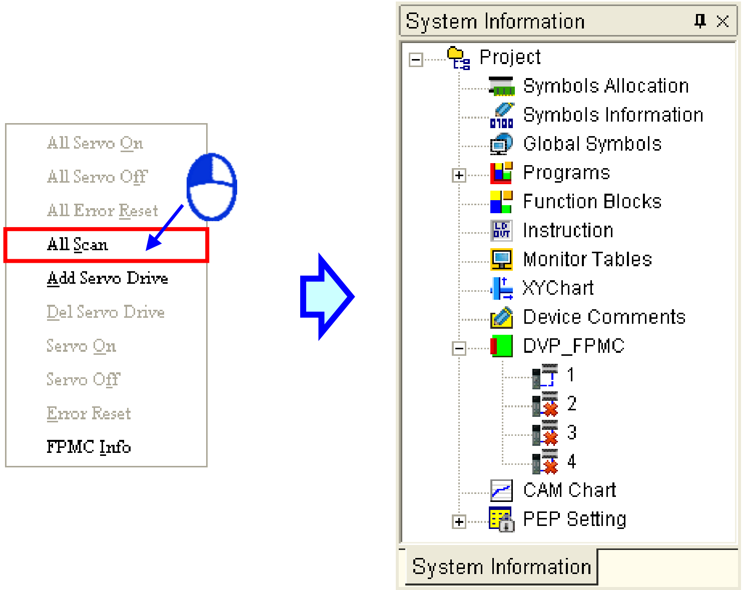
If the function card DVP-FPMC is installed in a DVP series motion controller, the DVP series motion controller can communication with a device by means of Ethernet, and it can be connected to a ASDA-A2 series servo drive through CANopen. By means of PMSfot, users can scan the servo drives which are connected to a motion controller equipped with DVP-FPMC, add a servo drive, and turn on/off the servo drives which are connected to the motion controller.

(1) After the users right-click **DVP_FPMC** in the system information area, a context menu will appear.

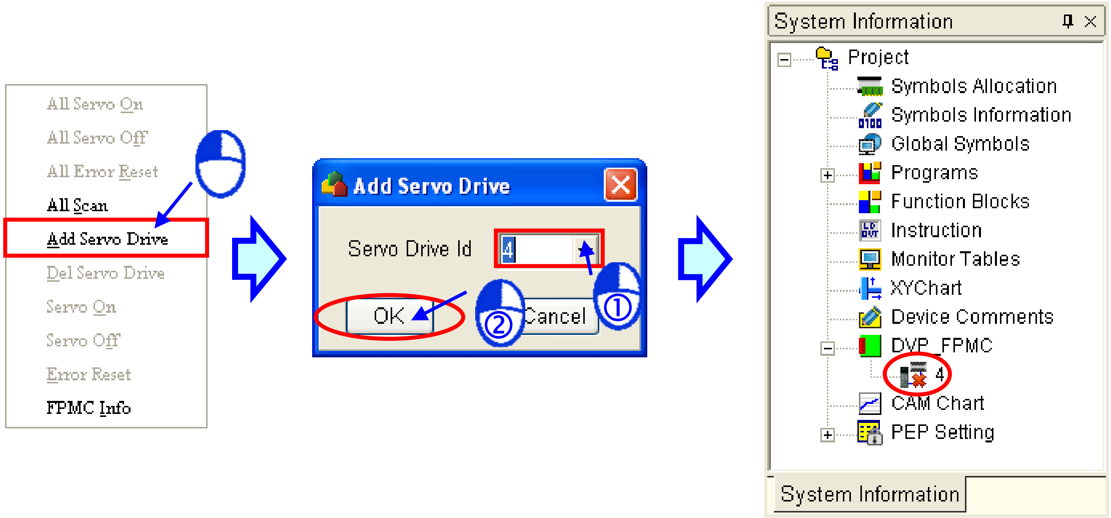


***. Some items on the context menu are shown in grayscale. They can be clicked only when a connection is created, or a device is added.**

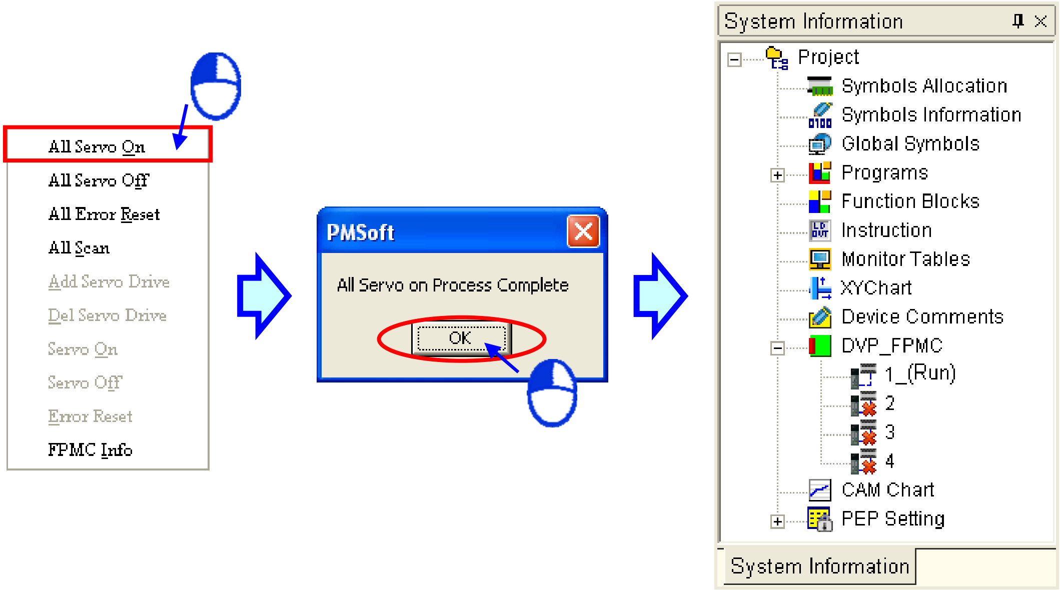
(2) If the users click **All Scan** on the context menu, the function card will scan the servo drives which are connected to the motion controller. After the function card scans the servo drives which are connected to the motion controller, four nodes and the connection statuses of the nodes will be displayed.



(3) If the users click **Add Servo Drive** on the context menu, they can type a node number in the **Add Servo Drive** window.



(4) If there are servo drives which are connected to the motion controller, the servo drives will be started after the users click **All Servo On** on the context menu. If the servo drives which are connected to the motion controller are started successfully, the **PMSoft** window shown below will appear, and the statuses of the servo drives will be displayed in the system information area.



All Servo on Process Complete

(5) If there are servo drives which are connected to the motion controller, the servo drives will be stopped after the users click **All Servo Off** on the context menu. Besides, if the users want to clear the error states which appears when servo motors are connected to the motion controller, they have to click **All Error Reset** on the context menu.

(6) If the users click **FPMC Info** on the context menu, they can view the status of the function card.

Chapter 12 Continuous Function Chart

Table of Contents

12.1 Introduction of a Continuous Function Chart

12.1.1 Continuous Function Chart

A continuous function chart is a programming language further defined by IEC 61131-3. Continuous function charts are frequently used in the field of motion control. The main character of a continuous function chart is that a program is represented by a diagram resembling a circuit diagram. Compared with other programming languages, a continuous function chart defines the relation between the inputs and the outputs in a program more clearly, and the order in which the instructions in the program are executed

The creation of a continuous function chart in PMSoft will be introduced in the following sections. The principle of a continuous function chart will be not be described further in this chapter.

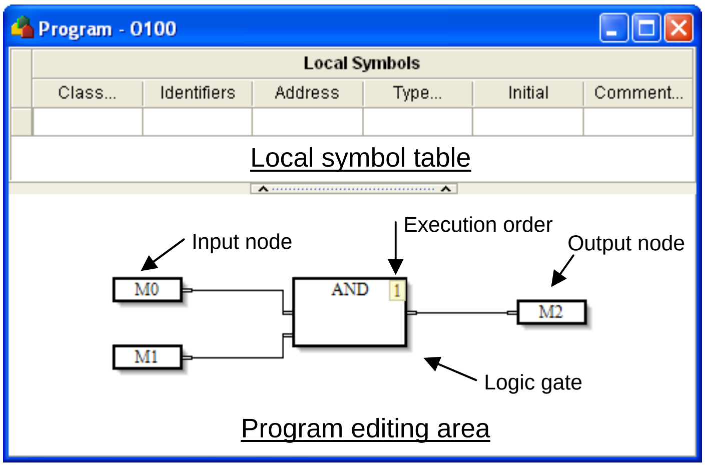
12.1.2 Important Points about Creating a Continuous Function Chart

- Users can enter instructions by means of a keyboard, but the instructions LD, LDI, LDP, and LDF are not supported. After the users type a register name or a variable, an input node will be pasted automatically.
- A continuous function chart is case-insensitive. That is, the words, OUT, Out, and out have the same meaning.
- Constants are represented in the following ways.
  - ➢ Decimal system: 2345
  - ➢ Hexadecimal system: 16#5BA0
  - ➢ ➢ Floating-point number: 4.123

  *. A continuous function chart still supports the use of K and the use of H.
- There is no limit on the number of loops which can be edited in a continuous function chart, buts users still have to consider whether the size of the program compiled exceeds the capacity of the memory in the motion controller used.
- Each input pin of a node, logic gate, instruction, or function block can only be connected to one source, but each output pin of the node, logic gate, instruction, or function block can be connected to several targets.
- A continuous function chart has not supported G-codes, M-codes, and comparison instructions yet.

12.1.3 Editing Environment

The environment in which a continuous function chart can be edited is shown in figure 12-1. Users can declare local symbols in the local symbol table at the top of the window, and create a continuous function chart in the program editing area at the bottom of the window. The objects in a continuous function charts are nodes and logic gates. The numbers in the upper right corners of the objects in a continuous function chart indicate the order in which the objects are executed.



Program - 0100

Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...

Figure 12-1 Working area

12.1.4 Toolbar

After a program editing window in which a continuous function chart can be created is opened, a toolbar will appear in the **Delta PMSoft** window, as shown in figure 12-2. The functions of the toolbar are described in table 12-1.

- PMSoft toolbar

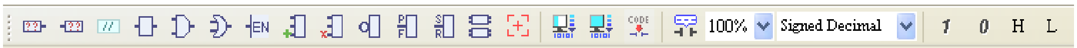


Figure 12-2 PMSoft toolbar

Table 12-1 Descriptions of the PMSoft toolbar

Icon	Keyboard shortcut	Function
??	Shift+F9	Input node
??	Shift+F3	Output node
//	Shift+F10	Comment
□	Shift+F11	OUT gate
D	Shift+F1	AND gate
⊃	Shift+F2	OR gate
EN	Shift+F12	Enabling/Disabling the EN pin of a function block or an instruction
+	Shift+F4	Adding an input pin
×	Shift+F5	Deleting an input pin
o	Shift+F6	Negation
P F	Shift+F8	Rising/Falling edge
S R	Shift+F7	Setting/Resetting a pin

Icon	Keyboard shortcut	Function
	None	Objects are reassigned numbers according to their data flow, and then all objects are arranged according to the order in which they are executed.
	None	PMSoft zooms out on all objects automatically so that all the objects can be seen.
	Alt+F7	Checking a program
	Ctrl+F7	Compiling a program
CODE	None	Transforming the instruction list into a ladder diagram (The function is not applicable to continuous function charts)
	None	Displaying information
100% ▾	None	Zooming in/Zooming out
Signed Decimal ▾	None	Converting the values monitored
1	Num+	When the online monitoring function is enabled, the node selected is set to ON.
0	Num-	When the online monitoring function is enabled, the node selected is set to OFF.
H	None	Maximum baud rate
L	None	Minimum baud rate

12.1.5 Context Menu

After users right-click the working area in a program editing window, a context menu will appear, as shown in figure 12-3. The functions of the items on the context menu are described in table 12-2.

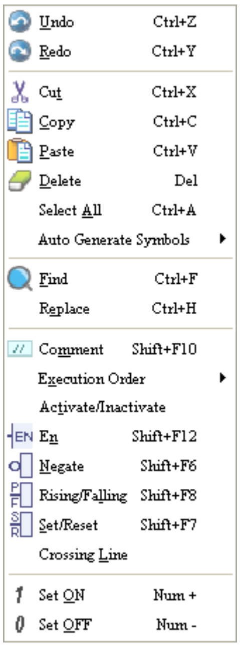


Figure 12-3 Context menu

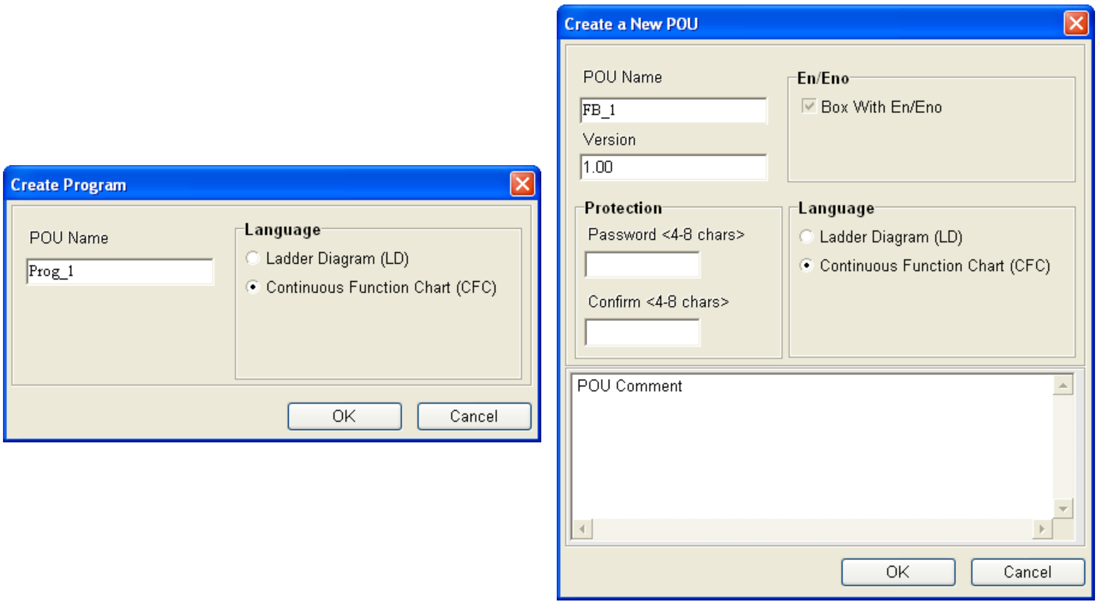
Table 12-2 Description of the context menu

Item	Function
Undo	Undoing the last action (The number of previous actions which can be undone is 20.)
Redo	Redoing an action which has been undone
Cut	Cutting the object selected
Copy	Copying the object selected
Paste	Pasting the object which has been copied or cut on the present position
Delete	Deleting the object selected
Select All	Selecting all the objects in the program editing area
Auto Generate Symbols	Automatically assigning symbols to the pins of a function block (Please refer to chapter 13 for more information.)
Find	Searching for an item (Please refer to chapter 9 for more information.)
Replace	Replacing the item found with another item (Please refer to chapter 9 for more information.)
Comment	Inserting a comment in the position to which the mouse cursor moves
Execution Order	Users can set the order in which logic gates, instructions, or function block are executed. If **Data Flow** is clicked, the order in which objects are executed will be determined by the data flow of the objects. If **Topology** is clicked, the order in which all objects are executed will be determined by the relative positions of the objects.
Activate/Inactivate	Activating/Inactivating an object
EN	Enabling/Disabling the EN pin of a function block or an instruction
Negate	Negating the state of a pin
Rising/Falling	A pin is rising edge-triggered/falling edge-triggered.
Set/Reset	Setting/Resetting a pin
Crossing Line	Way in which the lines that cross are displayed
Set ON	When the online monitoring function is enabled, the node selected is set to ON.
Set OFF	When the online monitoring function is enabled, the node selected is set to OFF.

12.2 Creating a Continuous Function Chart in PMSoft

12.2.1 Adding a POU which is a Continuous Function Chart

(1) If users want to create a POU which is O100 or a POU which is a function block, they have to right-click **O100** or **Function Blocks** in the system information area, and then click **New POU…** on the context menu which appears. The users can select the **Continuous Function Chart (CFC)** option button in the **Create Program** window or the **Create a New POU** window, as shown in figure 12-4. If the users want to create a POU which is a program, they have to expand a program section in the system information area, and double-click a number. Please refer to chapter 13 for more information.



Create Program

POU Name: Prog_1

Language

○ Ladder Diagram (LD)

● Continuous Function Chart (CFC)

Create a New POU

POU Name: FB_1

Version: 1.00

En/Eno

☑ Box With En/Eno

Protection

Password <4-8 chars>

Confirm <4-8 chars>

Language

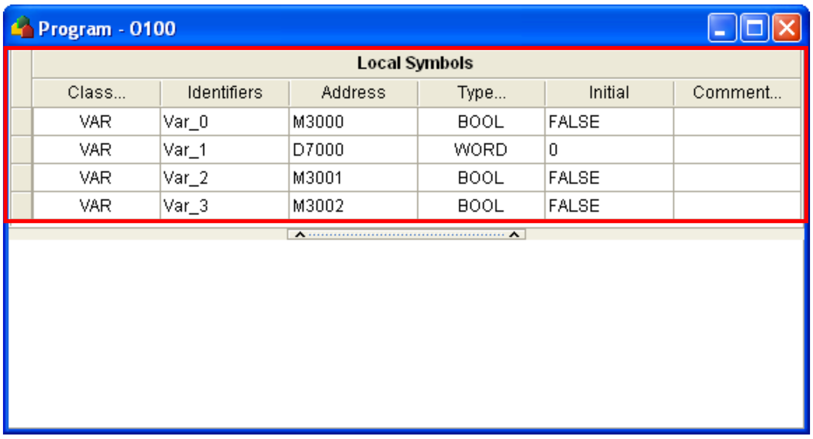
○ Ladder Diagram (LD)

● Continuous Function Chart (CFC)

POU Comment

Figure 12-4 Creating a POU which is O100 (left), and a POU which is a function block POU (right)

(2) After a POU is created, PMSoft will automatically open a program editing window. The users can used the PMSoft toolbar in the window, and create local symbols in the local symbol table in the window, as shown in figure 12-5. Please refer to chapter 4 fore more information.



Program - O100

Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	Var_0	M3000	BOOL	FALSE	
VAR	Var_1	D7000	WORD	0	
VAR	Var_2	M3001	BOOL	FALSE	
VAR	Var_3	M3002	BOOL	FALSE	

Figure 12-5 Creating local symbols in a program editing window

12.2.2 Selecting an Object

Users can select the objects in a continuous function chart by means of the mouse cursor. The objects selected can be cut/copied.

- Selecting an object

  Users have to move the mouse cursor to the object they want to select, and then click the left mouse button. The object which is selected is indicated by orange, and the objects which are not selected are indicated by white, as shown in 12-6. The users can also select an object by means of the direction keys.

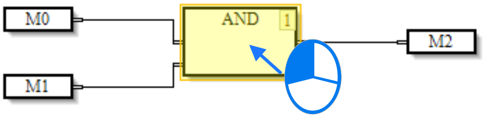


Figure 12-6 Selecting an object

- Selecting multiple objects

  If users want to select several objects, they have to click the blank in a program editing window, hold down the left mouse button, and drag the mouse cursor over the object that they want to select, as shown in figure 12-7.

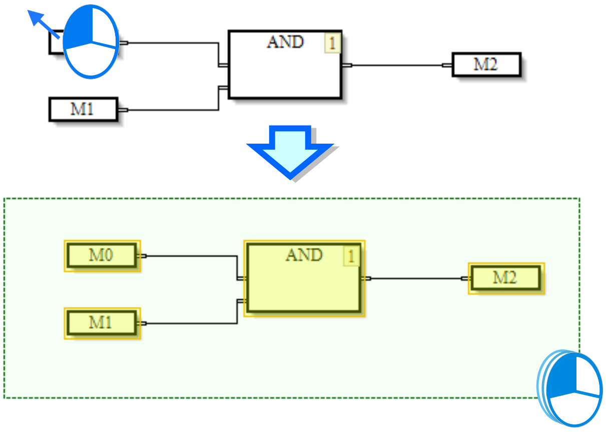


Figure 12-7 Selecting several objects

12.2.3 Input Nodes, Output Nodes, and Logic Gates

The principle of a continuous function chart is that logic gates perform operations on the states of variables or registers that input nodes put into the logic gates, and the outputs that the logic gates produce are sent to output nodes. An input node can not be connected to an output node directly. If a state needs to be moved, an OUT gate must be used. In figure 12-8, the state of the variable Var_1 is moved to the register M1. The ladder diagram corresponding to the movement is also shown in figure 12-8.

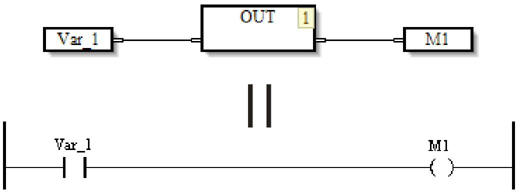


Figure 12-8 Use of an OUT gate

Other logic gates include AND gates and OR gates. Users can design complex loops by these gates, as shown in figure 12-9.

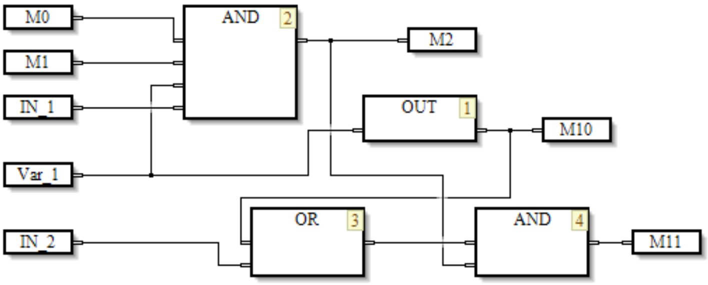


Figure 12-9 Continuous function chart

12.2.3.1 Inserting a Node or a Gate

There are two methods of inserting a node. Users can use one of the methods according to their habit.

- Method 1: PMSoft toolbar

(1) The users have to click an icon on the PMSoft toolbar, as shown in figure 12-10.

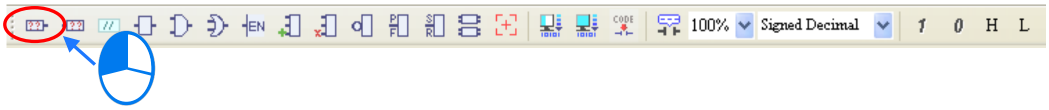


Figure 12-10 PMSoft toolbar

(2) After the users click a position in the program editing area, an object will be inserted, as shown in figure 12-11.

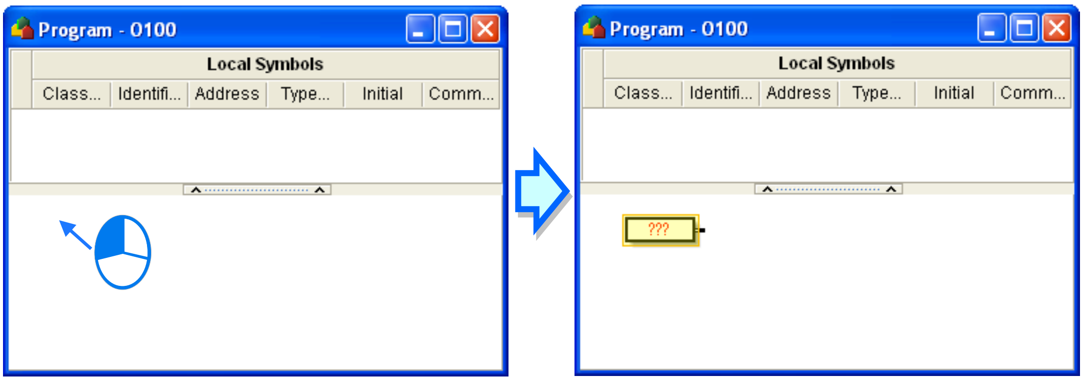


Figure 12-11 Inserting an object

- Method 2: Keyboard

  (1) When the users type an instruction, the **New Instruction** window appears. After the users type an instruction in the **Instruction** box, they have to press Enter on the keyboard, or click **OK** in the **New Instruction** window. If the users move the mouse cursor and click a position in the program editing area, the instruction will be inserted. (The instructions that the users type is case-insensitive. If the users type an incorrect applied instruction, or an incorrect device name, an error message will appear after they click **OK**.) Please see figure 12-12.

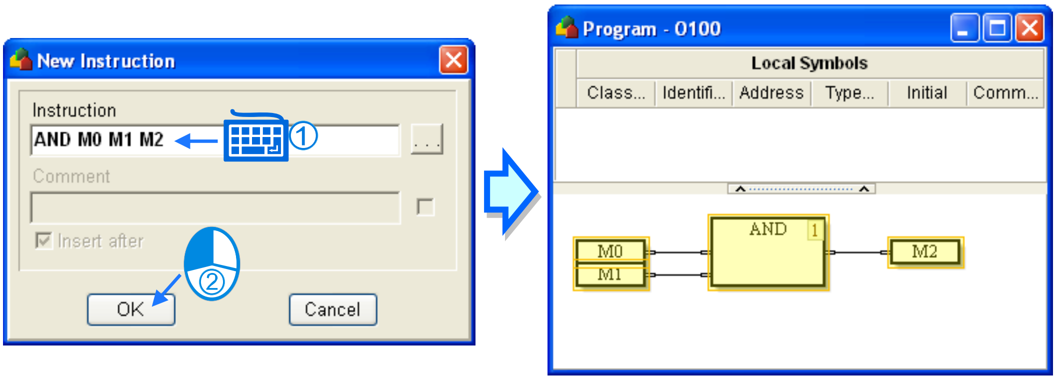


Figure 12-12 Inserting an object

The instructions which can be typed are shown in table 12-3.

Table 12-3 Instructions which can be typed

Instruction which can be typed	Function
OUT	Creating an OUT gate
AND	Creating an AND gate
OR	Creating an OR gate
Register name	Creating an input node representing a register
Instruction name	Creating a block representing an instruction
Function block name	Creating a block representing a function block

12.2.3.2 Adding / Deleting a Pin

Users can add an input pin to an AND gate/OR gate, and delete an input pin from an AND gate/OR gate by following the steps below.

(1) The users have to select a logic gate, as shown in figure 12-13.

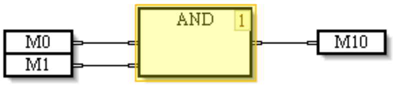


Figure 12-13 Selecting an object

(2) The users have to click ⊞ on the PMSoft toolbar, as shown in figure 12-14.

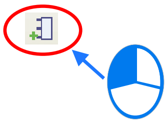


Figure 12-14 Clicking **Add Pin** on the PMSoft toolbar

(3) A new input pin is added to the bottom of the logic gate, as shown in 12-15.

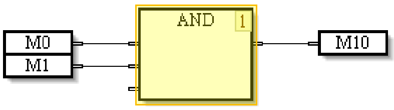


Figure 12-15 Adding a new input pin

(4) If users select an input pin, and then click ⊞, a new input pin will be put on the top of the input pin selected, as shown in figure 12-16.

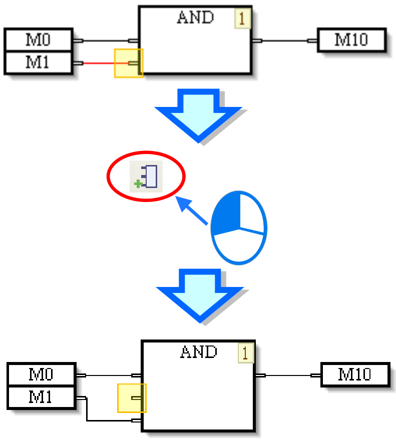


Figure 12-16 Adding a new input pin

(5) If the users click a logic gate, and then click ⊠ on the PMSoft toolbar, the input pin at the bottom of the logic gate will be deleted. If the users select an input pin, and then click ⊠, the input pin will be deleted. Please see figure 12-17.

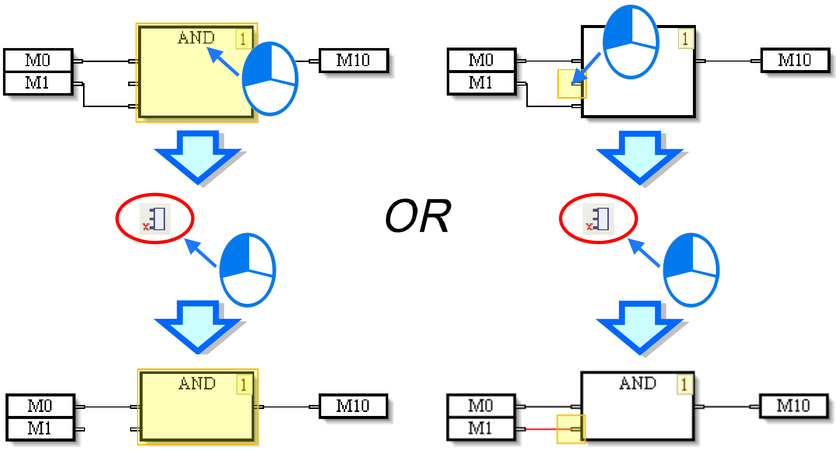


Figure 12-17 Deleting an input pin

12.2.4 Changing a Pin Type

Users can change a pin type in a way described below, and design a complex continuous function chart.

(1) If users select the pin of a node, a pin of a logic gate, a pin of a function block, or a pin of an instruction, and then click on the PMSoft toolbar or on the context menu which will appear after the users right-click the pin, the state of the pin will be negated. Please see figure 12-18. If the users select a pin whose state is negated, and then click , the state of the pin will not be negated.

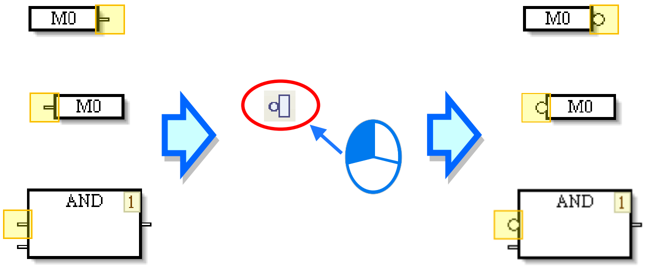


Figure 12-18 Negating the state of a pin

(2) If the users select the pin of an output node, and then click on the PMSoft toolbar or on the context menu which will appear after the users right-click the pin, the pin will be set. Please see figure 12-19. If the users select a pin which is set, and then click , the pin will be reset. If the users select a pin which is reset, and then click , the pin will not be reset.

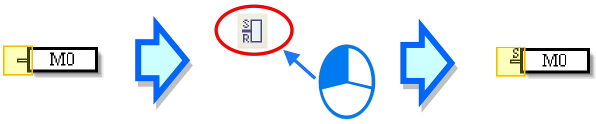


Figure 12-19 Setting/Resetting a pin

(3) If users select a pin of a logic gate, a pin of a function block, or a pin of an instruction, and then click on the PMSoft toolbar or on the context menu which will appear after the users right-click the pin, the pin will be rising-edge triggered. Please see figure 12-20. If the users select a pin which is rising edge-triggered, and then click , the pin will be falling edge-triggered. If the users select a pin which is falling edge-triggered, and then click , the pin will not be falling edge-triggered.

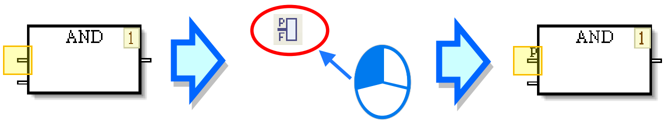


Figure 12-20 Rising edge/Falling edge

12.2.5 Connecting objects and Canceling Connections

When users click a pin of an object, and hold down the left mouse button, the target pins to which the pin can be connected are indicated by gray. When the users drag the mouse cursor to a target pin of an object, an arrow appears. After the users release the left mouse button, the two pins will be connected. Please see figure 12-21. If the users want to change the connection or cancel the connection, they can drag the line to a new target pin or to the blank.

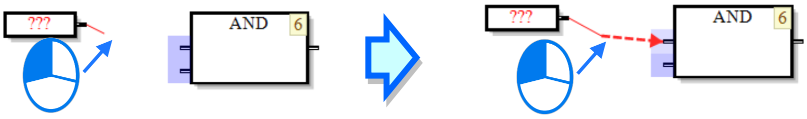


Figure 12-21 Connecting objects

12.2.6 Instructions and Function Blocks

The instructions in a continuous function chart in PMSoft are represented by blocks. An instruction name and operands are shown in a block. The order in which the instructions in a continuous function chart are executed are indicated by the numbers in the right corners of the blocks representing the instructions. An instruction is executed only when the logic state connected to the En pin of the block representing the instruction is ON. Please see figure 12-22.

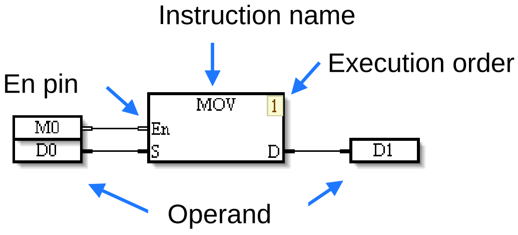


Figure 12-22 Instruction

A function block in a continuous function chart in PMSoft is shown in figure 12-23. It is executed only when the logic state connected to the En pin is ON. Besides, the text on the top of the function block is the name given to the function block, i.e. a function block instance. Please refer to chapter 5 for more information about function blocks. See figure 12-23.

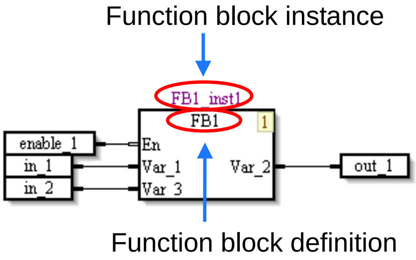


Figure 12-23 Function block

If users need to enable an instruction and a function block by a condition, they can connect the input node of the condition to the En pin of the block representing the instruction and the En pin of the function block. When the input node is ON, the instruction and the function block is executed in order. In figure 12-24, enable_1 controls the instruction and the function block.

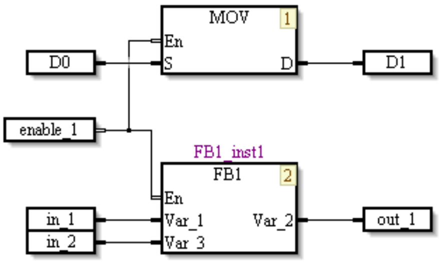


Figure 12-24 En pin

If an instruction or a function block does not need to be controlled by a condition, the users can select the block representing the instruction or the function block, and then click EN on the PMSoft toolbar or on the context menu which will appear after the users right-click the block representing the instruction or the function block. After the En pin of a block representing the instruction or the En pin of a function block is disabled, the instruction or the function block will be executed if it is the instruction's turn or the function block's turn to be executed. Please see figure 12-25. If the users select an instruction or a function block whose En pin is disabled, and then click EN, the En pin of the instruction or the En pin of the function block will be enabled.

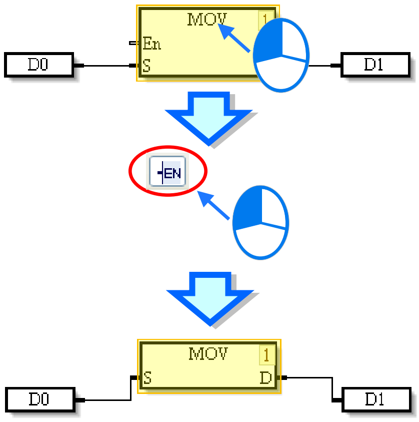


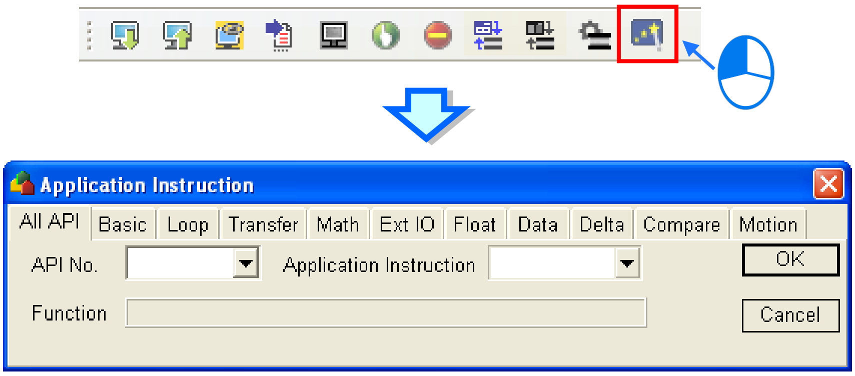
Figure 12-25 En pin

12.2.6.1 Inserting an Instruction

There are two methods of inserting an instruction. Users can use one of the methods according to their habit.

- Method 1: **Instruction Wizard**

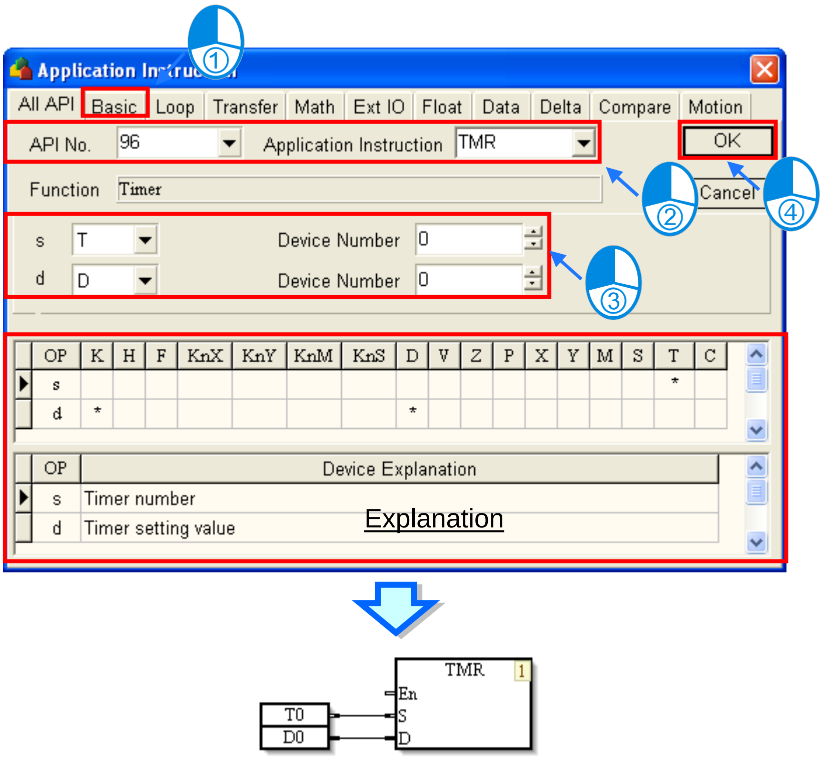
(1) After the users click on the fast toolbar, the **Application Instruction** window will appear, as shown in figure 12-26.



Application Instruction

Figure 12-26 **Instruction Wizard**

(2) The users have to select an instruction type, select an instruction in the **API No.** drop-down list box or the **Application Instruction** drop-down list box, select devices which are supported according to the explanation at the bottom of the window, click **OK**, move the mouse cursor, and click a position in the program editing area, as shown in figure 12-27.



Application Instruction

API No. 96

Application Instruction TMR

Function Timer

s T Device Number 0

d D Device Number 0

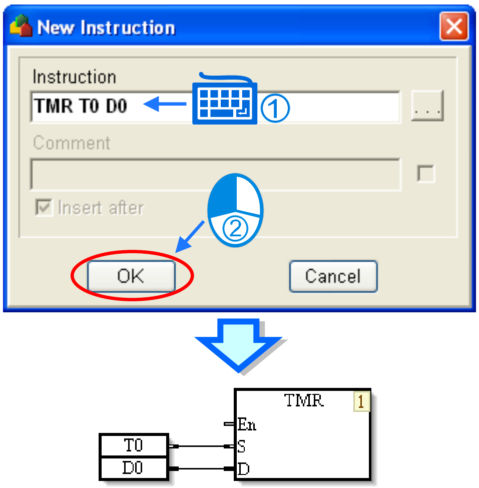
OP	K	H	F	KnX	KnY	KnM	KnS	D	V	Z	P	X	Y	M	S	T	C
s																*	
d	*							*									

OP	Device Explanation
s	Timer number
d	Timer setting value

Figure 12-27 **Instruction Wizard**

- Method 2: Keyboard

  When the users type an instruction, the **New Instruction** window appears. After the users type an instruction in the **Instruction** box, they have to press Enter on the keyboard, or click **OK** in the **New Instruction** window. If the users move the mouse cursor and click a position in the program editing area, the instruction will be inserted. (The instruction that the users type is case-insensitive. If the users type an incorrect applied instruction, or an incorrect device name, an error message will appear after they click **OK**.) Please see figure 12-28.



Instruction: TMR T0 D0

Figure 12-28 Keyboard

12.2.6.2 Inserting a Function Block

There are two methods of inserting a function block. Users can use one of the methods according to their habit.

- Method 1: Dragging a function block definition

  (1) The users have to press the left mouse button while the mouse cursor hovers over a function block definition. They have to move the mouse cursor to a position in the program editing area while holding the left mouse button down. The **Add Symbol** window will appear after the users release the left mouse button. Please see figure 12-29.

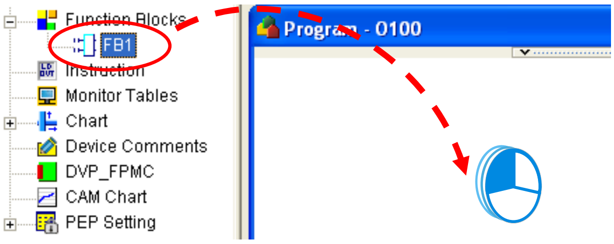
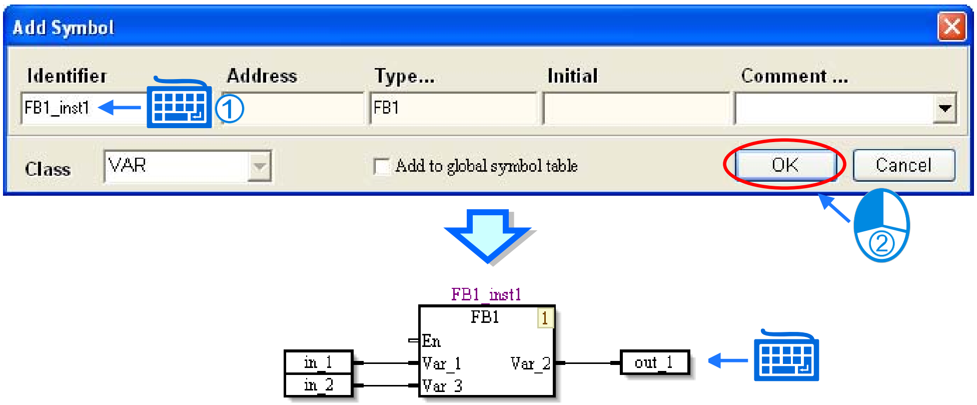


Figure 12-29 Dragging a function block definition

(2) The users have to type a function block instance in the **Add Symbol** window. (The users have to create a symbol whose data type is a function block.) Then, the users have to click **OK**, or press Enter on the keyboard, as shown in figure 12-30. Finally, the users have to assign devices or symbols to the pins of the function block inserted.



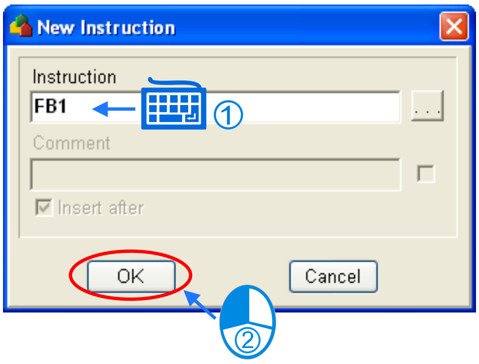
Add Symbol

Identifier: FB1_inst1
Address:
Type...: FB1
Initial:
Comment ...:
Class: VAR
□ Add to global symbol table

Figure 12-30 Declaring a function block instance

- Method 2: Typing a function block definition

(1) When the users type a function block definition, the **New Instruction** window appears. (The function block definition that the users type is case-insensitive.) After the users type a function block definition in the **Instruction** box, they have to press Enter on the keyboard, or click **OK** in the **New Instruction** window. Please see figure 12-31.

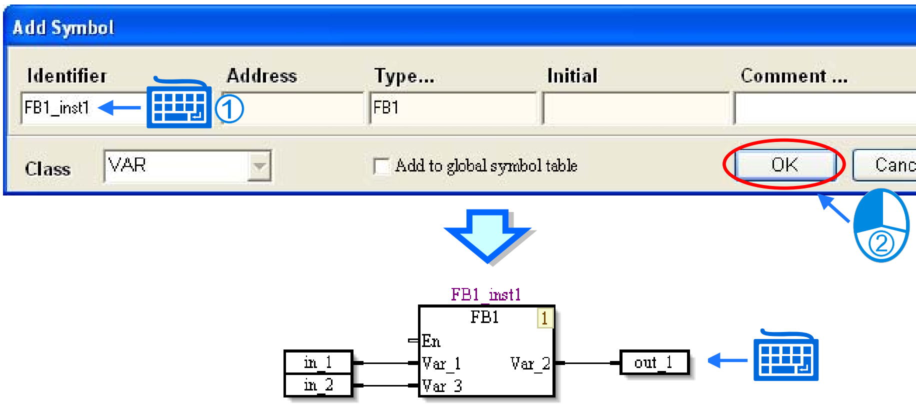


New Instruction

Instruction: FB1
Comment:
☑ Insert after

Figure 12-31 Typing a function block definition

(2) The users have to type a function block instance in the **Add Symbol** window. (The users have to create a symbol whose data type is a function block.) Then, the users have to click **OK**, or press Enter on the keyboard. If the users move the mouse cursor and click a position in the program editing area, a function block will be inserted. Finally, the users have to assign devices or symbols to the pins of the function block inserted. Please see figure 12-32.



Add Symbol

Identifier: FB1_inst1
Address:
Type...: FB1
Initial:
Comment ...:
Class: VAR
□ Add to global symbol table
OK | Cancel

Figure 12-32 Typing a function block definition

12.2.7 Deleting objects

If users want to delete a single object or several objects in a continuous function chart, they can use one of the methods below.

(1) The users have to click an object, and click **Delete** on the **Edit** menu, or on the standard toolbar, as shown in figure 12-33.

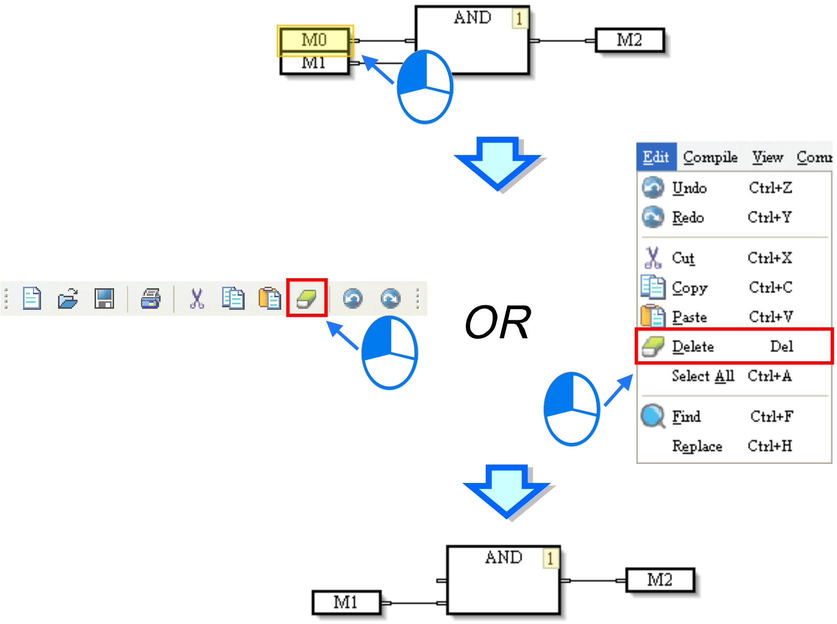


Figure 12-33 Deleting an object

(2) The users have to click an object, right-click the object, and click **Delete** on the context menu which appears, as shown in figure 12-34.

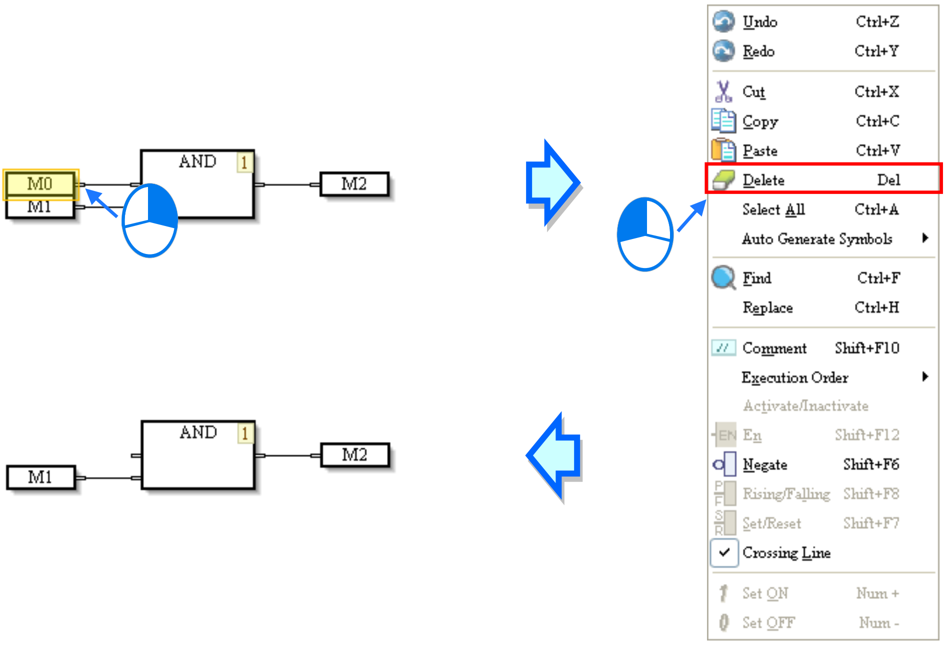


Figure 12-34 Deleting an object

(3) The users have to click an object, and press Delete on the keyboard, as shown in figure 12-35

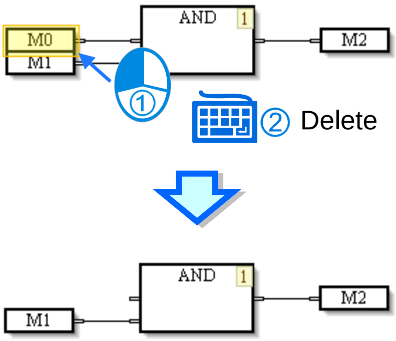


Figure 12-35 Deleting an object

12.2.8 Editing Devices or Symbols

If users want to edit a node, they have to double-click the node, type a device name, and click the blank in the program editing area or press Enter on the keyboard. Please see figure 12-36.

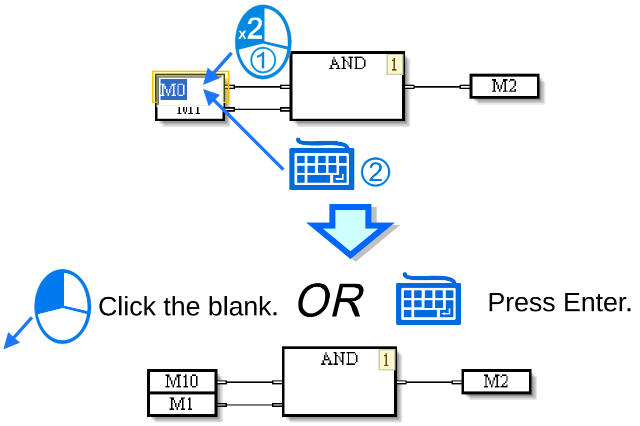
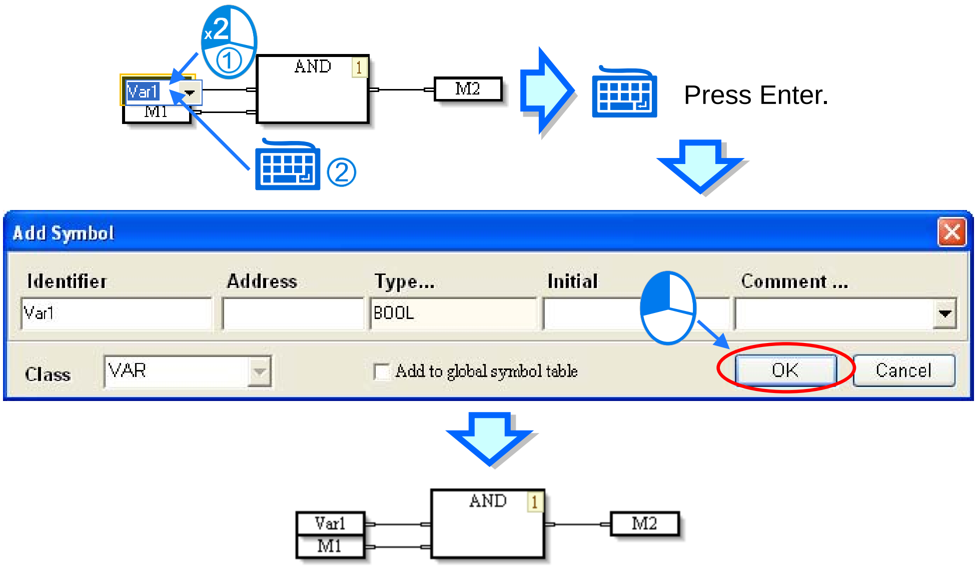


Figure 12-36 Editing a node

If the users type a symbol in a node, the **Add Symbol** window will appear after they press Enter on the keyboard. After the users set a symbol and click **OK** in the **Add symbol** window, they will finish declaring the symbol, as shown in figure 12-37. Please refer to chapter 4 for more information about symbols.



Identifier	Address	Type...	Initial	Comment ...
Var1		BOOL		

Class: VAR

□ Add to global symbol table

Figure 12-37 Symbol in a node

12.2.9 Activating / Inactivating an Object

If an object in a continuous function chart inactivated, the compiling of the continuous function chart will skip the object. Users can temporarily inactivate some parts of a program.

(1) The users have to right-click an object, as shown in figure 12-38.

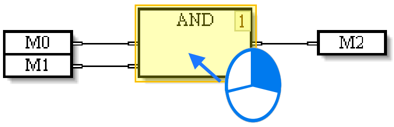
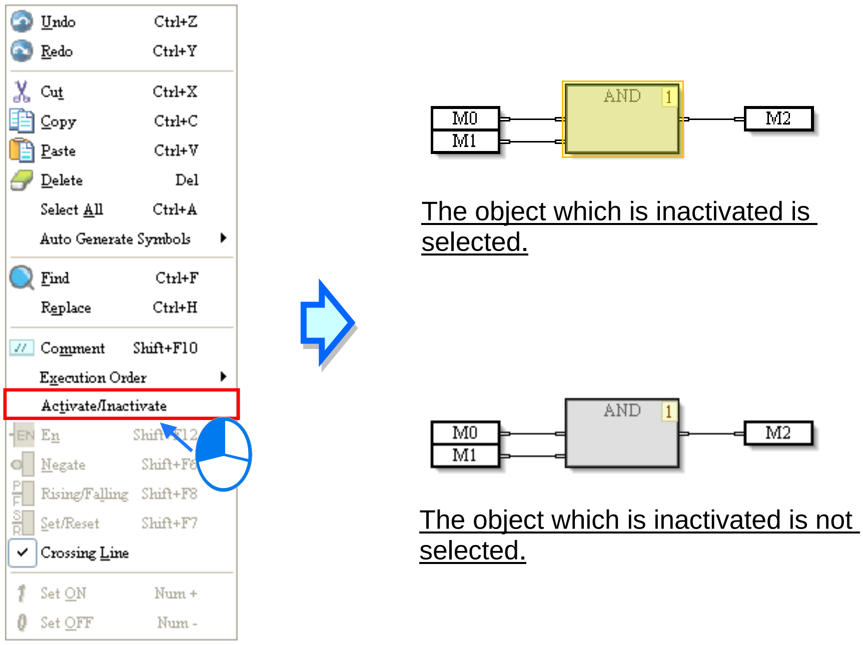


Figure 12-38 Right-clicking an object

(2) After the users click **Activate/Inactivate** on the context menu which appears, the object will be in a dark color, as shown in figure 12-39.



The object which is inactivated is selected.

The object which is inactivated is not selected.

Figure 12-39 Object which is inactivated

(3) If the users want to activate the object, they have to right-click the object, and click **Activate/Inactivate** on the context menu which appears.

12.2.10 Inserting a Comment

Users can insert comments in a continuous function chart. There are two methods of inserting a comment.

- Method 1: Inserting a comment in any position

  (1) The users have to click **Comment** on the PMSoft toolbar, as shown in figure 12-40.

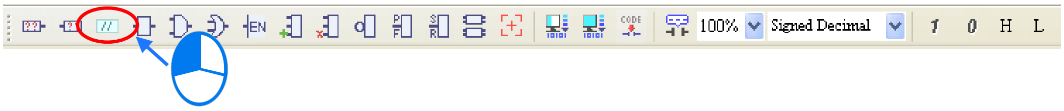


Figure 12-40 **Comment**

  (2) After the users click a position in the program editing area, an object will be inserted, as shown in figure 12-41.

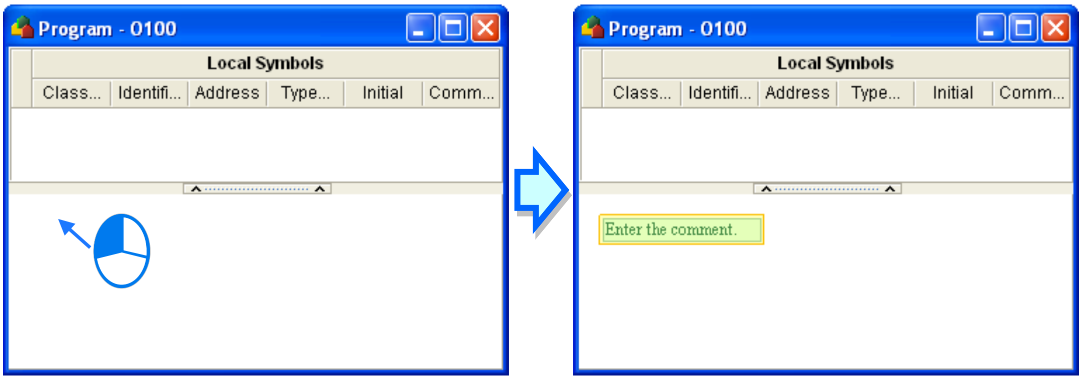


Figure 12-41 Inserting a comment

(3) After the users double-click the object inserted, a box will appear. The users can type a comment in the box, as shwon in figure 12-42. If the users want to start a new line of text at a specific point, they can press Shift+Enter on the keyboard.

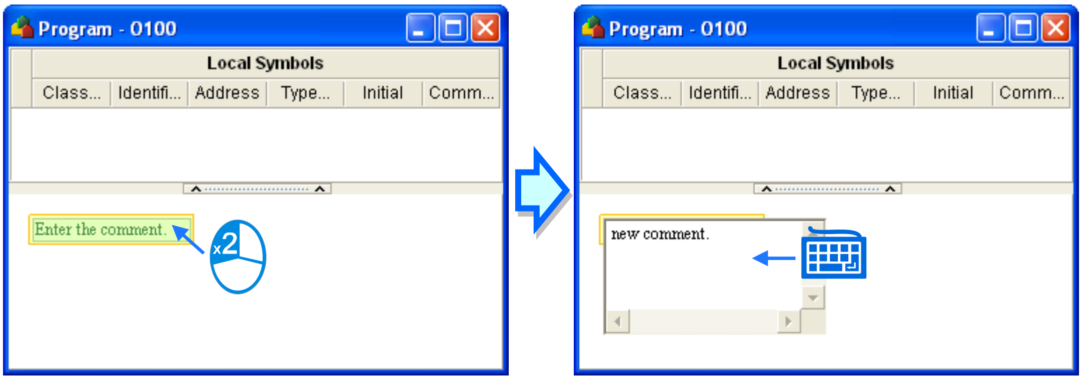


Figure 12-42 Typing a comment

(4) After the users type a comment, they have to click the blank in the program editing window, or press Enter on the keyboard, as shown in figure 12-43. This type of comment is independent of the other types of objects in the program editing area. It does not vary with the deletion or the movement of another type of object.

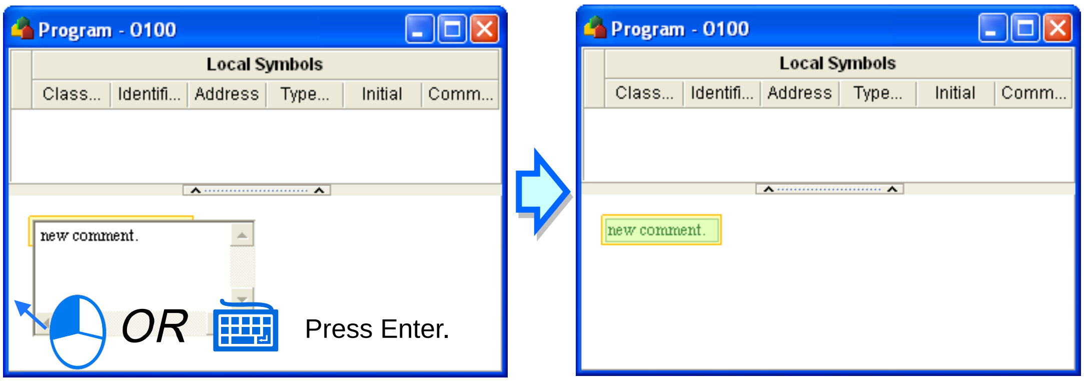


Figure 12-43 Comment which is inserted

- Method 2: Inserting a comment binding with an object

(1) The users have to click the object in which a comment will be inserted, as shown in figure 12-44.

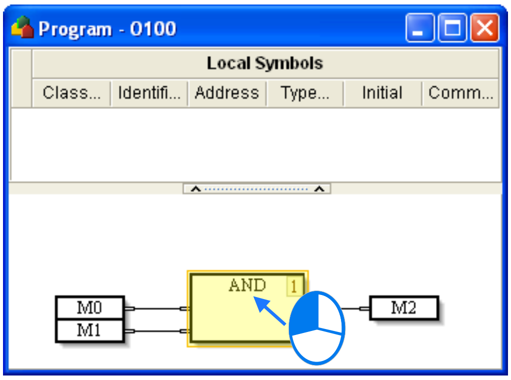


Figure 12-44 Clicking an object

(2) After the users click **Comment** on the PMSoft toolbar, the comment binding with the object will appear, as shown in figure 12-45. The users can edit the comment.

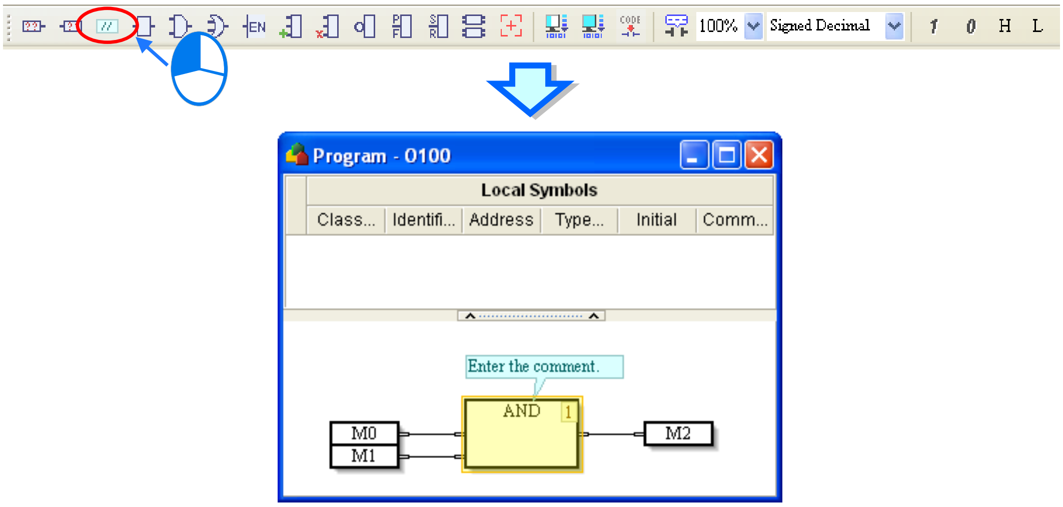


Figure 12-45 Comment binding with an object

(3) The position of the comment varies with the movement of the object. When the users move the object, the comment moves, as shown in figure 12-46.

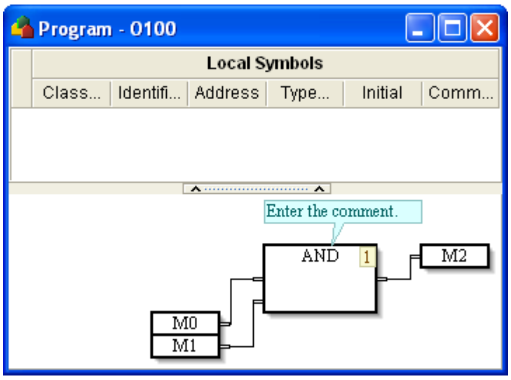


Figure 12-46 Comment binding with an object

12.2.11 Changing the Order in Which Objects are Executed

The numbers in the upper right corners of the objects in a continuous function chart indicate the order in which the objects are executed. If users want to change the number in the upper right corner of an object, they can right-click the object, point to **Execution Order** on the context menu which appears, and click **Up**, **Down**, **Front**, **Back**, **Insert After**, **Data Flow**, or **Topology**. Please see figure 12-47. If **Data Flow** is clicked, the order in which the objects in a continuous function chart are executed will be determined by the data flow of the objects. If **Topology** is clicked, the order in which the objects in a continuous function chart are executed will be determined by the relative positions of the objects.

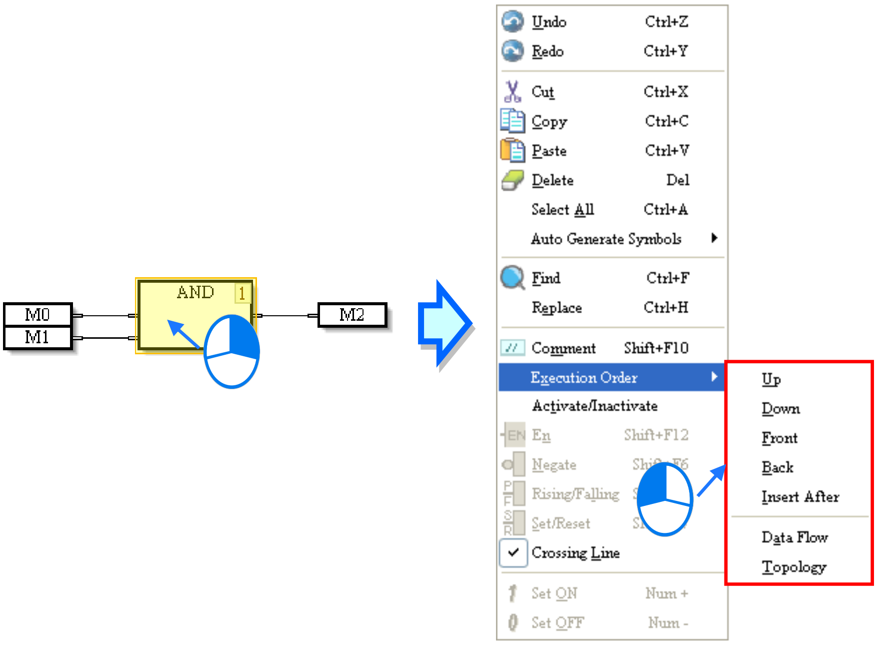
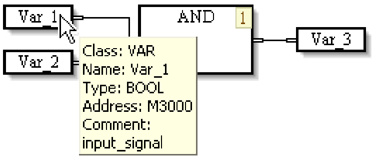


Figure 12-47 Changing the number in the upper right corner of an object

12.2.12 Displaying / Hiding Information

If users move the mouse cursor to a device or a symbol after they click on the PMSoft toolbar, the information about the device or the symbol will appear. The information related to a device or a symbol includes a device address and a comment, as shown in figure 12-48.



Class: VAR
Name: Var_1
Type: BOOL
Address: M3000
Comment:
input_signal

Figure 12-48 Displaying information

12.3 Monitoring a Continuous Function Chart

12.3.1 Screen

Users have to connect a motion controller to a computer, and download a program to the motion controller. When the state of the motion controller is monitored, the users can monitor a continuous function chart. After the users open a program editing window in which a continuous function chart is created, the text boxes showing the states of nodes and the values in nodes will appear, as shown in figure 12-49.

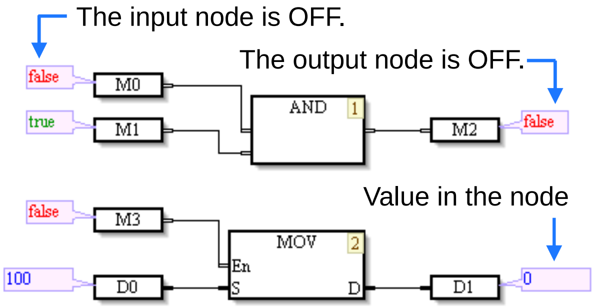


Figure 12-49 Monitoring a continuous function chart online

The users can select **Signed Decimal**, **Unsigned Decimal**, **Hexadecimal**, or **Float Point** in the **Monitoring Radix** drop-down list box on the **View** menu, or in Signed Decimal on the PMSoft toolbar, as shown in figure 12-50 and figure 12-51.

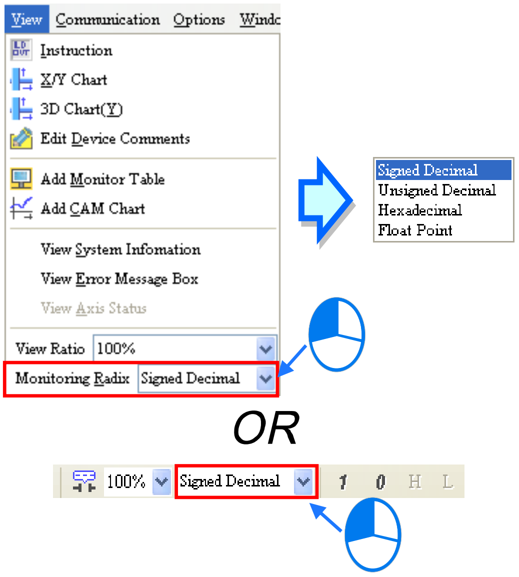
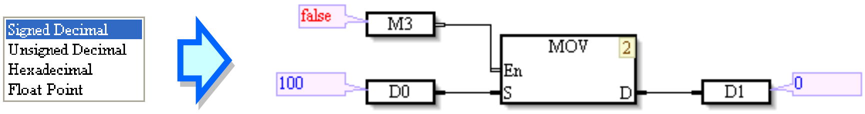


Figure 12-50 **Monitoring Radix**

- **Signed Decimal is selected.**



- **Hexadecimal** is selected.

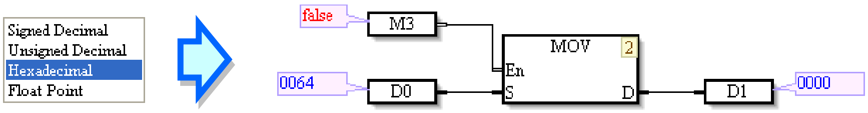


Figure 12-51 Changing a radix

12.3.2 Controlling a Node Online

(1) If users want to change the state of a node online, they have to select the text box showing the state of the node, right-click the text box, and click **Set ON** or **Set OFF** on the context menu which appears, or click 1 or 0 on the toolbar. Please see figure 12-52.

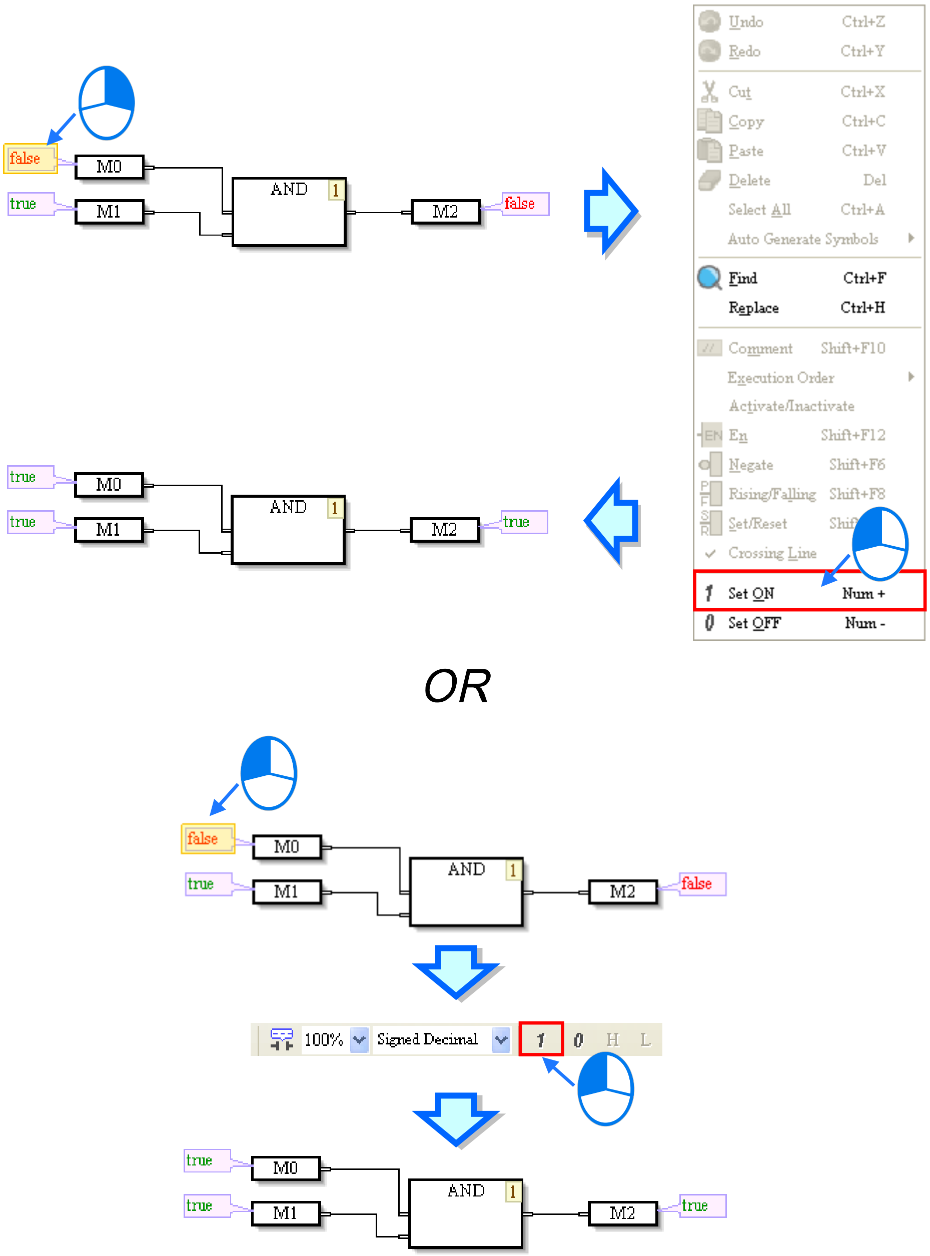


Figure 12-52 Changing the state of a node

- **Set ON**: The node selected is set to ON.

- **Set OFF**: The node selected is set to OFF.

The users can change the state of a node by double-clicking the text box showing the state of the node, as shown in figure 12-53.

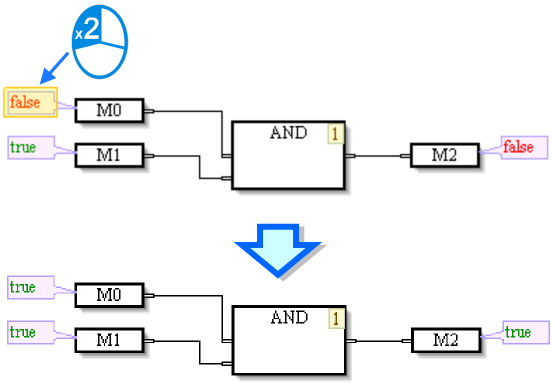


Figure 12-53 Changing the state of a node

(2) If the users want to change the value in a node, they have to double-click the text box showing the value in the node, type a value, and press Enter on the keyboard, or click the blank in the program editing area. Please see figure 12-54.

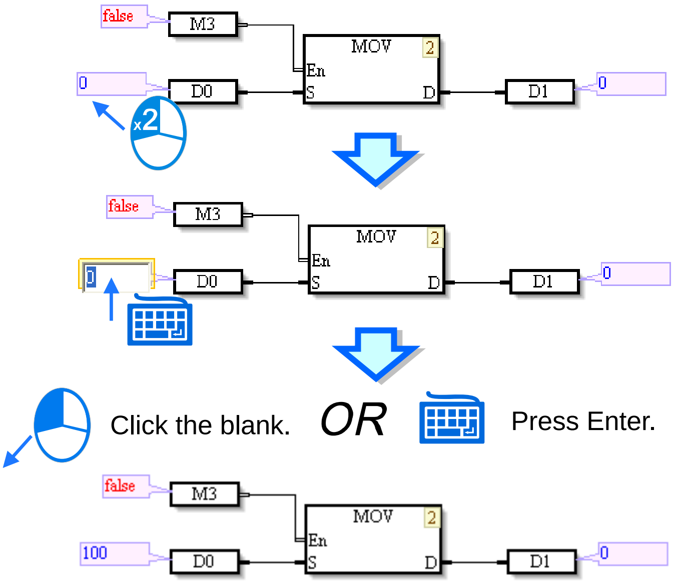


Figure 12-54 Changing the value in a node

Chapter 13 New Functions and Revised Functions

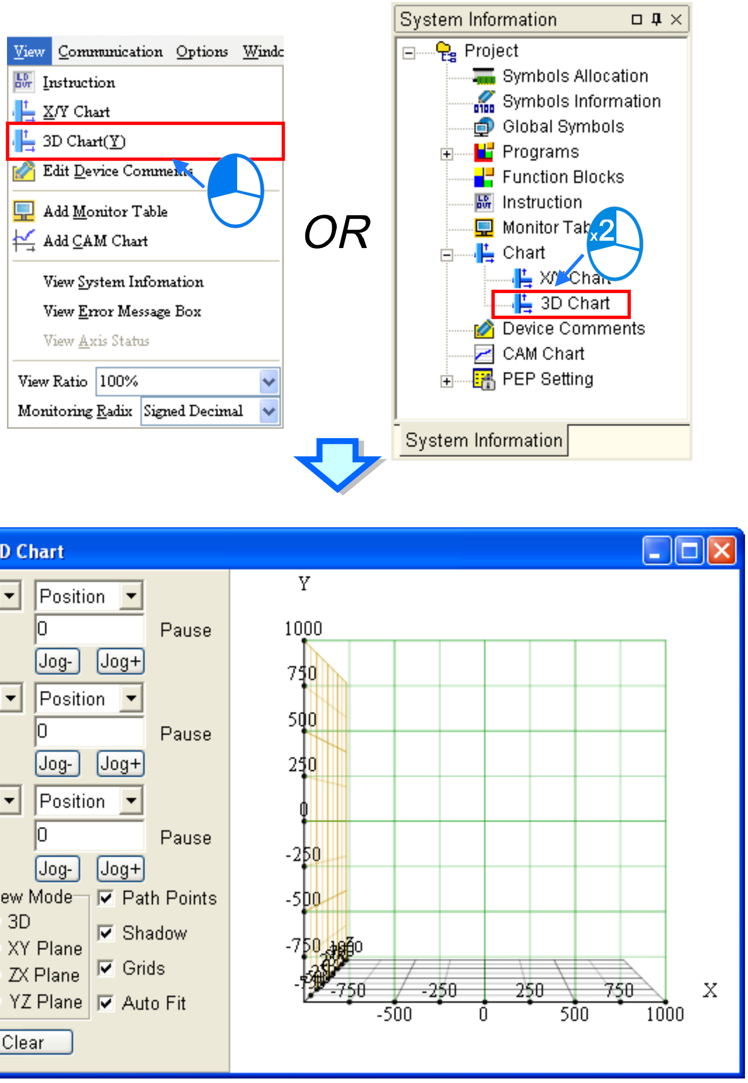
Table of Contents

13.1 Three-dimensional Chart

13.1.1 Functions

When the multiaxial motion controlled by a program is monitored, a motion path can be drawn on a three-dimensional chart. Users can check whether the multiaxial motion is correct by monitoring the three-dimensional chart. A motion controller has to be connected a computer, and a program has to be downloaded to the motion controller. When the state of the motion controller is monitored, a three-dimensional chart can be monitored.

(1) The users have to click **3D Chart** on the **View** menu, or double-click **3D Chart** in the system information area, as shown in figure 13-1.



View Ratio: 100%
Monitoring Radix: Signed Decimal

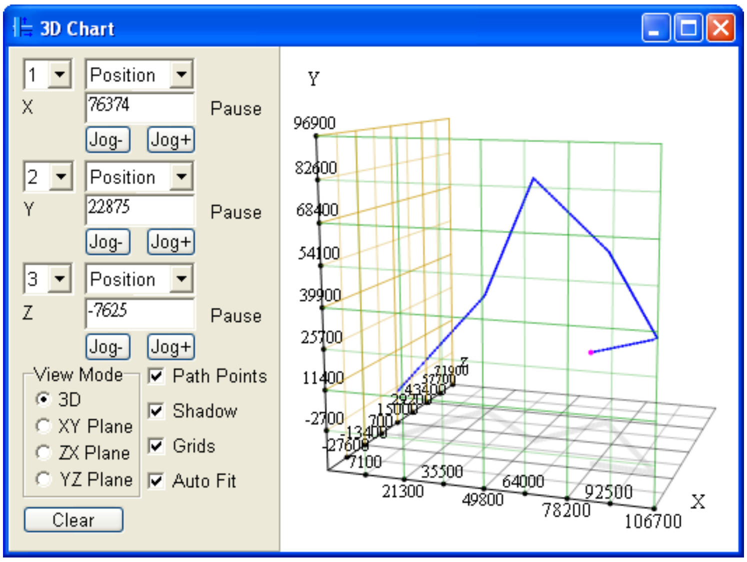
X: 1, Position, 0
Y: 2, Position, 0
Z: 3, Position, 0

View Mode: ● 3D ○ XY Plane ○ ZX Plane ○ YZ Plane

☑ Path Points
☑ Shadow
☑ Grids
☑ Auto Fit

Figure 13-1 Opening the **3D Chart** window

(2) Users have to make the motion controller operational. After the motion controller begins to execute the motion program, the three-dimensional motion path of the three axes of the motion controller will be drawn in the **3D Chart** window according to the actual output of the three axes. Please see figure 13-2.



1 Position
X 76374 Pause

2 Position
Y 22875 Pause

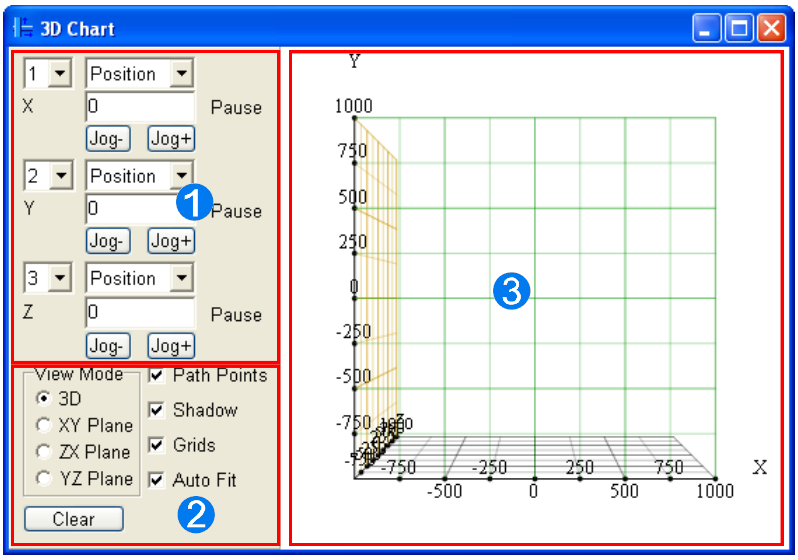
3 Position
Z -7625 Pause

View Mode: ● 3D ○ XY Plane ○ ZX Plane ○ YZ Plane

☑ Path Points ☑ Shadow ☑ Grids ☑ Auto Fit

Figure 13-2 Three-dimensional chart

(3) The **3D Chart** window shown in figure 13-3 is described below.



1 Position
X 0 Pause

2 Position
Y 0 Pause

3 Position
Z 0 Pause

View Mode: ● 3D ○ XY Plane ○ ZX Plane ○ YZ Plane

☑ Path Points ☑ Shadow ☑ Grids ☑ Auto Fit

Figure 13-3 Three-dimensional chart

➊ The users can select the channel or the station address corresponding to an axis in the left drop-down list box of the axis, and select **Position** or **Speed** in a right drop-down list box, as shown in figure 13-4.

1 Position

Figure 13-4 Setting a channel, and selecting **Position** or **Speed**

When an axis moves, the box under the right drop-down list box of the axis shows the position or the speed of the axis, as shown in figure 13-5. If the Jog- button or the Jog+ button of an axis is clicked, the axis will move in the positive direction or in the negative direction.

X 76374

Figure 13-5 Information which is displayed

❷ In the **View Mode** section, the users can select the 3D option button, the XY Plane option button, the ZX Plane option button, or the YZ option button. Besides, the users can select the **Path Points** checkbox, the **Shadow** checkbox, and the **Grids** checkbox. If the **Auto Fit** checkbox is selected, the coordinate ranges on the chart will be modified according to the size of the path drawn. **Clear** is used to clear the path which has been drawn.

❸ The path of the three axes of the motion controller is shown here. The users can turn the chart in the **3D Chart** window by dragging the chart, as shown in figure 13-6. Besides, the users can adjust the coordinate ranges on the chart by using the mouse wheel.

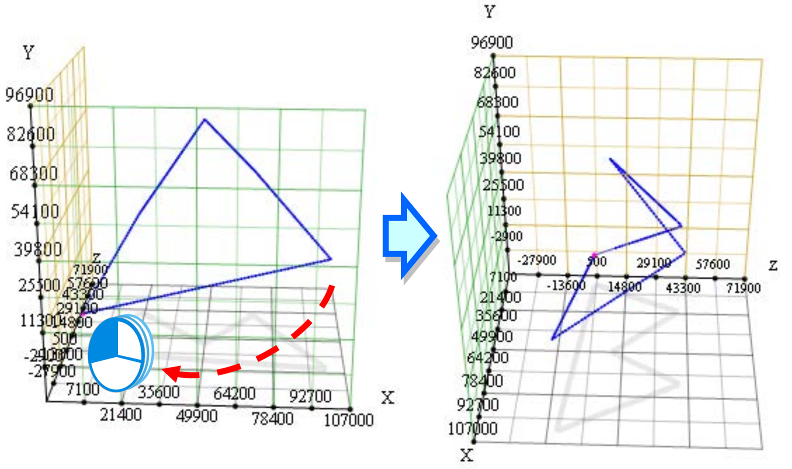


Figure 13-6 Turning the chart in the **3D Chart** window

When the mouse cursor hovers over the path on the chart, coordinates appear, as shown in figure 13-7.

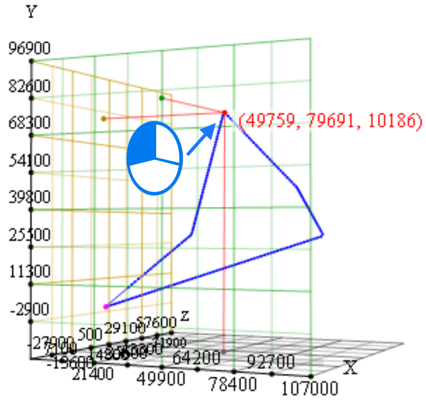


Figure 13-7 Coordinates

13.2 POUs Which are Programs

13.2.1 Creating POUs Which Are Programs

- O100 POU

  Users can manage the main program O100 by creating several O100 POUs. There is no limit on the number of O100 POUs which can be created, but users have to note the limit on the size of the program in a motion controller. The users have to right-click **O100** in the system information area, and click **New POU…** on the context menu which appears, as shown in figure 13-8.

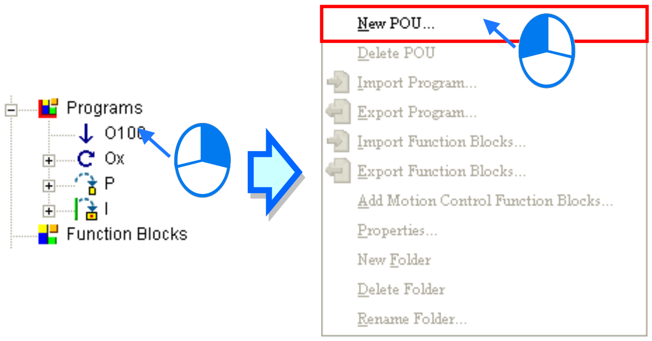
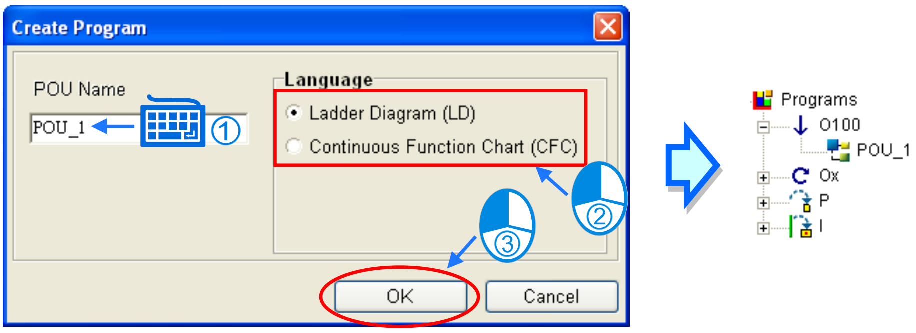


Figure 13-8 Adding a new POU

The users have to type a POU name in the **Create Program** window, select a programming language, and click **OK**, as shown in figure 13-9.



POU Name: POU_1

Language
● Ladder Diagram (LD)
○ Continuous Function Chart (CFC)

Figure 13-9 Items in the **Create Program** window

If the users repeat the steps above, they can create several O100 POUs, as shown in figure 13-10. When the O100 POUs created are compiled, they are transformed into execution code in order

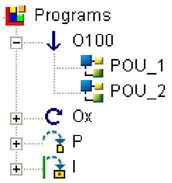


Figure 13-10 Several O100 POUs

If the users want to change the order in which the O100 POUs created are executed, they can drag an O100 POU to another position. In figure 13-11, the order in which POU_1, POU_2, and POU_3 are executed is POU_1→POU_3→POU_2.

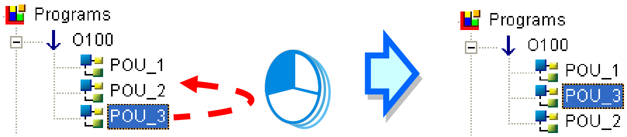


Figure 13-11 Change the order in which the O100 POUs created are executed

- Ox POU, P POU, and I POU

  If users want to create an Ox POU/P POU/I POU, they have to expand the **Ox/P/I** section in the system information area, and double-click a number. After the users select a programming language in the **Create Program** window, they have to click **OK**. (The POU name in the **Create Program** window can not be changed.) Please see figure 13-12.

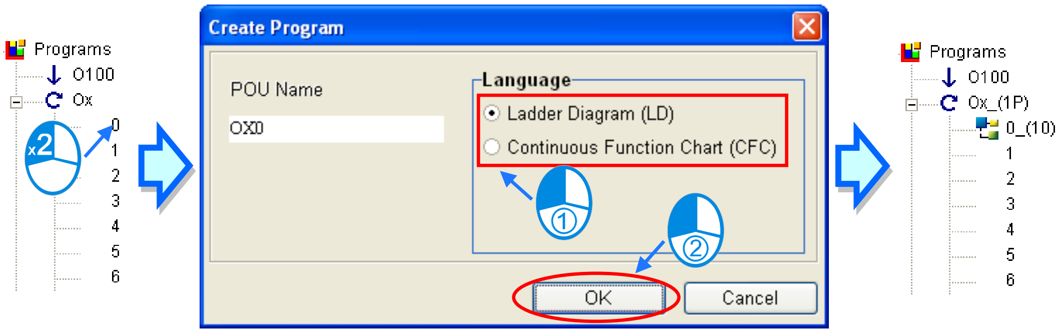


Figure 13-12 **Create Program** window

13.2.2 Changing the Property of an O100 POU

If users want to change the property of an O100 POU, they have to right-click the O100 POU, and click **Properties…** on the context menu which appears, as shown in figure 13-13.

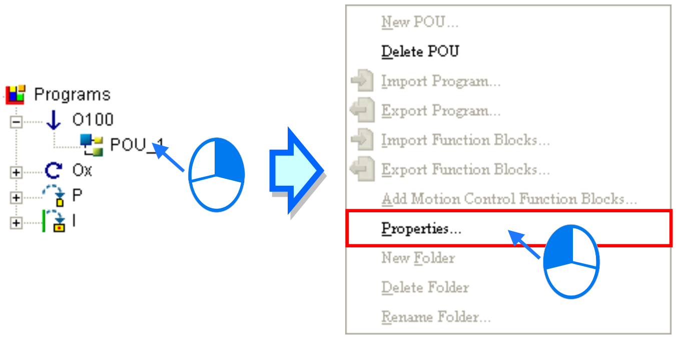


Figure 13-13 Changing the property of an O100 POU

The POU name in the **Property** window can be changed. Please see figure 13-14.

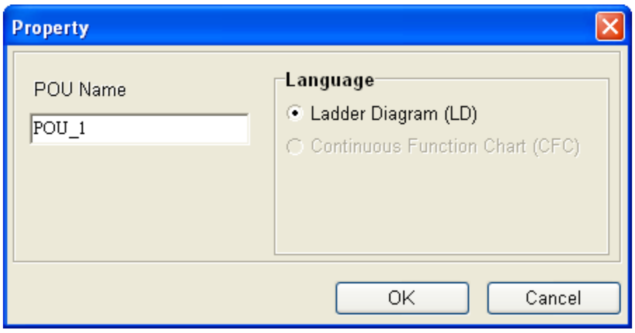
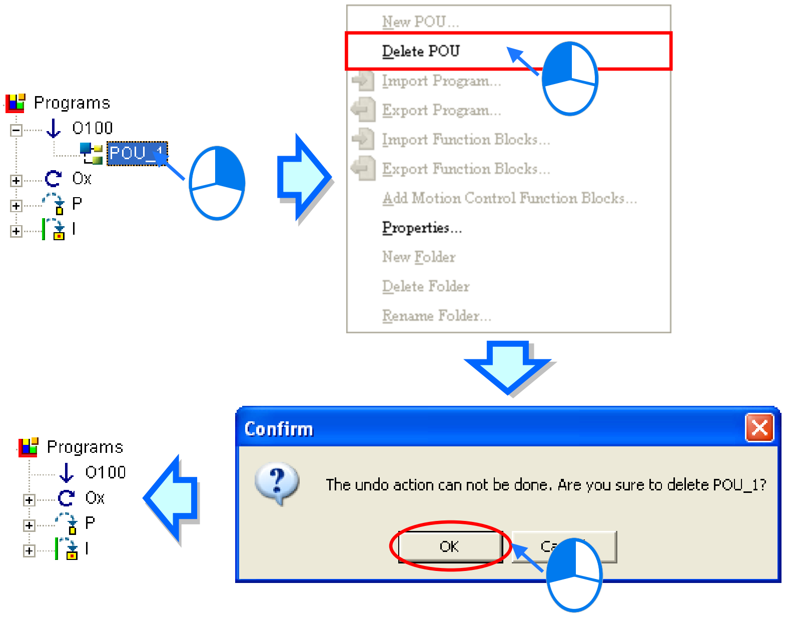


Figure 13-14 **Property** window

13.2.3 Deleting a POU

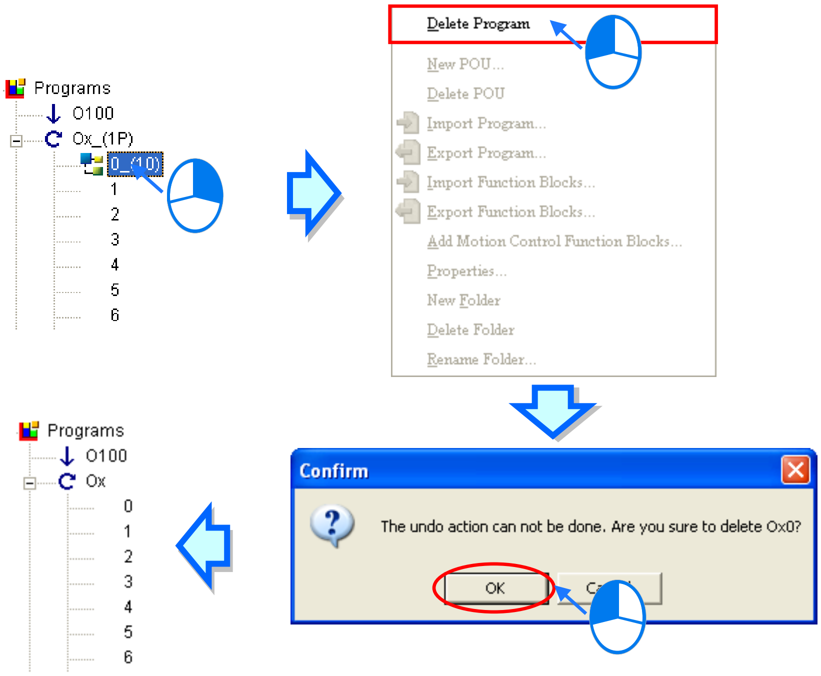
If users want to delete an O100 POU, they have to right-click the O100 POU, click **Delete POU** on the context menu which appears, and click **OK** in the **Confirm** window, as shown in figure 13-15.



The undo action can not be done. Are you sure to delete POU_1?

Figure 13-15 Deleting an O100 POU

If the users want to delete an Ox POU/P POU/I POU, they have to right-click the Ox POU/P POU/I POU, click **Delete Program** on the context menu which appears, and click **OK** in the **Confirm** window, as shown in figure 13-16.



The undo action can not be done. Are you sure to delete Ox0?

Figure 13-16 Deleting an Ox POU

13.2.4 Displaying/Hiding the POUs which Are Not Created

After the **Ox/P/I** section is expanded, a list of Ox/P/I POUs will be displayed. If users want to hide the POUs which are not created, they have to right-click **Programs**, and click **(Show/Hide) not created POUs** on the context menu which appears, as shown in figure 13-17.

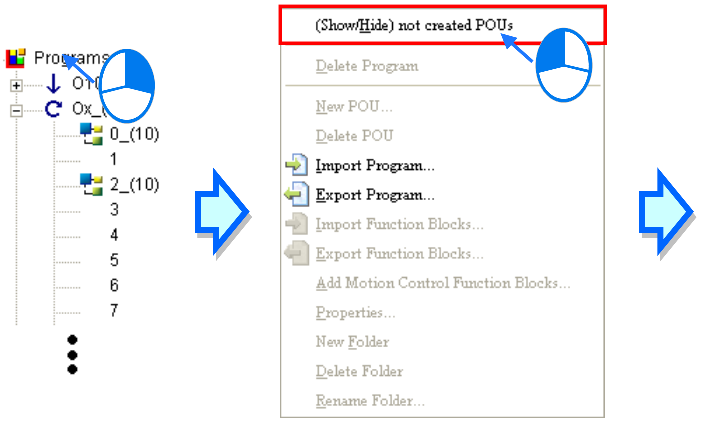


Figure 13-17 Displaying/Hiding the POUs which are not created

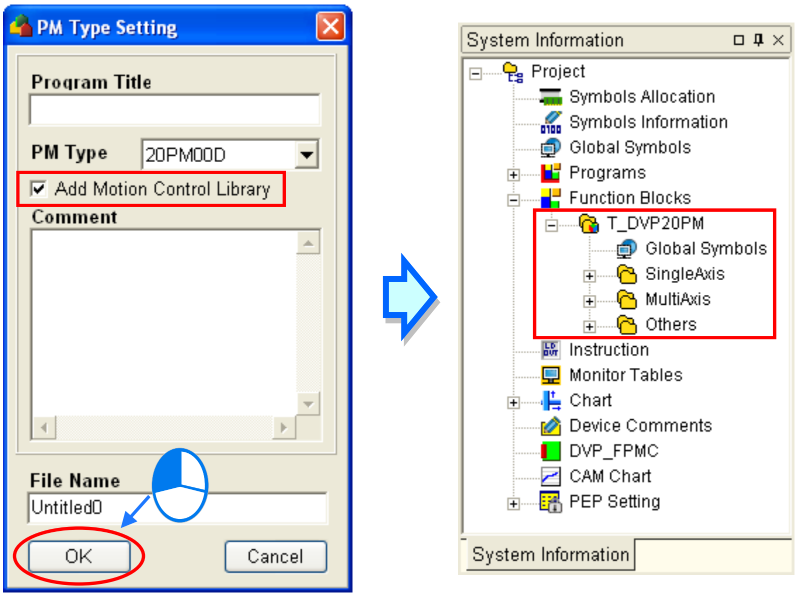
If the users want to create a new Ox/P/I POU, the POUs which are not created must be displayed. If the POUs which are not created need to be displayed, the users have to right-click **Programs** again, and click **(Show/Hide) not created POUs** on the context menu which appears.

13.3 Delta Libraries

13.3.1 Adding Delta Libraries

PMSoft provides various libraries for various models so that users can use motion control functions conveniently. A library can be used only after it is added to the **Function Blocks** section. There are two methods of adding libraries.

- Method 1: Libraries are added when a project is created.
  (1) When users create a project, the **Add Motion Control Library** checkbox in the **PM Type Setting** window is selected by default. After the users click **OK**, all Delta libraries will be added to the **Function Blocks** section, as shown in figure 13-18.


PM Type Setting

Program Title

PM Type: 20PM00D

☑ Add Motion Control Library

Comment

File Name: Untitled0

Figure 13-18 Adding motion control libraries

- Method 2: Libraries are added to a project.

  (1) If users do not need to add all libraries when they create a project, they have to unselect the **Add Motion Control Library** checkbox in the **PM Type Setting** window. Please see figure 13-19.

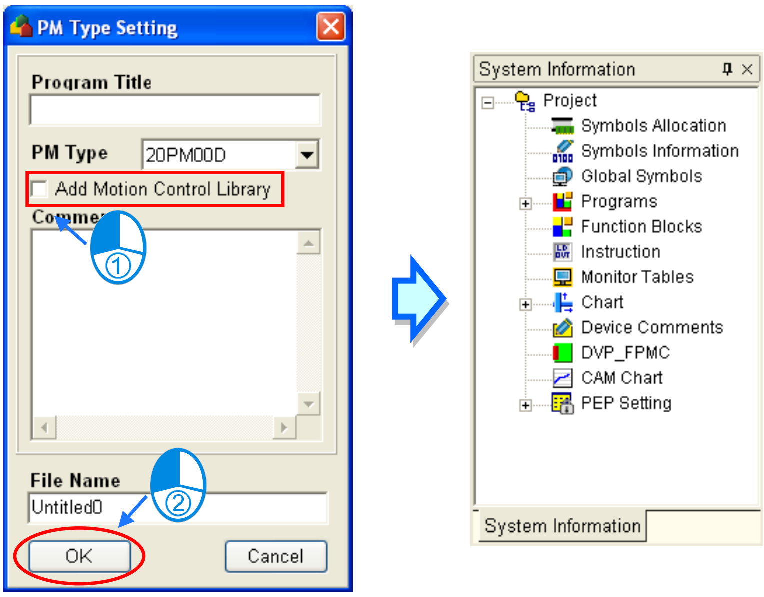

PM Type: 20PM00D
□ Add Motion Control Library
File Name: Untitled0

Figure 13-19 Not adding motion control libraries

  (2) The users have to right-click the item which needs to be added in the **Delta Libraries** section. After the users click **Add to Function Block Area** on the context menu which appears, the item will be added to the **Function Blocks** section in the system information area, as shown in figure 13-20.

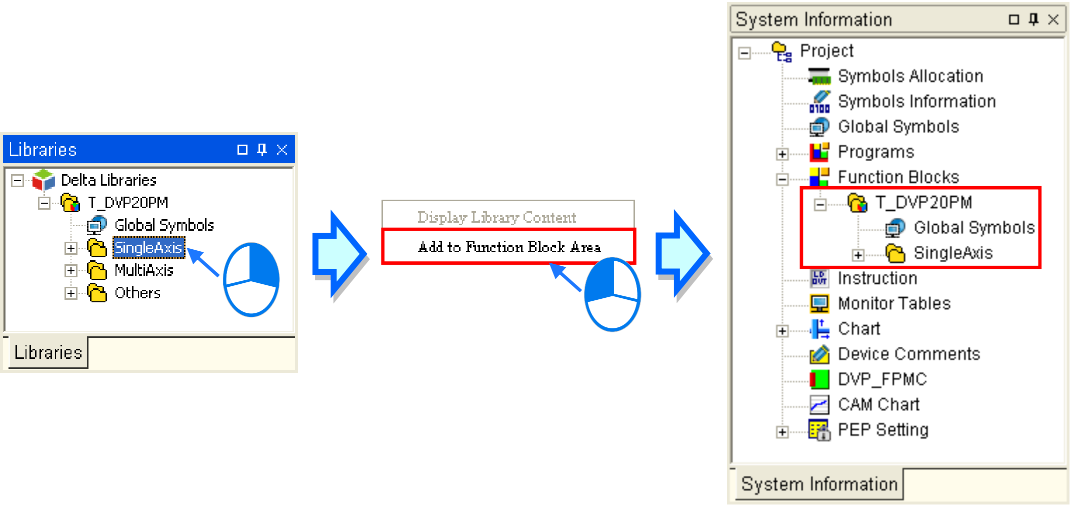


Figure 13-20 Adding libraries to the **Function Blocks** section

(3) The users can right-click **Function Blocks** in the system information area, and click **Add Motion Control Function Blocks…** on the context menu which appear, as shown in figure 13-21.

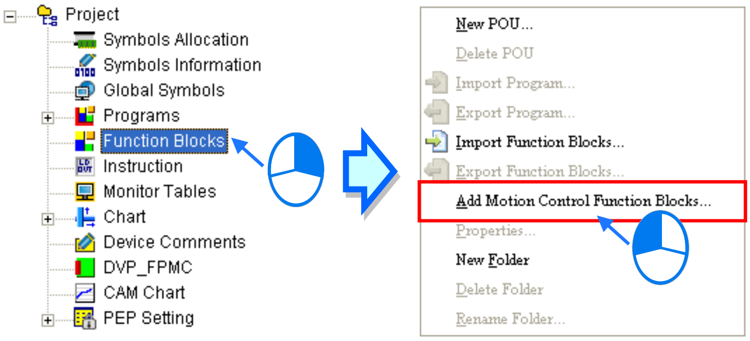


Figure 13-21 Adding motion control function blocks

(4) In the **Add Function Block window**, the users have to select a function block type in the **Function Block Info** section, and select function block definitions in the **Function Blocks** section. If the users click **Select All**, all the function block definitions in the **Function Blocks** section will be selected. If the users click **Deselect**, all the function block definitions in the **Function Blocks** section will be unselected. Finally, the users have to click **OK**. Please see figure 13-22.

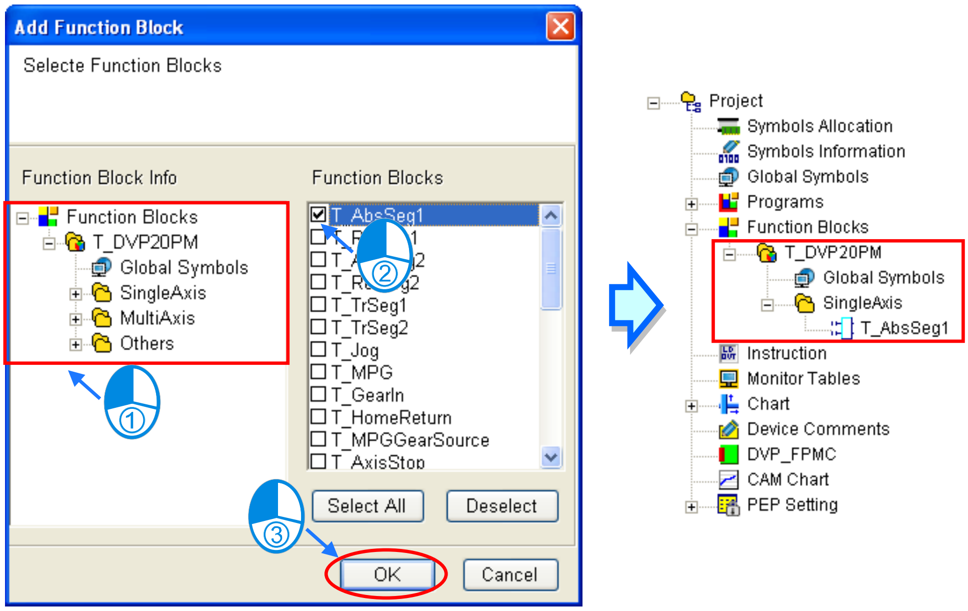


Figure 13-22 Selecting function blocks which need to be added

13.3.2 Using the Function Blocks in Delta Libraries

After users finish adding motion control function blocks, the motion control function blocks will be displayed in the **Function Blocks** section in the system information area, as shown in figure 13-23. In figure 13-23, the **T_DVP20PM** folder represents the function block type applicable to a DVP-20PM series motion controller. The function block definitions which can be added vary with the motion controller used. **Global Symbols** represents the global symbol table applicable to the function block definitions in the **T_DVP20PM** folder. The **SingleAxis** subfolder represents the function block type which is added. The **T_DVP20PM** folder, the **SingleAxis** subfolder, and **Global Symbols** are created when the function block definition **T_AbsSeg1** is added. The function block definitions and the folders which are created by the users can not be the same as the function block definition and the folders mentioned above. Otherwise, a message will appear, and the users will be asked to modify the function blocks and the folders they create.

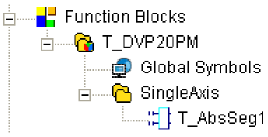


Figure 13-23 Selecting function blocks which need to be added

The use of the function block definitions which are added is the same as that of general function block definitions. Please refer to section 4.2.2 for more information. The function block definitions which are added can not be modified, and can only be used to produce function block instances.

Pins of function blocks in Delta libraries are provided with parameters which are defined so that users can use the function blocks conveniently. After the users type parameters which are defined in a function block, the function block can be executed. The users do not need to know the values or the states which should be set. In figure 13-24, TRUE, mcNegative, mcFalling are parameters which are defined. Please refer to appendix B for more information.

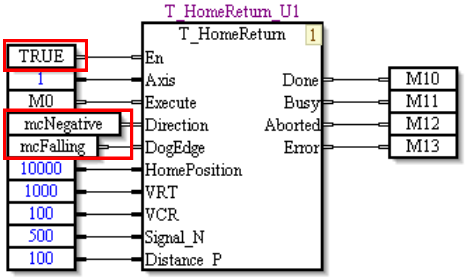


Figure 13-24 Parameters of a function block

13.4 New Functions Related to a Function Block

13.4.1 Condition Contact of a Function Block

The state of the condition contact preceding a function block determines whether the function block is executed. Pleas see figure 13-25

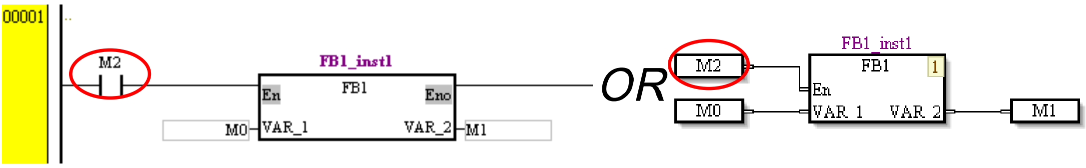


Figure 13-25 Condition contact preceding a function block

If a function block is not connected to a condition contact, it will always be executed. Please see figure 13-26. If a function block does not need to be connected to a condition contact in a ladder diagram, it will be connected to a busbar directly. If a function block does not need to be connected to a condition contact in a continuous function chart, the En pin of the function block will be disabled.

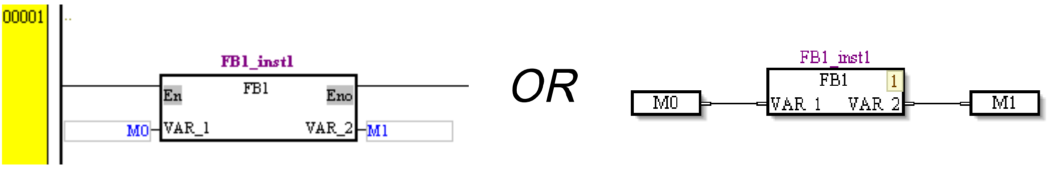


Figure 13-26 Canceling the condition contact preceding a function block

13.4.2 Omitting the Pins of a Function Block

The symbol or the device assigned to an input pin or an output pin of a function block is used as an input interface or an output interface of the function block. Users sometimes do not need to specify an input value, or can ignore output information, and therefore they do not need to assign a symbol or a device to a pin of a function block. After a function block is added to a program section, the pins of the function block will be marked with … , that is, the users does not need to define the pins of the function block.

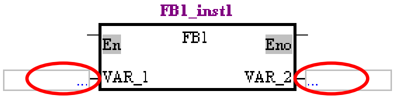


Figure 13-27 Mark indicating that a pin of a function block can be omitted

However, the users have to assign a symbol or a device to a general contact or coil. ??? on a general contact or coil reminds the users to assign a symbol or a device, as shown in figure 13-28.

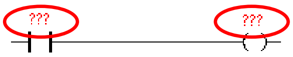
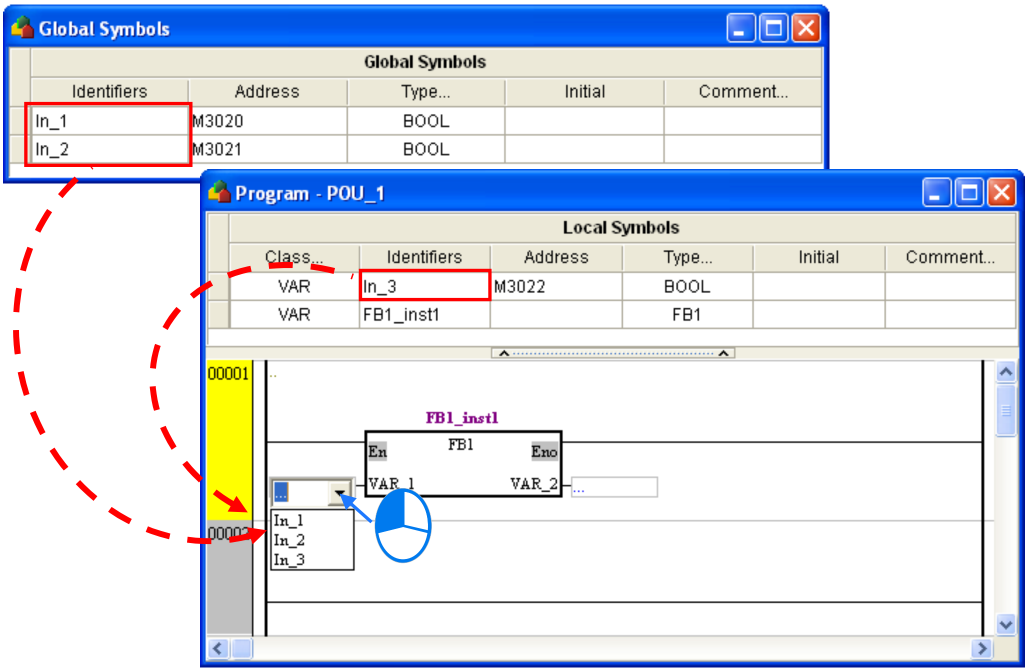


Figure 13-28 Mark indicating that a contact/coil can be omitted

13.4.3 Declaring the Symbol Assigned to a Pin of Function Block

A device or a symbol can be assigned to a pin of a function blocks. There are two methods of assigning a symbol to a pin of a function block.

- Method 1: A symbol which has been declared in a symbol table is used.
  (1) If users have declared symbols in the global symbol table or in the local symbol table in a POU, a drop-down list on which there are symbols whose data types are the same will appear after the users click a pin of a function block in the POU. Please see figure 13-29.



Global Symbols

Global Symbols

Identifiers	Address	Type...	Initial	Comment...
In_1	M3020	BOOL		
In_2	M3021	BOOL		

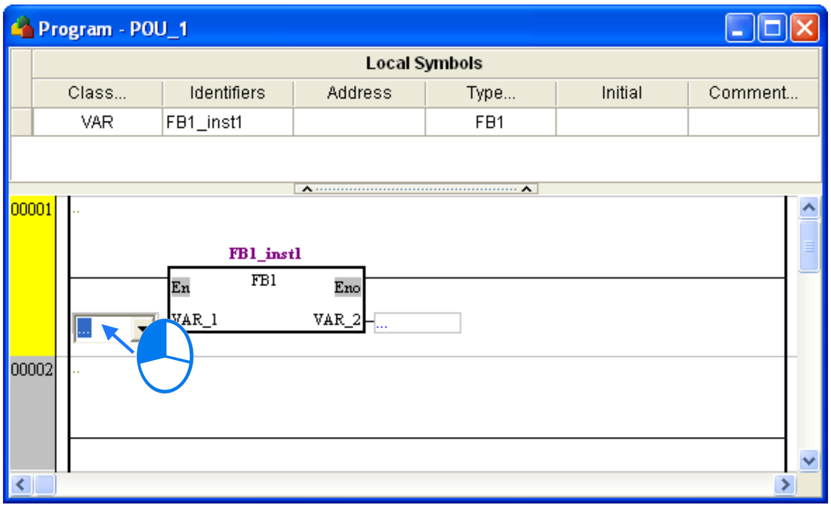
Program - POU_1

Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	In_3	M3022	BOOL		
VAR	FB1_inst1		FB1		

Figure 13-29 Selecting a symbol which has been declared for a pin

- Method 2: After a symbol is assigned to a pin, the symbol will be declared in a symbol table.
  (1) After users click a pin of a function block in a POU, they can type text, as shown in figure 13-30.



Program - POU_1

Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	FB1_inst1		FB1		

Figure 13-30 Clicking a node

(2) After the users type an identifier, and press Enter on the keyboard, the **Add Symbol** window will appear, as shown in figure 13-31. The identifier and the data type of the pin are automatically brought into the **Add Symbol** window. The users can type other information. If the **Add to global symbol table** checkbox is selected, the symbol will be declared in the global symbol table. If the **Add to global symbol table** checkbox is unselected, the symbol will be declared in the local symbol table in the POU.

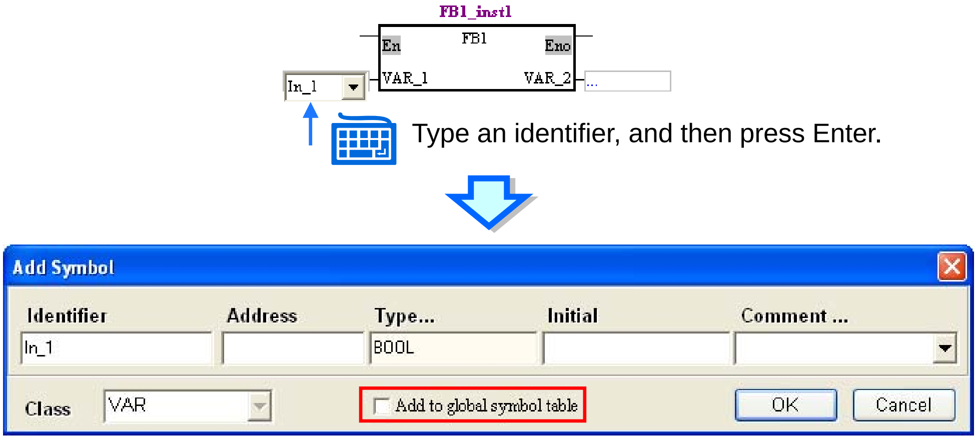


Figure 13-31 Adding a symbol

13.4.4 Automatically Assigning Symbols to the Pins of a Function Block

If users want to assign symbols to the pins of a function block in a POU at a time, they have to right-click the function block, point to **Auto Generate Symbols** on the context menu which appears, and click **Generate by Default** or **Input Words by Prefix**. Please see figure 13-32.

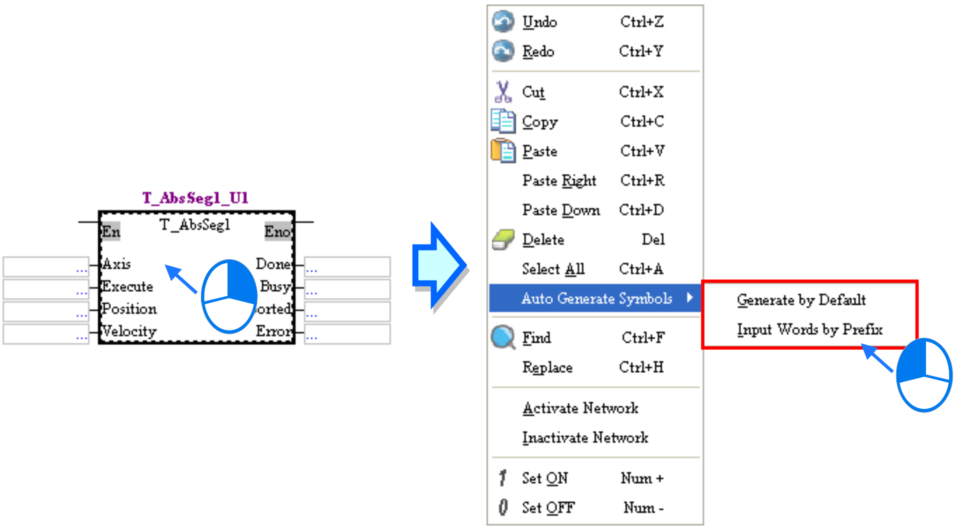


Figure 13-32 Automatically assigning symbols to the pins of a function block

- Function 1: **Generate by Default**

(1) After **Generate by Default** is clicked, symbols will be assigned to the pins of the function block, and they will be declared in the local symbol table in the POU, as shown in figure 13-33. The rule of generating a symbol is “First seven letters in a function block definition_Pin name”. For example, the symbol automatically assigned to the Axis pin of the function block definition T_AbsSeg1 is “T_AbsSe_Axis”.

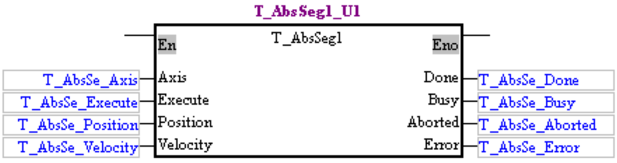


Figure 13-33 Automatically assigning default symbols to the pins of a function block

(2) If a function block definition has two function block instances in a program, the symbols assigned to the pins of the second function block instance will be generated according to the same rule after **Generate by Default** is clicked, but a number will be added to the ends of the symbols. Please see figure 13-34.

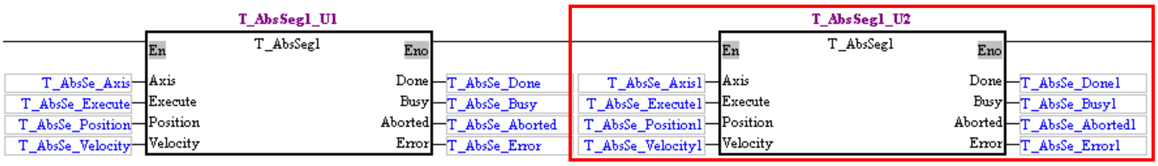


Figure 13-34 Automatically assigning default symbols to the pins of a function block again

(3) If a symbol which has been declared is assigned to a pin of a function block, the symbol will be retained after **Generate by Default** is clicked, as shown in figure 13-35.

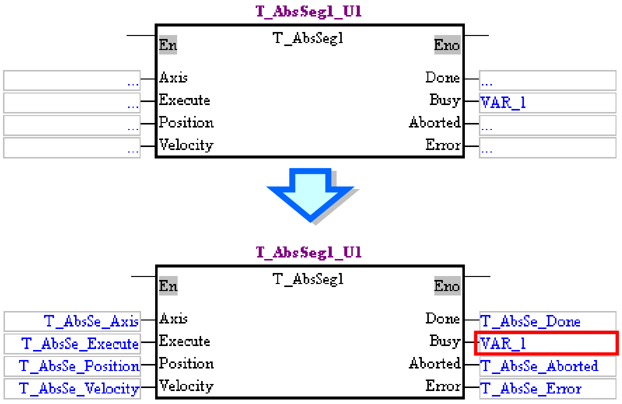


Figure 13-35 Automatically assigning default symbols to pins of a function block

- Function 2: **Input Words by Prefix**

(1) After **Input Words by Prefix** is clicked, the **Input Words by Prefix** window will appear. After the users type a prefix, and click **OK** in the **Input Words by Prefix** window, symbols will be assigned to the pins of the function block automatically, and they will be declared in the local symbol table in the POU, as shown in figure 13-36. The rule of generating a symbol is "Prefix_Pin name". For example, if the prefix typed in the **Input Words by Prefix** window is "Test", the symbol automatically assigned to the Axis pin of the function block definition T_AbsSeg1 will be "Test_Axis".

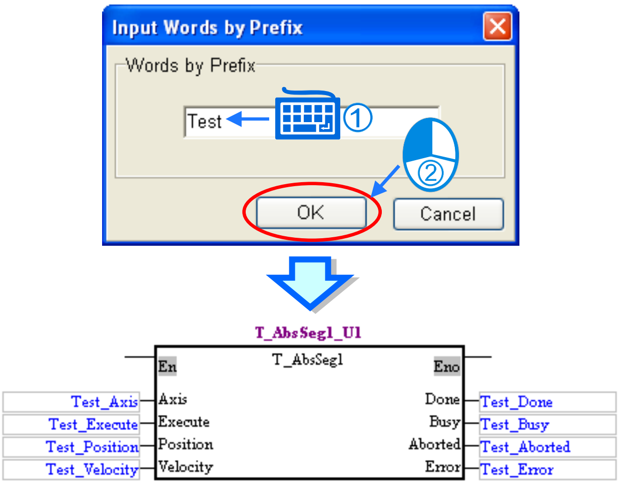


Figure 13-36 Typing a prefix

(2) If a function block definition has two function block instances in a program, the symbols assigned to the pins of the second function block instance will be generated according to the same rule after **Input Words by Prefix** is clicked, but a number will be added to the ends of the prefixes in the symbols (rather than the ends of the symbols). Please see figure 13-37.

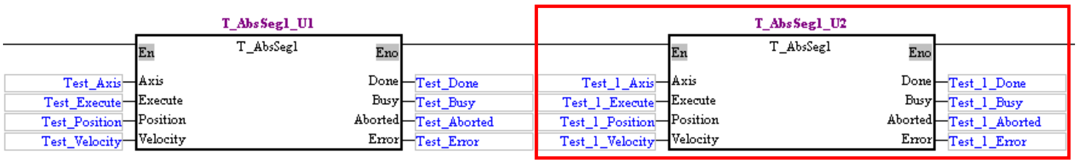
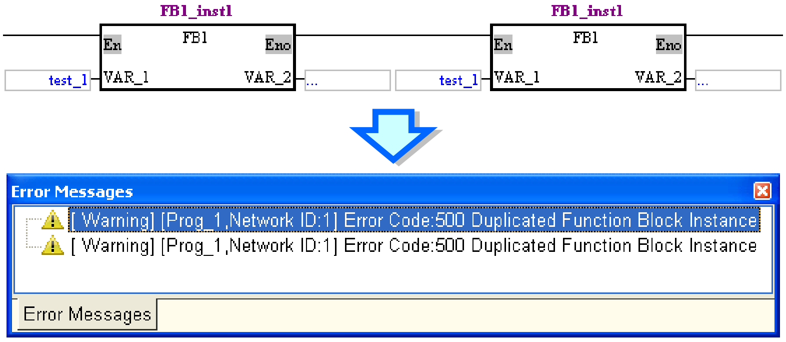


Figure 13-37 Using **Input Words by Prefix** again to assign symbols to the pins of a function block

(3) If a symbol which has been declared is assigned to a pin of a function block, the symbol will be retained after **Input Words by Prefix** is clicked.

13.4.5 Repeated Function Block Instances

A function block instance can be used several times. To prevent users from forgetting to declare a function block instance which is used repeatedly in a program, the warning message reminding the users that the function block instance is used repeatedly will appear after the program is compiled, as shown in figure 13-38. If the users make sure that the function block instance is used repeatedly, the warning message can be ignored.



[Warning] [Prog_1,Network ID:1] Error Code:500 Duplicated Function Block Instance

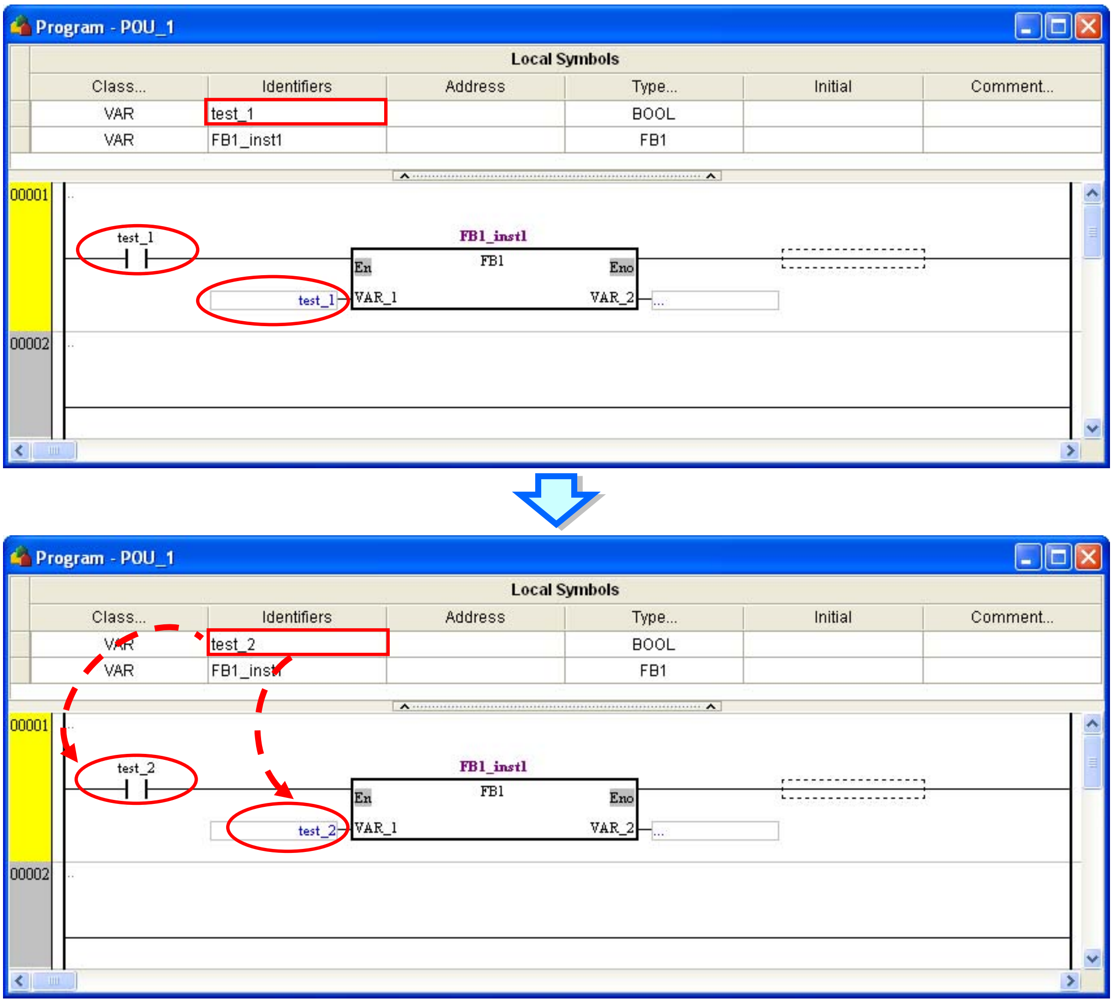
[Warning] [Prog_1,Network ID:1] Error Code:500 Duplicated Function Block Instance

Figure 13-38 Repeated function blocks

13.5 Other New Functions

13.5.1 Correspondence between the Symbols in the Local Symbol Table in a POU and the Symbols Used in the POU

If an identifier in the local symbol table in a POU is changed, the symbols corresponding to the identifier in the POU will also be changed, as shown in figure 13-39.



Program - POU_1

Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	test_1		BOOL		
VAR	FB1_inst1		FB1		

Program - POU_1

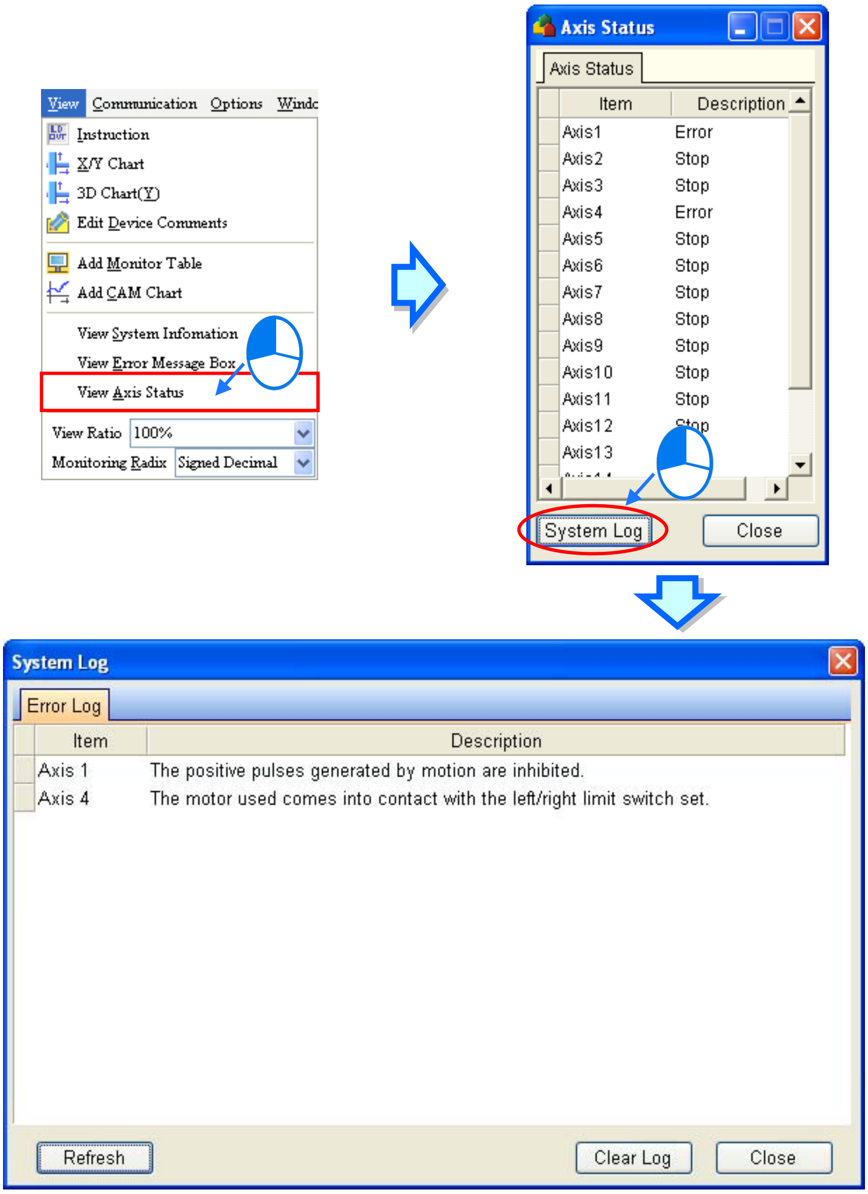
Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	test_2		BOOL		
VAR	FB1_inst1		FB1		

Figure 13-39 Automatically updating a symbol

13.5.2 Viewing the Statuses of the Axes of a Motion Controller

When a motion controller is monitored, users can view status of the axes of the motion controller by clicking **View Axis Status** on the **View** menu. If users find that an error occurs in an axis, they can click **System Log** in the **Axis Status** window, and view the error log in the **System Log** window, as shown in figure 13-40.



Item	Description
Axis1	Error
Axis2	Stop
Axis3	Stop
Axis4	Error
Axis5	Stop
Axis6	Stop
Axis7	Stop
Axis8	Stop
Axis9	Stop
Axis10	Stop
Axis11	Stop
Axis12	Stop
Axis13	

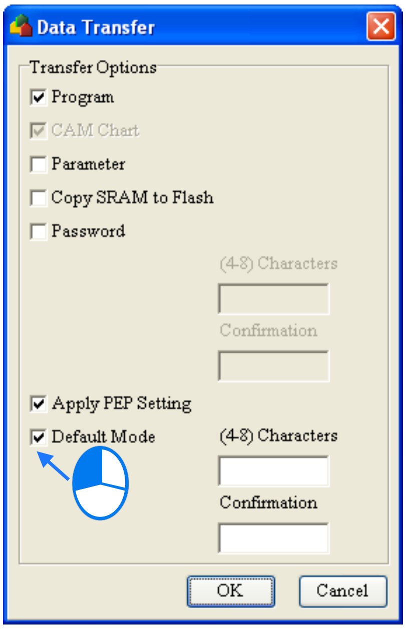
Item	Description
Axis 1	The positive pulses generated by motion are inhibited.
Axis 4	The motor used comes into contact with the left/right limit switch set.

Figure 13-40 Status of the axes of a motion controller

13.5.3 Default Mode of Program Encryption Protection

If users select the **Default Mode** checkbox in the **Data Transfer** window, all the POUs (excluding the I POUs) which are programs except Ox0 in the project created will be protected by a PEP password, all the POUs which are programs will be downloaded to a motion controller, and the users can save the time it takes to select the POUs which need to be protected by a PEP password. If the **Default Mode** checkbox in the **Data Transfer** window is unselected, the program in the project created will be stored in a motion controller according to the PEP setting in the project.

Users have to note that the PEP setting in a project will remain unchanged even if the **Default Mode** checkbox in the **Data Transfer** window is selected. Please refer to chapter 10 for more information about setting a PEP password.



Data Transfer

Transfer Options

☑ Program

☑ CAM Chart

□ Parameter

□ Copy SRAM to Flash

□ Password

(4-8) Characters

Confirmation

☑ Apply PEP Setting

☑ Default Mode

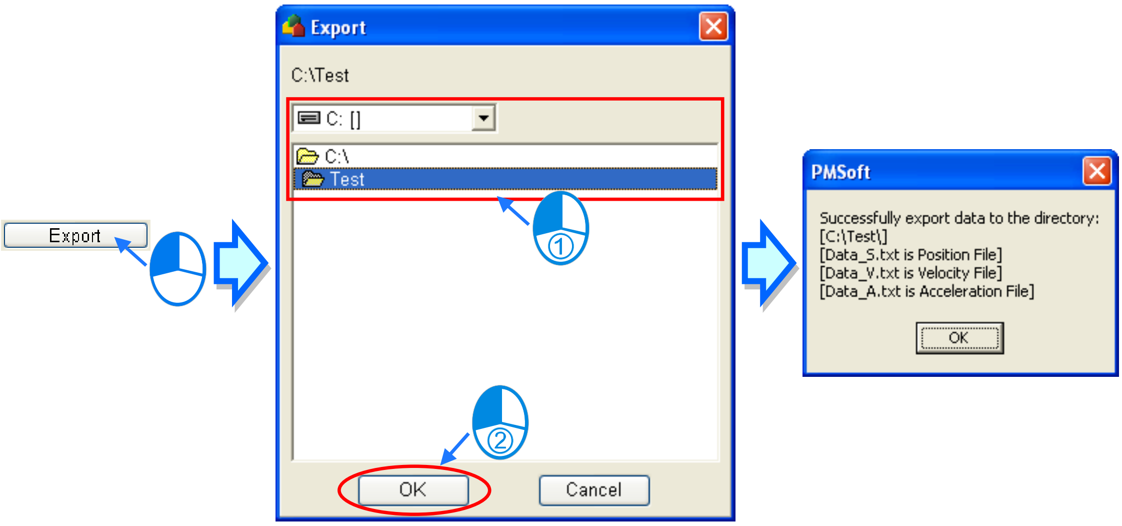
(4-8) Characters

Confirmation

Figure 13-41 Default mode

13.5.4 Exporting a Cam Chart to a Path / Importing a Cam Chart from a Path

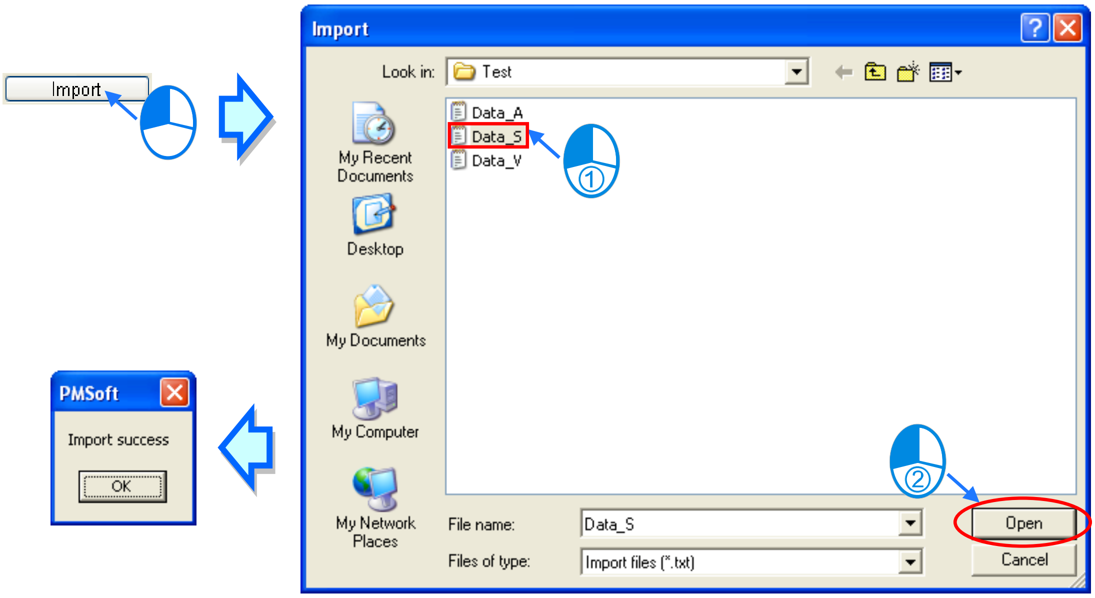
Users can manage cam charts conveniently in that they can export cam charts to specific paths or import cam charts from specific paths. If the users want to export a cam chart, they have to click **Export** on the cam chart, select a path in the **Export** window, and click **OK**. After **OK** in the **Export** window is clicked, a message will appear in the **PMSoft** window, and the cam chart will be exported to the path specified, as shown in figure 13-42.



Successfully export data to the directory:
[C:\Test\]
[Data_S.txt is Position File]
[Data_V.txt is Velocity File]
[Data_A.txt is Acceleration File]

Figure 13-42 Exporting a cam chart

If the users want to import a cam chart, they have to click **Import**, select the file corresponding to the cam chart in the **Import** window, and click **Open**. After **Open** is clicked, a message will appear, and the file will be imported, as shown in figure 13-43. Likewise, if users want to import speed data, they have to click **Import Speed Data**, select the file corresponding to the speed data in the **Import Speed Data** window, and click **Open**.



File name: Data_S

Files of type: Import files (*.txt)

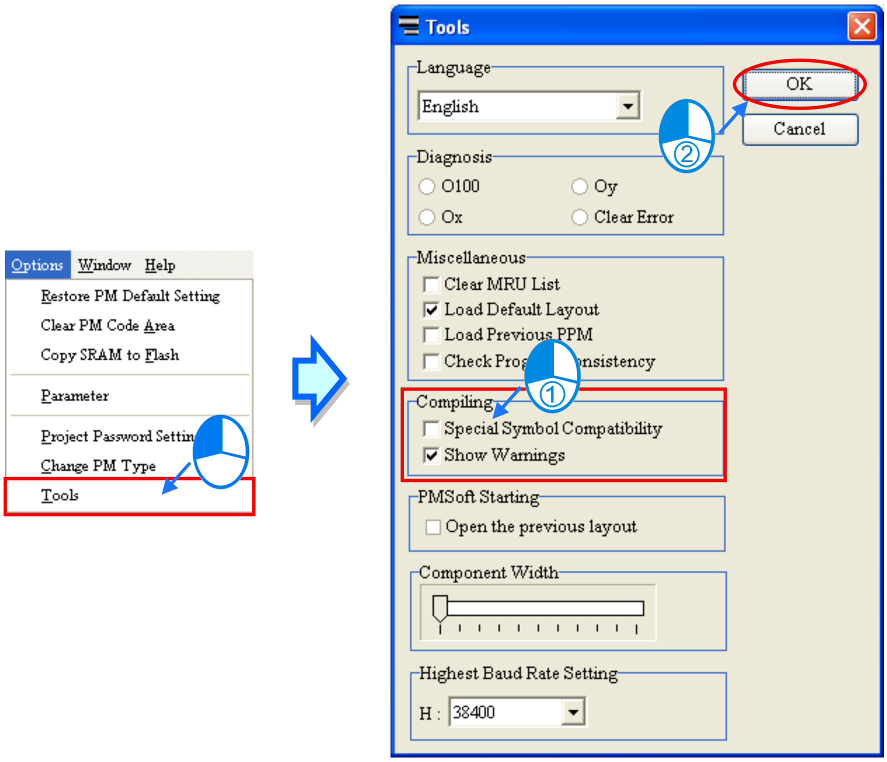
Import success

Figure 13-43 Importing a cam chart

13.5.5 Compatibility of Special Marks and Showing Warning Messages

Some special marks can not be used as symbols. If users want to use a special mark which is not allowed as a symbol, they have to click **Tools** on the **Options** menu, select the **Special Symbol Compatibility** checkbox in the **Tools** window, and click **OK**. Please see figure 13-44. Considering the compatibility of programs, it is suggested that users should not use special marks as symbols. The **Special Symbol Compatibility** checkbox in the **Tools** window are unselected by default.

If the **Show Warnings** checkbox in the **Tools** window is selected, and the situation described in section 13.4.5 arises (a function block instance is used repeatedly in the program created), a warning message will appear after the program created is compiled.



Language: English

Diagnosis: ○ O100 ○ Oy ○ Ox ○ Clear Error

Miscellaneous:
- □ Clear MRU List
- ☑ Load Default Layout
- □ Load Previous PPM
- □ Check Prog nsistency

Compiling:
- □ Special Symbol Compatibility
- ☑ Show Warnings

PMSoft Starting:
- □ Open the previous layout

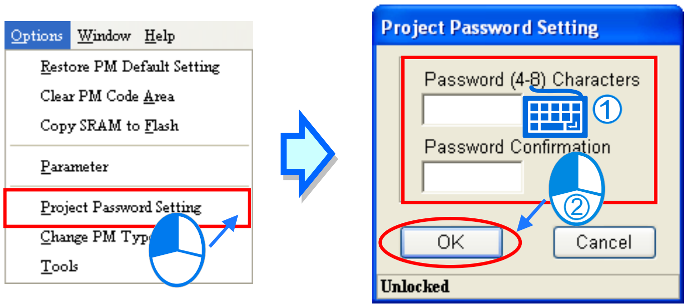
Component Width

Highest Baud Rate Setting: H : 38400

Figure 13-44 Compatibility of special marks and showing warning messages

13.5.6 Project Password

A project password is used to protect the program in a project. If users want to open a POU which is a program in a project which is protected by a project password, the system will ask the users to type the project password. If the users want to set a project password, they have to click **Project Password Setting** on the **Options** menu, type the project password in the **Project Password Setting** window, and click OK. Please see figure 13-45.



Password (4-8) Characters

Password Confirmation

Unlocked

Figure 13-45 Project password

After a project which is protected by a project password is closed, the **Project Password Setting** window will appear, and the users will be asked to type the project password if the users want to open a POU in the project. Please see figure 13-46.

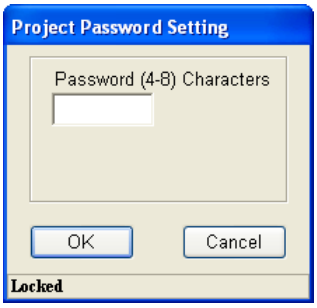


Figure 13-46 Project's being protected by a project password

If the users want to remove a project password, they have to click **Project Password Setting** on the **Options** menu, type the project password in the **Project Password Setting** window, and click **OK**. Please see figure 13-47.

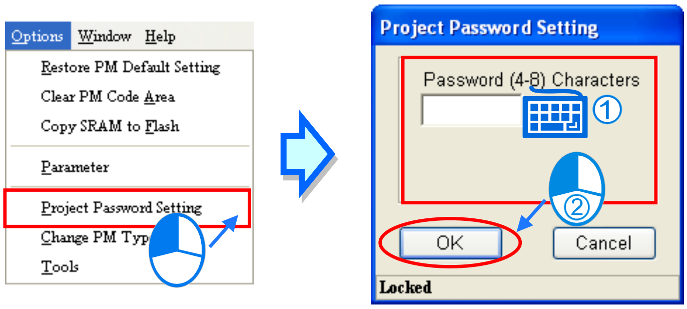


Figure 13-47 Removing a project password

If the users want to upload the program in a motion controller, or data from a motion controller, they have to click **Upload Program**. If the users click **OK** after they select the **Synchronize Project and PM Password** checkbox in the **Data Transfer** window, the PM password which protects the motion controller connected to PMSoft will be used as the project password which protects the project created.

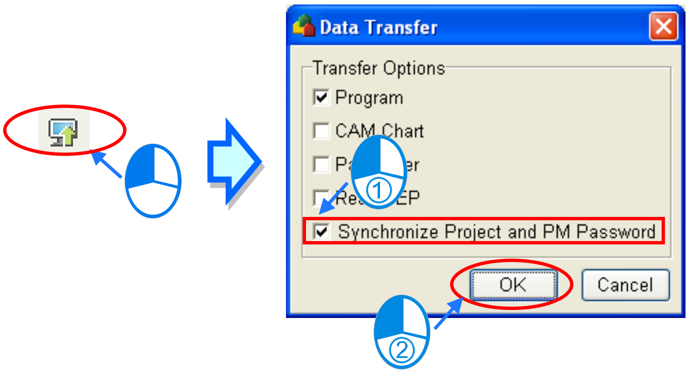
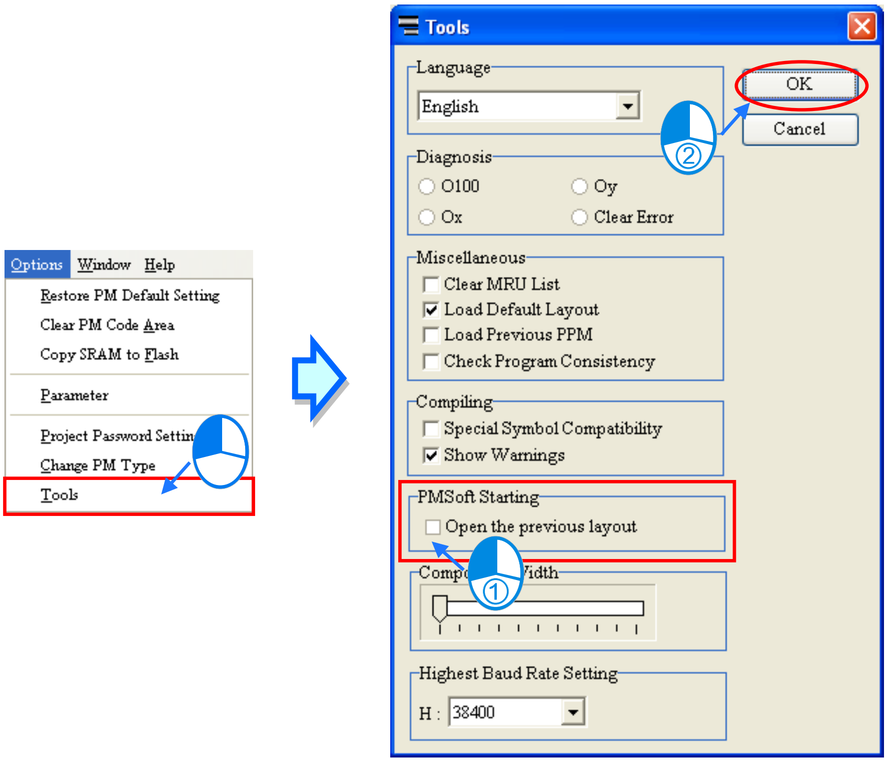


Figure 13-48 Using a PM password as a project password

13.5.7 Loading the Layout of an Old Project

If users click **Tools** on the **Options** menu, select the **Open the previous layout** checkbox in the **Tools** window, and click **OK**, the layout of an old project will be restored after the old project is opened. Please see figure 13-49. If the **Open the previous layout** checkbox in the **Tools** window is not selected, the default layout of an old project will appear after the old project is opened.



Tools

Language: English

Diagnosis
○ O100 ○ Oy
○ Ox ○ Clear Error

Miscellaneous
□ Clear MRU List
☑ Load Default Layout
□ Load Previous PPM
□ Check Program Consistency

Compiling
□ Special Symbol Compatibility
☑ Show Warnings

PMSoft Starting
□ Open the previous layout

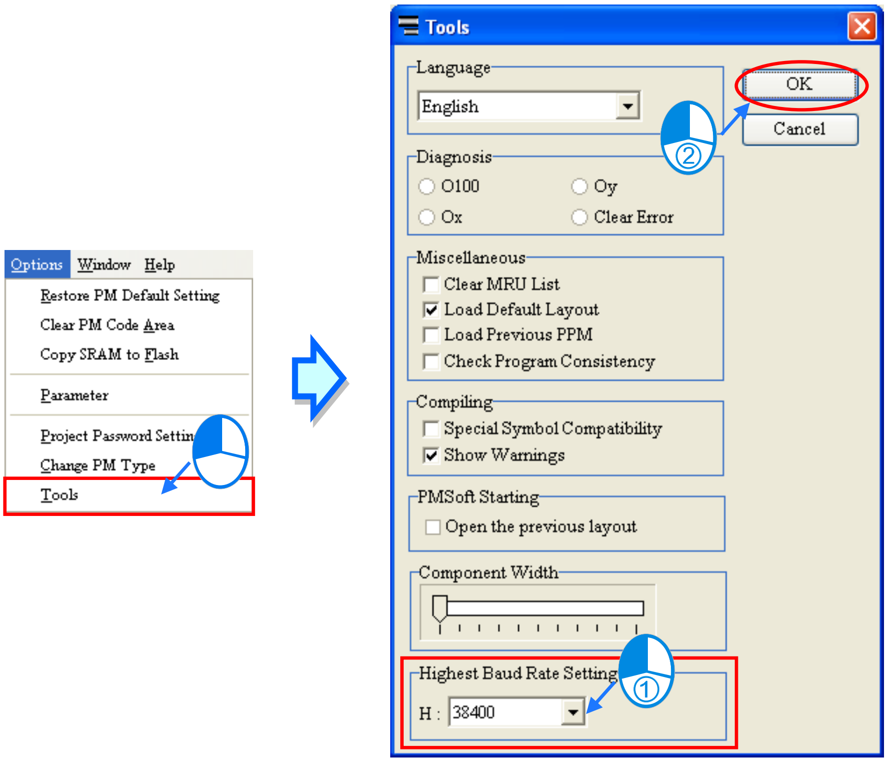
Comp... Width

Highest Baud Rate Setting
H : 38400

Figure 13-49 Loading the layout of an old project

13.5.8 Setting a Maximum Baud Rate

Users can set a maximum baud rate according to an environment and the lengths of cables. After the users click **Tools** on the **Options** menu, select a baud rate in the **Highest Baud Rate Setting** section in the **Tools** window, and click **OK**, the setting of a maximum baud rate will be complete. Please see figure 13-50.



Language: English

Diagnosis: ○ O100 ○ Oy ○ Ox ○ Clear Error

Miscellaneous:
- □ Clear MRU List
- ☑ Load Default Layout
- □ Load Previous PPM
- □ Check Program Consistency

Compiling:
- □ Special Symbol Compatibility
- ☑ Show Warnings

PMSoft Starting:
- □ Open the previous layout

Component Width

Highest Baud Rate Setting: H : 38400

Figure 13-50 Setting a maximum baud rate

After the users finishes setting a maximum baud rate, the maximum baud rate set will be written to the PLC connected after they click **H** on the PMSoft toolbar. Please see figure 13-51.

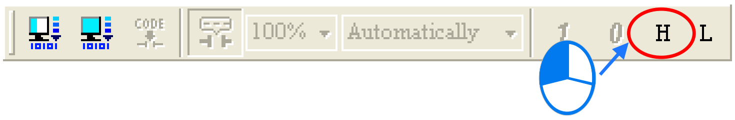


Figure 13-51 Switching to the maximum baud rate set

MEMO

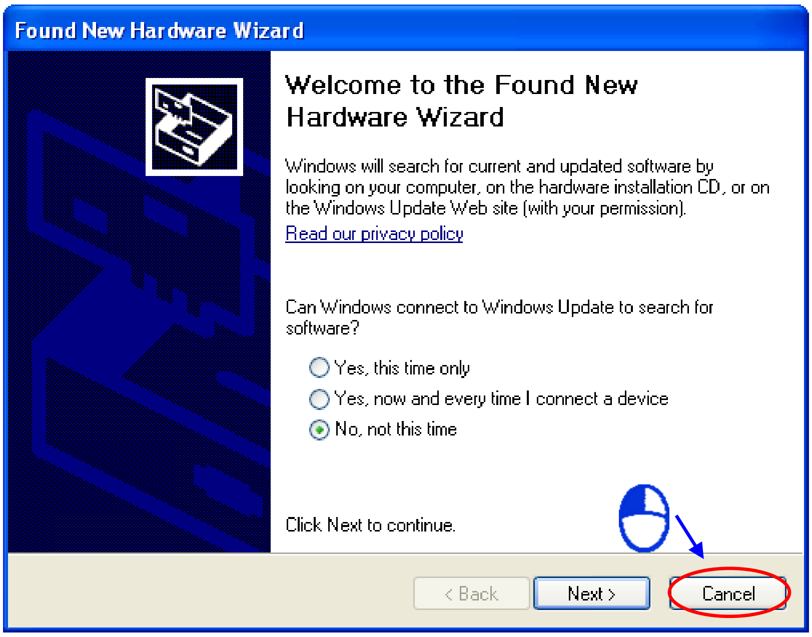
Appendix A Installing a USB Driver

Table of Contents

A.1 Installing the USB Driver for a Motion Controller

The installation of the USB driver for a motion controller on Windows XP is introduced below. If users want to install the USB driver for a motion controller on another operating system, they have to refer to the instructions in the operating system for more information about the installation of new hardware.

(1) The users have to make sure that the motion controller is supplied with power normally. They have to connect the motion controller to a USB port on the computer with a USB cable, and click **Cancel** in the **Found New Hardware Wizard** window.



Welcome to the Found New Hardware Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

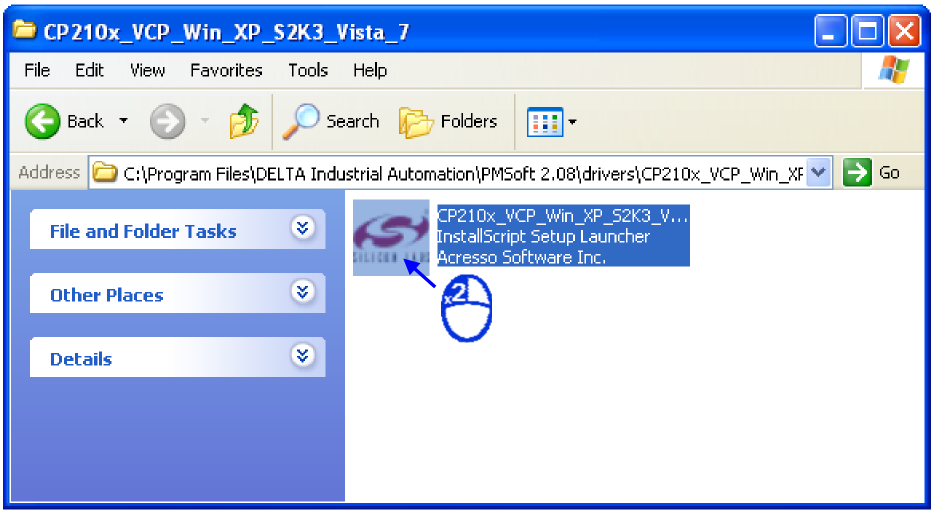
Read our privacy policy

Can Windows connect to Windows Update to search for software?

○ Yes, this time only
○ Yes, now and every time I connect a device
● No, not this time

Click Next to continue.

(2) The users have to double-click the USB driver in the **PMSoft x.xx** folder. The default path which denotes the folder in which the USB driver is saved is C:\Program Files\Delta Industrial Automation\PMSoft x.xx\drivers\CP210x_VCP_Win_XP_S2K3_Vista_7. x.xx is the version of PMSoft.

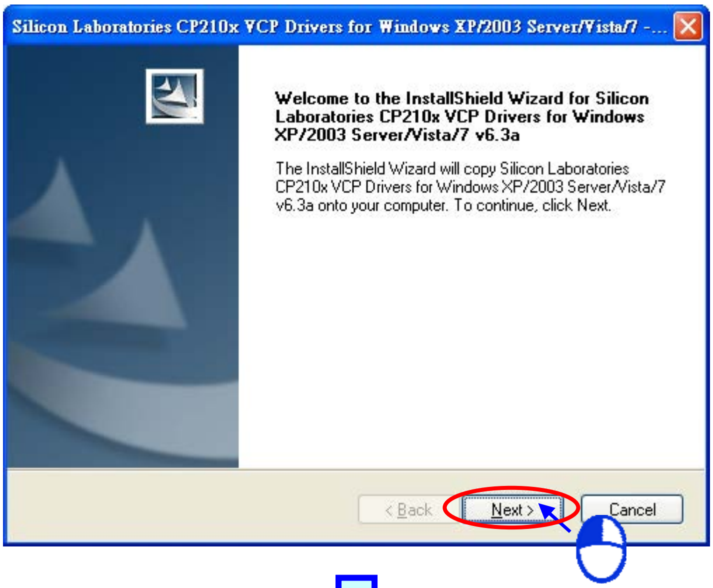


Address: C:\Program Files\DELTA Industrial Automation\PMSoft 2.08\drivers\CP210x_VCP_Win_XP

CP210x_VCP_Win_XP_S2K3_V...
InstallScript Setup Launcher
Acresso Software Inc.

(3) The installation of the USB driver is the same as the installation of a general program. After the

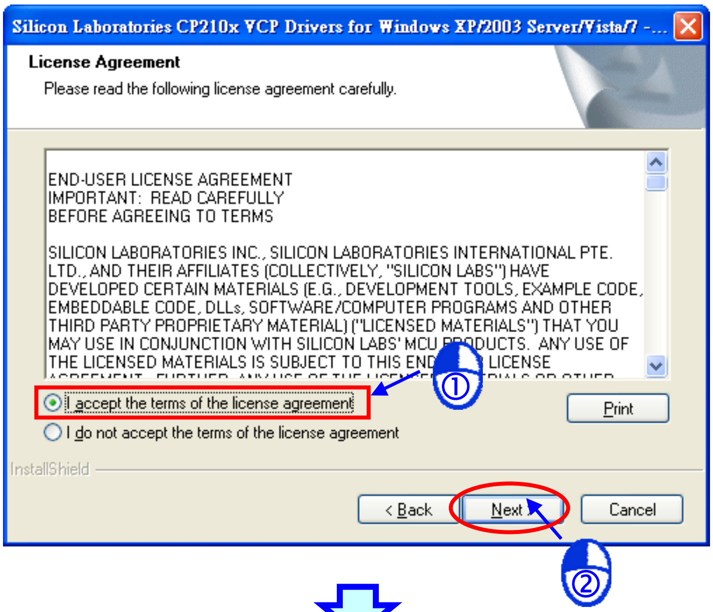
users accept the terms of the license agreement, they have to click **Next**. The users can select an installation path in the process of installing the USB driver. Finally, the users have to restart the computer to finish installing the USB driver.



Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7 -...

Welcome to the InstallShield Wizard for Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7 v6.3a

The InstallShield Wizard will copy Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7 v6.3a onto your computer. To continue, click Next.



Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7 -...

License Agreement

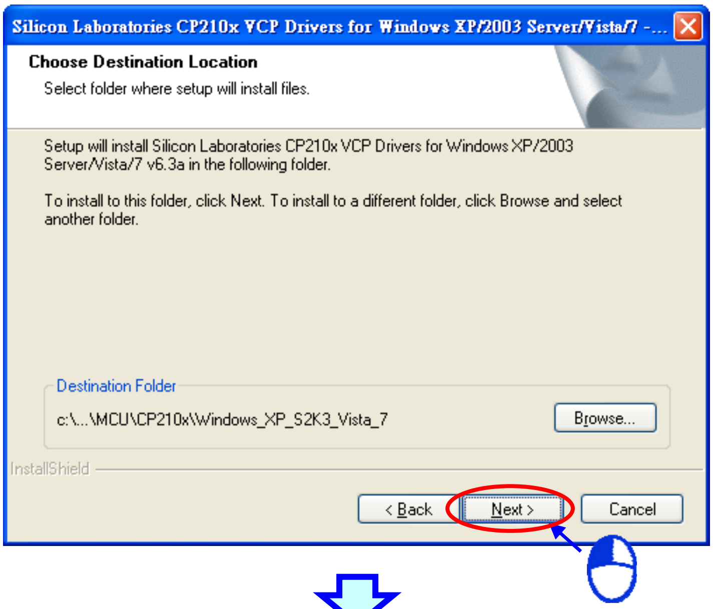
Please read the following license agreement carefully.

END-USER LICENSE AGREEMENT
IMPORTANT: READ CAREFULLY
BEFORE AGREEING TO TERMS

SILICON LABORATORIES INC., SILICON LABORATORIES INTERNATIONAL PTE. LTD., AND THEIR AFFILIATES (COLLECTIVELY, "SILICON LABS") HAVE DEVELOPED CERTAIN MATERIALS (E.G., DEVELOPMENT TOOLS, EXAMPLE CODE, EMBEDDABLE CODE, DLLs, SOFTWARE/COMPUTER PROGRAMS AND OTHER THIRD PARTY PROPRIETARY MATERIAL) ("LICENSED MATERIALS") THAT YOU MAY USE IN CONJUNCTION WITH SILICON LABS' MCU PRODUCTS. ANY USE OF THE LICENSED MATERIALS IS SUBJECT TO THIS END-USER LICENSE

● I accept the terms of the license agreement

○ I do not accept the terms of the license agreement



Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7 -...

Choose Destination Location

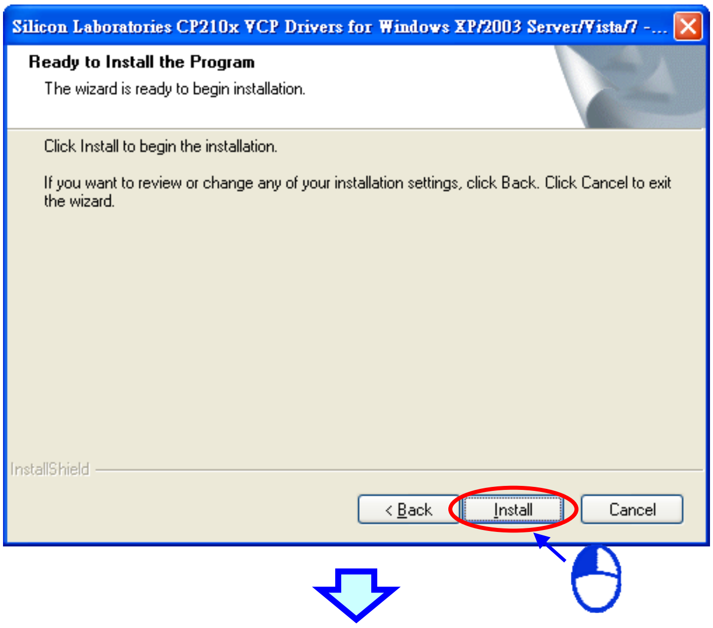
Select folder where setup will install files.

Setup will install Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7 v6.3a in the following folder.

To install to this folder, click Next. To install to a different folder, click Browse and select another folder.

Destination Folder

c:\...\MCU\CP210x\Windows_XP_S2K3_Vista_7



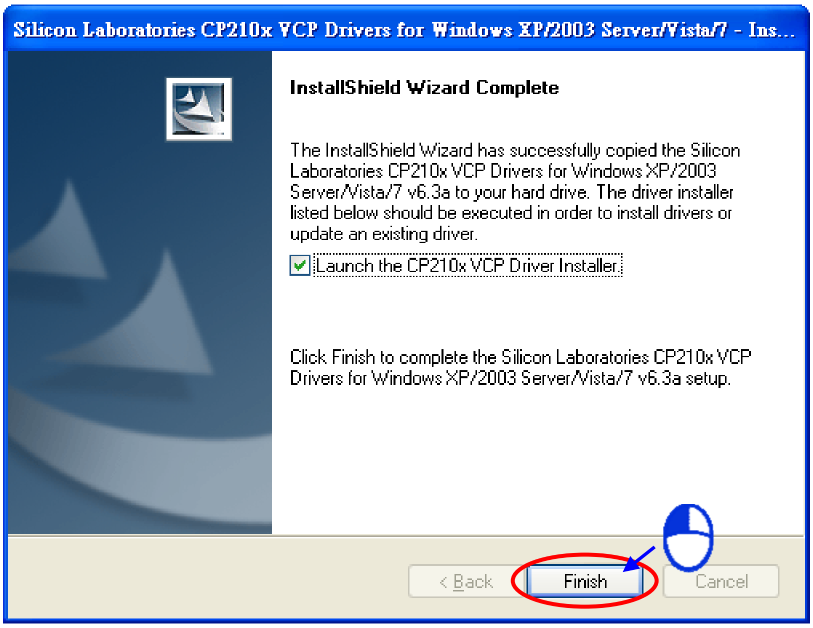
Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7 -...

Ready to Install the Program

The wizard is ready to begin installation.

Click Install to begin the installation.

If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.



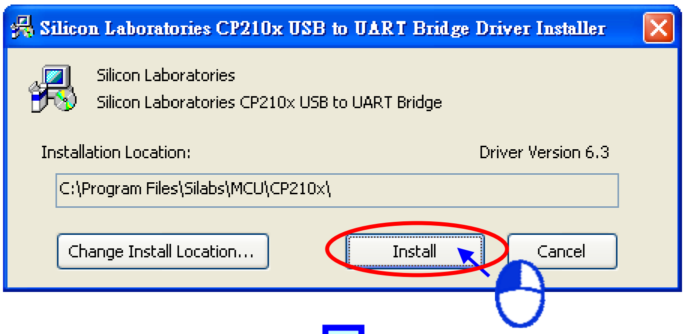
Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7 - Ins...

InstallShield Wizard Complete

The InstallShield Wizard has successfully copied the Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7 v6.3a to your hard drive. The driver installer listed below should be executed in order to install drivers or update an existing driver.

☑ Launch the CP210x VCP Driver Installer.

Click Finish to complete the Silicon Laboratories CP210x VCP Drivers for Windows XP/2003 Server/Vista/7 v6.3a setup.

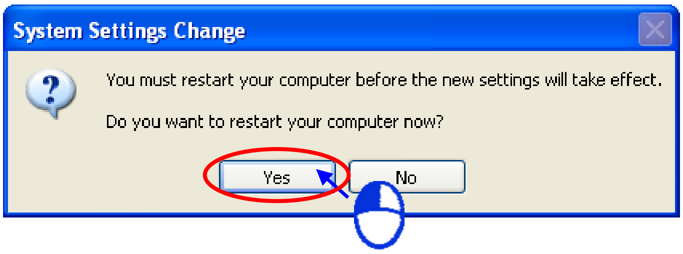


Silicon Laboratories CP210x USB to UART Bridge Driver Installer

Silicon Laboratories
Silicon Laboratories CP210x USB to UART Bridge

Installation Location: Driver Version 6.3

C:\Program Files\Silabs\MCU\CP210x\

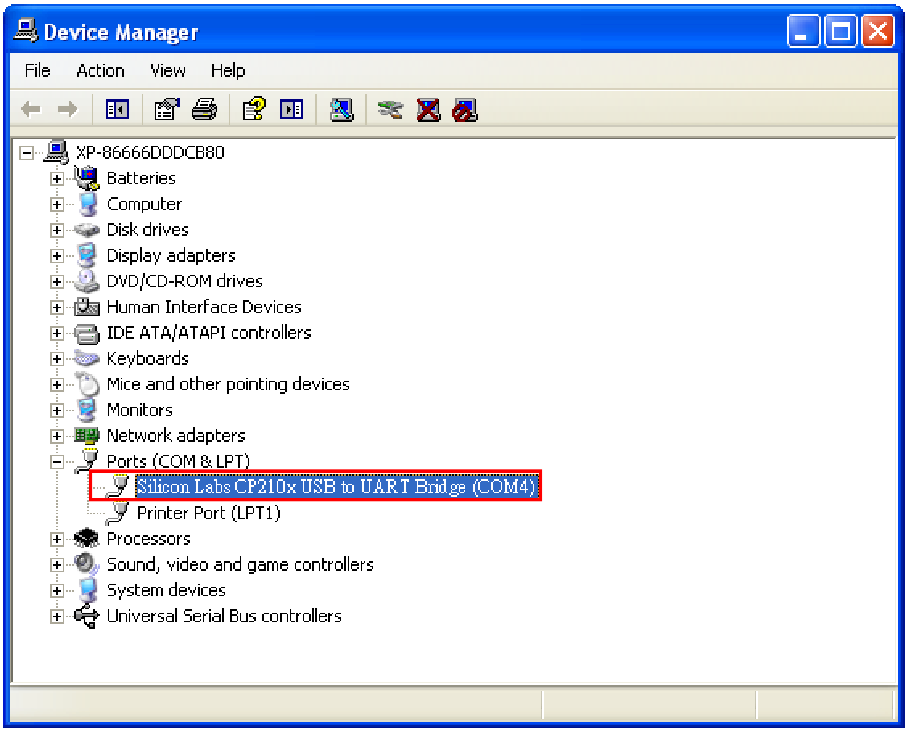


System Settings Change

You must restart your computer before the new settings will take effect.

Do you want to restart your computer now?

(4) After the users finish installing the USB driver, they have to open the **Device Manager** window. If there is device called **Silicon Labs CP210xUSB to UATY Bridge** in the **Ports (COM & LPT)** section, the USB driver is installed successfully, and the system assigns a communication port number to the device.



Device Manager

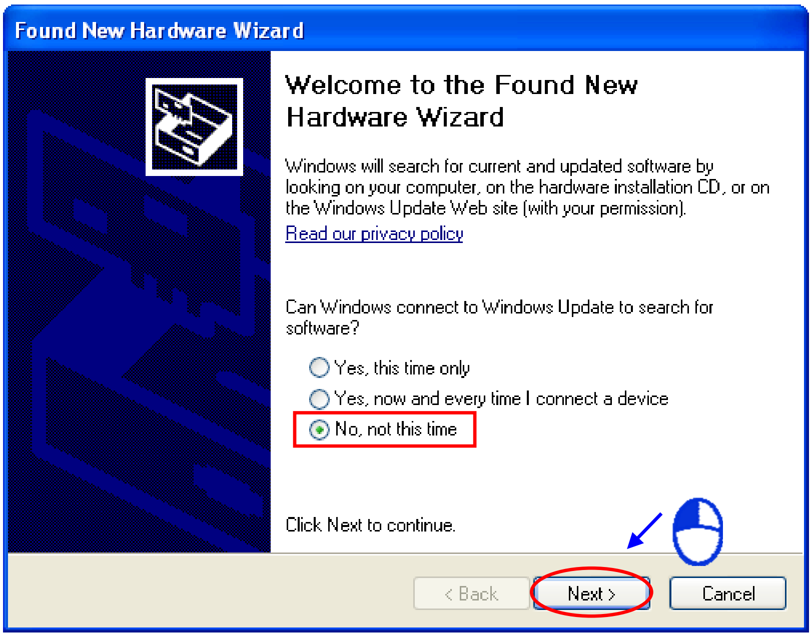
File Action View Help

XP-86666DDDCB80
- Batteries
- Computer
- Disk drives
- Display adapters
- DVD/CD-ROM drives
- Human Interface Devices
- IDE ATA/ATAPI controllers
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Ports (COM & LPT)
  - Silicon Labs CP210x USB to UART Bridge (COM4)
  - Printer Port (LPT1)
- Processors
- Sound, video and game controllers
- System devices
- Universal Serial Bus controllers

A.2 Installing the USB Driver for a PLC

The installation of the USB driver for a PLC on Windows XP is introduced below. If users want to install the USB driver for a PLC on another operating system, they have to refer to the instructions in the operating system for more information about the installation of new hardware.

(1) The users have to make sure that the PLC is supplied with power normally. They have to connect the PLC to a USB port on the computer with a USB cable. The users have to select the **No, not this time** option button in the **Found New Hardware Wizard** window, and then click **Next**.



Found New Hardware Wizard

Welcome to the Found New Hardware Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

Read our privacy policy

Can Windows connect to Windows Update to search for software?

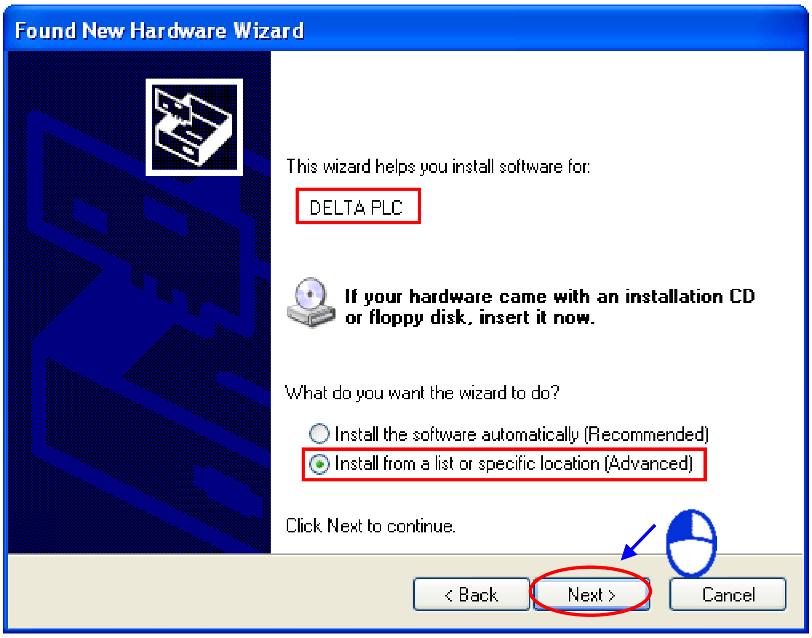
○ Yes, this time only

○ Yes, now and every time I connect a device

● No, not this time

Click Next to continue.

(2) The name of the USB device detected is displayed in the window. The device name shown in the figure below is the name of an AH500 series CPU module. Different models have different names. Please select the **Install from a lost or specific location (Advanced)** option button.



Found New Hardware Wizard

This wizard helps you install software for:

DELTA PLC

If your hardware came with an installation CD or floppy disk, insert it now.

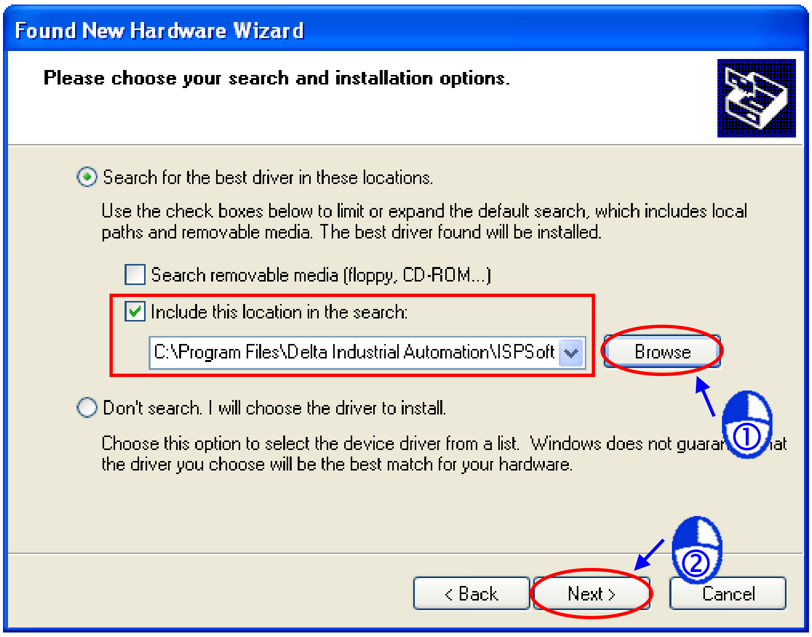
What do you want the wizard to do?

○ Install the software automatically (Recommended)

● Install from a list or specific location (Advanced)

Click Next to continue.

(3) After ISPSoft version 2.00 or above is installed, the USB driver for an AH500 series PLC will be installed in the folder denoted by the path C:\Program Files\Delta Industrial Automation\ISPSoftx.xx \drivers\Delta_PLC_USB_Driver. If the users get the driver in another way, they have to specify the path denoting the driver. After the users specify the path denoting the driver, they have to click **Next**.



Found New Hardware Wizard

Please choose your search and installation options.

● Search for the best driver in these locations.

Use the check boxes below to limit or expand the default search, which includes local paths and removable media. The best driver found will be installed.

□ Search removable media (floppy, CD-ROM...)

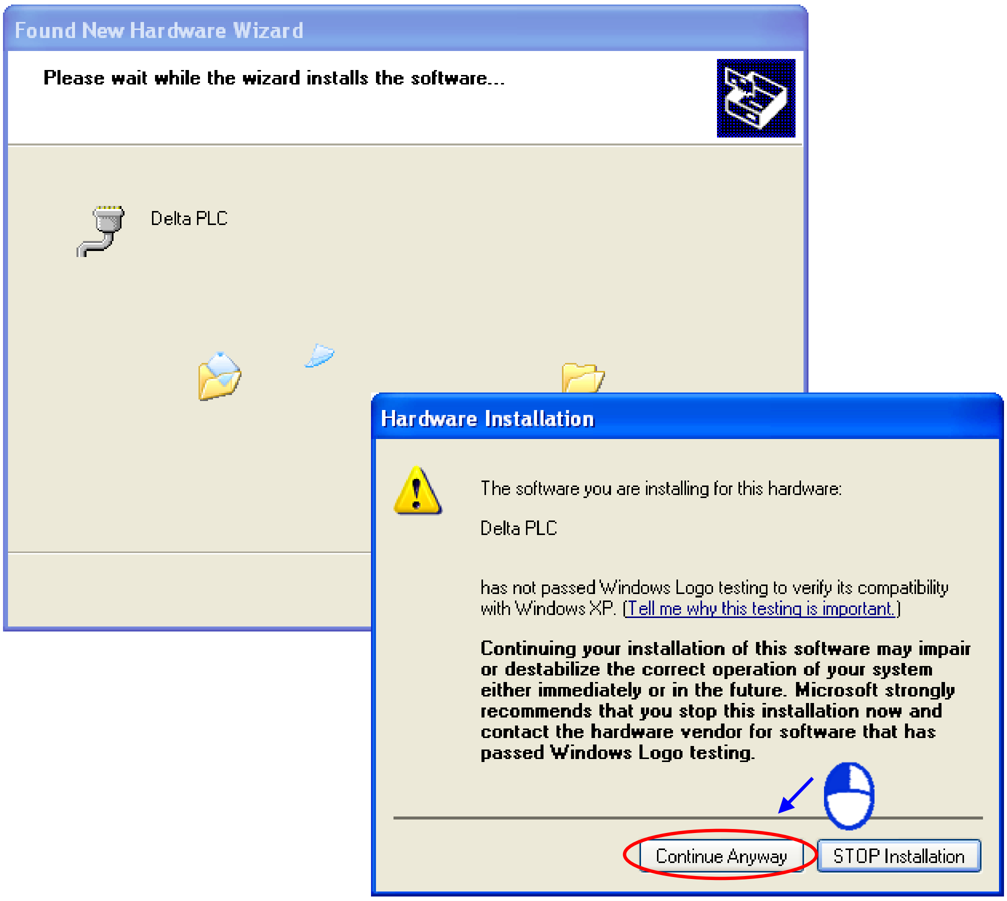
☑ Include this location in the search:

C:\Program Files\Delta Industrial Automation\ISPSoft

○ Don't search. I will choose the driver to install.

Choose this option to select the device driver from a list. Windows does not guaran[…]hat the driver you choose will be the best match for your hardware.

(4) After the driver denoted by the path which is specified by the users is found, the system will install the driver. If the **Hardware Installation** window appears during the installation, the users have to click **Continue Anyway**.



Found New Hardware Wizard

Please wait while the wizard installs the software...

Delta PLC

Hardware Installation

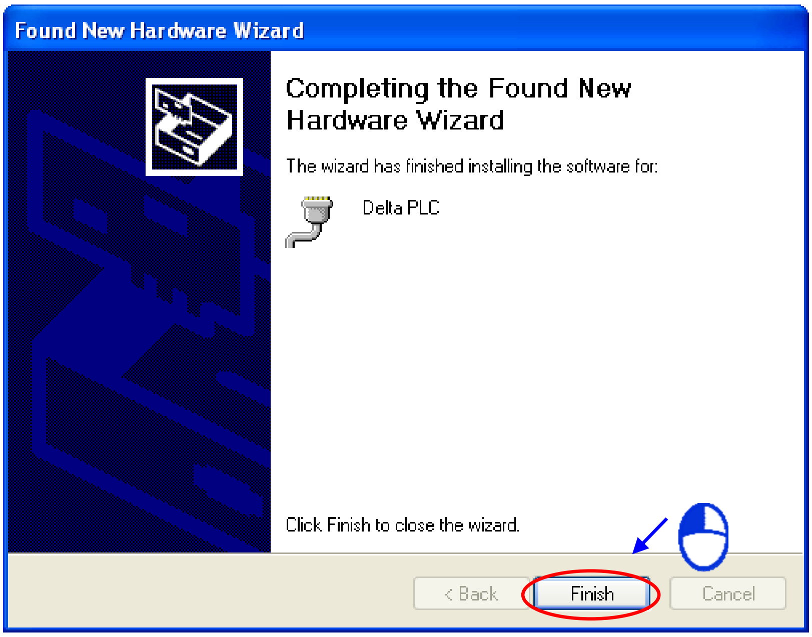
The software you are installing for this hardware:

Delta PLC

has not passed Windows Logo testing to verify its compatibility with Windows XP. (Tell me why this testing is important.)

Continuing your installation of this software may impair or destabilize the correct operation of your system either immediately or in the future. Microsoft strongly recommends that you stop this installation now and contact the hardware vendor for software that has passed Windows Logo testing.

(5) After the users finish the installation, they have to click **Finish**.



Found New Hardware Wizard

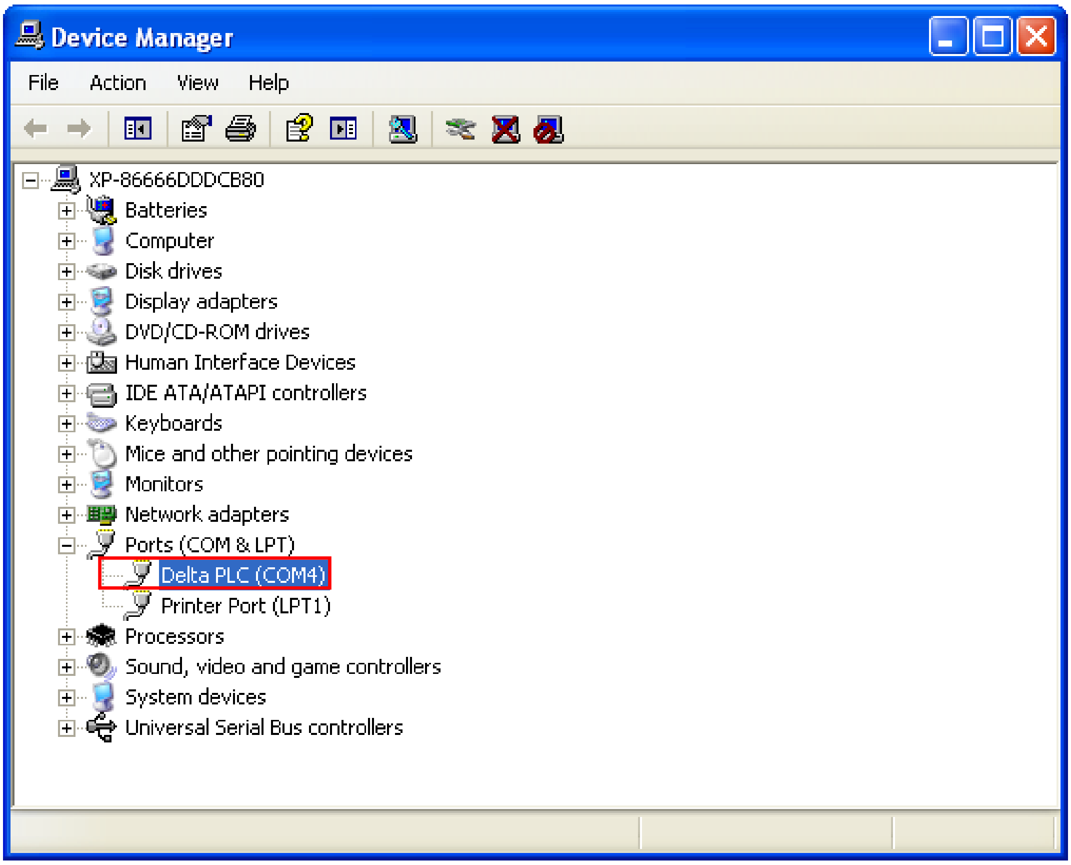
Completing the Found New Hardware Wizard

The wizard has finished installing the software for:

Delta PLC

Click Finish to close the wizard.

(6) After the users finish the installation, they have to open the **Device Manager** window. If the name of the USB device connected is in the **Ports (COM&LPT)** section, the driver is installed successfully, and the system assigns a communication port device to the USB device.



XP-86666DDDCB80
- Batteries
- Computer
- Disk drives
- Display adapters
- DVD/CD-ROM drives
- Human Interface Devices
- IDE ATA/ATAPI controllers
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Ports (COM & LPT)
    - Delta PLC (COM4)
    - Printer Port (LPT1)
- Processors
- Sound, video and game controllers
- System devices
- Universal Serial Bus controllers

***. The device name shown in the figure above is the name of an AH500 series CPU module. Different models have different names.**

Additional remark

- If the PLC is connected to another USB port on the computer, the system may ask the users to install the driver again. The users can follow the steps above, and install the driver again. After the driver is installed, the communication port number that the system assigns may be different.
- If Windows XP SP3 has not been installed on the computer, an error message will appear during the installation. The users can deal with the problem in either way below.
    a. Cancel the installation, install Windows XP SP3, and reinstall the driver according to the steps above.
    b. Get the file needed, and specify the path pointing to the file in the **Files Needed** window.

MEMO

Appendix B Delta-defined Parameter Table

Table of Contents

B.1 Delta-defined Parameter Table

Delta-defined parameters are for input pins in Delta motion control function blocks. Users can directly use Delta-defined parameters to operate motion control function blocks without having to know the descriptions of the input pins in the motion control function blocks. Delta-defined parameters are described below.

Table B-1 Delta-defined parameter table

Name	Type	Value	Motion control function block	Description
TRUE	BOOL	True	All motion control function blocks	Input pin
FALSE	BOOL	False		Input pin
mcRising	BOOL	True	T_TrSeg2, T_TrSeg1, T_HomeReturn	Transition in DOG's signal from low to high
mcFalling	BOOL	False		Transition in DOG's signal from high to low
mcPositive	BOOL	True	T_HomeReturn	Returning home in the positive direction
mcNegative	BOOL	False		Returning home in the negative direction
mcSCurve	BOOL	True	T_AxisSetting2	Speed curve: S curve
mcTrapezoid	BOOL	False		Speed curve: Trapezoid curve
mcNC	BOOL	True	T_InputPolatiry	Normally-closed contact
mcNO	BOOL	False		Normally-open contact
mc32bits	BOOL	True	T_DMCServoWrite	32-bit value
mc16bits	BOOL	False		16-bit value
mcUp_Up	BOOL	True	T_HTmr	A high-speed timer becomes active when its signal goes from low to high.
mcUp_Down	BOOL	False		A high-speed timer becomes active when its signal goes from high to low.
mcCmpSet	BOOL	True	T_Compare	An output is set when the condition of a comparison is met.
mcCmpRst	BOOL	False		An output is reset when the condition of a comparison is met.
mcMotor	WORD	0	T_AxisSetting2	Motor unit
mcMachine	WORD	1		Mechanical unit,
mcComp	WORD	2		Compound unit
mcUD	WORD	0	T_AxisSetting2, T_HCnt	Counting up/down
mcPD	WORD	1		Pulses+Directions
mcAB	WORD	2		A/B-phase pulses
mc4AB	WORD	3		Four times the frequency of A/B-phase pulses
mcSD_M	WORD	0	T_SDDevRead	Using M devices
mcSD_D	WORD	5		Using D devices
mcSD_W	WORD	6		Using W devices
IntTimer	WORD	0	T_Interrupt	An interrupt signal is triggered by a time interval.

Name	Type	Value	Motion control function block	Description
IntX8	WORD	1	T_Interrupt	The source of an interrupt signal is X0.8.
IntX9	WORD	2		The source of an interrupt signal is X0.9.
IntX10	WORD	3		The source of an interrupt signal is X0.10.
IntX11	WORD	4		The source of an interrupt signal is X0.11.
IntX12	WORD	5		The source of an interrupt signal is X0.12.
IntX13	WORD	6		The source of an interrupt signal is X0.13.
IntX14	WORD	7		The source of an interrupt signal is X0.14.
IntX15	WORD	8		The source of an interrupt signal is X0.15.
mcCmpAxis1	WORD	0	T_Compare	The source of a comparison is the present position of the first axis.
mcCmpAxis2	WORD	1		The source of a comparison is the present position of the second axis.
mcCmpAxis3	WORD	2		The source of a comparison is the present position of the third axis.
mcCmpAxis4	WORD	3		The source of a comparison is the present position of the fourth axis.
mcCmpC200	WORD	4		The source of a comparison is the value of C200.
mcCmpC204	WORD	5		The source of a comparison is the value of C204.
mcCmpC208	WORD	6		The source of a comparison is the value of C208.
mcCmpC212	WORD	7		The source of a comparison is the value of C212.
mcCmpY8	WORD	0		The device used for a comparison is Y0.8.
mcCmpY9	WORD	1		The device used for a comparison is Y0.9.
mcCmpY10	WORD	2		The device used for a comparison is Y0.10.
mcCmpY11	WORD	3		The device used for a comparison is Y0.11.
mcCmpRstC200	WORD	4		The device used for a comparison is C200.
mcCmpRstC204	WORD	5		The device used for a comparison is C204.
mcCmpRstC208	WORD	6		The device used for a comparison is C208.
mcCmpRstC212	WORD	7		The device used for a comparison is C212.

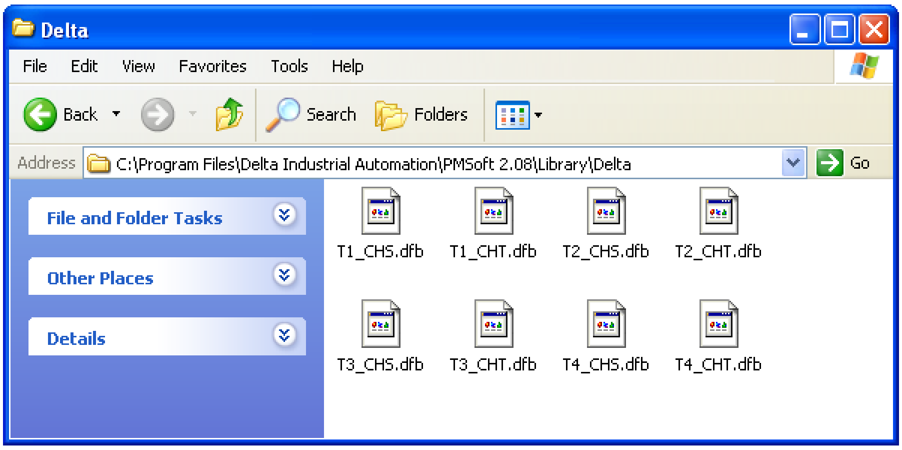
Name	Type	Value	Motion control function block	Description
mcCapAxis1	WORD	1	T_Capture	The source of capture is the present position of the first axis.
mcCapAxis2	WORD	2		The source of capture is the present position of the second axis.
mcCapAxis3	WORD	3		The source of capture is the present position of the third axis.
mcCapAxis4	WORD	4		The source of capture is the present position of the fourth axis.
mcCapC200	WORD	7		The source of capture is the value of C200.
mcCapC204	WORD	8		The source of capture is the value of C204.
mcCapC208	WORD	9		The source of capture is the value of C208.
mcCapC212	WORD	10		The source of capture is the value of C212.
mcCapX0	WORD	0		The source of a capture signal is X0.0.
mcCapX1	WORD	1		The source of a capture signal is X0.1.
mcCapX2	WORD	2		The source of a capture signal is X0.2.
mcCapX3	WORD	3		The source of a capture signal is X0.3.
mcCapX8	WORD	8		The source of a capture signal is X0.8.
mcCapX9	WORD	9		The source of a capture signal is X0.9.
mcCapX10	WORD	10		The source of a capture signal is X0.10.
mcCapX11	WORD	11		The source of a capture signal is X0.11.
mcCapX12	WORD	12		The source of a capture signal is X0.12.
mcCapX13	WORD	13		The source of a capture signal is X0.13.
mcCapX14	WORD	14		The source of a capture signal is X0.14.
mcCapX15	WORD	15		The source of a capture signal is X0.15.

Appendix C Overwriting .dfb Files in Delta Libraries

Table of Contents

C.1 Overwriting .dfb Files in Delta Libraries

Delta libraries are saved as .dfb files in the directory where PMSoft is installed, as shown in figure C-1. The default path where the Delta libraries are saved is C:\Program Files\Delta Industrial Automation\PMSoft x.xx\Library\Delta. Users can update the Delta libaries by replacing old .dfb files with new .dfb files.

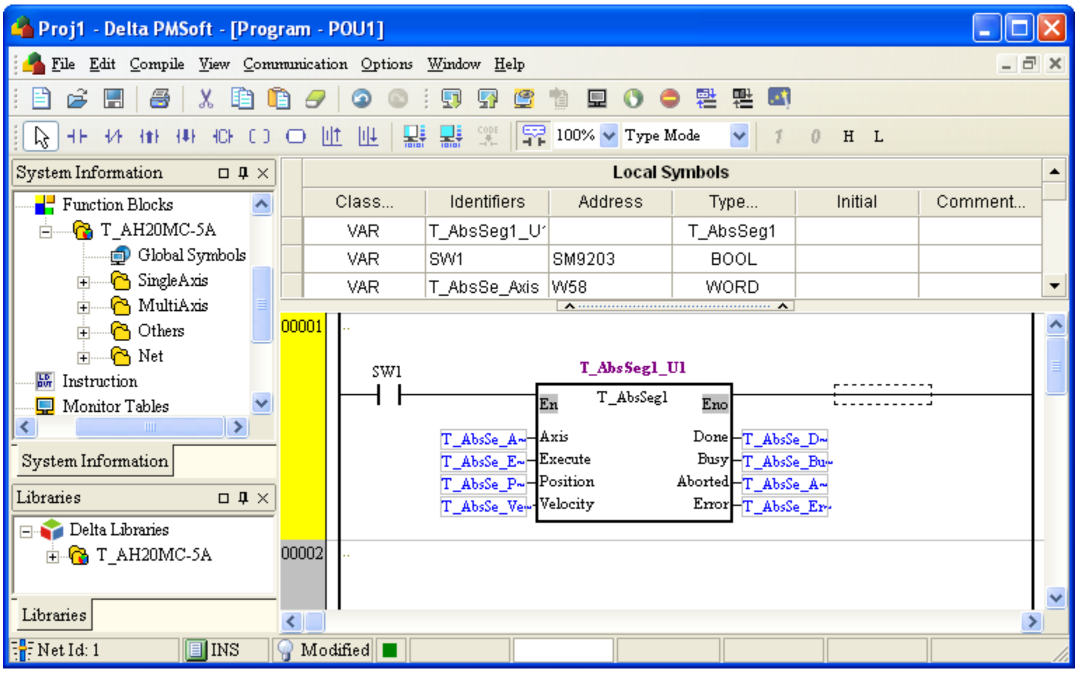


Address: C:\Program Files\Delta Industrial Automation\PMSoft 2.08\Library\Delta

Figure C-1 .dfb files

Example:

The function block T_AbsSeg1 in Delta libraries is used in an old project, as shown in figure C-2.



Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	T_AbsSeg1_U'		T_AbsSeg1		
VAR	SW1	SM9203	BOOL		
VAR	T_AbsSe_Axis	W58	WORD		

Figure C-2 Old project in which Delta libraries are used

If the Delta libraries need to be updated, users can replace old .dfb files contained inside the directory where PMSoft is installed with new .dfb files, as shown in figure C-3.

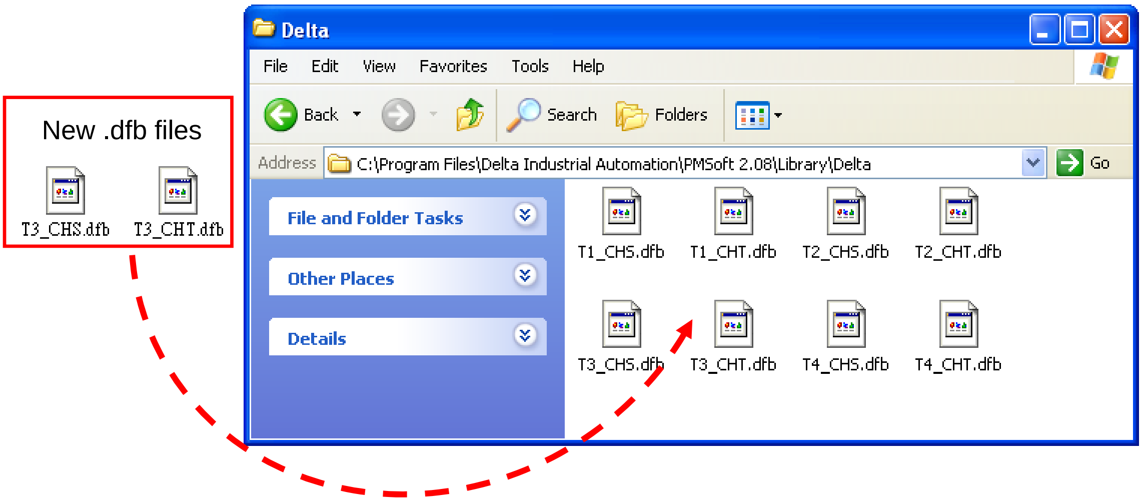
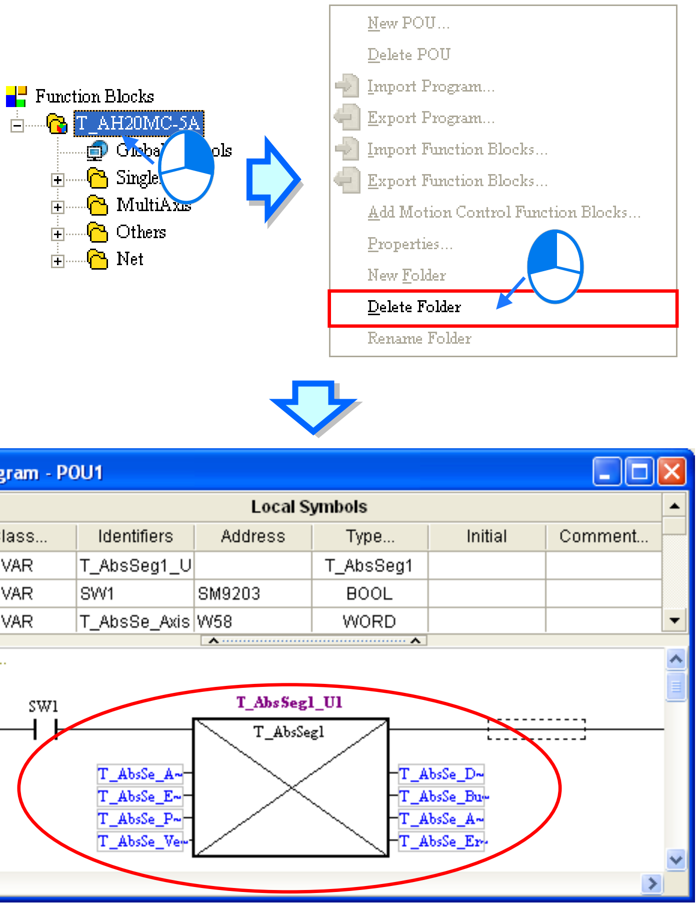


Figure C-3 Overwriting .dfb files

Although the old Delta libaries in PMSoft are updated, the function block used in the old project is still a function block in the old libraries. Therefore, the users have to delete the item which needs to be updated from the **Function Blocks** section. After the item which needs to be updated from the **Function Blocks** section is deleted, the function block will be marked with a cross, as shown in figure C-4.



Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	T_AbsSeg1_U		T_AbsSeg1		
VAR	SW1	SM9203	BOOL		
VAR	T_AbsSe_Axis	W58	WORD		

Figure C-4 Deleting an old item

The new libraries in the **Delta Llibaries** section need to be added to the **Function Blocks** section. The users can right-click the item in the **Delta Llibaries** section, and click **Add to Function Block Area** on the context menu which appears, as shown in figure C-5. Alternatively, the users can right-click Function Blocks in the system information area, and click **Add Motion Control Function Blocks...** on the context menu which appears.

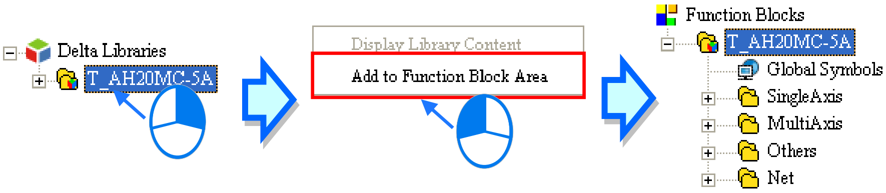
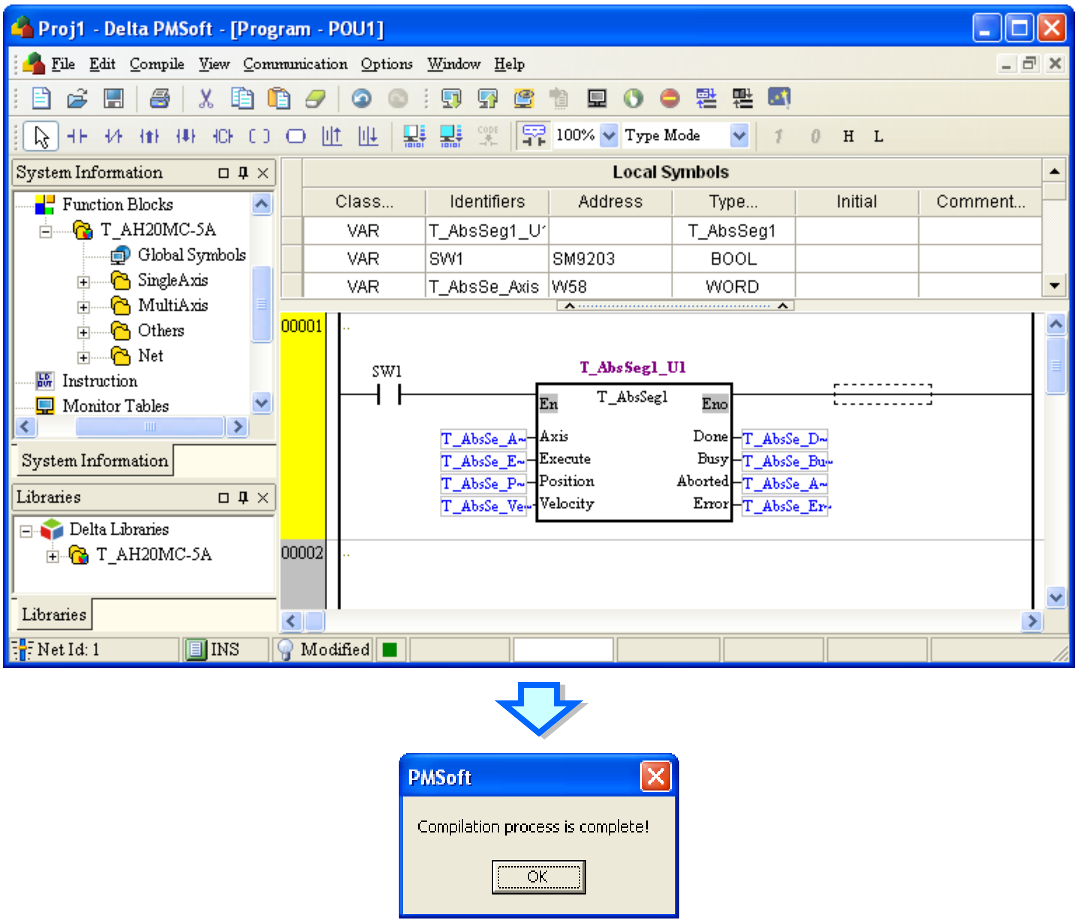


Figure C-5 Adding Delta libraries to the **Function Blocks** section

After the new libraries in the **Delta Llibaries** section are added to the **Function Blocks** section, the cross on the function block will disappear. After the users click **Compile Program** on the **Compile** menu, the function block compiled will be a function block in the nwe libraries, and the updating of the old project will be complete, as shown in figure C-6.



Local Symbols

Class...	Identifiers	Address	Type...	Initial	Comment...
VAR	T_AbsSeg1_U'		T_AbsSeg1		
VAR	SW1	SM9203	BOOL		
VAR	T_AbsSe_Axis	W58	WORD		

Compilation process is complete!

Figure C-6 Finishing updating a function block