Motors | Automation | Energy | Transmission & Distribution | Coatings

EasyBuilder Pro

Ver. 5.00.01

Installation, Configuration and Operation Guide

1. INDEX

1. EASYBUILDER PRO INSTALLATION AND STARTUP GUIDE

This chapter explains how to install EasyBuilder Pro.

1.1. INSTALLATION REQUIREMENTS

1.1.1. Software
Download EasyBuilder Pro from WEG website at www.weg.net. The latest upgraded files can be downloaded too.

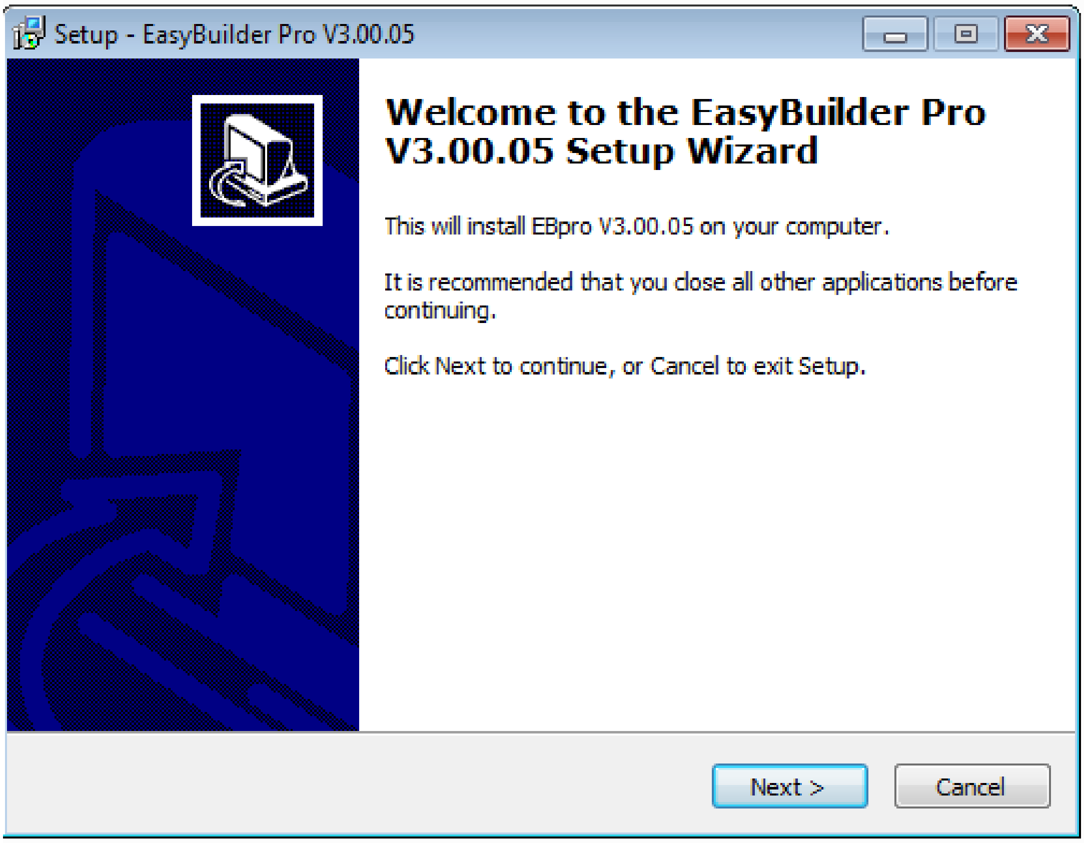
1.1.2. Hardware (Recommended)
- CPU: INTEL Pentium II or higher
- Memory: 256 MB or higher
- Hard Disk: 2.5 GB or higher (Disc space available at least 500 MB)
- CD-ROM: 4X or higher
- Display: 1,024 x 768 resolution or greater
- Keyboard and mouse
- Ethernet: for project downloading/uploading
- USB Port 2.0: for project downloading/uploading
- RS232 COM: for on-line simulation
- Printer

1.1.3. Operating System
- Windows XP
- Windows Vista
- Windows 7 (32bit / 64bit)
- Windows 8 (32bit / 64bit)
- Windows 8.1 (32bit / 64bit)

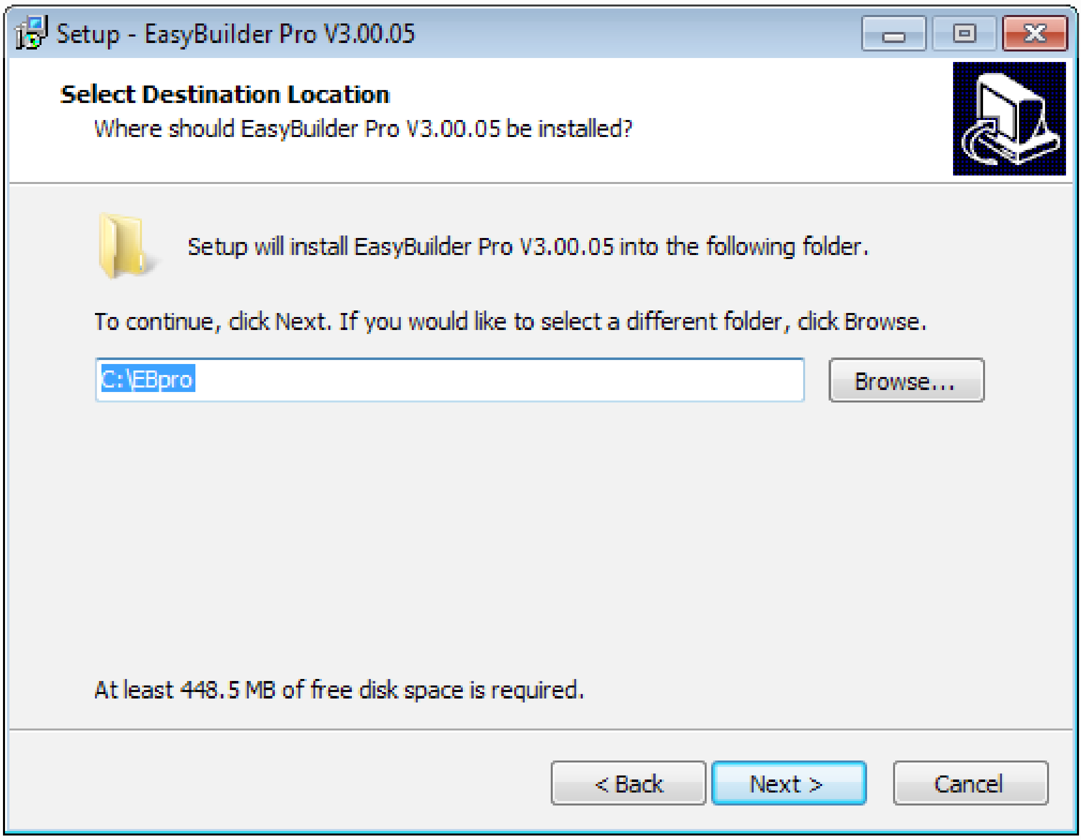
1.2. STEPS TO INSTALL EASYBUILDER PRO

1. Insert the CD-ROM into your CD-ROM drive. The computer will automatically install EasyBuilder Pro driver. Or, you can manually execute (Autorun.exe) file under the root directory. The installation screen is shown as the following figure.

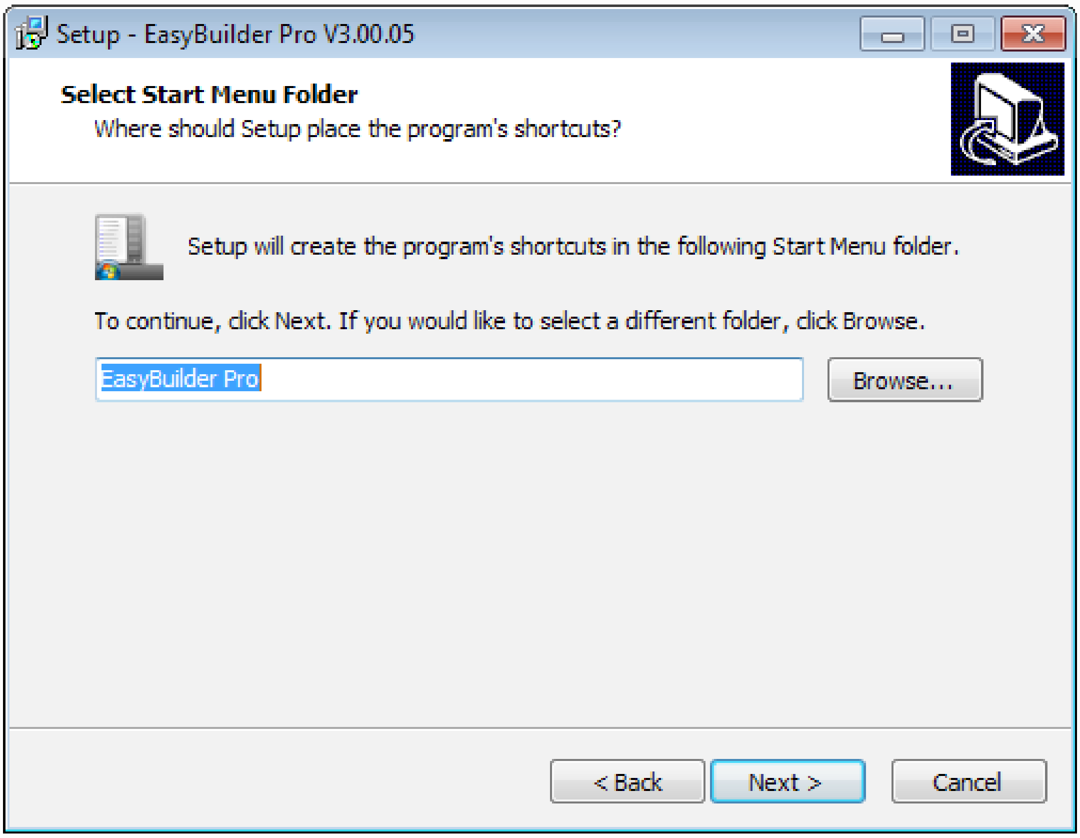
2. Click (Install) and select the language, then click (Next).



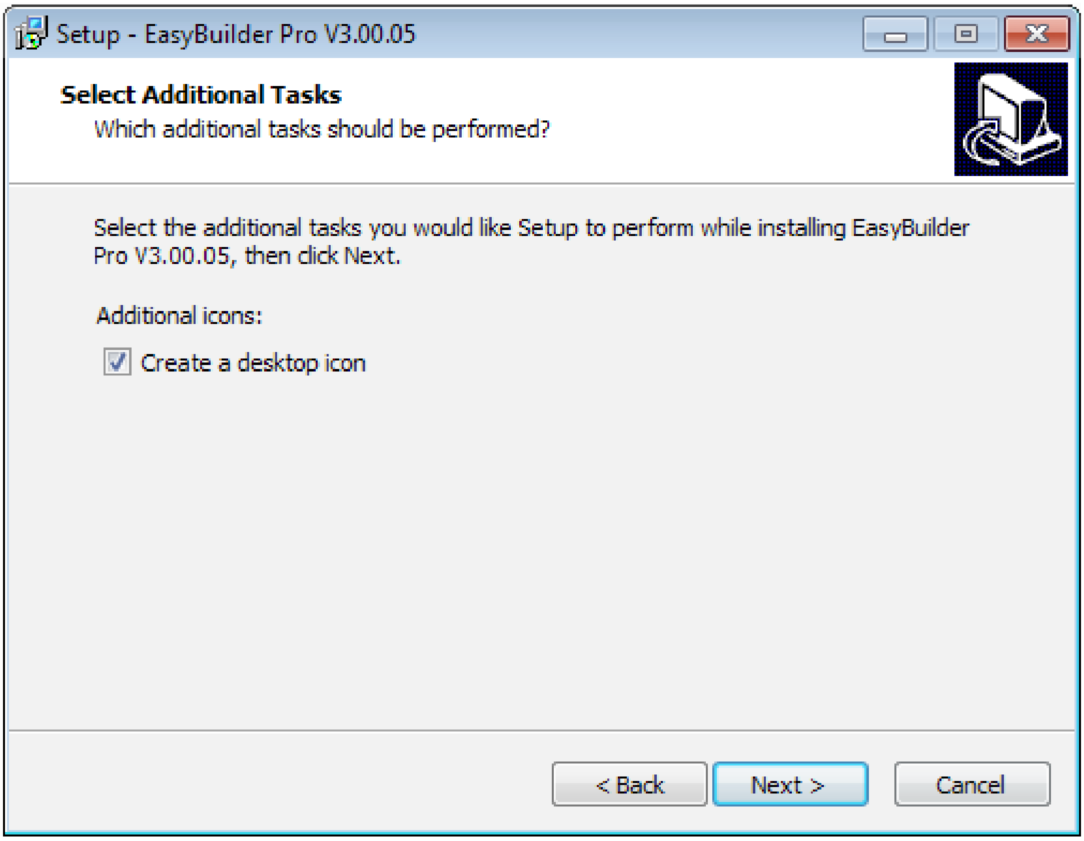
3. Remove the old version of EasyBuilder Pro if needed, and click (Next).
4. Designate a new folder for EasyBuilder Pro installation or use the recommended folder then click (Next).



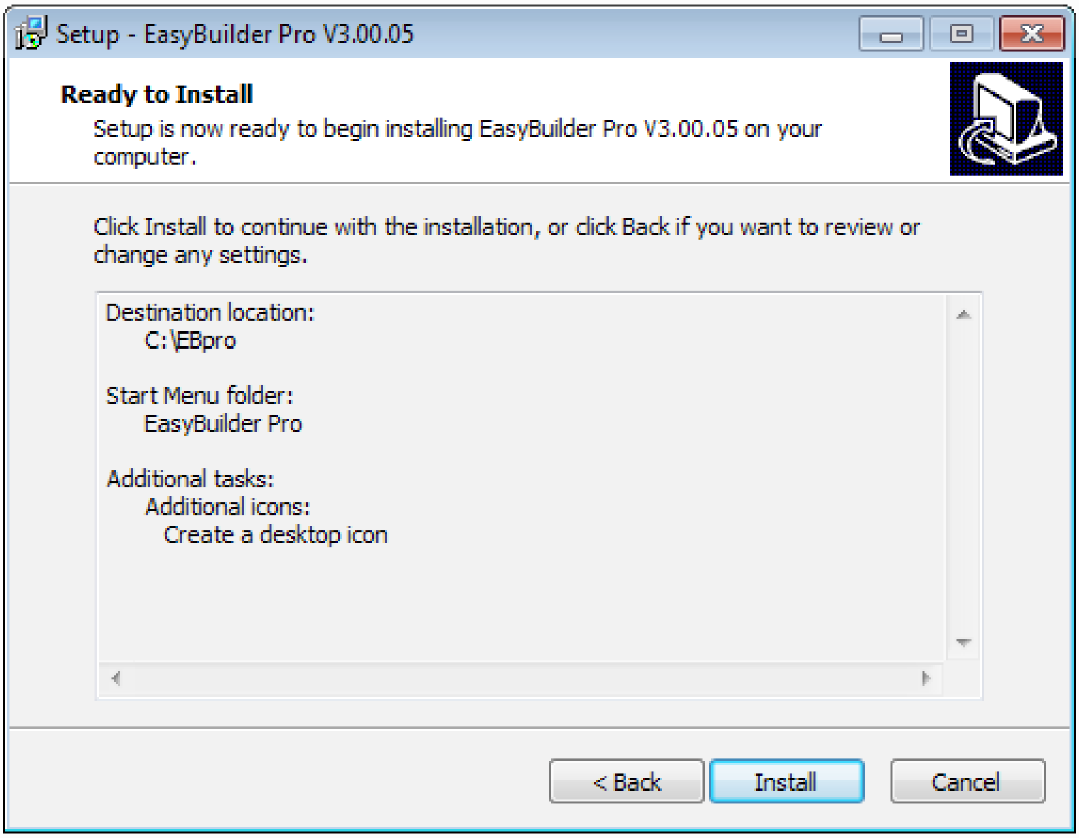
5. Select a start menu folder to save the program's shortcut. Click (Browse) to designate a folder or use the recommended folder then click (Next).



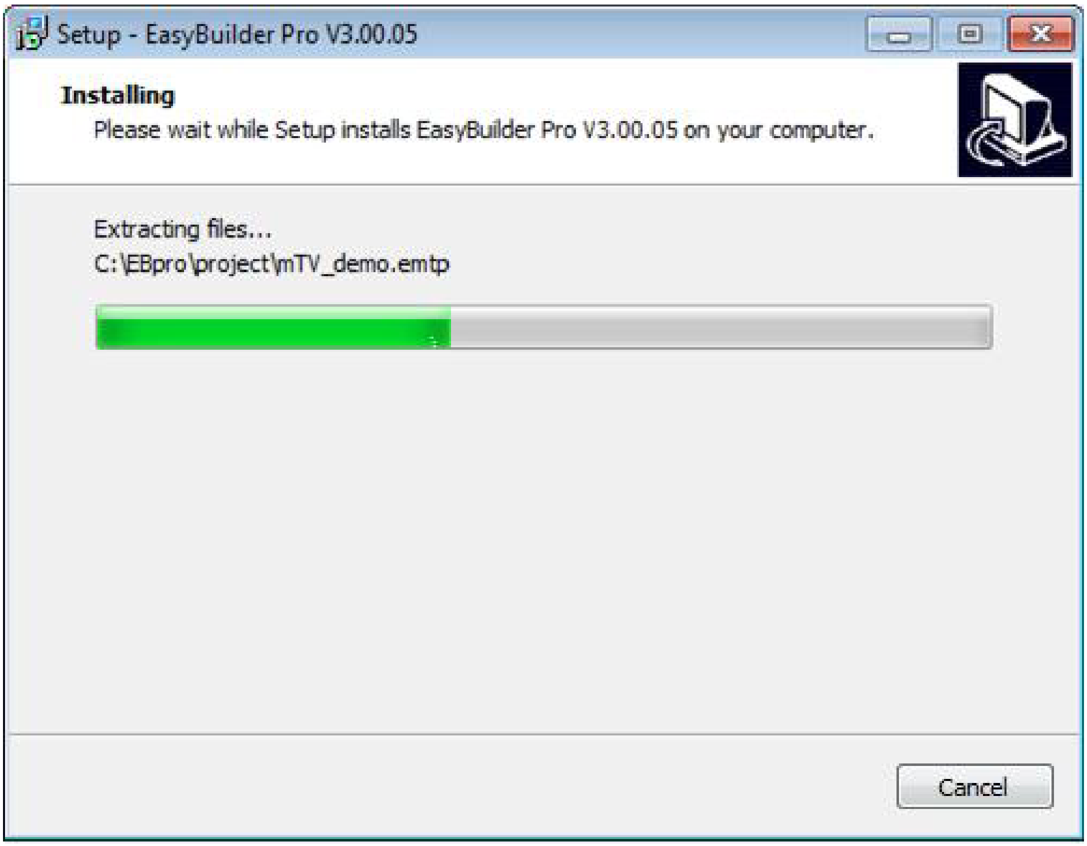
6. Users will be enquired if there are any additional tasks to be done. For example: (Create a desktop icon). Select it if needed then click (Next) to continue.



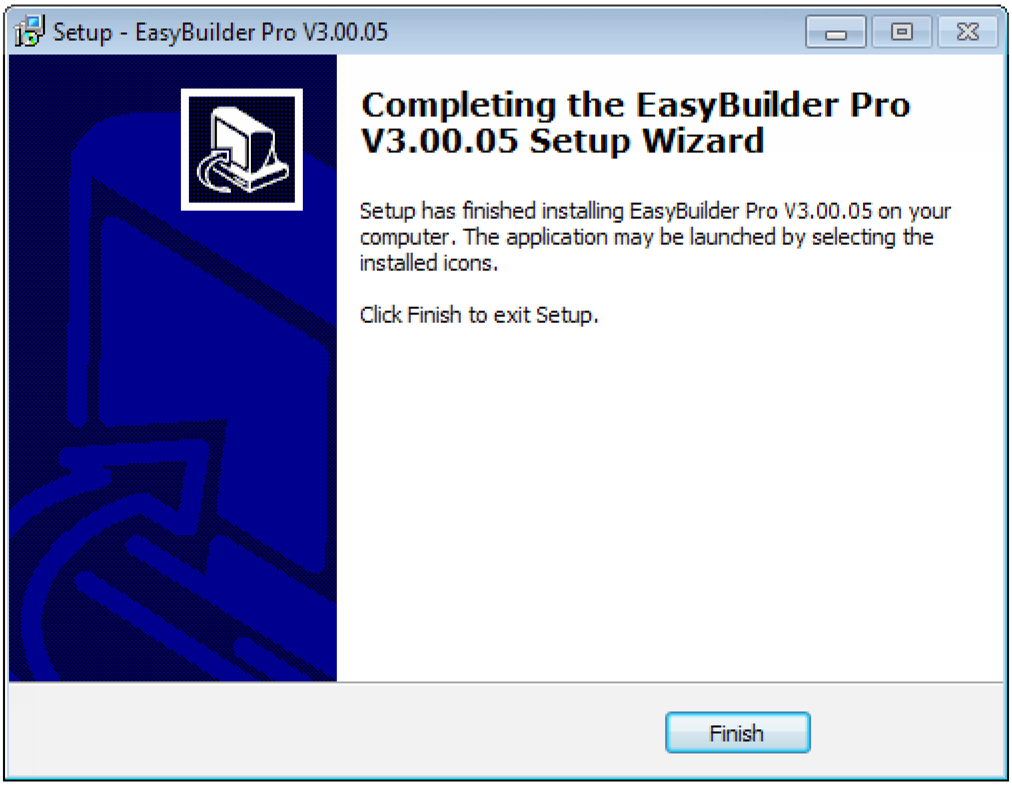
7. Please check if the setting is correct. If any changes need to be made, click (Back) to change the setting or click (Install) to start installation.



8. Installation processing.



9. Click (Finish) to complete the installation.



10. The EasyBuilder shortcut can be found in (Start) » (All Programs) » (EasyBuilder Pro).

The description of each item in EasyBuilder Pro menu:

Installed file	Description
Administrator tool	Saves the data of User Accounts, USB Security Key, e-Mail SMTP Server Setting, e-Mail Contacts to USB disk and import to HMI.
Easy access	Supports access to any HMI connected to the internet. The HMI can be operated on PC.
EasyBuilder Pro	EasyBuilder Pro project editor.
EasyConverter	Conversion tool for Data Sampling and Event Log.
EasyDiagnoser	Monitoring and debugging tool operated on HMI.
EasyPrinter	Remote screen hardcopy and backup server.
EasySimulator	Executes simulation.
EasyWatch	On PC to monitor or set HMI and PLC address value.
Recipe editor	Tool for setting the format of Recipe data. Users can open Recipe data or data in the External Memory.
Release note	Software release notes.
Structure editor	Supports AB TAG and improve the flexibility to read / write an object.
Utility manager	EasyBuilder Pro management tool.

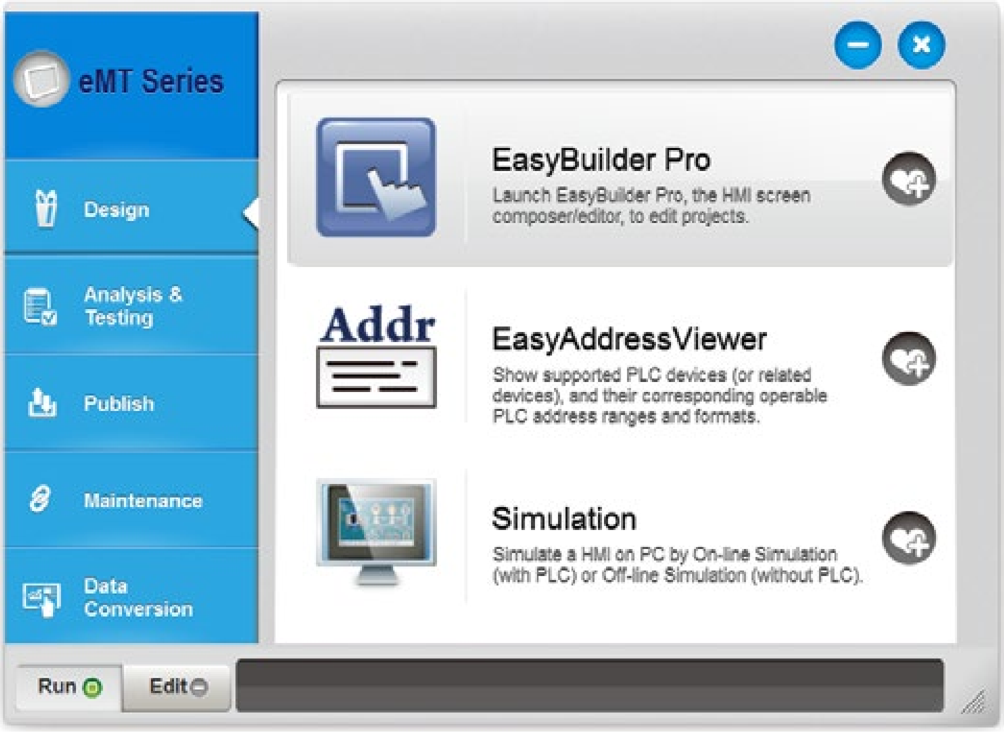
Note
HMI supports downloading/uploading projects via USB cable. After installing EasyBuilder Pro, please go to (Computer Management) » (Device Manager) to check if the USB driver is installed, if not, please install it manually.

2. UTILITY MANAGER

This chapter explains how to use Utility Manager.

2.1. OVERVIEW

After installing EasyBuilder Pro, double click (Utility Manager) shortcut on the desktop to start.
Utility Manager is for launching several utilities and it is a stand-alone program.



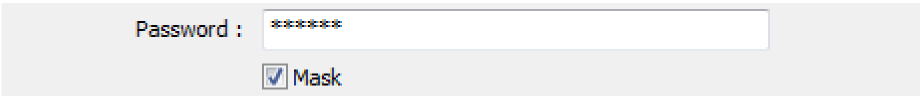
Utilities	Description
Select model	Select your HMI model. Please note that if the model is incorrect, certain features may not work correctly.
Design	**EasyBuilder Pro:** launch EasyBuilder Pro to edit projects. **EasyAddressViewer:** review the address ranges and formats of supported PLCs. **Simulation:** simulate a HMI on PC by On-line Simulation (with PLC) or Off-line Simulation (without PLC).
Analysis & Testing	**EasyDiagnoser:** on-line monitoring and debugging tool. Diagnose the connection status between PC/HMI and PLC. ☞ See “33 EasyDiagnoser”. **EasyWatch:** allows users to monitor HMI or PLC address values via Ethernet on PC. ☞ See “35 EasyWatch”. **Reboot HMI:** restart a HMI to its initial condition by Ethernet or USB connection. **Pass-Through:** allows PC applications to control PLC via HMI. In this case the HMI is an adaptor. ☞ See “29 Pass-through”.
Publish	**Download:** download project file to HMI via Ethernet. **Upload:** upload files on HMI to PC via Ethernet. **Build Download Data for SD/USB Disk:** build the data to be saved in SD card / USB drive and then insert the device to HMI to download the data. This feature is not supported by cMT Series.
Maintenance	**EasyPrinter, Backup/Printer Server:** a backup/printer server on PC, which receives backups from HMI and run a defined batch to convert, or HMI screenshots to print out on PC. **Administrator Tools:** allows storing the data of (User Accounts), (USB Security Key), (e-Mail SMTP Server Settings), and (e-Mail Contacts) to USB. This feature is not supported by cMT Series. ☞ See “36 Administrator Tools”. **CloudHMI:** connect to a cMT-SVR machine. PC acts as a display terminal for cMT-SVR. **Data/Event log Information:** connect with HMI via USB cable or Ethernet to check the number of history files in HMI. This feature is not supported by cMT Series. **EasyAccess 1.0:** control remote HMI instantly and conveniently no matter which corners in the world you are. You can find more information at: www.ihmi.net

Data conversion	**Database Editor:** used to edit recipe data. Click the icon to download the document about Recipe Database. **Easy Converter:** reads the data sampling file (.dtl) and event log file (.evt) in HMI and convert the files to Excel (.xls) format. See “25 EasyConverter”. **Recipe Editor:** used to create, view, and edit recipe data. See “24 Recipe Editor”.
	Minimize the window.
	Close Utility Manager.
	Add the frequently used utilities to the toolbar at the bottom of the window.
Run	Run the selected utility on the toolbar.
Edit	Delete the selected utility on the toolbar.

2.2. HMI IP, PASSWORD

2.2.1. Settings

When operating HMI via Ethernet or USB cable, please set the password for HMI to protect against unauthorized access.



Set the download password. To use masking password, select (Mask) check box.

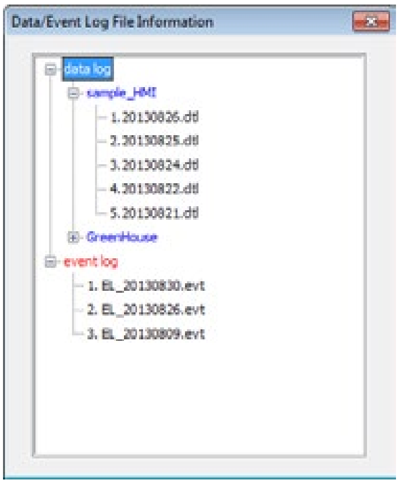
	Note Please remember the password, otherwise, while restoring HMI default settings, the project files and data in HMI will be completely erased.

2.2.2. Reboot HMI

Reboot the HMI without unplugging. After reboot, the system returns to the initial state. Set the correct IP address when rebooting HMI via Ethernet.

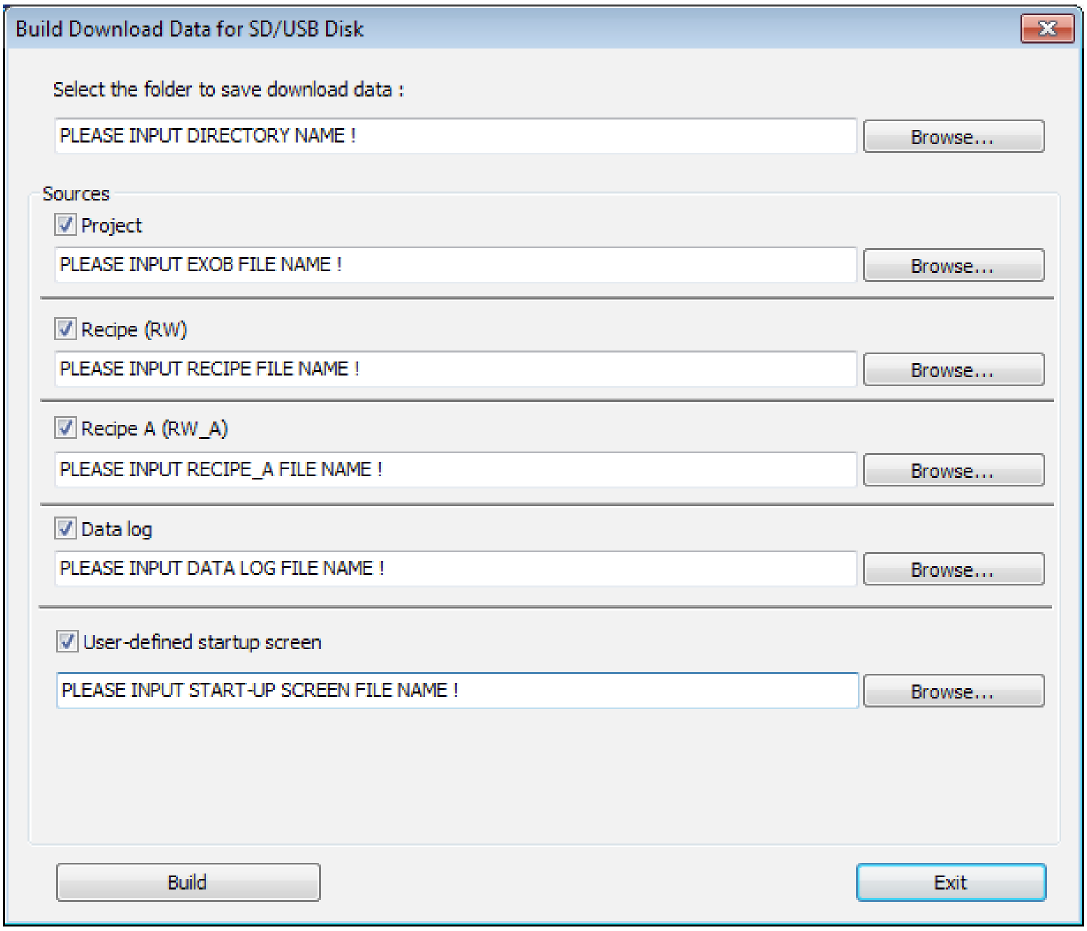
2.2.3. Data/Event Log File Information

After setting, connect with HMI to check the number of history files in HMI.



2.3. EDITING TOOLS

2.3.1. Build Download Data for SD/USB Disk



1. Insert an external device (SD card or USB drive) to PC.
2. Assign the directory to store data.
3. Select the directory of the source file.
4. Click (Build) to create files for downloading.

Files will then be store to the inserted device for users to download to HMI without connecting via a USB cable or Ethernet.

2.3.2. Steps to Download Project to HMI via USB Disk or SD Card

Assume we will download data in the folder named “123” (K:\123) on an USB disk.

1. Insert USB (in which the project is saved) to HMI.
2. In (Download / Upload) dialog box select (Download).
3. Enter Download Password.
4. In (Download Settings) dialog box, select (Download project files) and (Download history files) check boxes.
5. Press (OK).
6. In (Pick a Directory) dialog box, select directory: usbdisk\disk_a_1\123.
7. Press (OK).

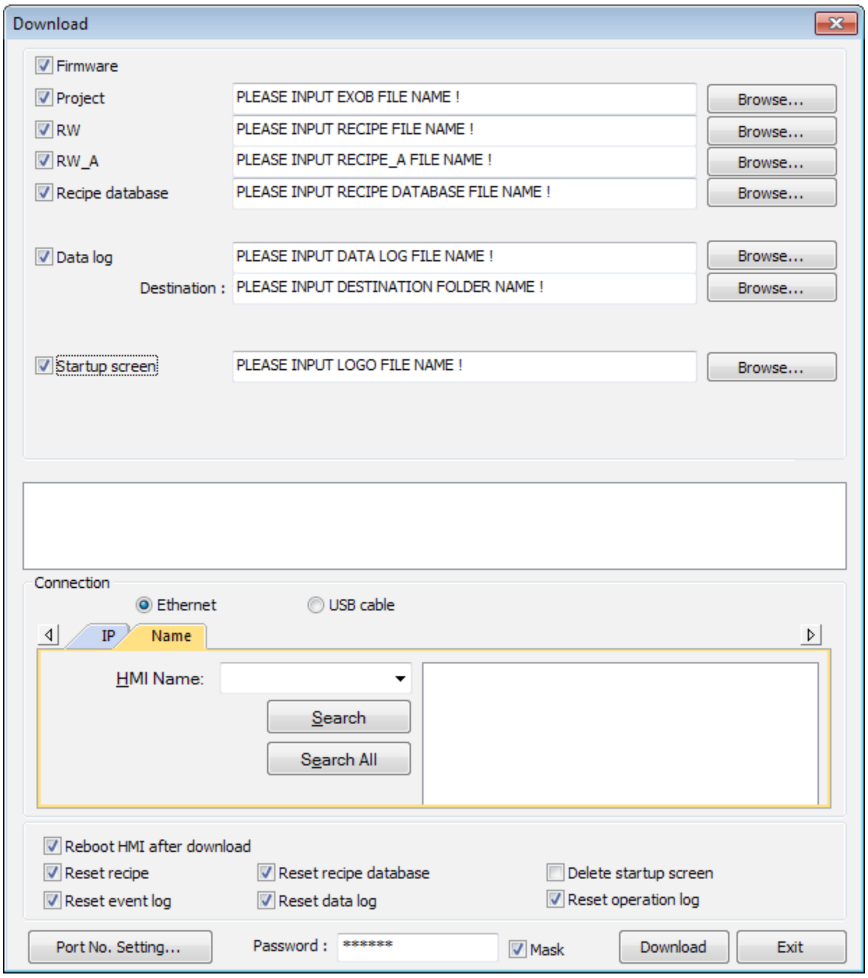
Project will then be updated.

	Note If only the history files are downloaded, it is necessary to reboot HMI to update files.

2.4. TRANSFER

2.4.1. Download

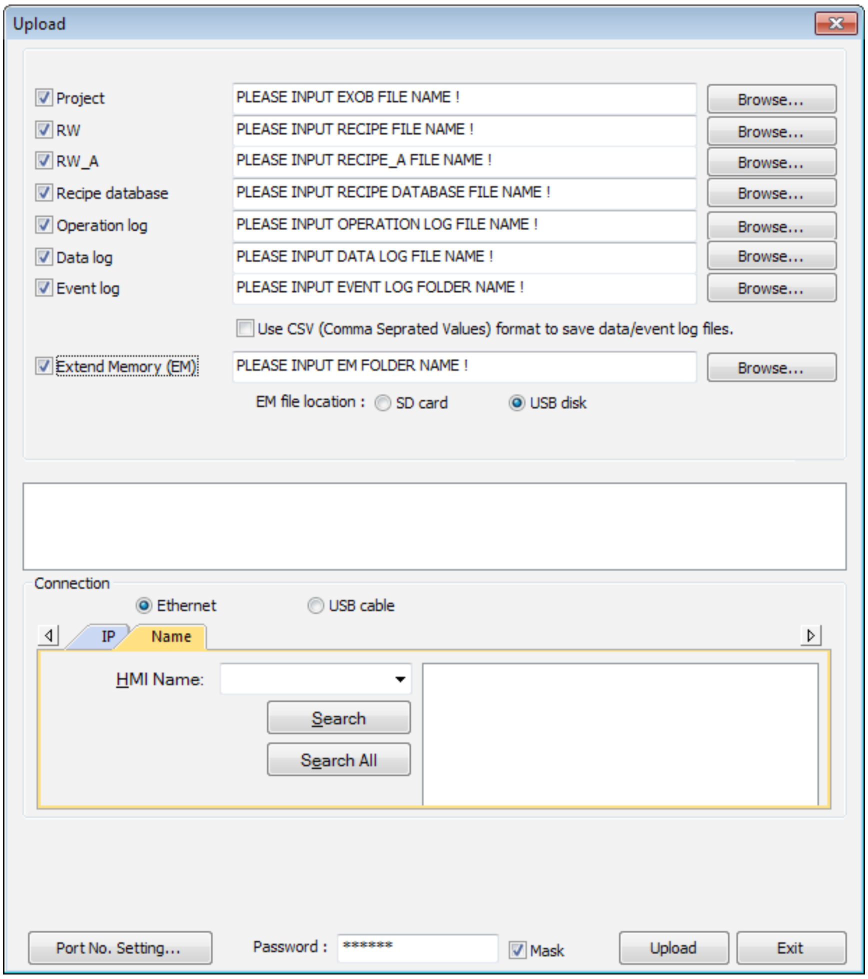
Download files to HMI via Ethernet or USB cable.



Setting	Description
Firmware	Update HMI kernel programs. The firmware must be downloaded at the first time downloading data to HMI.
Project	Select an .exob project file.
RW / RW_A	Select a .rcp recipe file.
Recipe database	Select a .db file of Recipe Database.
Data log	Select the data sampling folder in HMI and then select a .dtl file.
Startup screen	Download a .bmp bitmap file to HMI. After HMI is rebooted, this .bmp file will be shown before project starts.
Reboot HMI after download	Automatically reboot after download.
Port No. Setting	Select the port by which to download the project file via Ethernet.
Reset recipe / recipe database / event log / data log / operation log / delete startup screen	Erase the selected files in HMI before download.

2.4.2. Upload

Upload files from HMI to PC via Ethernet or USB cable. Click (Browse) and assign the file path before uploading.



Setting	Description
Event log	Upload the .evt file in HMI to PC.
Extended Memory (EM)	Upload the .emi file saved in SD card or USB disk to PC.

☞ For information about (Project), (RW / RW_A), (Recipe database) or (Data log), see “2.4.1 Download” in this chapter.

	Note The file will be uploaded to PC in .exob format. Please decompile it into editable .emtp file first and open the .emtp file in EasyBuilder Pro.

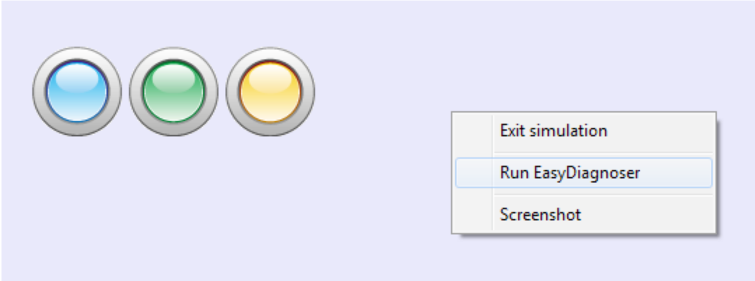
2.5. SIMULATION

2.5.1. Off-Line Simulation / On-Line Simulation

- Off-line simulation: simulate project operation on PC without any connection
- On-line simulation: simulate project operation on PC and PLCs are directly connected with PC

	Note When using (On-line simulation) on PC, if the target device is a local PLC (the PLC directly connected to PC), there is a 10 minutes simulation limit.

Before executing On-line/Off-line Simulation, please select the source .exob file.
When executing On-line/Off-line Simulation, right click to use these functions:



Setting	Description
Exit simulation	Stop simulating.
Run EasyDiagnoser	To monitor current communication status.
Screenshot	Capture and save current screen image as a picture file in the screenshot folder under the installation directory.

2.6. PASS-THROUGH

This function allows the PC application to connect PLC via HMI. In this case, the HMI works like a converter.

Pass-through provides two modes: (Ethernet) and (COM port).
When using (Ethernet), please install the virtual serial port driver first.

☞ For more detail, please refer to “Chapter 29 Pass-Through Function”.

3. CREATE AN EASYBUILDER PRO PROJECT

This chapter explains the basic steps to create an EasyBuilder Pro project.

3.1. OVERVIEW

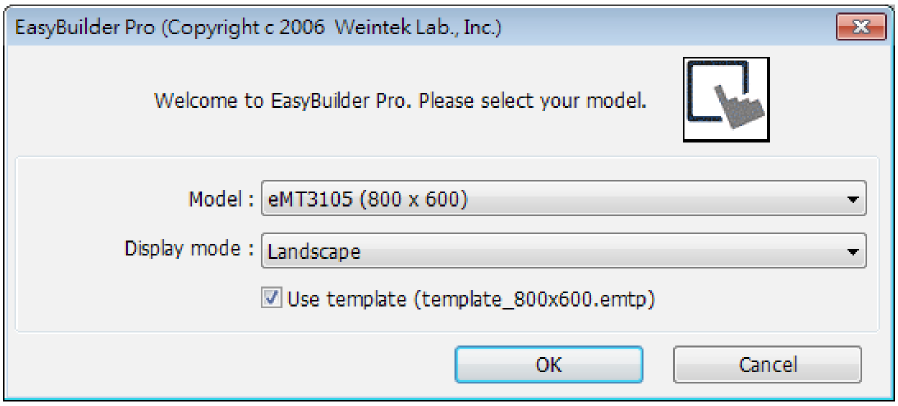
The following is the process of creating a project.

1. Create a new project file.
2. Save and compile the project file.
3. Run On-line or Off-line simulation.
4. Download the project file to HMI.

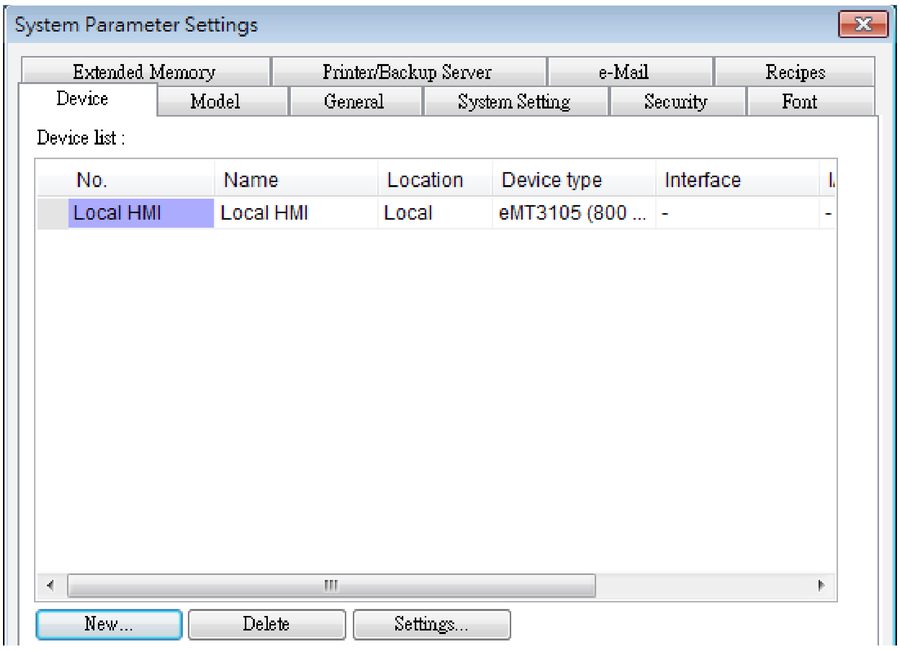
The following describes each process.

3.2. CREATE A NEW PROJECT FILE

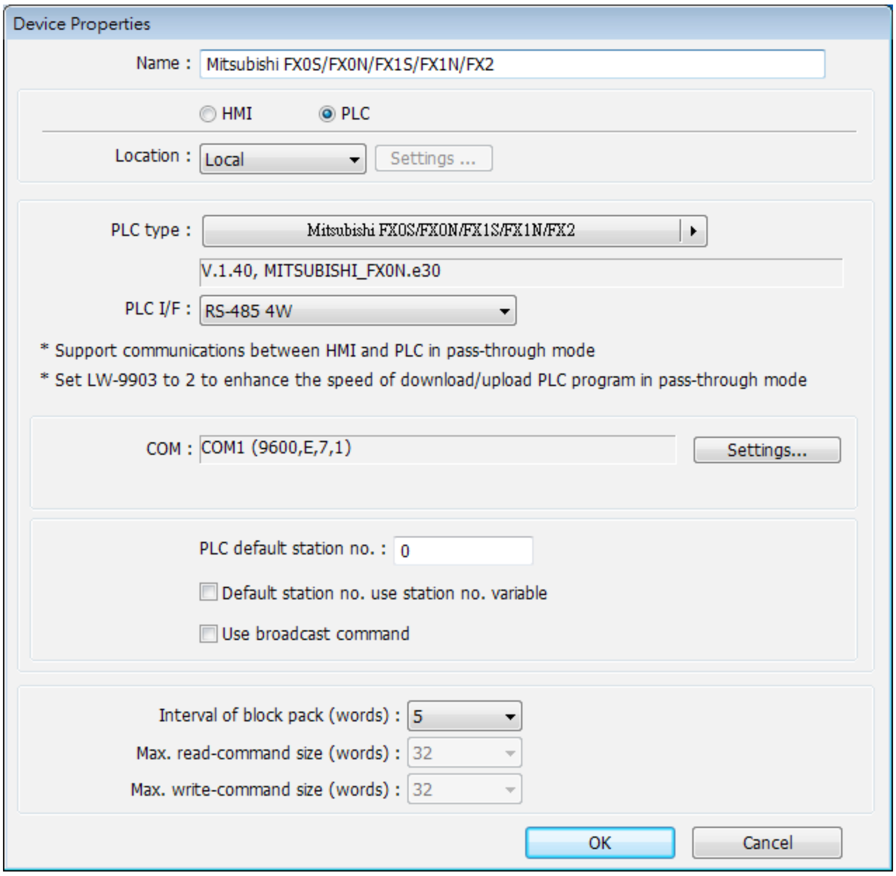
1. Launch EasyBuilder Pro and open a new file.
2. Select (Model) and select (Use template) check box.



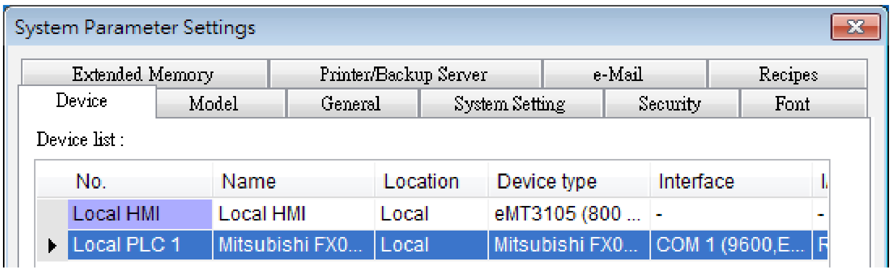
3. Click (New).



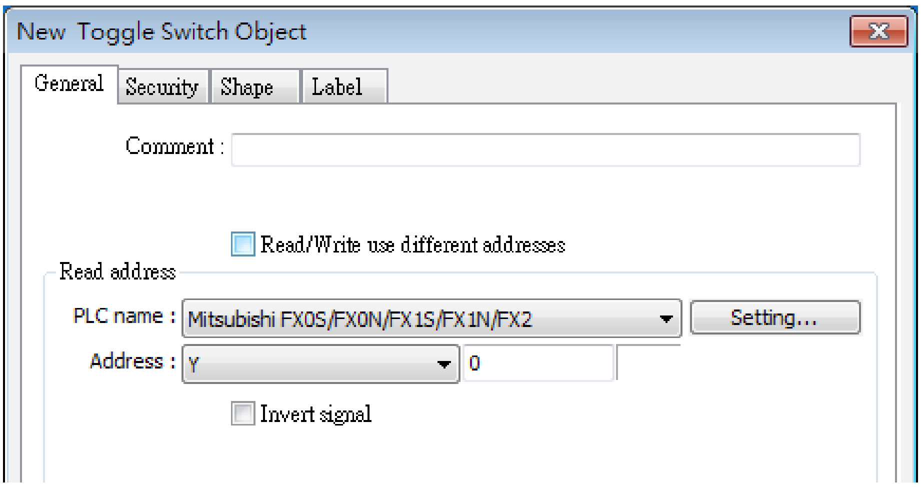
4. Configure parameters.



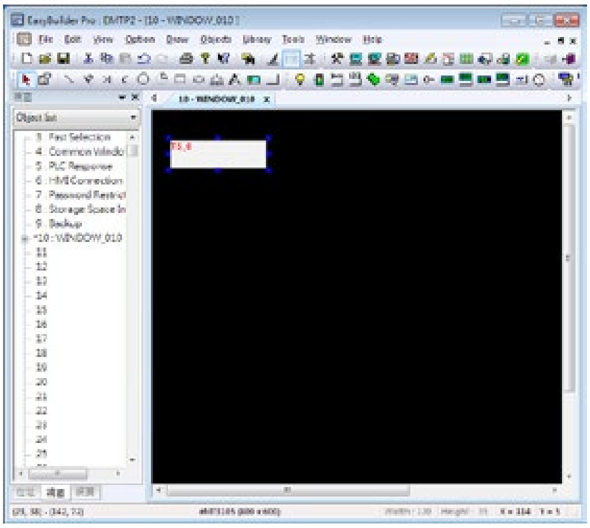
5. A new device is added to the (Device List).



6. Create an object, take Toggle Switch  as an example, and set the address.



7. Place the object in the edit window. A project is now created.

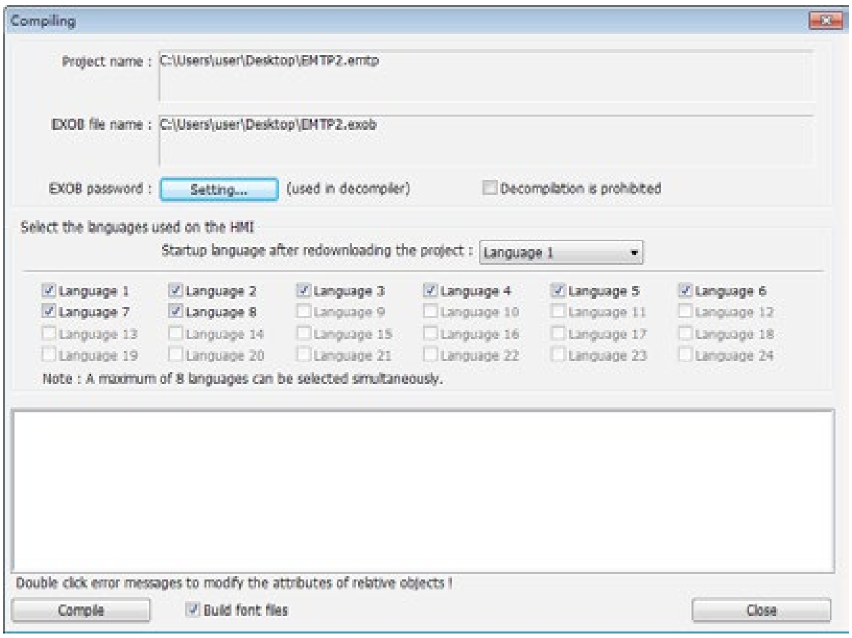


3.3. SAVE AND COMPILE THE PROJECT FILE

1. In EasyBuilder Pro toolbar click (File) » (Save) to save the .emtp file.
2. In EasyBuilder Pro toolbar click (Tools) » (Compile) to compile .emtp file as .exob file, which could be downloaded to HMI. This also checks if the project can run correctly.

	Note For cMT-SVR, the project file extension is .cmtp, and the compiled file extension is .cxob.

3. To use multiple languages, all languages must be configured in Label Tag Library first. When downloading the project to HMI, select the needed languages only. A successful compilation is shown in the following figure.



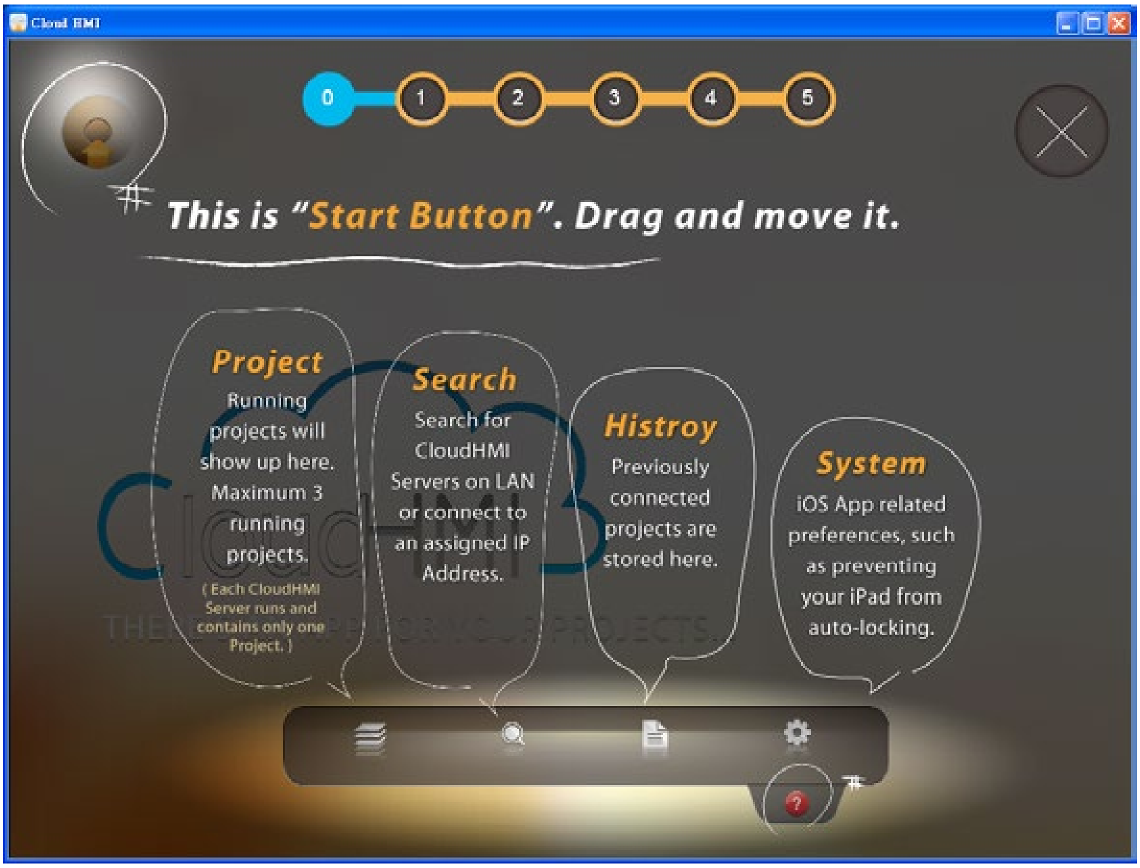
3.4. RUN ON-LINE OR OFF-LINE SIMULATION

	Off-line simulation: simulate project operation on PC without connecting any device.
	On-line simulation: simulate project operation on PC without downloading the project to HMI. The PLC is connected to PC, please set correct parameters.

	Note When using On-line Simulation on PC, if the target device is the PLC directly connected to PC, there is a 10-minute simulation limit.

3.5. CLOUDHMI

This program connects to CloudHMI Server (cMT-SVR) via network, similar to CloudHMI App on iPad. To run this program, execute CloudHMI.exe under installation directory. Or, in EasyBuilder Pro toolbar, click (Tools) » (CloudHMI).



Setting	Description
Start button	Click to enter the main configuration screen, users can drag and move the button.
Project	Running projects will show up here. Maximum 3 running projects. Each CloudHMI Server runs and contains only one project.
Search	Search CloudHMI Servers on LAN or connects to an assigned IP Address.
History	Previously connected projects are stored here.
System	Related preferences, such as using portrait mode.

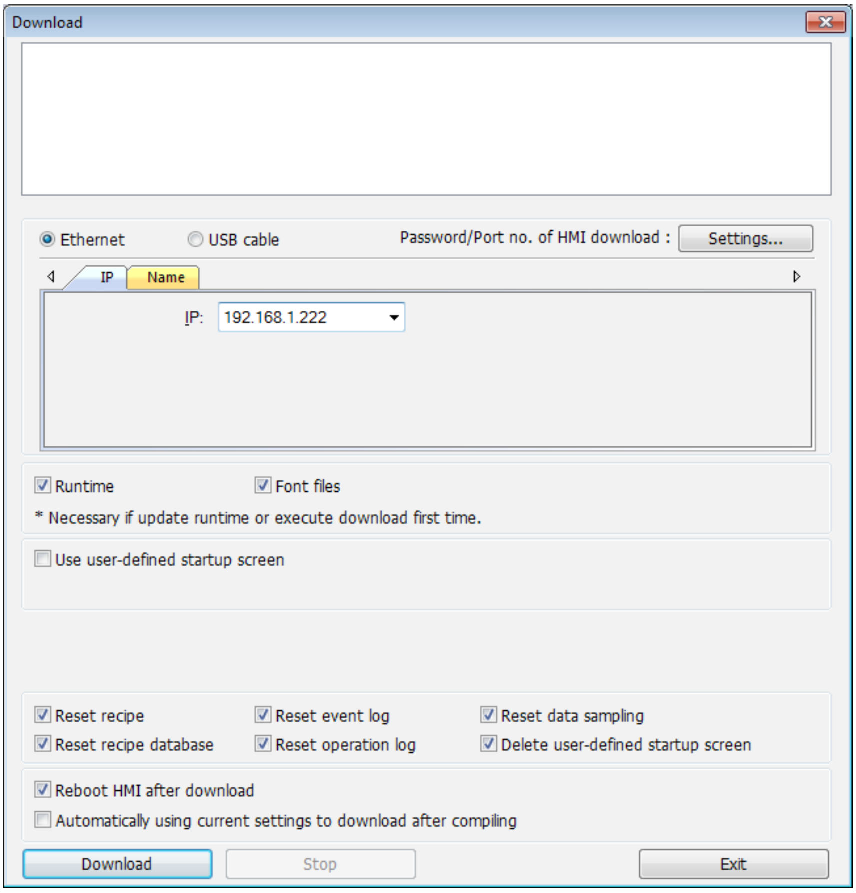
3.6. DOWNLOAD THE PROJECT FILE TO HMI

The following explains four ways to download the project file to HMI.

	Note For cMT-SVR, only the way described in 3.6.1 is available.

3.6.1. Configure in EasyBuilder Pro

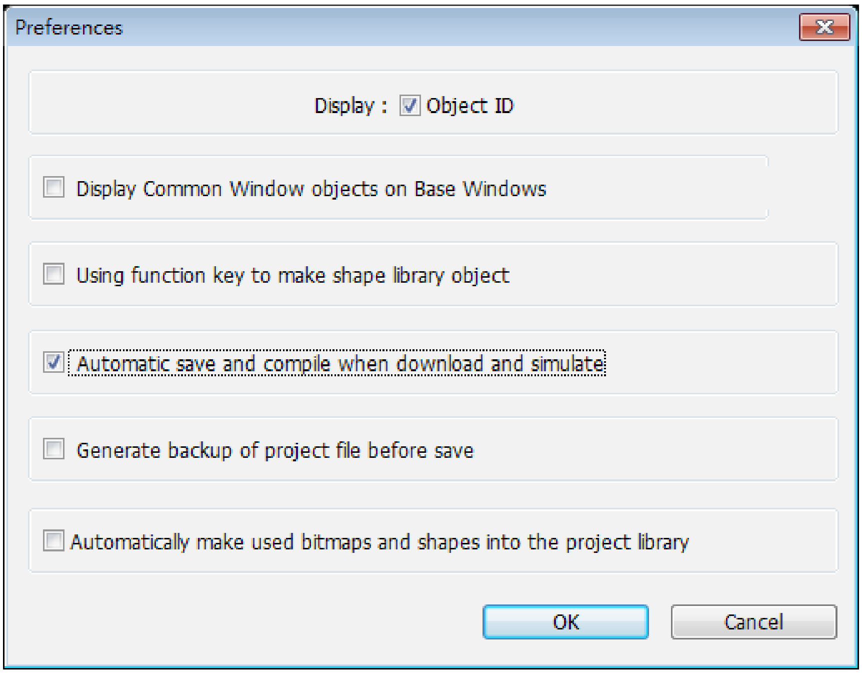
1. In EasyBuilder Pro toolbar, click (Tools) » (Download). Make sure that all the settings are correct.
2. Select (Ethernet), set password and HMI IP.



Setting	Description
Firmware	Select the check box to update the HMI kernel programs. If this is the first time downloading file or EasyBuilder Pro version is updated, please download the firmware before downloading files to HMI.
Font files	Download the font used in the project.
Reset recipe / event log / data sampling / recipe database / operation log / user-defined startup screen	The selected files will be erased before downloading.
Reboot HMI after download	HMI will reboot after the downloading process is done.
Automatically using current settings to download after compiling	The system will compile the project and download it to the latest target HMI. The way to enable this function is described in the following part.

	Note ▪ For cMT-SVR, the (Firmware) and (Font files) check boxes are not available. ▪ (Automatically using current settings to download after compiling): if this check box is selected, EasyBuilder Pro will compile the project and download it to the latest target HMI.

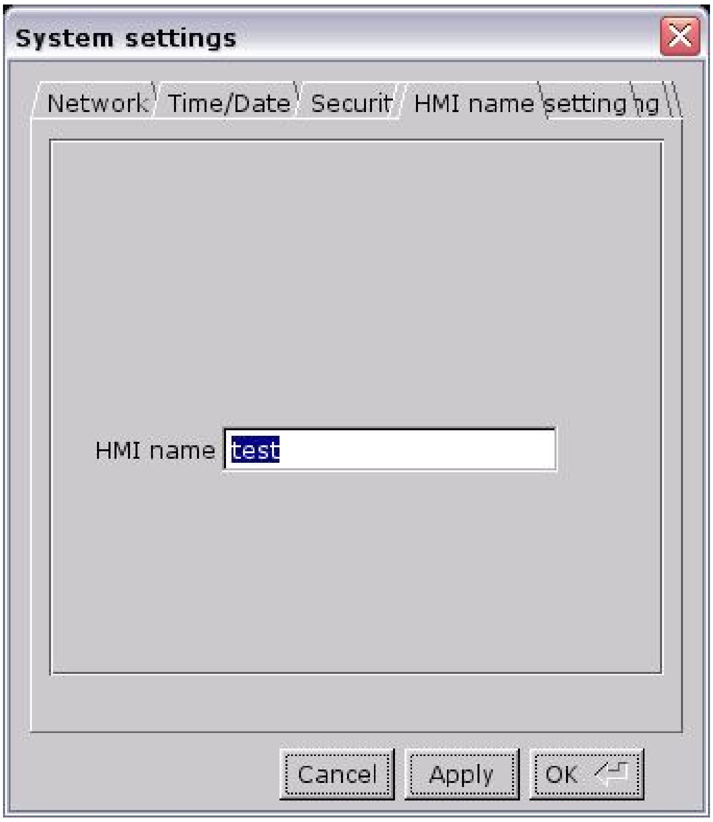
3. In EasyBuilder Pro toolbar, click (Option) » (Preferences).
4. Select (Automatic save and compile when download and simulate) check box.



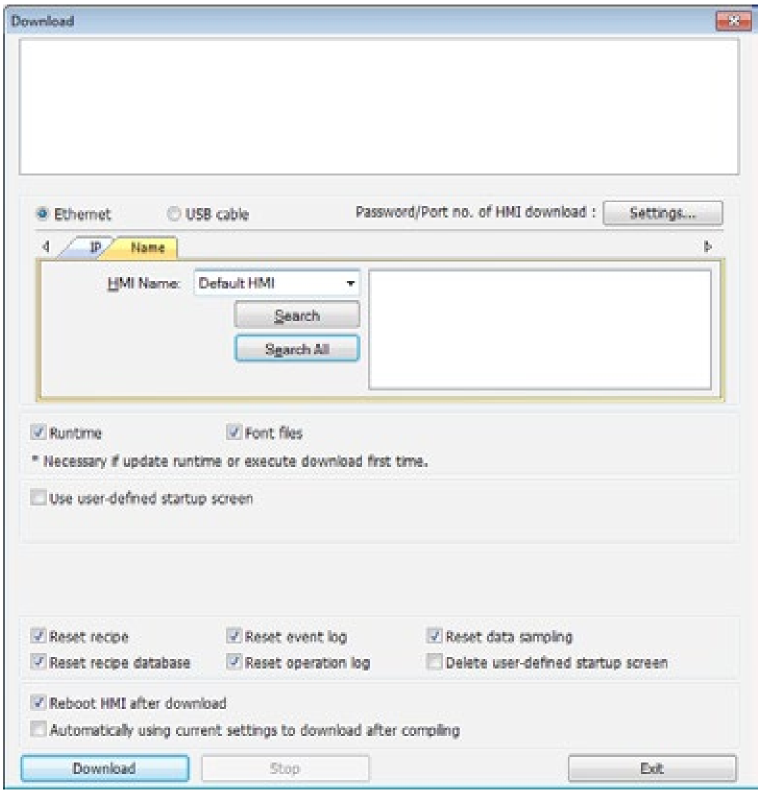
5. In EasyBuilder Pro toolbar, click (Save) and then (Download).
6. In the dialog box, select (Automatically using current settings to download after compiling) check box.
7. Click (Download).
8. When finished, next time when (Download) is clicked, EasyBuilder Pro will automatically compile and download the project to the latest target HMI.

3.6.2. Use HMI Name

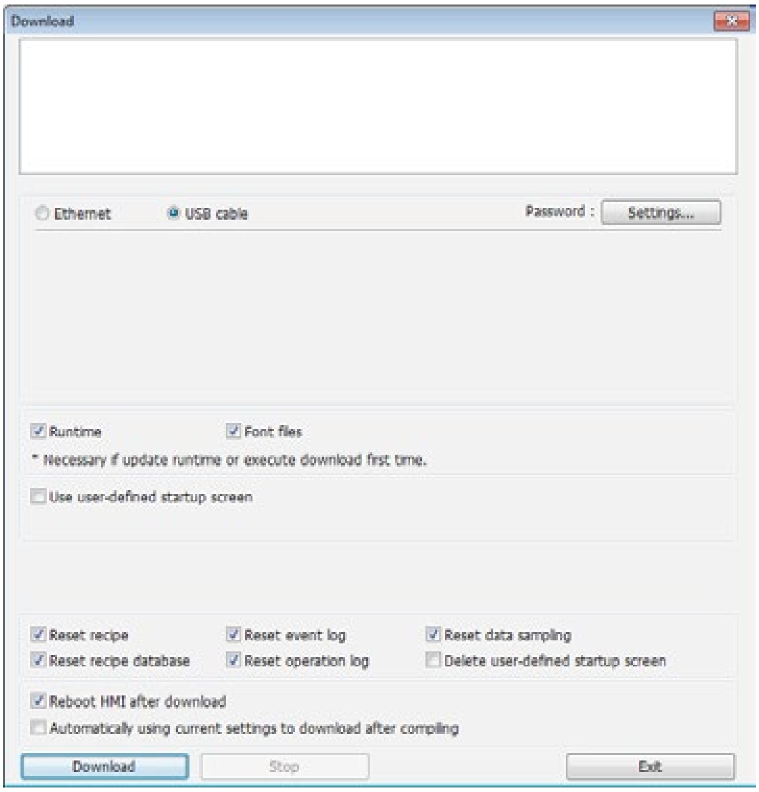
1. Go to (System settings) on HMI and then set HMI name first.



2. On PC, select the HMI name and start downloading. To use (Search), enter the HMI name first to search for the HMI. (Search all) searches for all HMIs in the same subnet network.



3.6.3. Use USB Cable

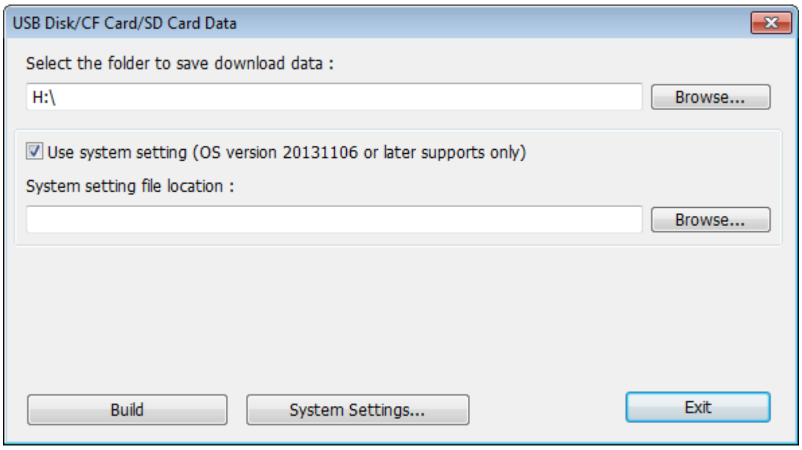


Select USB cable to download project to HMI. The way of setting is same as “3.6.1 Configure in EasyBuilder Pro”. Before downloading via USB cable, please make sure the USB driver is installed. Go to (Computer Management) » (Device Manager) to check if USB driver is installed. If it is not installed, please refer to installation steps to install manually.

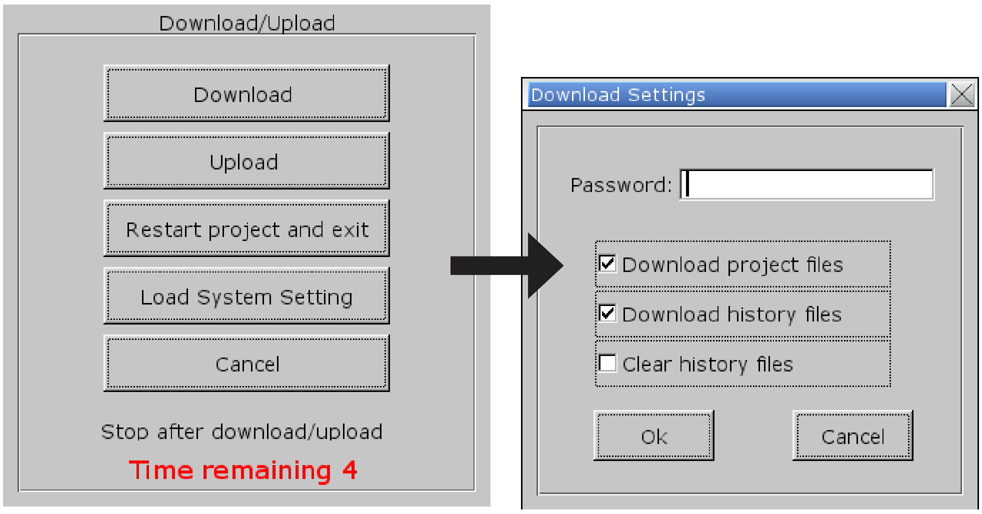
3.6.4. Use USB Drive or SD Card

The following explains how to download project file by using USB drive or SD card.

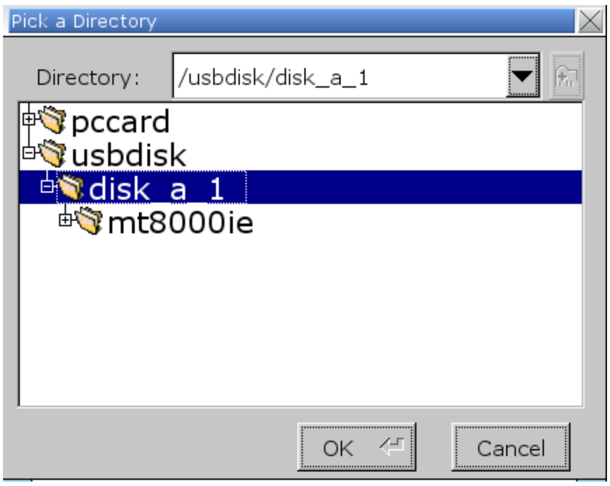
1. On EasyBuilder Pro toolbar click (Tools) » (Build Download Data for SD / USB Disk).



2. Insert an external device, such as SD card or USB disk to HMI.
3. Select (Download) and enter password.



4. After password is confirmed, it will show the directories in the external device. (pccard: SD Card; usbdisk: USB Drive)



5. Select the directory that contains project, and then click (OK) to start downloading.

	Note
	▪ Please select the parent directory of the generated files when downloading. For the structure above, please select disk_a_1, not mt8000ie. ▪ You may click (System Settings) to save the hardware settings configured in EasyBuilder Pro into SD card or USB disk, and then download the settings file to HMI. See "4 Hardware Settings" for more information.

4. HARDWARE SETTINGS

This chapter explains HMI settings.

4.1. OVERVIEW

This chapter discusses the HMI settings.

	Download Click the icon to download the cMT-SVR User Manual if needed. Please confirm your internet connection before downloading the document.

4.2. I/O PORTS

The I/O ports are different from one HMI type to another; please see the relevant datasheet for more information. The I/O Ports include:

- SD card slot: download / Upload project via SD card, including recipe transfer, event log, data log…etc and to backup or record history data
- COM Port: connects PLC or other peripheral devices. The type of serial port include: RS232, RS485 2W and RS485 4W
- Ethernet: download / Upload project including recipe transfer, event log, data log…etc. Connects to Ethernet devices, such as PLC, laptop
- USB Host: supports USB devices, such as mouse, keyboard, USB disk, printer or barcode device
- USB Client: download / Upload project including recipe transfer, event log, and data log…etc

For the first time operating HMI, please complete the following system settings. When finished, the project files designed using EasyBuilder Pro can be used on HMI.

4.3. LED INDICATORS

The LED indicators on the HMI indicate:

- Models: MT8121XE, MT8150XE, MT8121iE, MT8150iE

LED	Description
PWR (orange)	Indicates power status.
CPU (green)	Blinks when read/write Flash Memory.
COM (blue)	Indicates COM port communication status, blinks during communication. When communication is good, it may stay on. (Not including network communication).

- Other Models

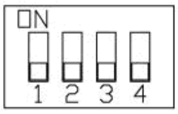
LED	Description
PWR (orange)	Indicates power status.
CPU (green)	Indicates CPU status. If it blinks or goes out, there may be a CPU error.
COM (blue/red)	Indicates communication status, blinks during communication. When communication is good, it may stay on.

4.4. SYSTEM RESET

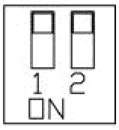
Each HMI is equipped with a reset button and a set of DIP switches. When using the DIP switches to change modes, the corresponding functions will be triggered. If system password is lost or forgotten, see the following steps to restore factory default.

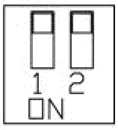
1. Flip DIP Switch 1 to ON and the rest to OFF, and then reboot HMI. HMI will switch to touch screen calibration mode.
2. A “+” sign appears on the screen, touch the center of the sign, after all 5 signs are touched, “+” disappears and the touch screen parameters will be stored in HMI system.
3. After calibration, confirm to restore the default password, select (Yes).
4. Confirm to restore the default password again by typing (yes) and clicking (OK). The project files and history records stored in HMI will all be removed. (The default Local Password is 111111. However, other passwords, such as Download/Upload passwords have to be reset.)

The following lists the DIP switch settings of different models. Please see the relevant installation instruction.

eMT / iE Dip Switch	**SW1**	**SW2**	**SW3**	**SW4**	**Mode**
	**ON**	OFF	OFF	OFF	Touch screen calibration mode
	OFF	**ON**	OFF	OFF	Hide hmi system setting bar
	OFF	OFF	**ON**	OFF	Boot loader mode
	OFF	OFF	OFF	**ON**	Reserved
	OFF	OFF	OFF	OFF	Normal

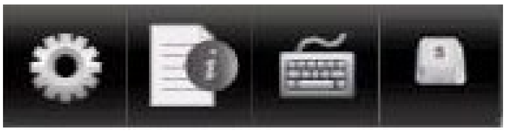
Note
The state of DIP Switch 4 on each unit may be different. If it should be ON when out from factory, the Dip Switch 4 would be set ON and cut off. If it should be OFF, the Dip Switch 4 would be set OFF but the switch is not cut.

mTV Dip Switch	**SW1**	**SW2**	**Mode**
	**ON**	**ON**	Restore factory default
	ON	OFF	Hide system setting bar
	OFF	**ON**	Boot loader mode
	OFF	OFF	Normal mode

cMT-SVR Dip Switch	**SW1**	**SW2**	**Mode**
	**ON**	**ON**	Restore factory default
	ON	OFF	Restore Ethernet IP settings
	OFF	**ON**	Boot loader mode
	OFF	OFF	Normal mode

4.5. SYSTEM TOOLBAR

After rebooting HMI, you can set the system with (System Toolbar) at the bottom of the screen. Normally, this bar is hidden automatically. Only by touching the arrow icon at the bottom-right corner of the screen will the System Toolbar pop up. From left to right the icons are: System Settings, System Information, Text Keyboard, and Number Keyboard.

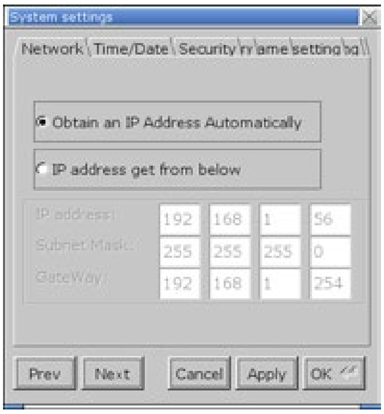


How to hide HMI System Setting Toolbar:

- When (DIP Switch 2) is set ON, the system setting toolbar is disabled. When set OFF; the system setting toolbar is enabled. Please restart HMI to enable/disable the toolbar
- For mTV Series, flip DIP Switch 1 to ON to hide system setting toolbar
- System register (LB-9020) can also enable/disable system setting toolbar. When (LB-9020) is set ON, the toolbar is displayed, and set OFF to hide the toolbar

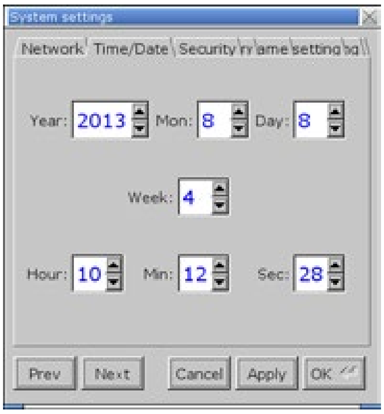
4.5.1. System Setting

Set or modify system parameters. Confirm password for security first. The factory default password is 111111.



Network

When downloading project file to HMI via Ethernet, set the correct IP of the target HMI. You can obtain an IP address automatically or enter the IP address manually.



Time / Date

Set HMI local time/date.

Security

Password protection, the default is 111111. Please click the buttons to set the passwords, and finish password confirmation.

(Password for entering system)

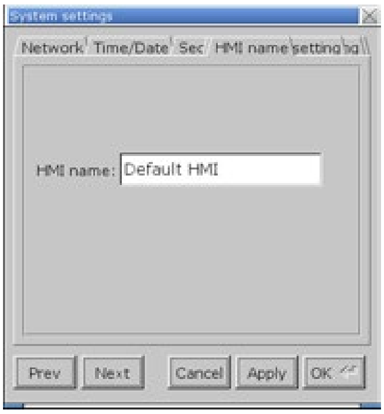
(Password for uploading project)

(Password for downloading project)

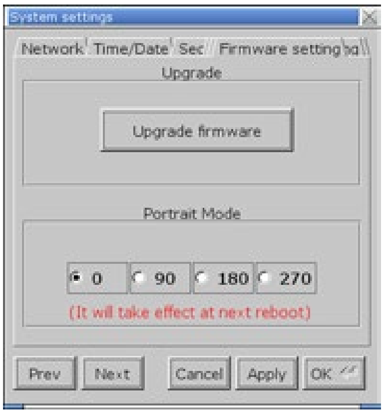
(Password for uploading history data)

History

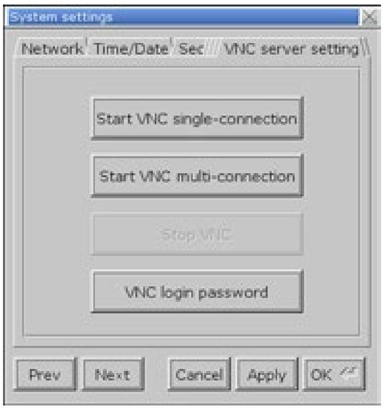
Clears history data in HMI.



HMI name
Set HMI name to be used when download/upload project.



Firmware setting
Upgrade firmware and select the display mode. The display mode will take effect at next reboot.



VNC server
Remote HMI monitoring and controlling via Ethernet.
(Start VNC single-connection)
Allows connection with one VNC client.
(Start VNC multi-connection)
Allows connection with multiple VNC clients.
Connecting more VNC clients may slow down the communication speed.
Please see the settings steps in the later part.

Miscellaneous
Rotary switch for adjusting LCD brightness.
(Popup download window)
If selected, after inserting USB disk or SD card to HMI, the Upload / Download dialog box is displayed.
(Restart after download/upload)
If selected, restarts HMI automatically after uploading / downloading project.

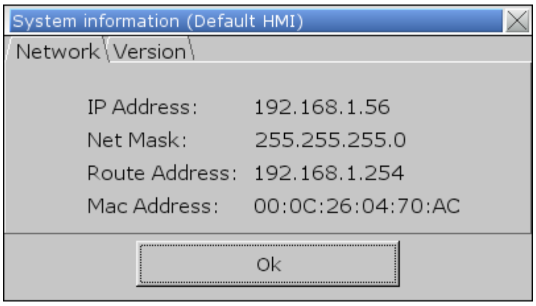
The following steps explain how to set VNC server.

1. Enable HMI VNC server, set password.
2. Install Java IE or VNC Viewer on PC.
3. Enter remote HMI IP in Internet Browser. Or, in VNC Viewer enter remote HMI IP and password.

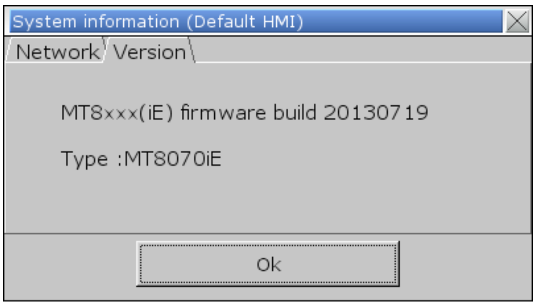
	Note ▪ If there is no activity of VNC client for more than one hour, HMI system will log out automatically. ▪ The mTV Series does not support VNC server.

4.5.2. System Information

▪ Network: displays network information & HMI IP



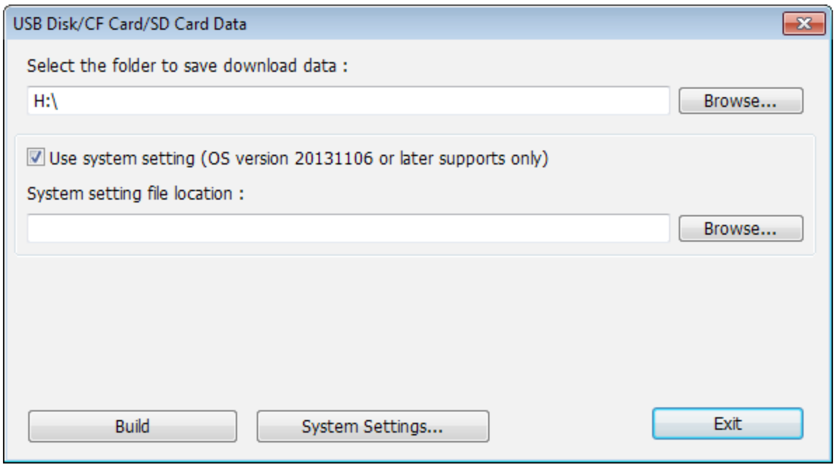
▪ Version: displays HMI firmware version and model type



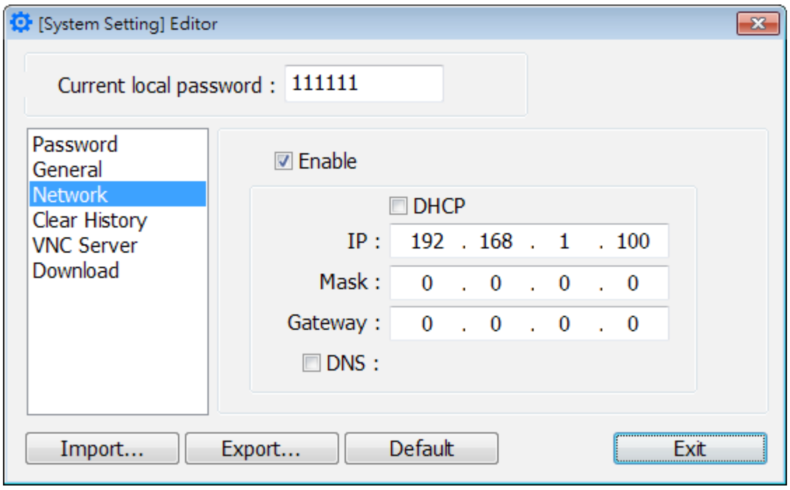
4.6. EASYSYSTEMSETTING

EasySystemSetting allows updating hardware system settings by using SD card or USB drive. The feature is available for HMI OS version 20131106 or later. The following explains how to update HMI IP address by using SD card or USB drive.

1. On EasyBuilder Pro toolbar click (Tools) » (Build Download Data for SD / USB Disk), and then select (Use system setting) check box.

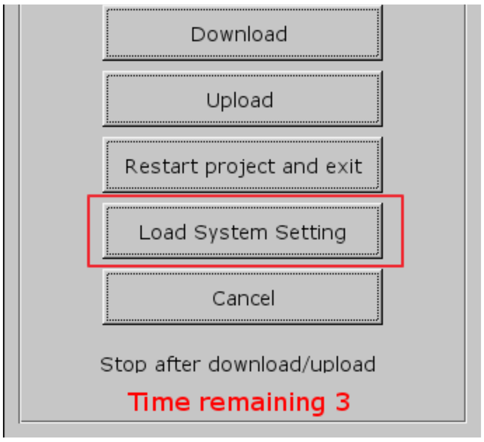


2. Click (System Settings) to open EasySystemSetting. Specify HMI network information as shown in the following figure.



Setting	Description
Import	Import and edit an existing .conf file.
Export	Export the configured data to a .conf file.
Default	Restore default.

3. Click (Export) to generate a “systemsetting.conf” file.
4. Click (Exit) to leave EasySystemSetting.
5. Click (Build) button in (USB Disk/CF Card/SD Card Data) dialog box to generate the file for download by using SD card or USB disk.
6. Insert the storage device that stores the download file to HMI and the Download/Upload dialog box appears.



7. Press (Load System setting) and then the (Download Config Settings) message appears. The project file will be updated after finishing system settings.

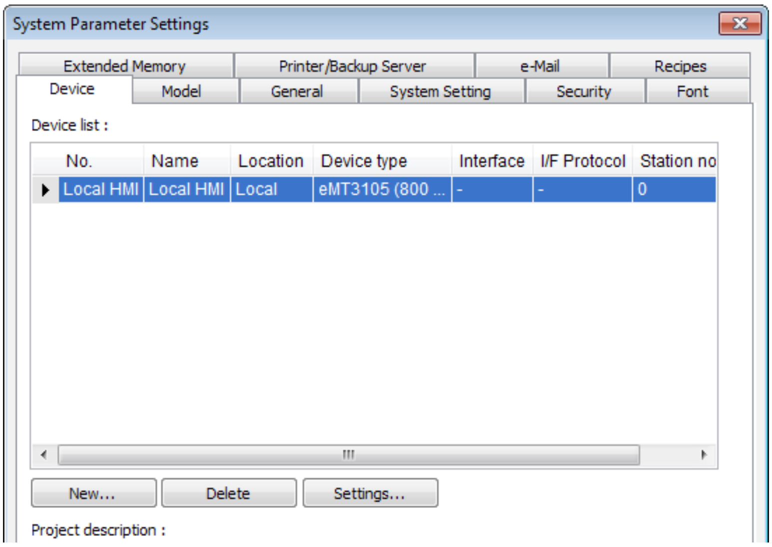
5. SYSTEM PARAMETER SETTINGS

This chapter introduces the system parameter settings.

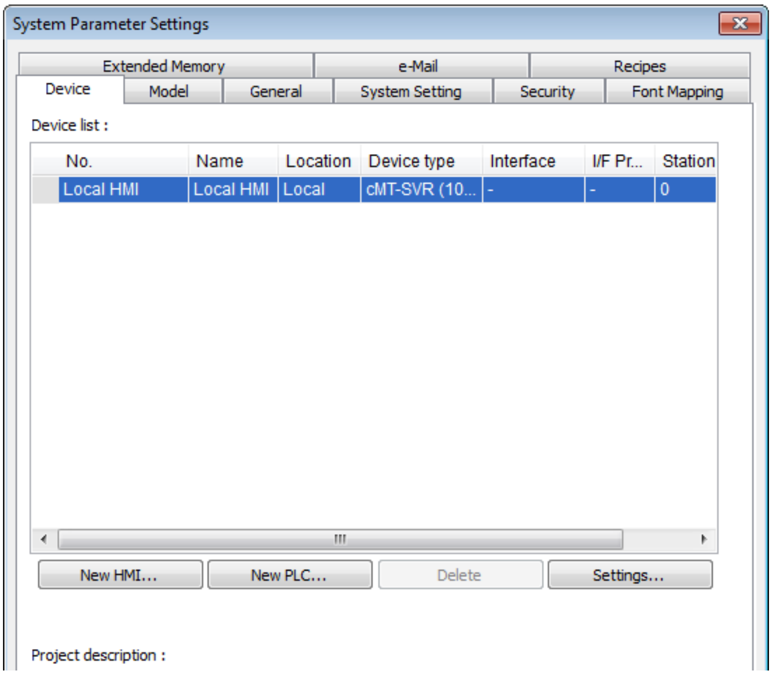
5.1. OVERVIEW

Launch EasyBuilder Pro, in the main menu select (Edit) » (System Parameters) to open the (System Parameter Settings) dialog box. System Parameter Settings are divided into several tabs as shown in the following figures. These tabs will be introduced respectively in this chapter.

- eMT, iE, XE, mTV Series



- cMT Series

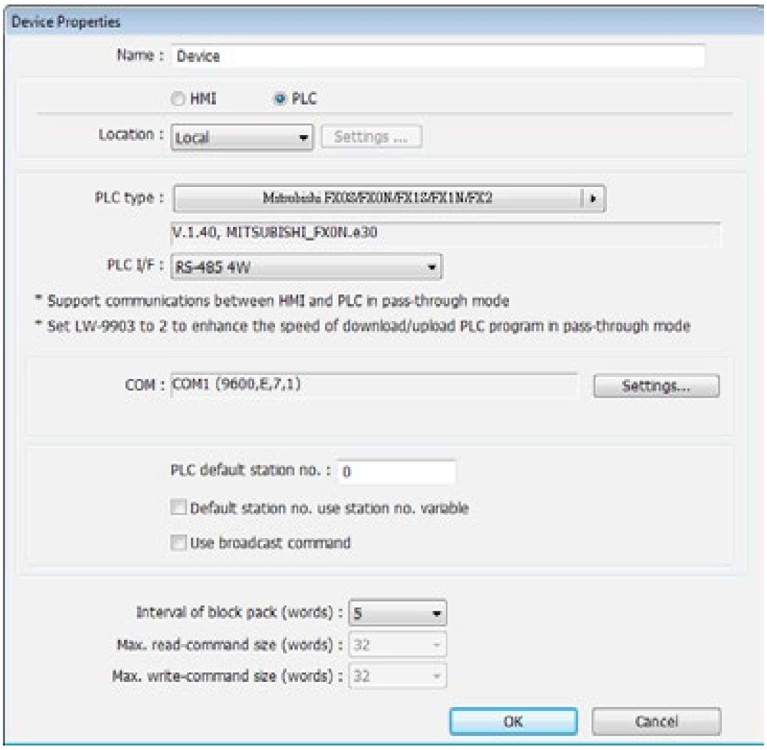


5.2. DEVICE

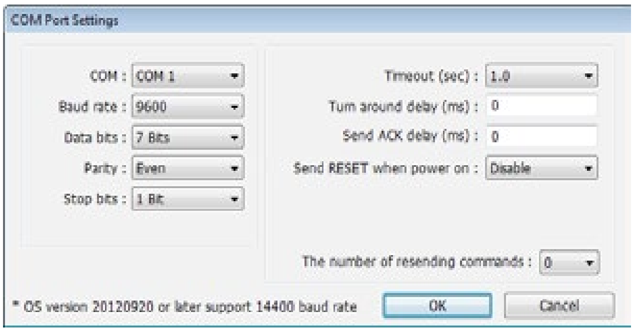
Parameters in this tab determine the attributes of each device connected with HMI. The device can be a Local / Remote HMI / PLC. When creating a new project file, there is a default device “Local HMI” which indicates the HMI that will be updated and programmed. To change the relevant device settings, click (System Parameter Settings) » (Settings) to open (Device Properties) dialog box.

5.2.1. How to Control a Local PLC

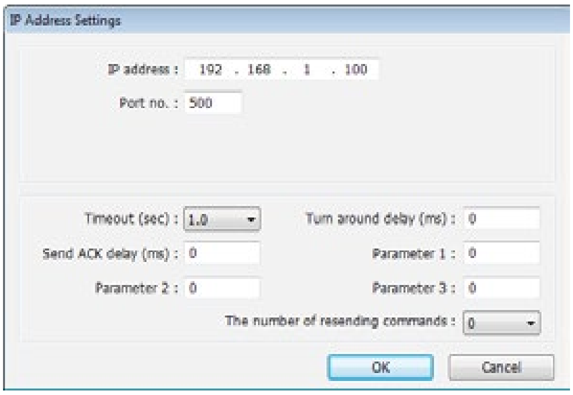
“Local PLC” means the PLC is connected to the local HMI. To control/connect a Local PLC, add this type of device first. Click (System Parameter Settings) » (New) to open (Device Properties) dialog box. For example, when connecting “Mitsubishi FX0s/FX0n/FX1s/FX1n/FX2” as a Local PLC:

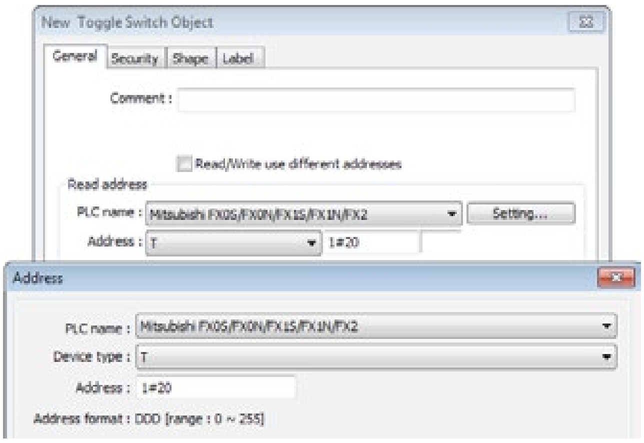


Setting	Description
Name	The name of the device.
HMI / PLC	In this example the device used is a PLC, so select (PLC).
Location	Select (Local) or (Remote). In this example the PLC is connected to the Local HMI, so select (Local).
PLC type	Select the type of the PLC.

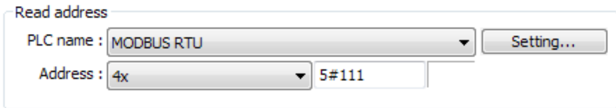
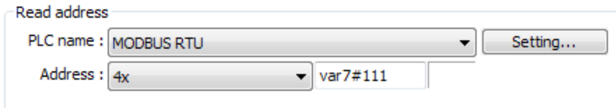
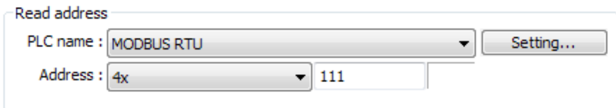
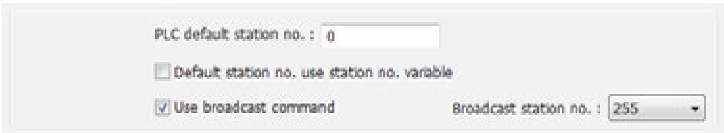
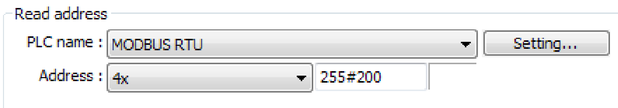
<table>
<tr>
<td>PLC I/F</td>
<td>
The available PLC interface: (RS232), (RS485 2W), (RS485 4W), (Ethernet), (USB) and (CAN Bus).
▪ If the interface used is (RS232), (RS485 2W) or (RS485 4W), configure communication parameters by clicking (Device Properties) » (Settings) and then (Com Port Settings) dialog box opens.


Timeout
If the communication has been disconnected for more than preset time limit configured in (Timeout) (in sec), Window No. 5 will pop up and show “PLC No Response” message.

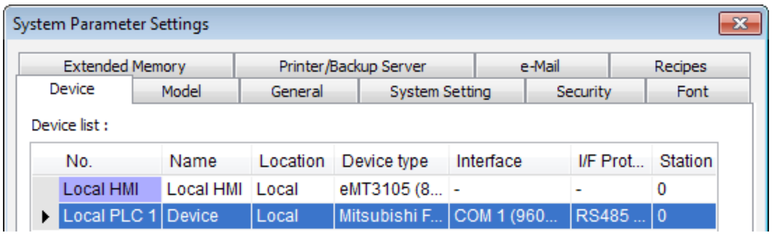
Turn around delay
While sending the next command to PLC, HMI will delay the sending according to the time interval set in (Turn around delay). This may influence the efficiency of the communication between HMI and PLC. Default value is “0”.
Note: if the PLC used is SIEMENS S7-200 Series, it is recommended to assign “5” to (Turn around delay) and “30” to (Send ACK delay).
▪ If the interface used is (Ethernet), click (Device Properties) » (Settings) and the (IP Address Settings) dialog box opens. Please set correct PLC IP address and port number.


▪ If the interface is (USB), no further setting is required. Please check the settings in (Device Properties).
▪ If the interface is (CAN (Controller Area Network) Bus), please see “PLC Connect Guide” for “CANopen” and import the .eds device file.
</td>
</tr>
<tr>
<td>PLC default station no.</td>
<td>
The default station number for PLC address if the PLC station number is not included in the address. PLC station no. can be set in PLC address. The address format: ABC#DEFGH
ABC stands for PLC station number and ranges from 0 to 255. DEFGH stands for PLC address. And the “#” sign separates the station number and the address. As shown in the following figure, the data is read from PLC station number 1, and address T-20.


</td>
</tr>
</table>

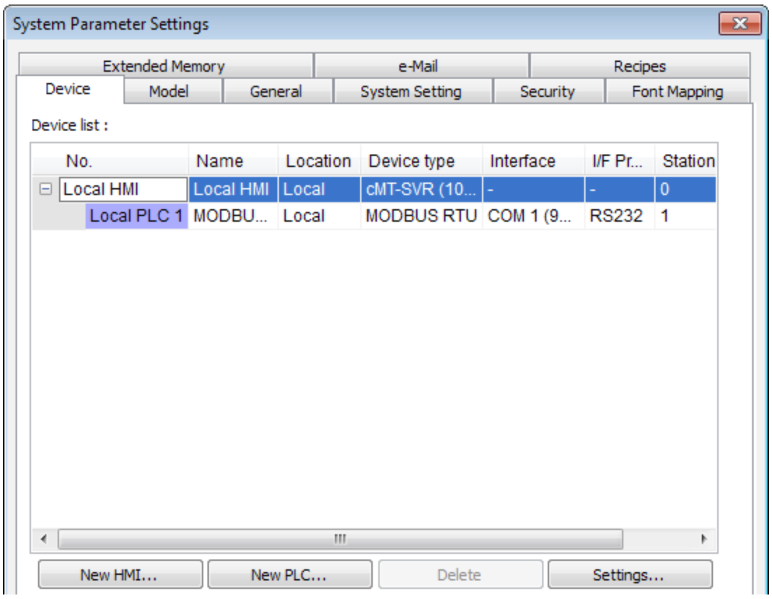
Default station no. use station no. variable	Use the station number variables as the default PLC station number. Select one from LW-10000 to LW-10015 (var0 to var15) as the station number variables. If the station no. is not specified in PLC address, the station number will be determined by the station no. variable. For example, if var3 is set for default station no.:  The followings demonstrate some examples: ▪ The PLC station number is "5"  ▪ The PLC station number is determined by var7 (LW-10007)  ▪ PLC address is set to "111", since PLC station no. is not specified, and the default station no. is var3, the PLC station no. is determined by var3 (LW-10003) 
Use broadcast command	When (Use broadcast command) check box is selected, please fill in (Broadcast station no.) according to the broadcast station number defined by PLC. When HMI sends a broadcast command to the station number set here, PLC will only receive the command and not reply to HMI.  As shown in the following figure:  When HMI sends a command to address 255#200, all the PLCs will receive this command and will not reply. Only PLCs that support broadcast command can use this feature.
Interval of block pack (words)	If the interval between read addresses of different commands is less than this value, the commands can be combined to one. The combining function is disabled if this value is set to "0". For example, the interval value is set to "5", to read 1 word from LW-3 and 2 words from LW-6 respectively (read from LW-6 to LW-7), since the interval of addresses between LW-3 and LW-6 is less than 5, these two commands can be combined to one. The result is to read 5 consecutive words from LW-3 to LW-7. Note: the maximum size of command combination data must be less than (Max. read-command size).
Max. read - command size (words)	The maximum data size to read from the device at one time. Unit: word
Max. write - command size (words)	The maximum data size to write to the device at one time. Unit: word.

After all settings are completed, a new device named “Local PLC 1” is added to the (Device list).

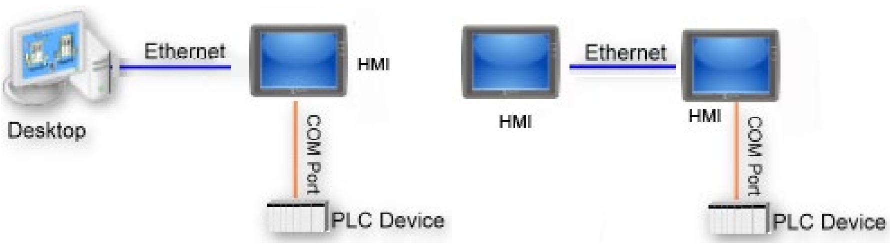


Note

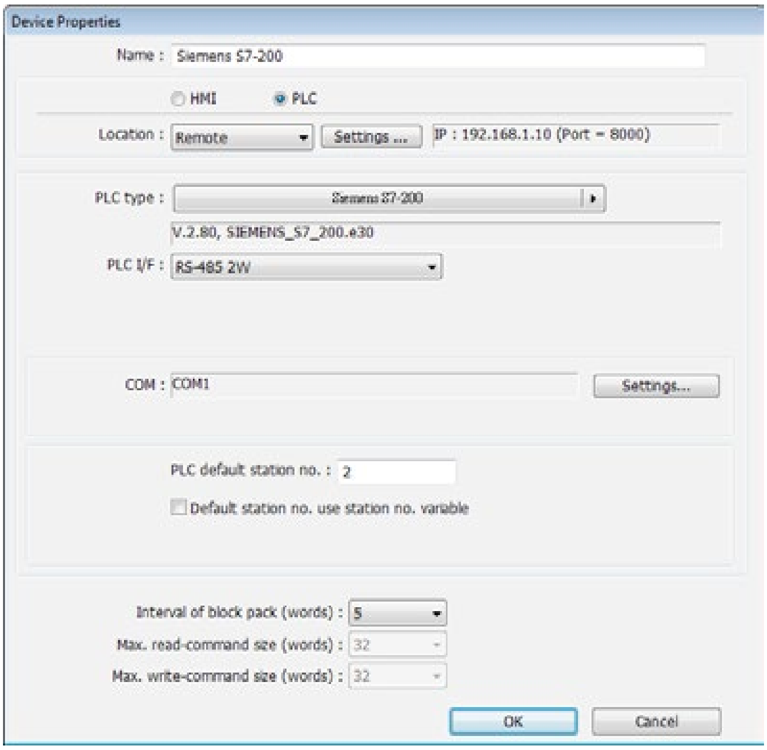
When using cMT-SVR, select “Local HMI” in (System Parameter Settings) dialog box and then click (New PLC) to add a “Local PLC 1” under “Local HMI”.

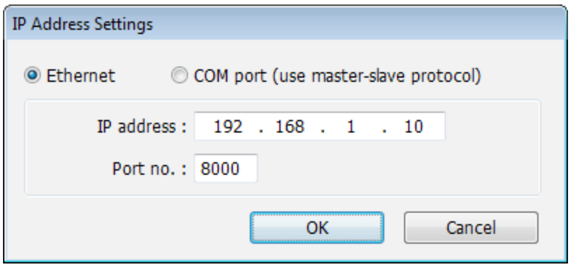


5.2.2. How to Control a Remote PLC

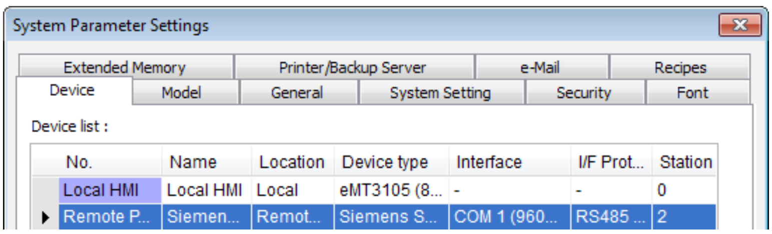


“Remote PLC” is a PLC being connected to a remote HMI. To control a remote PLC, add this type of device first. Please click (System Parameter Settings) » (New) to open (Device Properties) dialog box. For example, use SIEMENS S7-200 as the Remote PLC:

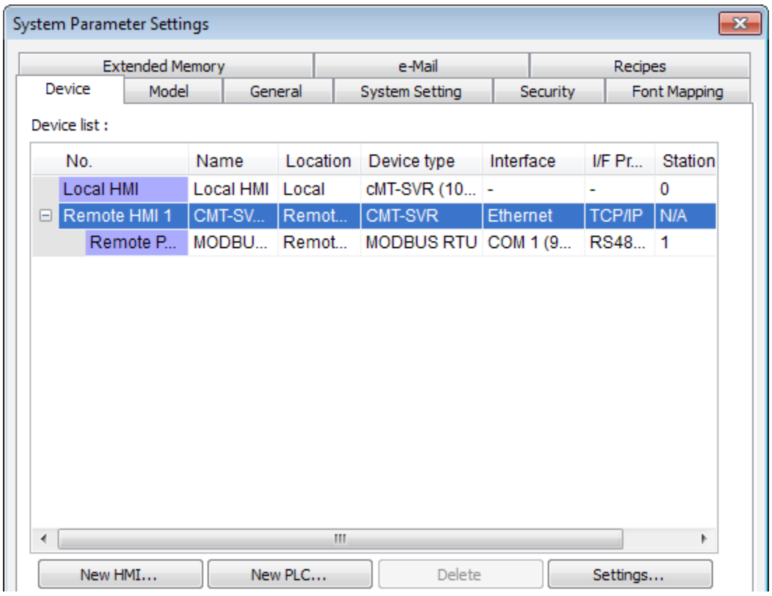


Setting	Description
HMI or PLC	In this example, the device used is a PLC, so select (PLC).
Location	Select (Local) or (Remote). In this example the PLC is connected to Remote HMI, so select (Remote). Set the IP address and port number of the Remote HMI by clicking (Settings) next to (Remote). 
PLC type	Select the type of the PLC.
PLC I/F	The interface used for Remote PLC. If the remote PLC uses a COM port, select (RS232), (RS485 2W) or (RS485 4W).
COM	Set the correct COM port used by the Remote PLC.
PLC default station no.	Set the station number of Remote PLC.

After all settings are completed, a new device named “Remote PLC 1” is added to the (Device list).

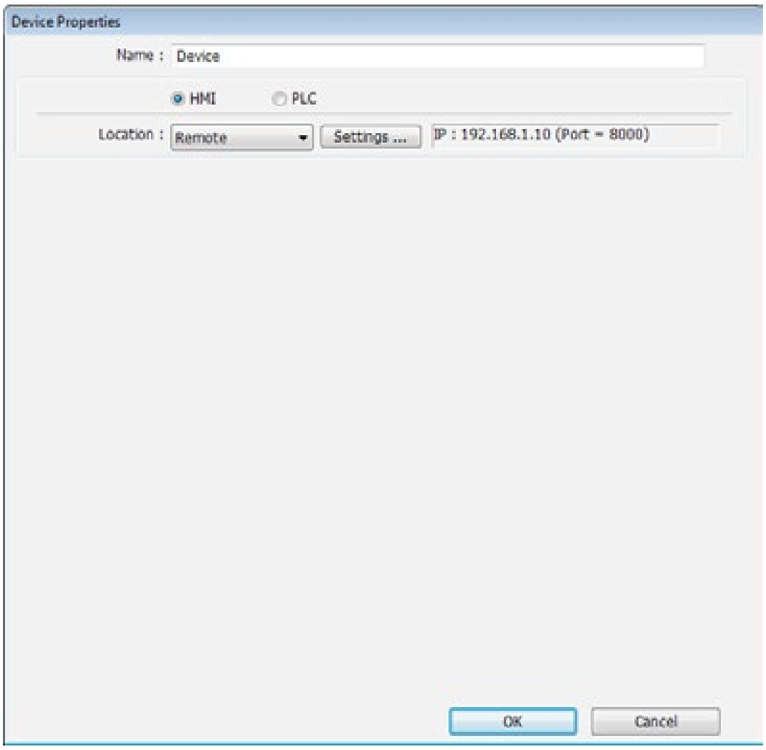


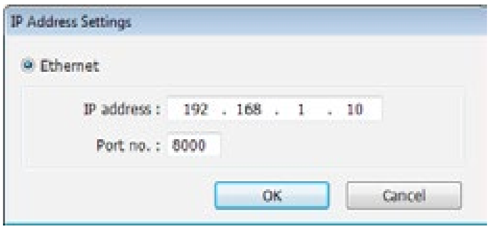
Note
When using cMT-SVR, select the defined “Remote HMI 1” in (System Parameter Settings) dialog box and then click (New PLC) to add a “Remote PLC 1” under “Remote HMI 1”.



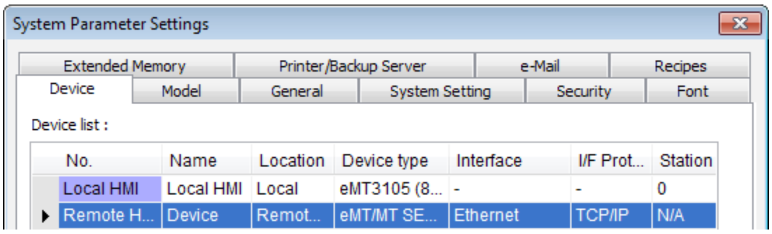
5.2.3. How to Control a Remote HMI

“Remote HMI” is the HMI other than “Local HMI”, and PC is also a “Remote HMI”. To control a Remote HMI, add this type of device first. Click (System Parameter Settings) » (New) to open (Device Properties) dialog box as shown in the following figure:

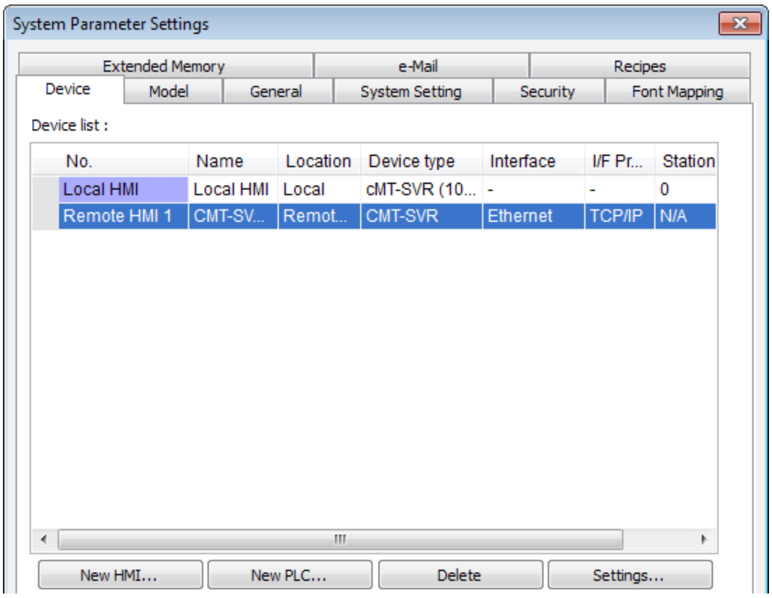


Setting	Description
HMI or PLC	In this example, the device used is a HMI, so select (HMI).
Location	Select (Local) or (Remote). In this example Remote HMI is used, select (Remote). Set the IP address and port number of the Remote HMI by clicking (Device Properties) » (Settings). 

After all settings are completed, a new device named “Remote HMI 1” is added to the (Device list).

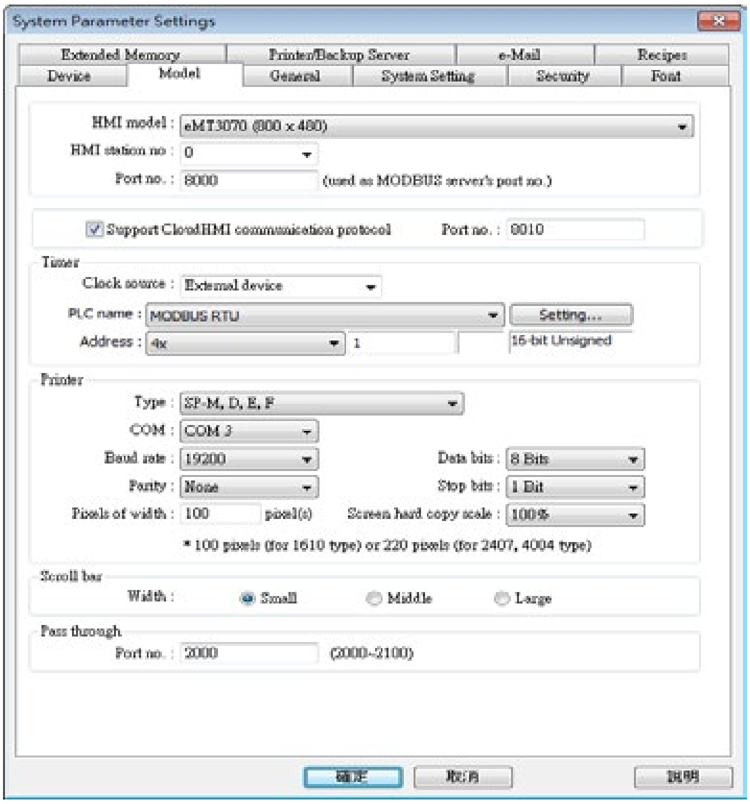


Note
When using cMT-SVR, in (System Parameter Settings) dialog box click (New HMI) to add a “Remote HMI 1”.

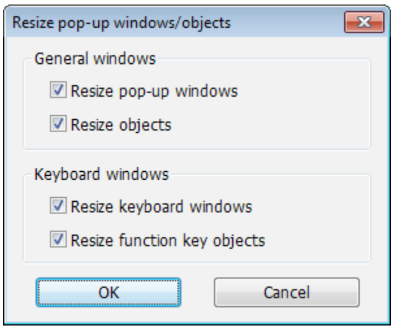


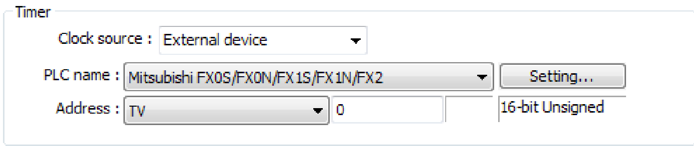
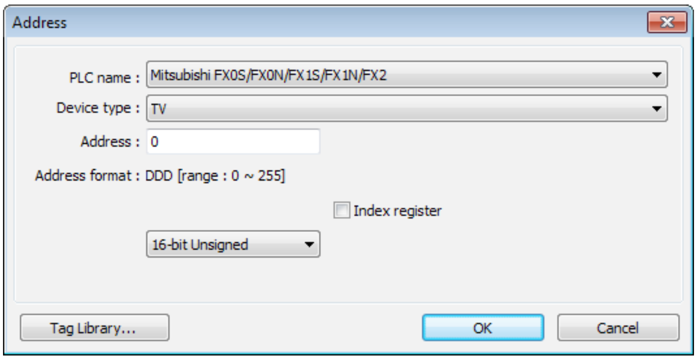
5.3. MODEL

Configure the (HMI model), (Timer), (Printer) and (Scroll bar) settings.



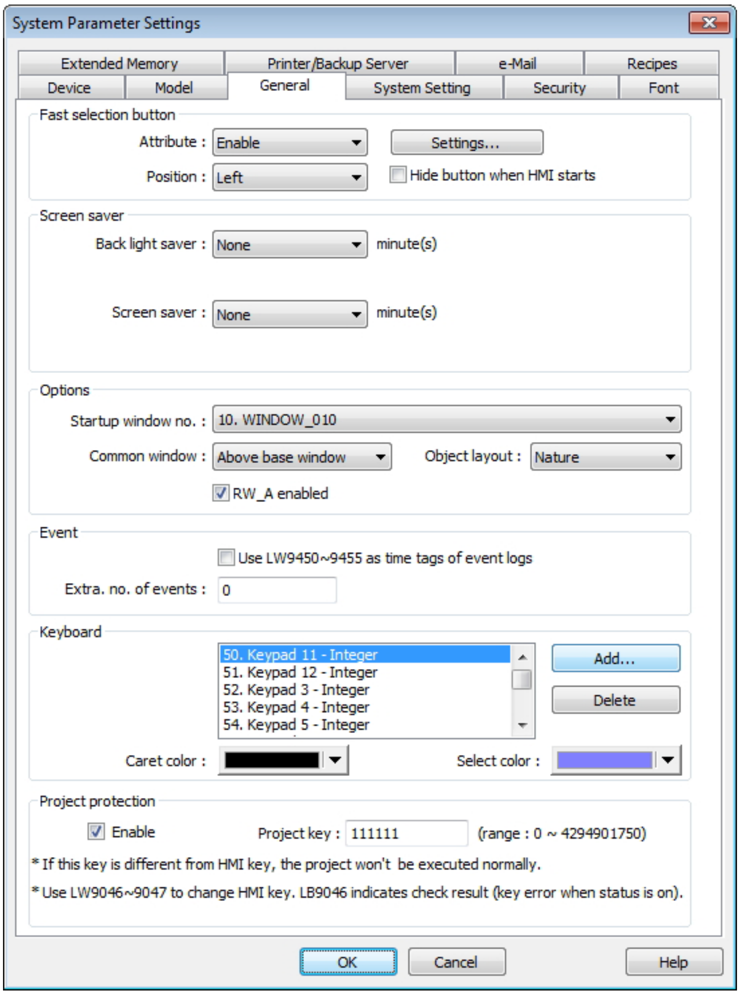
	Note When using cMT-SVR, configure only (HMI model), (HMI station no.), and (Port no.).

Setting	Description
HMI model	Select the HMI model to use. If the HMI model is changed, the (Resize pop-up windows / objects) dialog box will pop up. Select required adjustment and click (OK). In most cases, select all options. 
HMI station no.	Set the station number of current HMI. The default value is “0”.
Port no.	Set the port number of current HMI. It is also used in Modbus server. The default value is “8,000”.
Support CloudHMI communication protocol	Supports CloudHMI using this HMI as a Remote HMI. The Remote HMI port number in cMT-SVR project must be identical to the port number set here.

Timer	**Clock source** Set the source device of the clock/time information. It is used by (Data Sampling), (Event Log), etc. ▪ If (HMI RTC) is selected, the time information comes from the internal clock of the HMI ▪ If (External device) is selected, the time information comes from an external device. The address of the source device must set correctly. As shown below, the time information is from “TV” address type of the “Local PLC”. The addresses of “TV” start from 0 and contain 6 consecutive words and each of them store the following information: TV 0 → Second (range: 0~59) TV 1 → Minute (range: 0~59) TV 2 → Hour (range: 0~23) TV 3 → Day (range: 1~31) TV 4 → Month (range: 1~12) TV 5 → Year (range: 1970~2037)  
Printer	**Type** A printer can be connected with the HMI. The HP PCL Series printer is connected through USB interface while other printers through a COM port. For more information, see “23 HMI Supported Printers”. If the printer is connected through (COM), configure the parameters correctly. If the printer type is (SP-M, D, E, F), the (pixels of width) has to be set accurately, i.e. the set pixel(s) cannot exceed printer’s default setting, or the HMI will fail to print data.
Scroll bar	Set the width of Scroll Bar, when the size of the object is too small to display the contents, a scroll bar is displayed in the object. This feature can be applied to objects that allow scrolling, such as Alarm Display, Event Display, History Data Display, and Option List.
Pass through	Set the port number for Pass-through communication.

5.4. GENERAL

Configure the properties related to screen display.



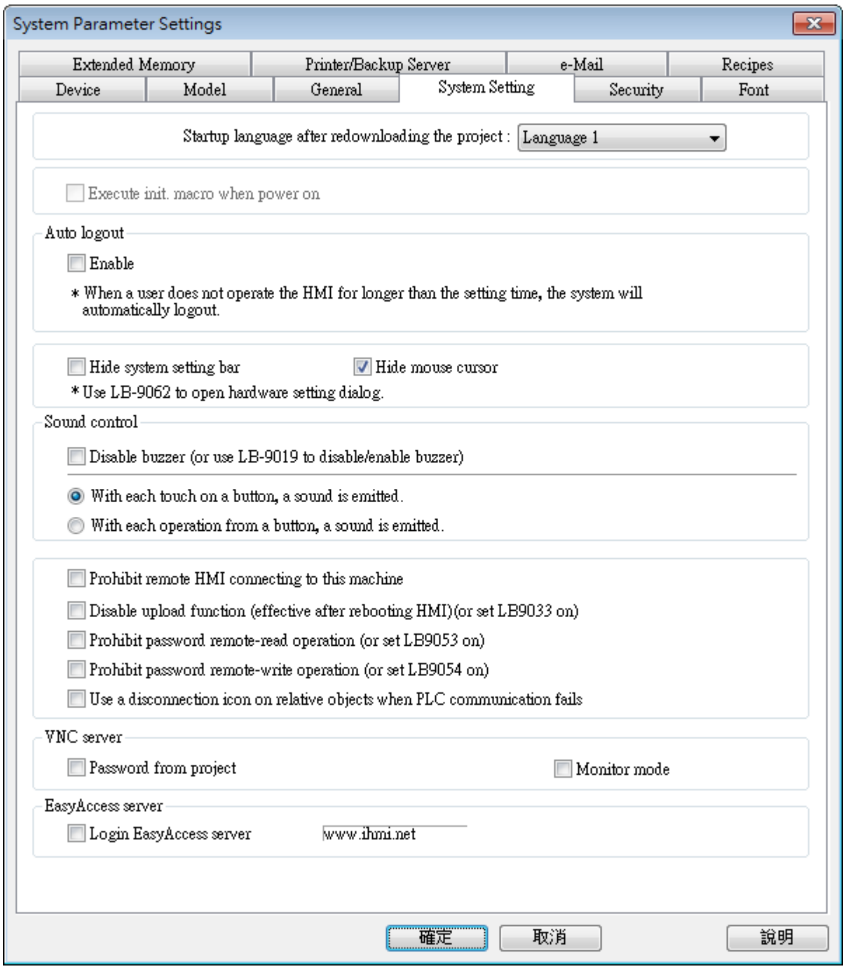
Setting	Description
Fast selection button	Setting the attributes for fast selection button for Window No. 3. To use the fast selection button, create Window No. 3 first. **Attribute** Enable or disable fast selection window. Select (Enable) and click (Settings) to set the attributes, including color and text of the button. **Position** Select the button position on the screen. If (Left) is chosen, the button will show up in at bottom left side of the screen; if (Right) is chosen, the button will show at the bottom right side of the screen.
Screen saver	**Back light saver** If the screen is left untouched and reaches the time limit set here, the back light will be turned off. The unit is minute. Back light will be on again once the screen is touched. If (none) is set, the back light will always be on. **Screen saver** If the screen is left untouched and reaches the time limit set here. The current screen will automatically switch to a window assigned in (Saver window no.).The setting unit is minute. If (none) is set, this feature is disabled. **Saver window no.** To assign a window for screen saver.

Option	**Startup window no.** Designate the window shown when start up HMI. **Common window** The objects in the common window (Window No. 4) will be shown in each base window. This determines that the objects in common window are placed above or below the objects in the base window. **Object layout** If (Control) mode is selected, when operating HMI, (Animation) and (Moving Shape) objects will be displayed above other kinds of objects neglecting the sequence that the objects are created. If (Nature) mode is selected, the display will follow the sequence that the objects are created, the first created will be displayed first. **RW_A enabled** Enable or disable recipe data RW_A. Enable this, the objects can then control RW_A .The size of RW_A is 64 K.
Event	**Extra no. of events** The default number of the events in the system is 1,000. For additional number of events, modify this setting. The maximal is 10,000.
Keyboard	The window number in which the keyboard is placed. When using Numeric Input or ASCII Input objects, the type of keyboards can be selected. Up to 32 keyboards can be added. To design a keyboard, a window should be designated for creating it. Press (add) after creating, and add the window to the list. See "12 Keyboard Design and Usage". **Caret color** Set the color of caret that appears when entering data in Numeric Input and Word Input objects.
Project protection	Projects can be restricted to be executed by a specific HMI. See "30 Project Protection".

Note
cMT Series does not support (Fast selection button) and (Keyboard).

5.5. SYSTEM SETTING

(System Setting) is used to configure different features of EasyBuilder Pro.



Some features are duplicated from system registers, such as (Hide system setting bar (LB-9020)), (Hide mouse cursor (LB-9018)), (Disable buzzer (LB-9019)), (Prohibit remote HMI connecting to this machine (LB-9044)) and (Disable upload function (LB-9033)). Users can also set these features via system tag.
To select a system tag, select (Address) » (System tag) check box when adding a new object and then select the (Device Type).
To browse all the system tags, Select (Library) » (Tag) » (System) from the main menu of EasyBuilder Pro.

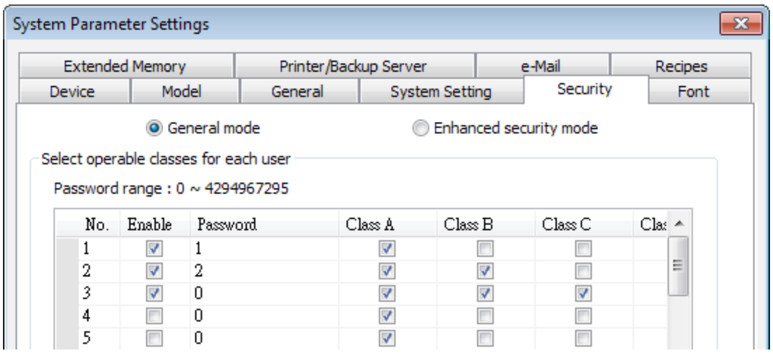
Setting	Description
Startup language after redownloading the project	Set the language to use when HMI starts after the project is redownloaded.
Execute init. MACRO when power on	Designate the macro to be executed when HMI power on.
Auto logout	If leaving HMI untouched for longer than the set time, the objects protected by security classes will not be able to operate. The user ID and password must be entered again to unlock it.
Hide system setting bar	Hide the system setting bar in the bottom right corner of the HMI screen.
Hide mouse cursor	Hide the mouse cursor in HMI screen.
Sound control	**Disable Buzzer:** mute HMI. **With each touch on a button, a sound is emitted:** a sound is emitted when touching a button. **With Each operation from a button, a sound is emitted:** when the (Min. press time) is specified, there may be a time gap between touching the object and the action of the object. This setting can control the timing to emit a sound.
Prohibit remote HMI connecting to this machine	Prohibit the connection with a remote HMI. The remote HMI will not be able to control the local HMI.
Disable upload function (effective after rebooting HMI) (or set LB9033 ON)	Disable HMI to upload project, after downloading, HMI must be rebooted to disable uploading project.
Prohibit password remote-read operation (or set LB9053 ON)	Prohibit Remote HMI to read Local HMI.
Prohibit password remote-write operation (or set LB9054 ON)	Prohibit Remote HMI to write Local HMI.
Use a disconnection icon or relative objects when PLC communication fails	If selected, displays a disconnection icon on relevant objects when failing to communicate with PLC. This icon will be shown in the lower right corner of the object. 
VNC Server	Set the password to log in VNC server. If (Monitor mode) check box is selected, the HMI connected via VNC can only be monitored but not controlled.
LW protection **RW protection**	If select (Disable LW/RW remote-write) check boxes and set the protect range in (LW/RW range), values within the protected range cannot be adjusted using Remote HMI.
Easy access server	Through this technology, users can easily access to any HMI connected to the internet and operate them on PC just like holding touch screen in hand. Easy Access does not transmit updated graphic images directly but only the real time data. This makes transmission really quick and efficient.

Note
cMT Series does not support VNC Server.

5.6. SECURITY

Parameters in this tab configure the user passwords and security classes. There are two authentication modes: General Mode and Enhanced Security Mode.
For more information, see “10. User Password and Object Security”.

5.6.1. General Mode

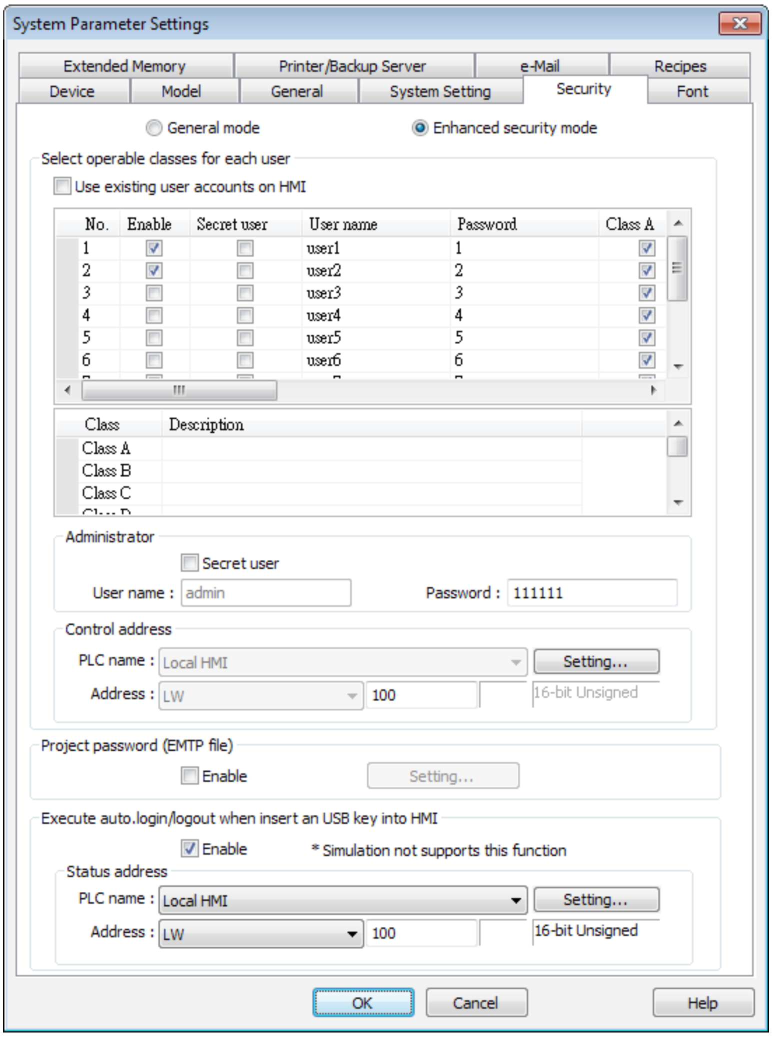


Up to 12 sets of user and password are available. Password should be one non-negative integer. Once the password is entered, the objects that the user can operate are classified.
There are six security classes available: A to F.
If (None) is selected for an object, every user can access this object.
For example, when the security class of User No. 3 is set as the preceding figure, User No. 3 could only access objects of classes A, B, C and "none".

Users can set password to protect the project (.emtp) files. The password set here must be entered when editing the project file.
Select (Enable) then click (Setting) to set the password.
Before editing a project, a popup window is shown for entering the password. Only when the password is correct can the user edit this project.

5.6.2. Enhanced Security Mode

At most 11 users can be set here. An (Administrator) user is provided in this mode. An (Administrator) has all privileges and can operate all object classes. User passwords must be alphanumeric and each user can have up to 12 classes: A to L.



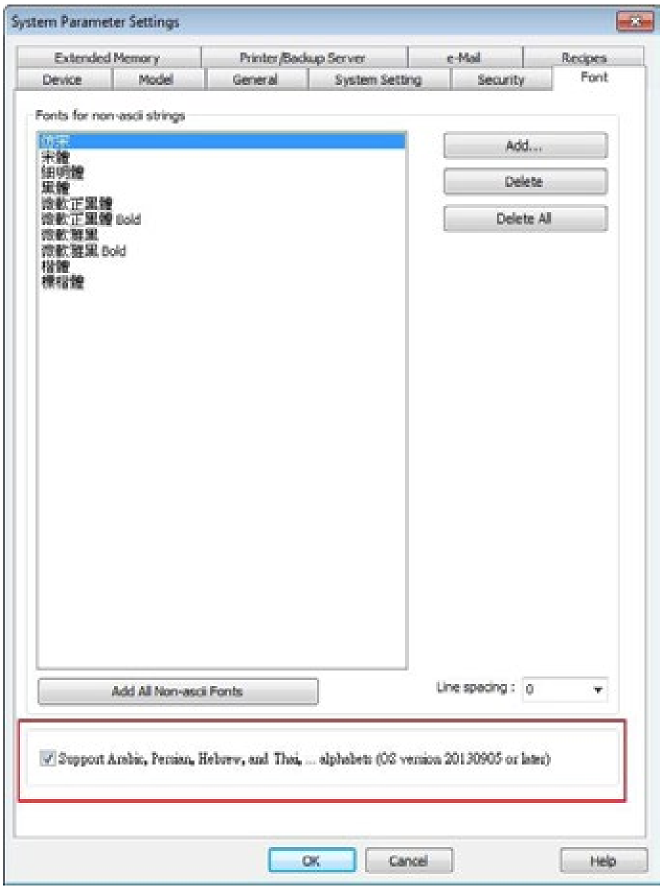
Setting	Description
Select operable classes for each user	When selecting (Use existing user accounts on HMI) check box, the operable objects for each user will be decided by the account settings in HMI.
Administrator	Default administrator account, cannot be deleted, has all privileges, and cannot be changed. Enhanced Security Mode can be used with Option List object. It displays the account names and privileges. If (Secret user) is checked, the account names and privileges will be hidden in Option List.
Control address	An address for users to manage the accounts directly on HMI.
Execute auto. login/logout when insert an USB key into HMI	This feature allows automatic login / logout using an USB security key. The login / logout status will be written into a designated address. Insert the USB disk to HMI to log in, and remove the USB disk to log out. The result codes of login / logout: 0x00: no action, 0x01: login succeeds, 0x04: login fails, 0x08: logout succeeds, 0x10: logout fails.

Note
cMT Series only supports Enhanced Security Mode, but does not support (Execute auto. Login/Logout when insert an USB key into HMI.)

5.7. FONT

5.7.1. eMT, iE, XE, mTV Series

Parameters in (Font) tab determine the non-ASCII fonts.



The non-ASCII fonts are listed here. When using non-ASCII characters or double byte characters (including Simplified or Traditional Chinese, Japanese, or Korean) which are not listed in (Fonts for non-ascii strings) table, EasyBuilder Pro will select a font from the list to substitute for it automatically.

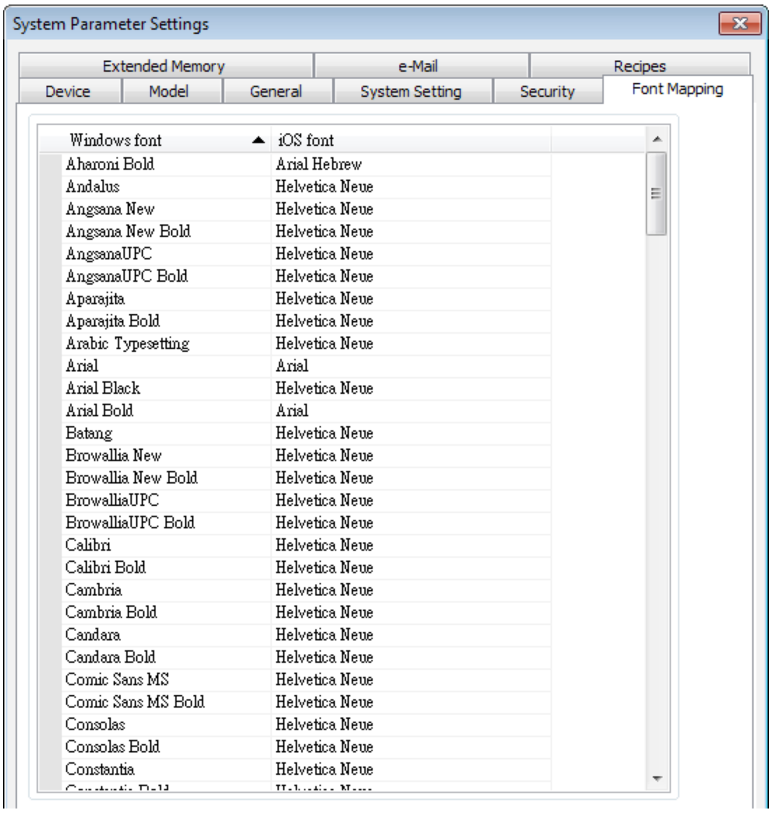
The non-ASCII fonts in Windows can be added to the (Fonts for non-ascii strings) table.

Decide the space between lines in the text in (Line spacing) field.

Select (Support Arabic, Persian, Hebrew, and Thai, … alphabets) check box to correctly display these alphabets.

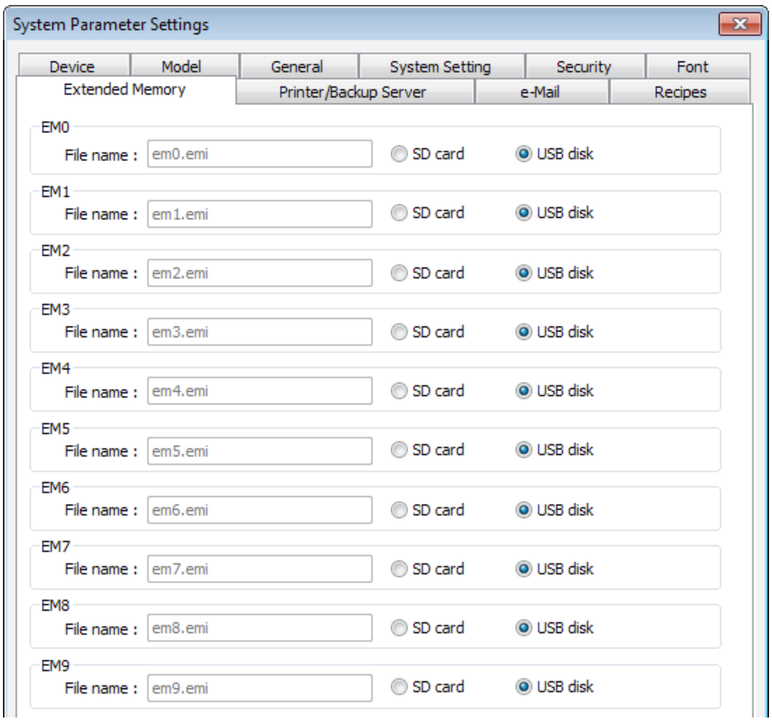
5.7.2. cMT Series

This tab lists the fonts used in Windows and the corresponding fonts displayed on iPad.

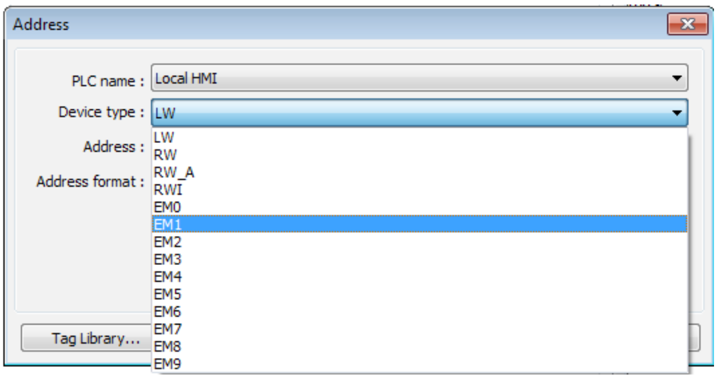


5.8. EXTENDED MEMORY

Parameters in this tab determine the location of the extended memory.



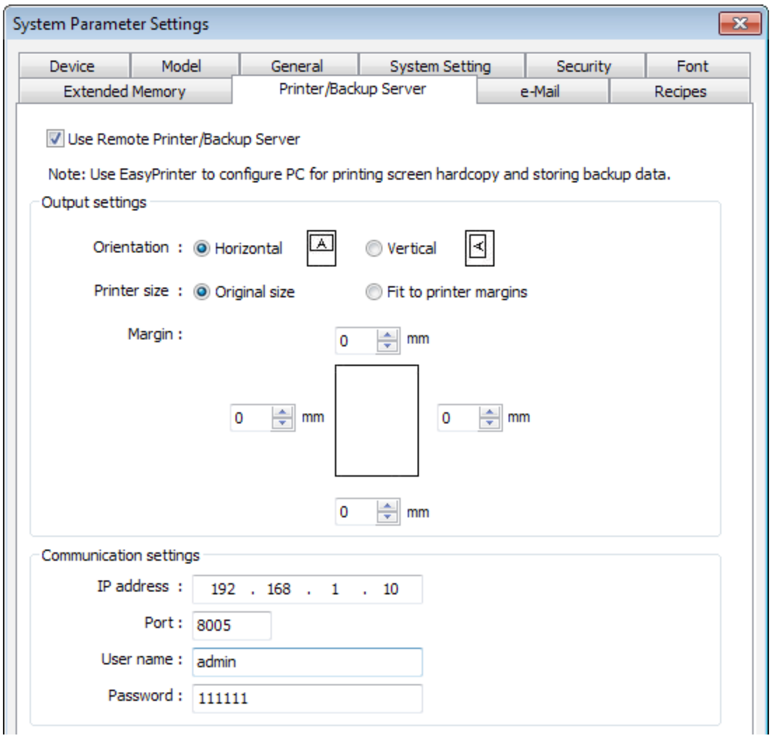
Extended Memory is numbered from EM0 to EM9. It works in a way similar to other device types (i.e. LW or RW address). Users can simply select from (Device type) list while adding a new object. Size of each extended memory is up to 2G word.



Extended memories are saved as files in (SD card) or (USB disk). (EM0) to (EM9) are saved as “em0.emi” to “em9.emi” respectively. Users can use RecipeEditor.exe to open these files and edit the data in the extended memory.
Data in extended memory will not be erased when power is cut, which means next time when start up HMI again, data in the extended memory remains the same as before power off. This is similar to recipe data (RW, RW_A). The difference is that users can specify the location to store data. (SD card, USB disk)
When the device of extended memory does not exist and to read data in it, the data content will be “0”; to write data to a device that does not exist, the “PLC no response” message will be shown in HMI.
Users can insert or remove the external device to or from HMI without cutting the HMI power to update or take data in extended memory.

5.9. PRINTER / BACKUP SERVER

Configure remote printer / backup server.

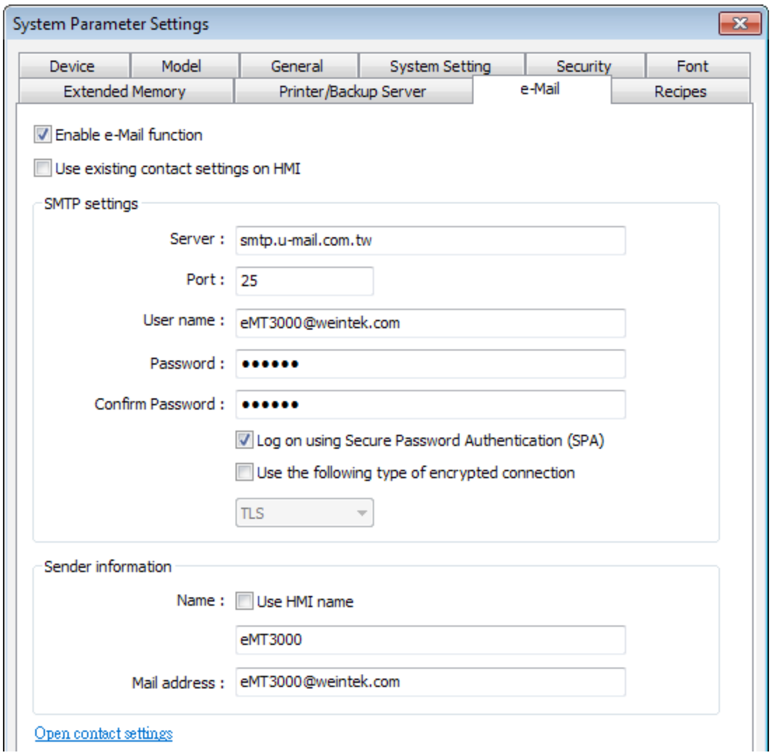


Setting	Description
Output settings	**Orientation** Set how will words or pictures be printed out, (horizontal) or (vertical). **Printer size** Set to print out in (Original size) or to (Fit to printer margins). **Margin** Set the top, bottom, right and left margin width.
Communication settings	**IP address** Assign the IP address of the printer via network. **(Port), (User name), (Password)** Specify the data to log in printer. Port can be set from 1 to 65535. Maximum length of user name or password is 12 characters.

☞ For more information, see “26 EasyPrinter”.

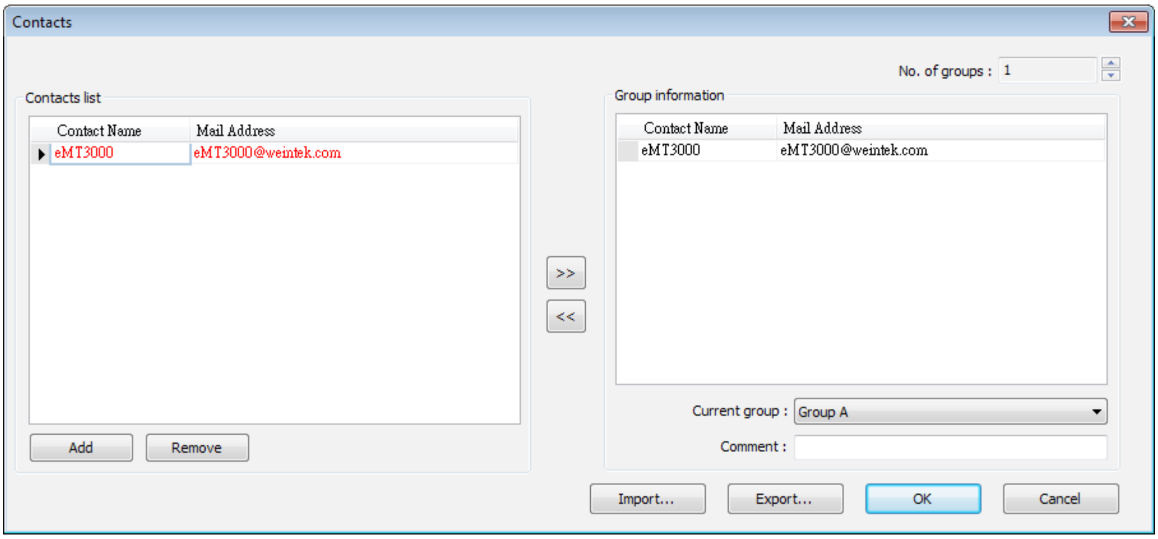
5.10. E-MAIL

Select (Enable e-Mail function) check box to configure the parameters.
If (Use existing contact settings on HMI) check box is selected, the system will use the contact settings in HMI.



Setting	Description
SMTP settings	**Server:** set SMTP Server. **Port:** set communication port. **User name:** set e-mail address. **Password:** set e-mail password. **Confirm Password:** confirm e-mail password. **Log on using Secure Password Authentication (SPA):** decide whether SPA is needed when log in e-mail. **Use the following type of encrypted connection:** decide whether the encrypted connection (TLS, SSL) is needed when sending e-mail.
Sender information	**Name:** specify the sender name or use HMI name. **Mail Address:** setting e-mail address.

Click (Open Contact settings) to open the following dialog to edit contacts:

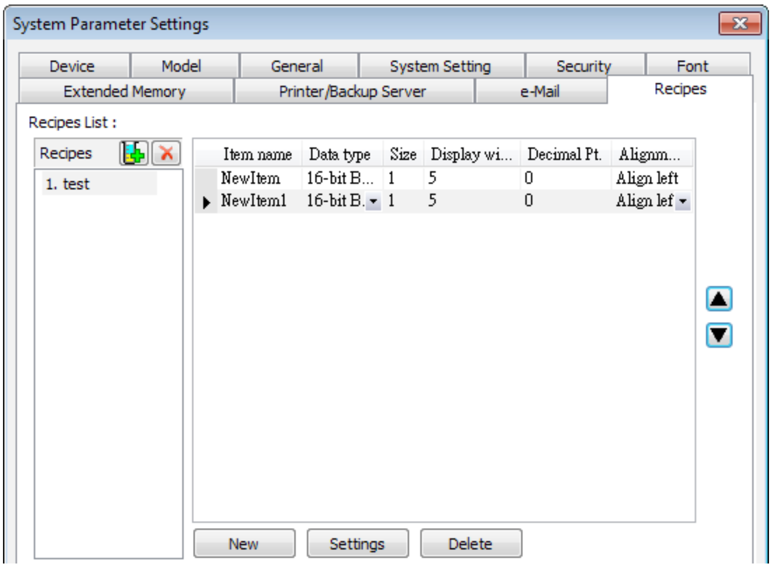


Setting	Description
Contact list	Add or remove contacts from the list.
Group Information	Group up contacts. **No. of groups:** set no. of contact groups, according to the number, the groups are named from A to P and up to 16 groups can be set. **Current group:** displays the group that includes the contacts above. **Comment:** enter a description for the current group.

☞ For more information, see "7 Event Log".

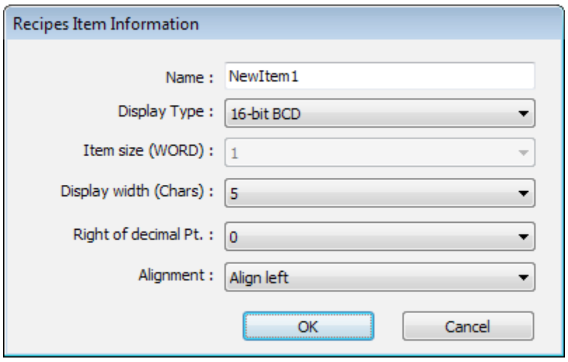
5.11. RECIPES

Configure the recipe list for (Recipe Database).



Setting	Description
Recipes List	Add or delete a recipe.
New	Add a new item.
Settings	All the item information can be user-defined and modified, see the following description.
Delete	Delete the selected items.

Click (Settings):



Setting	Description
Name	Enter recipe item name.
Display type	Setting item data type.
Item size (WORD)	Setting the size of the item.
Display with (Chars)	Setting the number of characters of the item to be displayed.
Right of decimal pt.	Setting the decimal place when displaying data.
Alignment	Setting the alignment when displaying data. (Align left), (Align center), and (Align right) can be selected.

☞ For more information, see “24 Recipe Editor”.

6. WINDOW OPERATIONS

This chapter describes different types of windows and how to create, set and delete a window.

6.1. OVERVIEW

A window is a basic element in a project. With a window, all kinds of information like objects, pictures, and texts can be displayed on HMI screen. In total, 1997 windows numbered from 3 ~ 1999 in EasyBuilder Pro can be built and edited.

6.2. WINDOW TYPES

There are 4 types of windows, each with different functions and usages:
- Base Window
- Fast Selection Window
- Common Window
- System Message Window

6.2.1. Base Window

The most frequently used window, except for main screen, it can also be:
- A background of other windows
- A keyboard window
- A pop-up window of Function Key object
- A pop-up window of Direct Window and Indirect Window objects
- A screen saver

	Note Base Window should be in same size as the HMI screen. Therefore, the resolution of the base window should be set to the resolution of HMI.

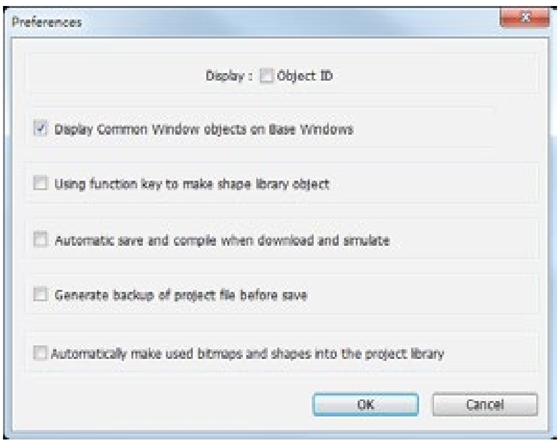
6.2.2. Fast Selection Window

Window no. 3 is the default Fast Selection Window. This window can co-exist with base window. Generally, it is used to place the frequently-used buttons on the lower-left side or the lower-right side on the screen. Please create window no. 3 first, and set the relevant properties in (System Parameter Settings) » (General) tab. Apart from showing or hiding fast selection window with the button on the screen, there are system registers to do so:
- (LB-9013) Fast Selection window control (hide(ON)/show(OFF))
- (LB-9014) Fast Selection button control (hide(ON)/show(OFF))
- (LB-9015) Fast Selection window/button control (hide(ON)/show(OFF))

	Note cMT-SVR Series does not support Fast Selection Window.

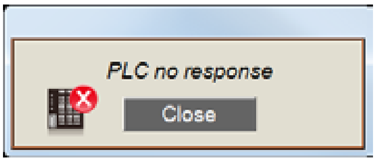
6.2.3. Common Window

Window no. 4 is the default Common Window. Objects in this window will be displayed in other base windows, not including pop-up windows. Therefore, the common objects in different windows are often placed in common window.
When operating HMI, select (Function Key) » (Change common window) to change the source of common window.
In menu (Option) » (Preferences) select whether or not to (Display Common Window objects on Base Windows) when editing a project. This can avoid overlapping objects in base window with objects in common window.

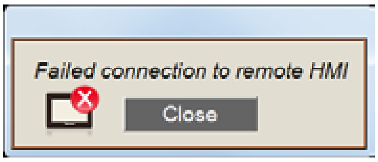


6.2.4. System Message Window

Windows No. 5, 6, 7, 8 are the default System Message Windows:



Window No. 5: PLC Response
When the communication between PLC and HMI is disconnected, this message window will pop up automatically right on the base window currently opened. This window can be disabled by system registers.



Window No. 6: HMI Connection
When failing to connect HMI with a remote HMI, this message window will pop up automatically.

Window No. 7: Password Restriction
When attempting to control an object without authorization, this window may pop up as a warning depending on the settings of the object.

Window No.8: Storage Space Insufficient
When HMI flash memory, USB disk or SD card run out of storage space, this message window will pop up automatically. (When the memory space is under 4 MB)

The following system registers can be used to check the free memory space in HMI, USB disk or SD card:

- (LW-9072) HMI current free space (K bytes)
- (LW-9074) SD current free space (K bytes)
- (LW-9076) USB disk current free space (K bytes)

To check if there is sufficient storage in the devices, the following system registers can be used. These addresses will set ON when the space is under 4 MB.

- (LB-9035) HMI free space insufficiency alarm (when ON)
- (LB-9036) SD card free space insufficiency alarm (when ON)
- (LB-9037) USB disk free space insufficiency alarm (when ON)

For more information, see “22 System Registers”.

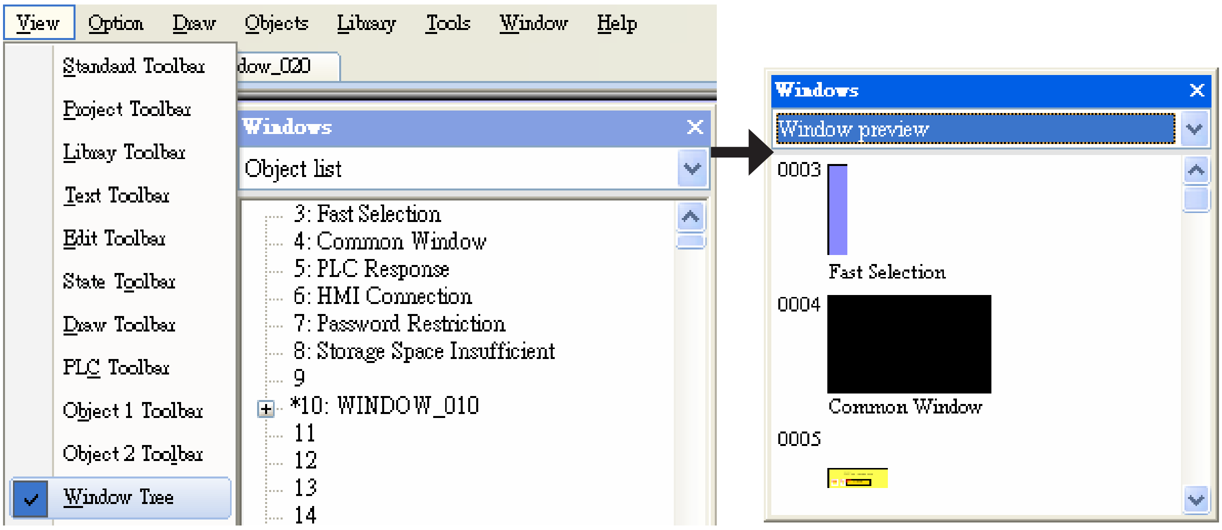
The text shown in windows no. 5 ~ 8 can be edited for easier reference.

Note

- A screen can display up to 16 pop-up windows simultaneous including System Message Window, Direct Window and Indirect Window.
- The system does not allow opening the same window with two Direct (or Indirect) Windows in one base window.
- Windows no. 3 to 9 are used by the system only, and windows no. 10 to 1999 can be edited based on actual usage.
- For cMT-SVR Series, only 1 pop-up window can be displayed simultaneously.

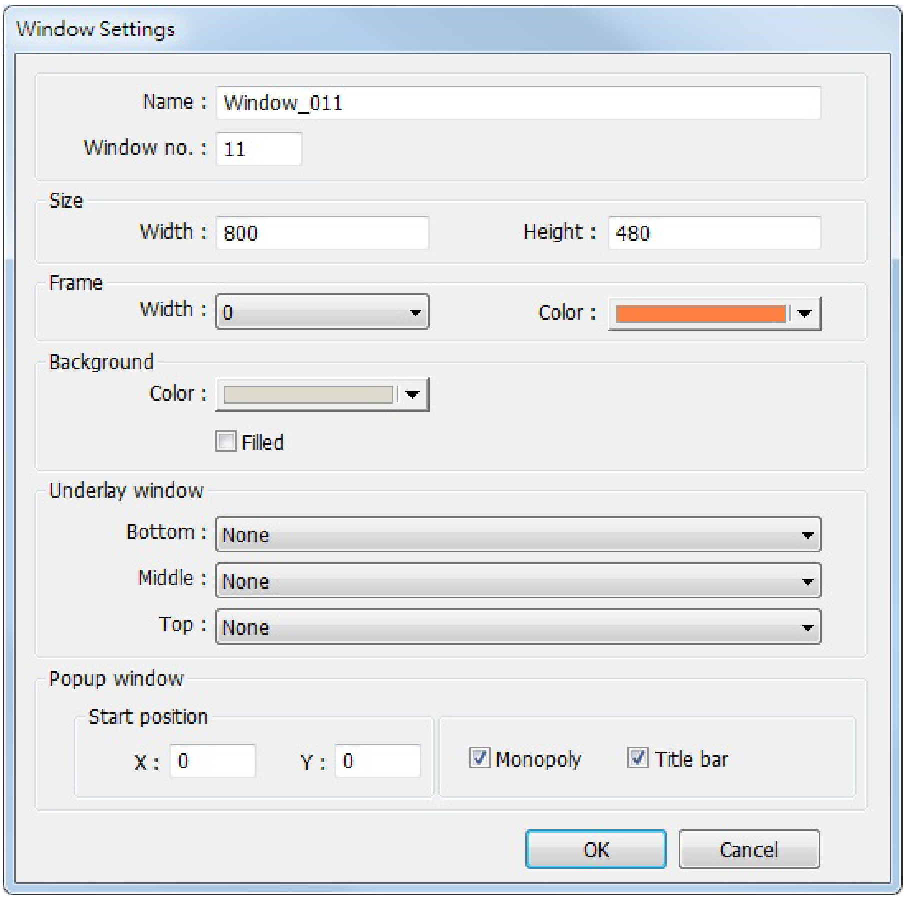
6.3. CREATE, SET AND DELETE A WINDOW

Check the existing windows in (View) » (Window Tree).
(Object list) displays window numbers and window names. Opened windows are marked with (*) sign. Press the (+) sign to see the object ID, address and description in this window.
(Window preview) displays the thumbnails of windows.



6.3.1. Creating and Setting a Window

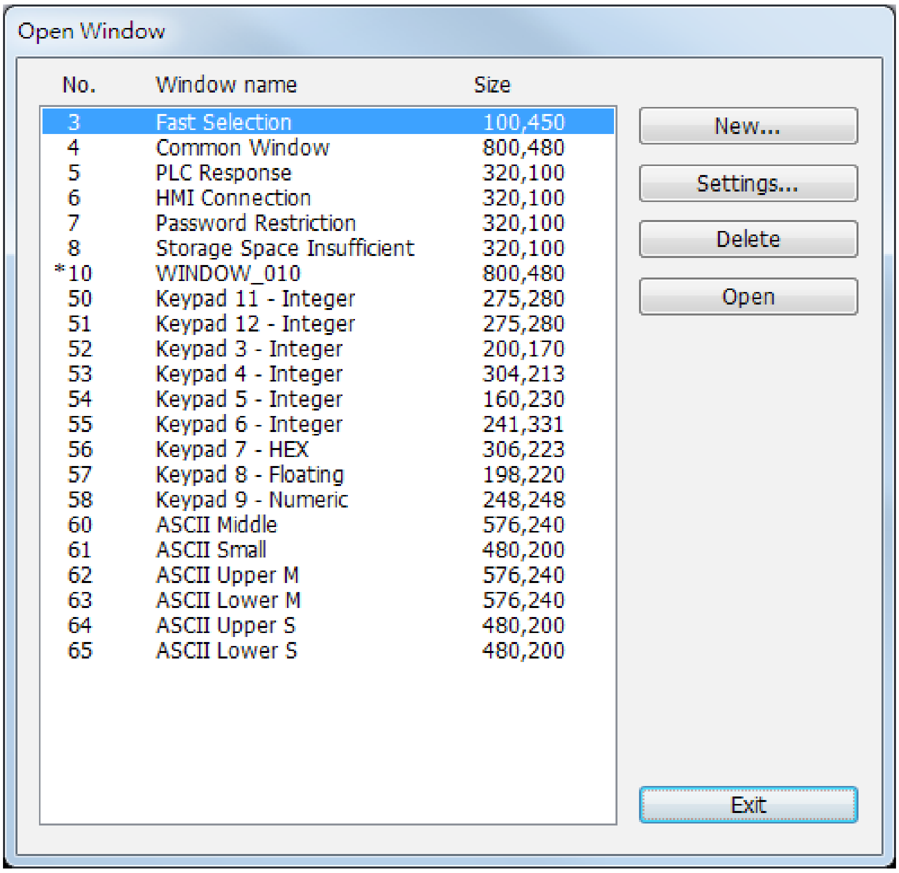
In window tree right click on a window number then select (New).



Setting	Description
Name	The name appears on the title bar and also in window tree.
Window no.	Can be 3 to 1999.
Size	Set the window size in accordance with the HMI resolution.
Underlay window	Underlay Window can be seen as an extra Common Window. When designing the project, some commonly used objects are used in some windows but not all. These objects can be placed in underlay window. Each base window can set three underlay windows as background, from (Bottom) to (Top). The objects in underlay windows are displayed in this order in base window.
Pop-up window	Base window can also be used as a pop-up window. Use (X) and (Y) to set the coordinates indicating where in the screen will this base window pop up. The origin of the coordinates is the upper-left corner of the window.
Monopoly	If the option is selected, when the base window pops up, no operations of other pop-up windows and background windows are allowed until the monopoly window is closed. If a base window is used as a keyboard window, “Monopoly” is automatically enabled.
Title bar	If the option is selected, a title bar appears on a system message window. This feature is only available for system message windows no. 5 to no. 8.

Note
- The objects in underlay window cannot be edited from the base window that displays them. To edit those objects, please open the underlay window where they are located.
- When the window number of the underlay window used by the base window is identical to the pop-up window, the pop-up window is disabled.
- When base window and pop-up window use the same underlay window, the objects in the underlay window cannot be displayed in pop-up window.

Or, from the main menu click (Window) » (Open Window) and then click (New) and select the type of the window and click (OK).



Ways to call up (Window Settings) dialog:
- Right click on the window number in the window tree and select (Settings)
- In (Window) » (Open Window) select the window then click (Settings)
- In the window, right click when no object is selected, and select (Attribute)

6.3.2. Open, Close and Delete a Window

The ways to open an existing window:

- Double click on the window number in the window tree
- In the window tree, select the window, right click, and then select (Open)
- In (Window) » (Open Window) select the window then click (Open)

The ways to close or delete an existing window:

- In the window tree, select the window; right click, then select (Close) or (Delete)
- In (Window) » (Open Window) select the window then click (Delete)
- To delete a window, please close it first

7. EVENT LOG

This chapter explains how to set and use Event Log.

7.1. OVERVIEW

The following are the basic steps to use Event Log:

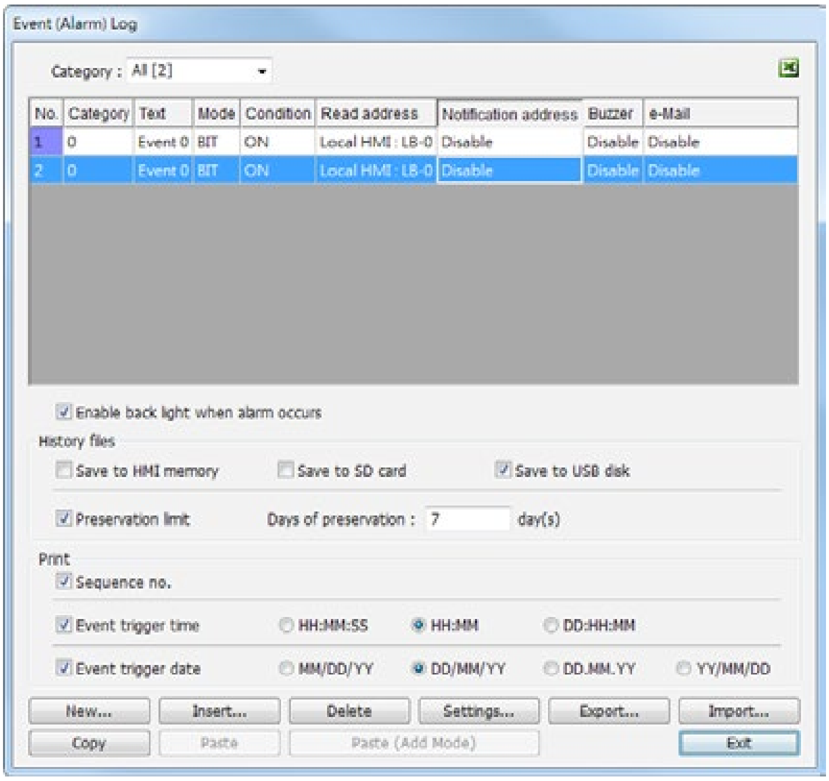
1. Define event content and trigger condition.
2. Trigger event according to the condition.
3. Save the event log to the specified device.
4. View the process of event by using the relevant objects.

This chapter will explain how to set and use Event Log.

7.2. EVENT LOG MANAGEMENT

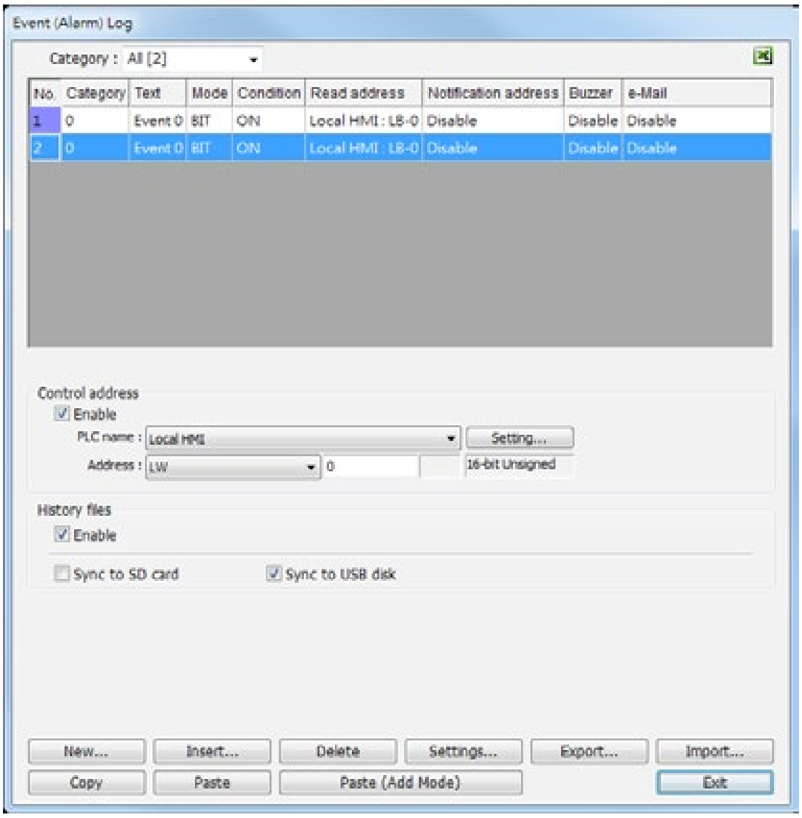
Firstly, define the event content then use Alarm Bar, Alarm Display, Event Display objects to view the process of the whole event from triggering waiting to be processed return to normal.

7.2.1. eMT, iE, XE, mTV Series



Setting	Description
Category	Classifies events by dividing them into 0 ~ 255 categories. Select one category to add or view event log. In the bracket "( )", it shows the number of events are in this category.
History files	To specify a location to save event log files. However, when executing On-line or Off-line Simulation on PC, the files will be saved in the HMI_memory / SD_card / USB folder under the installation directory. **Preservation limit** This setting determines the length of the data can be preserved. For example, the (Days of preservation) is set to two days; the data of yesterday and the day before yesterday will be kept. The history data in other days will be deleted automatically for saving the storage space.
Print	In (System Parameter Settings) » (Model), select a printer and set the printing format.
Paste	Overwrites the selected item with the new items. A message window will pop up to confirm this operation.
Paste (Add Mode)	Appends as a new entry.

7.2.2. cMT Series



Setting	Description
Category	Classifies events by dividing them into 0 ~ 255 categories. Select one category to add or view event log. In the bracket "( )", it shows the number of events are in this category.

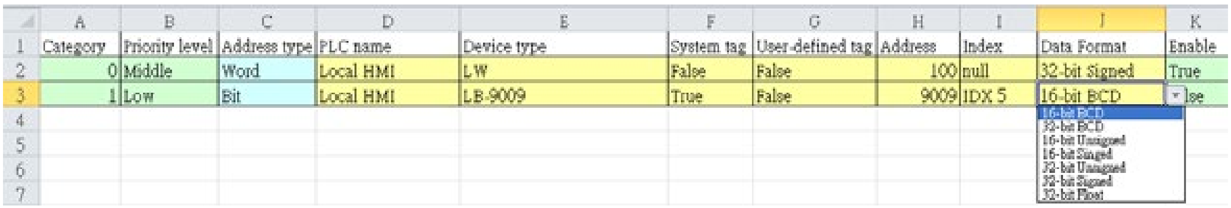
Control address	There are two ways to save the event log file:
	Automatically saved by system
	If (Enable) check box under (Control address) is not selected, the system will automatically save the event log file to HMI_memory folder. The maximum is 10,000 records, when there are more than 10,000 records, the earliest 1,000 records will be deleted.
	If (Enable) check box under (Control address) is selected, selecting the (Enable) check box under (History file) can save data to HMI_memory / SD_card / USB folder.
	Manually saved by user
	If select (Enable) check boxes under both (Control address) and (History files), entering a specific value in the control address sends the corresponding command.
	When entering 1, clears the event log on cMT-SVR.
	When entering 2, synchronizes event log to the external device.
	When entering 3, synchronizes event log to the external device and then clears the event log on cMT-SVR.
	If none of these values is entered, the system will save the event log file in HMI_memory.

History files	If enabled, saves the synchronized data to SD card or USB disk.
Paste	Overwrite the selected items with the clipboard contents. A message window will pop up to confirm this operation.
Paste (Add Mode)	Append the clipboard contents to the end of the list.

	Note To remove SD card or USB disk, event log data can be synchronized by using control address first.

7.2.3. Excel Editing

Click on the Excel icon in Event Log setting dialog box to open the Excel template for a reference of editing. This template is under the installation directory, the file name is EventLogExample.xls. This template includes the ready-made dropdown lists and validation mechanism.



	A	B	C	D	E	F	G	H	I	J	K
1	Category	Priority level	Address type	PLC name	Device type	System tag	User-defined tag	Address	Index	Data Format	Enable
2	0	Middle	Word	Local HMI	LW	False	False	100	null	32-bit Signed	True
3	1	Low	Bit	Local HMI	LB-9009	True	False	9009	IDX 5	16-bit BCD	lse
4											
5											
6											
7											

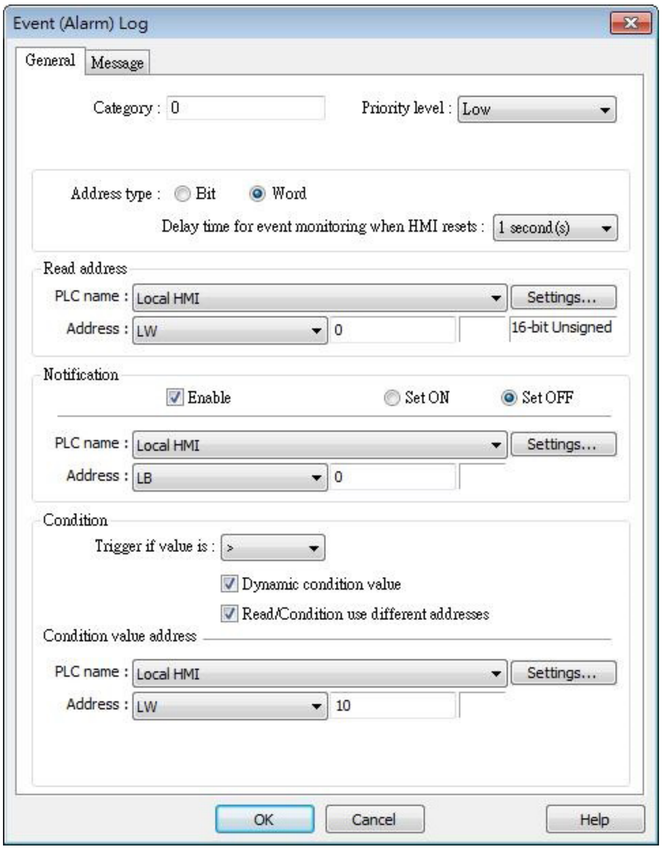
Note

- (System tag) and (User-defined tag) cannot be set to true simultaneously, otherwise, the system will view the User-defined tag to be a System tag, and (User-defined tag) to be false. If setting (Device type) to (User-defined tag), please set (System tag) to false.
- When setting (User-defined tag) to true, if the system compares the (Device type) with the user-defined tag in the system, and no suitable tag is found, the system will set the (User-defined tag) in event log to false.
- (Color) format is R:G:B, each should be an integer form 0 to 255.
- Before importing Label Library / Sound Library, please make sure the library names exist in the system.

7.3. CREATING A NEW EVENT LOG

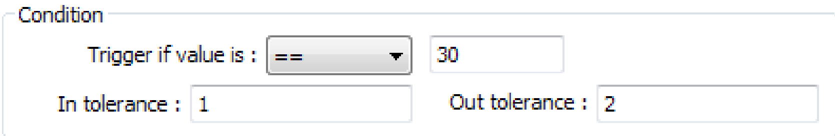
7.3.1. General Tab

Click (New) in the (Event (Alarm) Log) dialog box.



Setting	Description
Category	Select event category, the range is from 0 to 255.
Priority level	When the number of events equals to the max number of the system (default 1,000), the lower priority events will be deleted and new events will be added in.
Delay time for event monitoring when HMI resets	This feature is used to set the delay time of Event Log after HMI reboot, in order to avoid false alarm that occurs upon HMI reboot due to uninitialized values. This feature is often used with (Dynamic condition value). The delay time only occurs once upon HMI reboot.
Read address	The system reads data from this address to check if the event matches the trigger condition.
Notification	When enabled, the system will set the specified address ON or OFF when the event is triggered.
Condition	When (Bit) is selected, Event Log will detect the state of a Bit address. When (Word) is selected, Event Log will detect the value of a Word address to check if it is greater than, less than, or equals to a specified value. See Example 1 and Example 2. **Dynamic condition value** Allows online change of the comparison value for trigger condition when the condition is a Word address type. If (Read/Condition use different addresses) is not selected, the source of condition value will be the next consecutive address from (Read address). **Read/Condition use different addresses** Allows selecting the Word address type to be the source of condition value.

Example 1



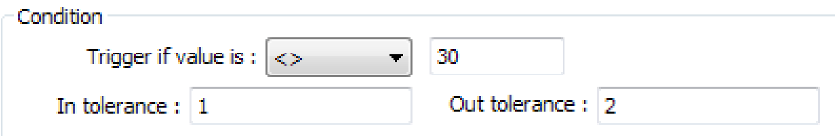
The setting above indicates:
When (Read address) value is greater than or equals to 29 (= 30 - 1)
Or less than or equals to 31 (= 30 + 1), the event will be triggered. The trigger condition:

$$29 \leq [\text{Read address}] \text{ value} \leq 31$$

After the event is triggered, when (Read address) value is greater than 32 (= 30 + 2) or less than 28 (= 30 - 2) the system will return to normal condition:

$$[\text{Read address}] \text{ value} < 28 \text{ or } [\text{Read address}] \text{ value} > 32$$

Example 2



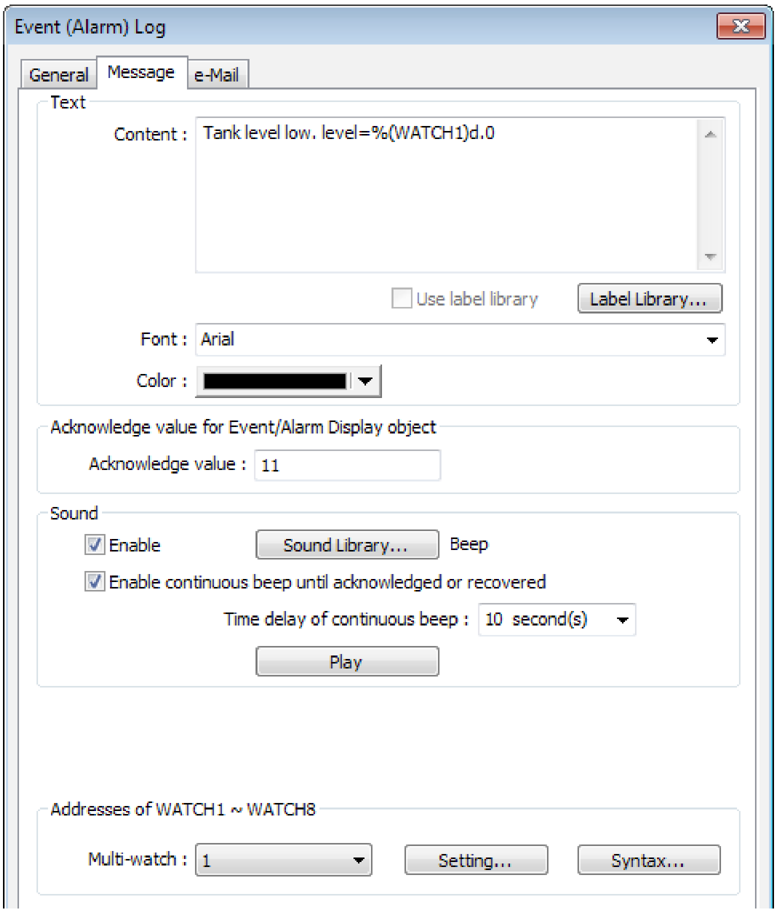
The setting above indicates:
When (Read address) value is less than 29 (= 30 - 1)
or greater than 31 (= 30 + 1), the event will be triggered. The trigger condition:

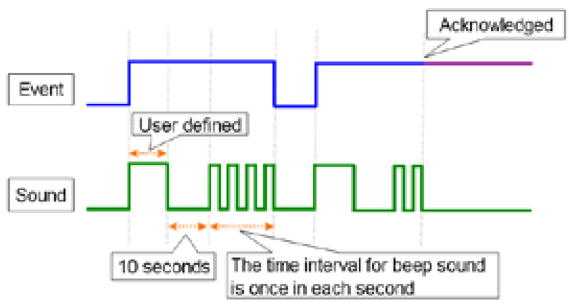
$$[\text{Read address}] \text{ value} < 29 \text{ or } [\text{Read address}] \text{ value} > 31$$

After the event is triggered, when (Read address) value is greater than or equals to 28 (= 30 - 2) or less than or equals to 32 (= 30 + 2) the system will return to normal condition:

$$28 \leq [\text{Read address}] \text{ value} \leq 32$$

7.3.2. Message Tab



Setting	Description
Content	The text content displayed in (Alarm Bar), (Alarm Display), and (Event Display) objects. Use the formats in the following two examples of WATCH addresses to use register data in content. See Example 3 and Example 4.
Font / color	The font and color can be set differently for each event. The setting determines the font and color shown in (Alarm Bar), (Alarm Display) or (Event Display) objects.
Write value for event/alarm display object	When an event in (Event Display) or (Alarm Display) is acknowledged, the value is written to the assigned (Write address).
Sound	If enabled, the selected sound will be played when an event is triggered. Continuous beep can also be enabled, which only stops when the event is acknowledged or recovered. For continuous beep, a delay time can be set between triggering the alarm and the start of beeping. 
Address of WATCH 1 ~ 8	Click (Syntax) to edit and display the value in watch address when the event is triggered. Up to 8 watch addresses can be set.

Example 3

The data of the LW register can be used in the content displayed when an event is triggered:

Format: %#d (% -> initial sign, # -> address, d -> end sign)

When an event is triggered, if the value in LW-20 is 13:

Setting: "High Temperature = %20d"→ Display: "High Temperature = 13"

Example 4

The data in the specified address when the event is triggered can be included in the content displayed.
The address should be set to the (Read address) of Event Log, take Modbus RTU 4x address as an example:

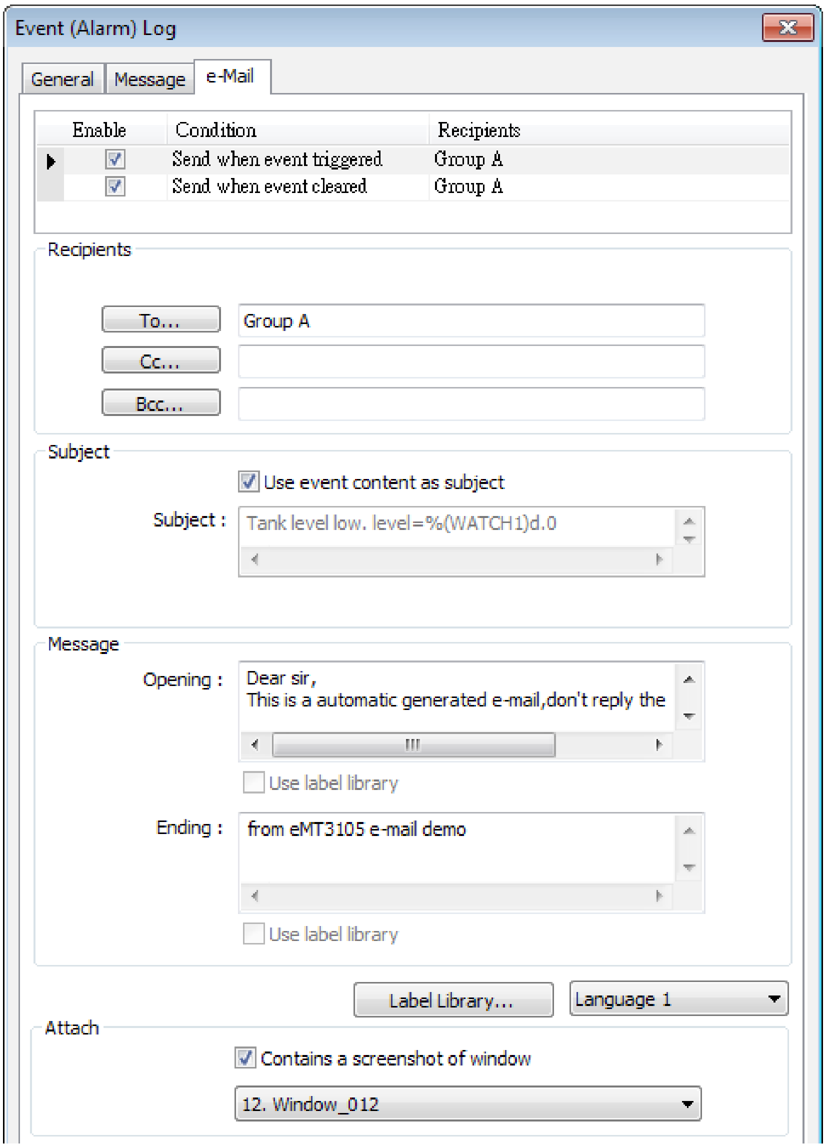
Format: $#d ($ -> initial sign, # -> address, d -> end sign)

When an event is triggered, if the value in Modbus 4x-15 is 42:

Setting: “High Temperature = $15d” → Display: “High Temperature = 42”

7.3.3. e-Mail Tab

Please enable this function in (System Parameter Settings) » (e-Mail) first.



Setting	Description
Recipients	Select the (To), (Cc), and (Bcc) recipients.
Subject	Enter the subject of the e-mail.
Message	Enter the (Opening) and (Ending) content of an E-mail.
Attach	If the (Contains a screenshot of window) check box is selected, the screenshot of the selected window will be attached.

8. DATA SAMPLING

This chapter explains how to set and use Data Sampling.

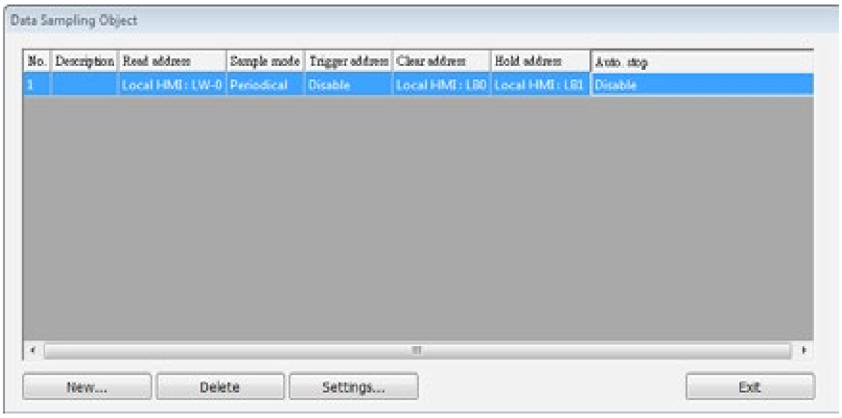
8.1. OVERVIEW

After defining how the data is sampled, by sampling time, address, or data length, the sampled data can be saved to the designated location, such as HMI memory, SD card, or USB disk. Trend Display and History Data Display objects can be used to display sampling records.

8.2. DATA SAMPLING MANAGEMENT

Create a new Data Sampling object first by the following steps:

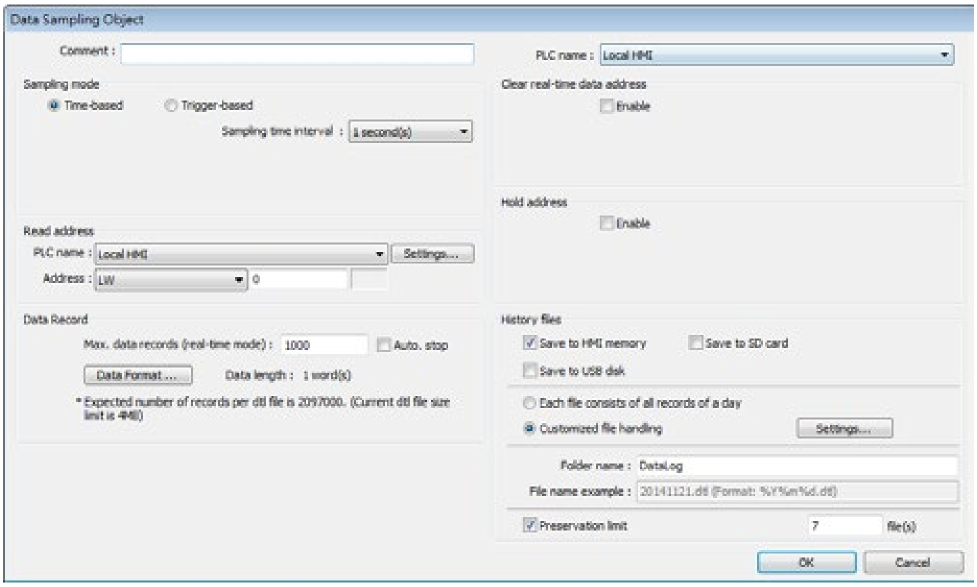
1. From the menu select (Objects) and click (Data Sampling).
2. Click (New) to finish relevant settings.



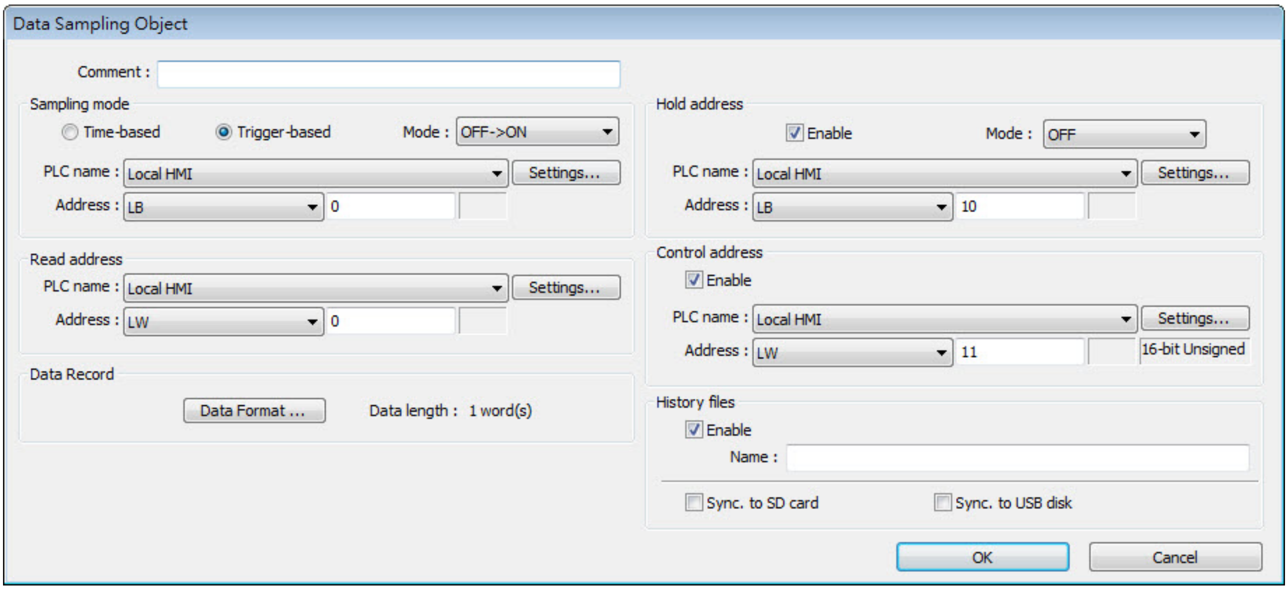
8.3. CREATING A NEW DATA SAMPLING

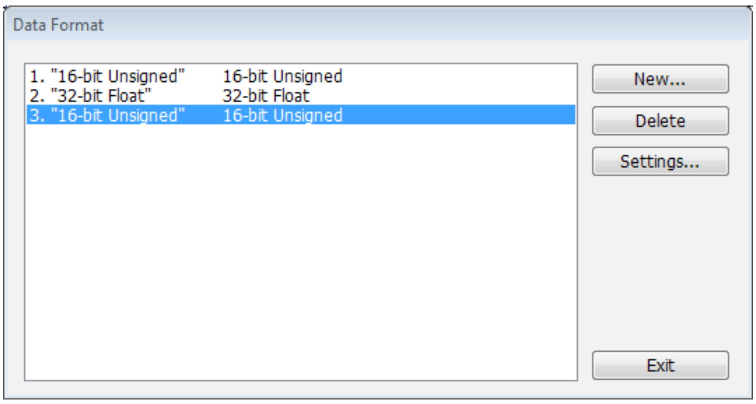
The following introduces how to set a new Data Sampling.

▪ eMT, iE, XE, mTV Series



▪ cMT Series



Setting	Description
Sampling mode	**Time-based** Samples data in a fixed frequency. The (Sampling time interval) can be set from "0.1 second(s) to 120 minutes". **Trigger-based** Triggers data sampling by the status of a designated bit address. **Mode Conditions to trigger Data Sampling:** (OFF -> ON) Triggers sampling when the status of the address changes from OFF to ON. (ON -> OFF) Triggers sampling when the status of the address changes from ON to OFF. (OFF <-> ON) Triggers sampling when the status of the address changes. **Set ON/OFF after triggered** If selected, after triggering Data Sampling, the system will set the designated bit address back to ON/OFF state.
Read address	Specify an address to be the source of Data Sampling.
Data record (real-time)	In Real-time Mode, the max. number of data records can be saved by one Data Sampling in one day is 86,400. (1 record per second for 24hours) If (sampling time interval) is set to "0.1 second", the max. number of data records is still 86,400. **Data Format** Data of different formats in consecutive registers can be sampled. As shown in the following figure: LW-0 (16-bit Unsigned), LW-1 (32-bit Float), and LW-3 (16-bit Unsigned).  **Auto. stop** This function depends on the arrangement of different objects and modes. See "8.3.1 Demonstration of Auto. stop".
Clear real-time data address	Set when the bit address status changes from (OFF -> ON) or (ON -> OFF), clear the sampled data in Trend Display Real-time Mode. The number of data records returns zero but the data records that are already saved as history files will not be cleared.
Hold address	If the status of the designated address is set ON or OFF, sampling will be paused until the status of the designated address returns.

Control address (cMT Series)	Entering a value in the control address sends the corresponding command. Enter 1: clears the sampled data in HMI. Enter 2: synchronizes data to the external device. Enter 3: synchronizes data to the external device and then clears the sampled data in HMI.
History files	**eMT, iE, XE, mTV Series** ▪ Save to HMI momery Saves Data Sampling to HMI only when its size reaches 4KB. Or, use system register (LB-9034) to force storing data. ▪ Save to SD card / USB disk Saves Data Sampling to the specified external device. ▪ Each file consists of all records of a day The data sampling file will be saved on a daily basis into the specified folder, and the file name will be yyyymmdd.dtl, indicating the date of the file. ▪ Cutomized file handling This feature can be used to customize naming and management of data sampling files (*dtl). See “8.3.2 Customized File Handling”. ▪ Folder name Specify Data Sampling file name which must be all in ASCII characters. The folder name will be written as: (Storage Location) \ (Folder Name) \ (File Name) ▪ Preservation limit Determines the number of data sampling files to be preserved. **cMT Series** History data can be saved to USB disk or SD card. When the sampled data reaches 10,000 records, the sampled data is automatically saved to the selected external device, and the earliest 1,000 records are deleted. See “8.4 Synchronizing CloudHMI data and Saving to External Device”.

Note

- A Data Sampling may include more than one type of records. Data Sampling can retrieve different types of records at the same time. For example, if define three types of data, 4 words in total, the system retrieves a 4-word data each time from the designated address to be the content in one Data Sampling.
- When using (Each file consists of all records of a day) and set (Preservation limit) to 2 files, the data of yesterday and the day before yesterday will be kept. Data that is not built in this period will be deleted to prevent the storage space from running out.
- If using (Customized file handling) and set (Preservation limit) to 2 files, not only the currently sampled file, another 2 newest files (3 files in total) will be kept. The rest of the data will be deleted to prevent the storage space from running out.
- When running simulation on PC, all data sampling will be saved to the datalog folder in C:\EBPro\ (Storage Location) \ datalog. If you change the data format of data sampling, delete the previous data records in the installation directory to prevent the system from reading the old records.

8.3.1. Demonstration of Auto. Stop

This feature depends on the arrangement of different objects and modes. (Set (Max. data records) to n.)

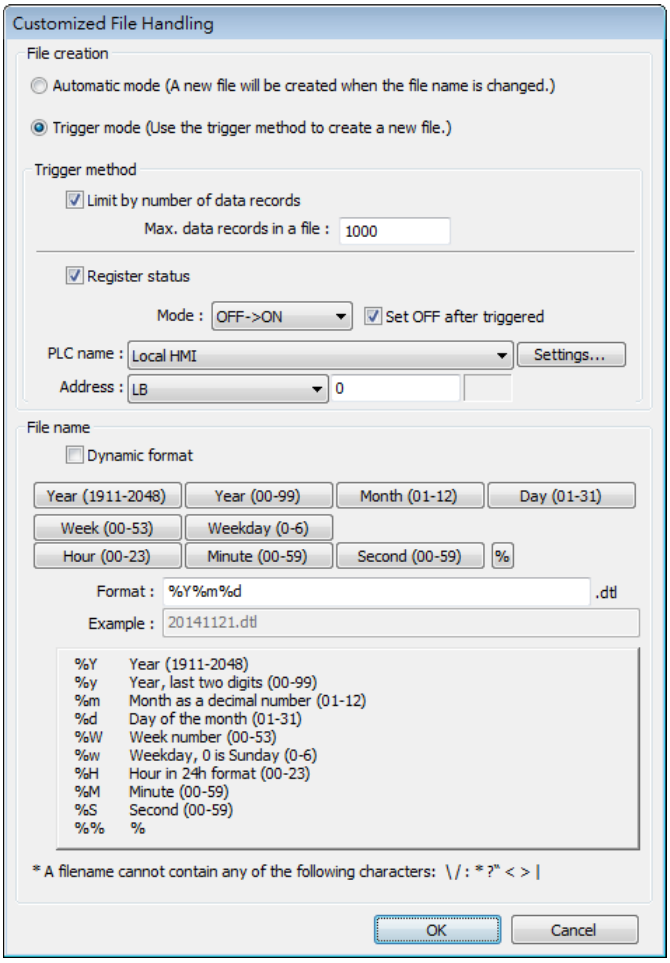
Object	(Auto. stop) not selected	(Auto. stop) selected
Trend display-real time	Deletes the earlier records and displays the latest number of records (n) in Trend Display. See the following figure.	Stops after reaching the specified number of data records (n).
Trend display-historical	Keeps on sampling data and displays all history data in Trend Display.	Irrelevant.
History data display	Keeps on sampling data and displays all history data in History Data Display.	Irrelevant.
Data sampling	Keeps on sampling new data.	Irrelevant.

The figure illustrates how the data is sampled in Trend Display – Real Time mode when (Auto. stop check) box is not selected. Set the number of data records to 10, when the 11th data is generated, the earliest record is deleted and the newest record is added.

Record Number	Data	Not selecting [Auto. stop]
1	101	102
2	102	103
3	103	104
4	104	105
5	105	106
6	106	107
7	107	108
8	108	109
9	109	110
10	110	111
11	111	

8.3.2. Customized File Handling

This feature can be used to customize naming and management of data sampling files (*.dtl).

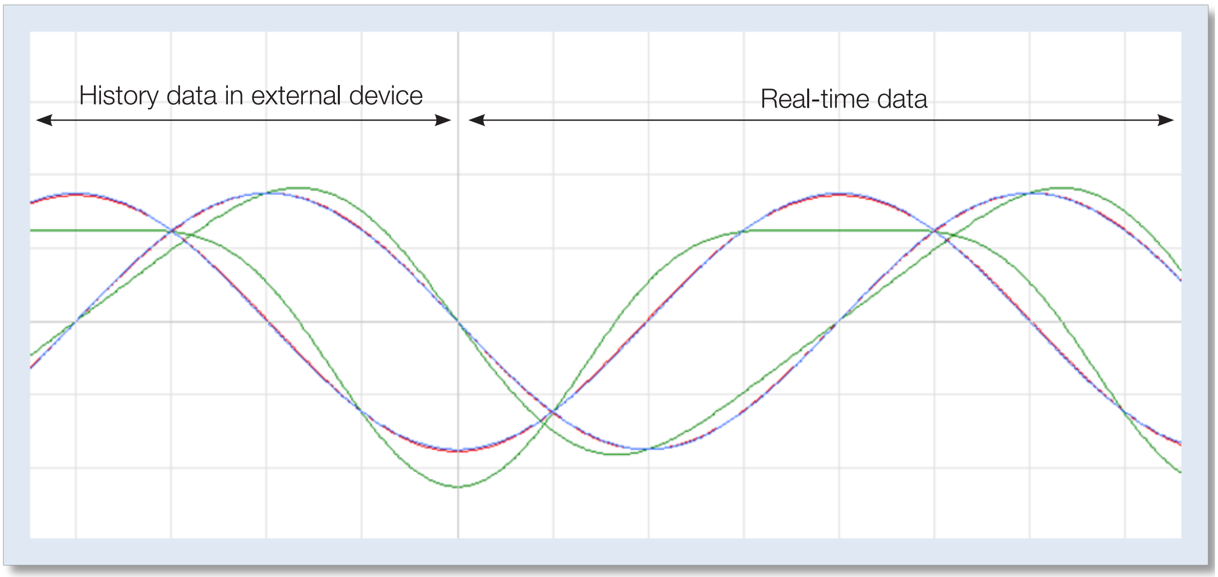


Setting	Description
File creation	**Automatic mode** A new file will be created when the name of an existing file is changed. **Trigger mode** A new file will be created according to the (Trigger method) settings.
Trigger method	**Limit by number of data records** A new file will be created when the number of data sampling records reaches the specified “Max. data records in a file” **Register status** A new file will be created when the status of a designated bit address meets the specified condition. The condition is specified in Mode field. **Set ON/OFF after triggered** If selected, after the new file is created, the system will set the designated bit address back to ON/OFF state.
File name	The file name can be an alphanumeric name, and certain half-width symbols are allowed. The file name can also be specified by a file name syntax. **Dynamic format** The file names can be set by a designated word address, or by a file name syntax indicating the current system time. The file name syntax can be specified by selecting time buttons or entering the syntax in Format field. The length limit is from 1 to 25. The following half-width characters are not allowed: \ /: * ? “ < > |

Note

- If both (Limit by number of data records) and (Dynamic format) check boxes are selected, before startup HMI, please enter the name in the designated register for Dynamic Format, otherwise, it is impossible the reach the “Max. data records in a file”, and the data sampling file will not be generated.
- A data sampling file (*.dtl) cannot be written when it’s size exceeds 4 MB.
- When a new file is generated, the systm will first detect if the filename already exists. If the file name does exist, the newly sampled data will be appended to the existing file.

8.4. SYNCHRONIZING CLOUDHMI DATA AND SAVING TO EXTERNAL DEVICE



For other series, when displaying the sampled data in Trend Display object, it is necessary to select from Real-time mode or History mode and the two modes cannot simultaneously be displayed in one object.
cMT Series allows displaying history data and at the same time updates real-time data in one Trend Display or History Data Display object. The data saved in the external device can be updated.
The rule of synchronizing the data saved in the external device:

1. When the sampled data reaches 10,000 records, HMI will automatically saves data to the external device and deletes the earliest 1,000 records in HMI.
2. If the external device is removed from HMI, and inserted back again at the time when the sampled data is under 9,000 records, the data generated during the time the external device is removed is saved in HMI and is not cleared. If the data exceeds 9,000 records during the time the external device is removed, the earlier data is cleared and cannot be synchronized even to insert the external device back to the HMI.
3. If there already exists sampled data in the external device, the new data is appended without overwriting the original data each time in synchronization.

8.5. CHECKING HISTORY DATA OF A SPECIFIC DATE ON CLOUDHMI

To check the history data, see the following steps (Use Trend Display object as example).

1. Tap the icon in the upper-right corner of the Trend Display object.
2. The following dialog box appears.

3. Specify the (Begin Date) and (Ended Date).

4. Tap (Done) button to finish setting.

9. OBJECT GENERAL PROPERTIES

This chapter explains the basic settings of an object.

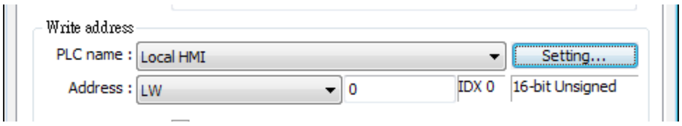
9.1. OVERVIEW

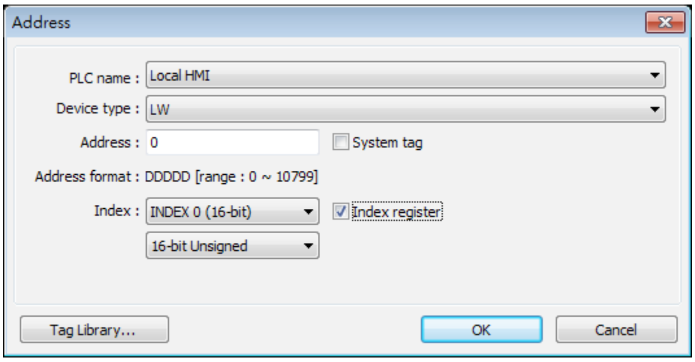
The basic steps to create an object:

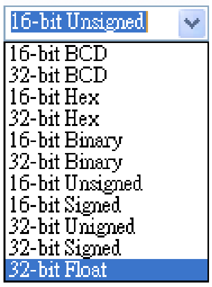
1. Selecting the PLC device and setting the read/write address.
2. Using Shape Library and Picture Library.
3. Setting label text.
4. Adjusting profile size.
5. This chapter explains the basic settings of an object.

9.2. SELECTING PLC AND SETTING READ/WRITE ADDRESS

Most objects read data from PLC devices, so a properly configured PLC address is needed. Select the PLC to control at (PLC name) which comes from (System Parameters Settings) » (Device List).



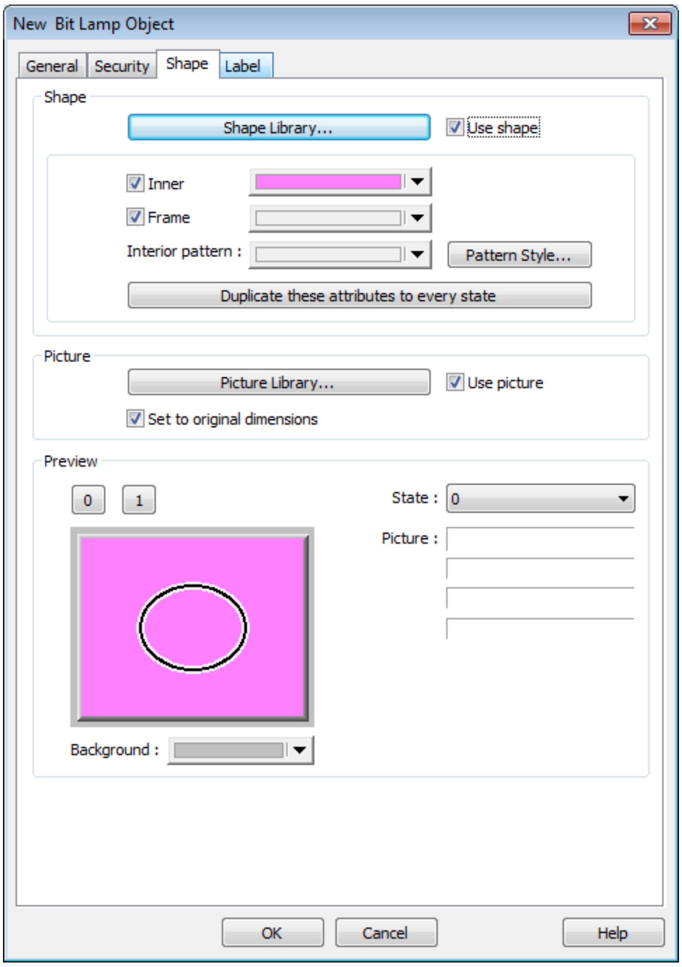


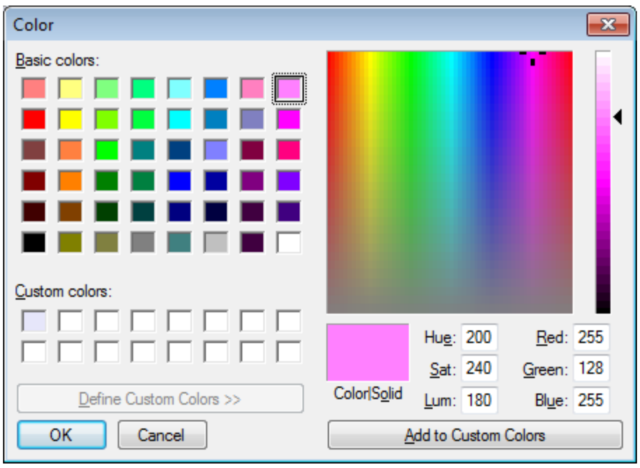
Setting	Description
PLC name	Select the PLC type.
Device type	Different PLCs have different device types.
Address	Set the read/write address.
System tag	Address tags include (System Tag) and (User-defined Tag). This option allows users to use (System Tag). (System Tag) consists of the preserved addresses by system for particular purposes. The address tags include bit registers or word registers (LB or LW). After selecting (System tag), not only will the (Device type) field display the chosen tag, but (Address) field will also display the chosen register.
Index register	Select this check box to use the (Index Register). Certain objects require data type setting. The following data types are supported in EasyBuilder Pro, selecting correct data type is necessary especially when using address tag. 

☞ For more information about System Tag, see “22 System Registers”.
☞ For more information about Index Register, see “11 Index Register”
☞ For more information about Tag Library, see “16 Address Tag Library”

9.3. USING SHAPE LIBRARY AND PICTURE LIBRARY

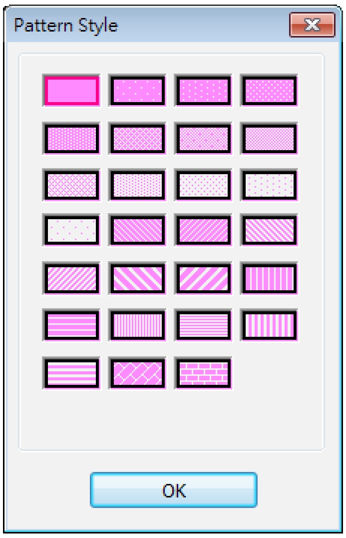
Shape Library and Picture Library are used to add visual effects on objects. Select (Shape) tab to use the libraries when creating an object.



Setting	Description
Shape library	Select (Use shape) check box and select a shape from the library. See next page.
Inner	Select this check box to set the inner part of a shape. Click the drop down button to select a color or customize a color and click (Add to Custom Colors). EasyBuilder Pro will save this color. 
Frame	Select this check box to set the frame of a shape. Click the drop down button to select a color.

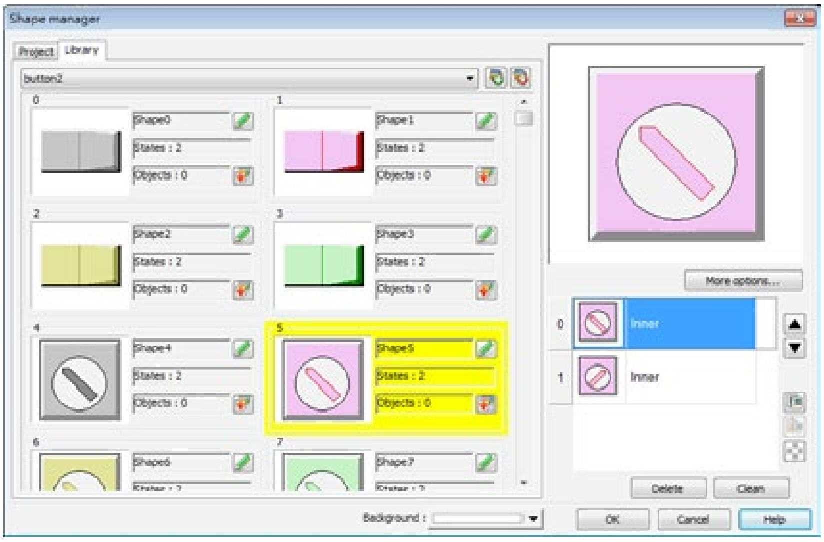
<table>
<tr><td>Inner pattern</td><td>Set the color of the interior pattern of the shape.

Pattern Style
Click this button to select a pattern.


</td></tr>
<tr><td>Duplicate these attributes to every state</td><td>Duplicate all attributes of the current state to other states.</td></tr>
</table>

9.3.1. Shape Manager

Click (Shape Library…) button to open the (Shape manager) dialog box. The currently selected shape is highlighted yellow.



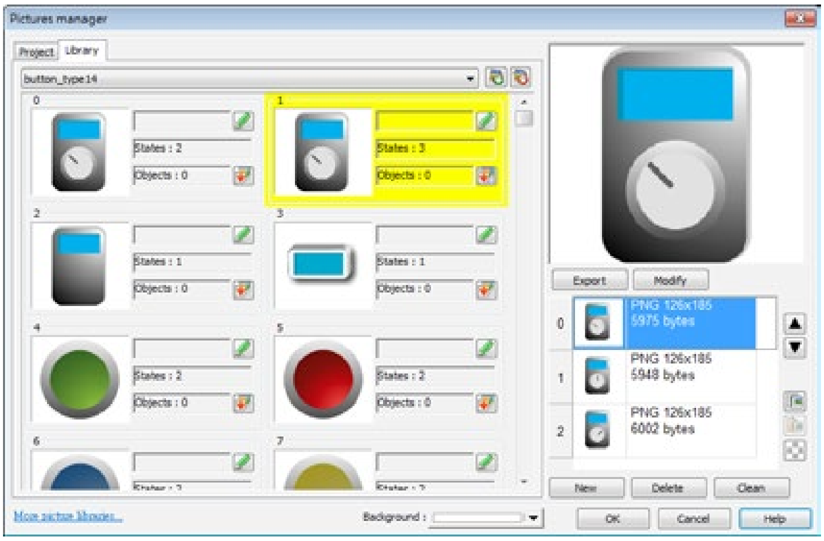
The illustration above provides information of one of the Shapes in the Shape Library:

- Shape5 Name of the shape
- States: 2 Number of states of the shape
- Objects: 1 This shape is used by 1 object in the project

The illustration above shows the shape has two states, State 0 and State 1, and contains only “inner” but not “frame.” When finished, click (OK), and the object will use the selected shape.

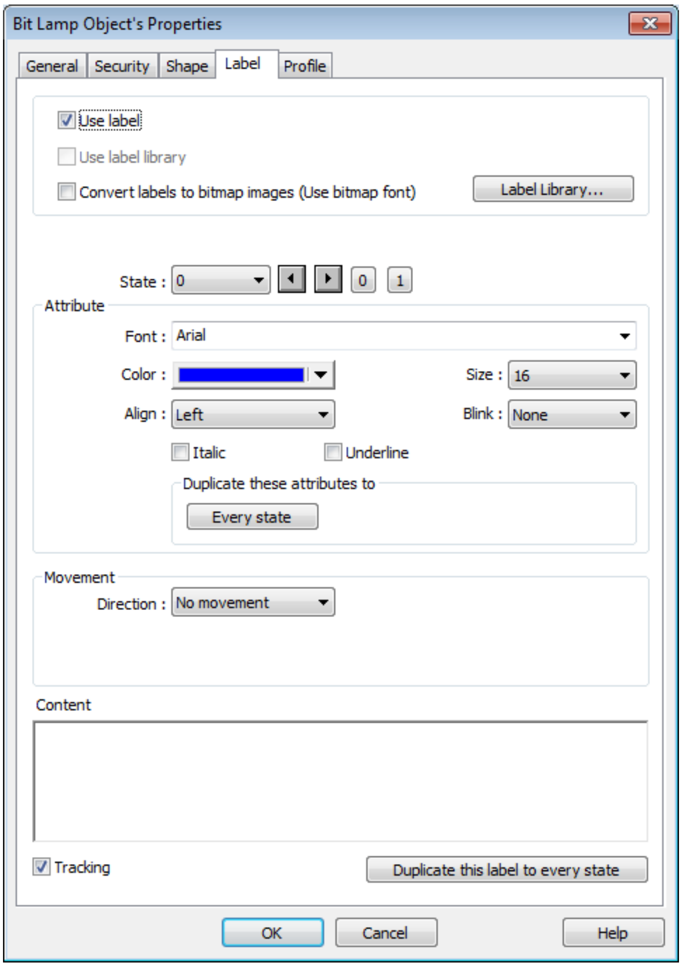
9.3.2. Pictures Manager

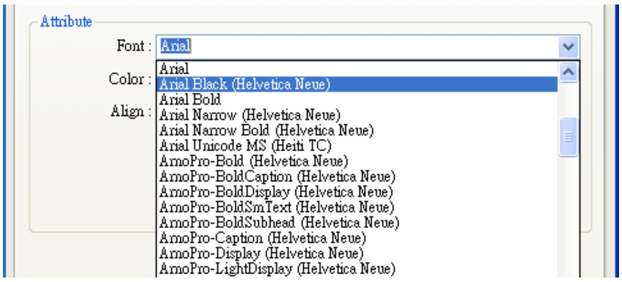
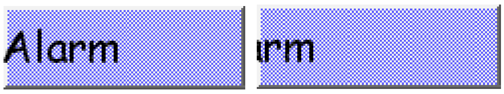
Click (Picture Library…) button to open the (Pictures manager) dialog. The currently selected picture is highlighted yellow.



☞ For more information, see “14 Shape Library and Picture Library”.

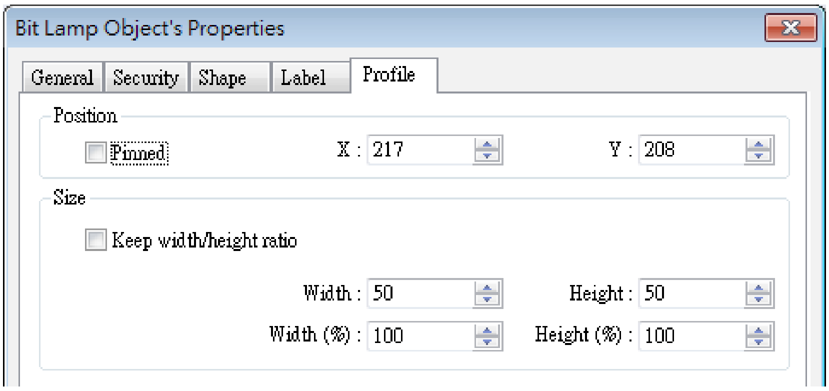
9.4. SETTING LABEL TEXT



Setting	Description
Use label	Select this check box to use labels for the object.
Use label library	Select this check box to choose a label in Label Library.
Use bitmap font	Select this check box to convert the label text into bitmap format.
Label library	Browse Label Library For more information, see “15 Label Tag Library and Multi Language”.
Font	Select a font from the list. EasyBuilder Pro supports Windows True-type fonts. For cMT-SVR Series, if the font is followed by another font enclosed by parentheses, the enclosed font is the one used in iPad. 
Color	Select the font color.
Size	Select the font size.
Align	The text aligned (Left) (Center) (Right) 111 111 111 222222 222222 222222 333333333 333333333 333333333
Blink	Specify the way the text blinks. Choose (None) to disable this feature or set the blinking interval to (1 second) or (0.5 seconds).
Italic	Use Italic font. *Italic Label*
Underline	Use Underline font. Underline Label
Movement	**Direction** Set the direction of the marquee effect. The directions include: (No movement), (Left), (Right), (Up), (Down). **Continuous** Specify how the marquee effect is displayed. If not selecting this check box, the next text appears only when the previous text disappears completely.  If selecting this check box, the text will be displayed continuously.  **Speed** Adjust the speed of the text movement.
Content	Set the content of the text. If (Label Library) is used, it will automatically use the text defined in Label Library.
Tracking	If this check box is selected, when changing the position of the text in one state, the text position in the other states will also change in accordance.
Duplicate this label to other state	Duplicate the current text to the other states.

9.5. ADJUSTING PROFILE SIZE

When an object is created and placed in the editing screen, double click it and select the (Profile) tab to adjust the position and size of the object.



Setting	Description
Position	**Pinned** When this check box is selected, the position and the size of the object cannot be changed. **(X) and (Y)** The coordinates of the position of the object in the editing screen.
Size	Adjust the (width) and (height) of the object.

10. USER PASSWORD AND OBJECT SECURITY

This chapter discusses the protection for operations provided by setting up user passwords and security classes.

10.1. OVERVIEW

This chapter discusses the protection for operations provided by setting up user passwords and security classes. There are two authentication modes:
- General Mode
- Enhanced Security Mode

Each mode will be introduced later.
To set up the protection system, please:

1. Set user password and operable classes.
2. Set object class for objects.

An object belongs only to one security class. Setting the object class to “None” means any user can operate this object.

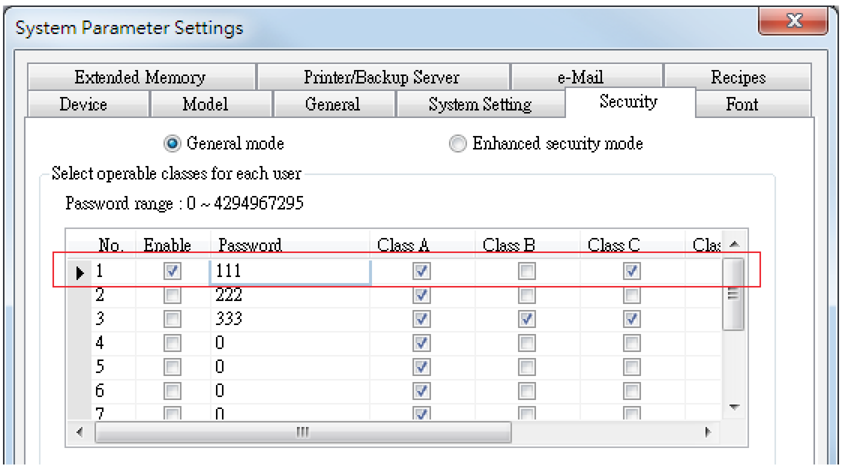
10.2. USER PASSWORD AND OPERABLE OBJECT CLASSES

The security parameters can be found in (Edit) » (System Parameter Settings) » (Security). Two modes are available: General Mode and Enhanced Security Mode.

10.2.1. General Mode

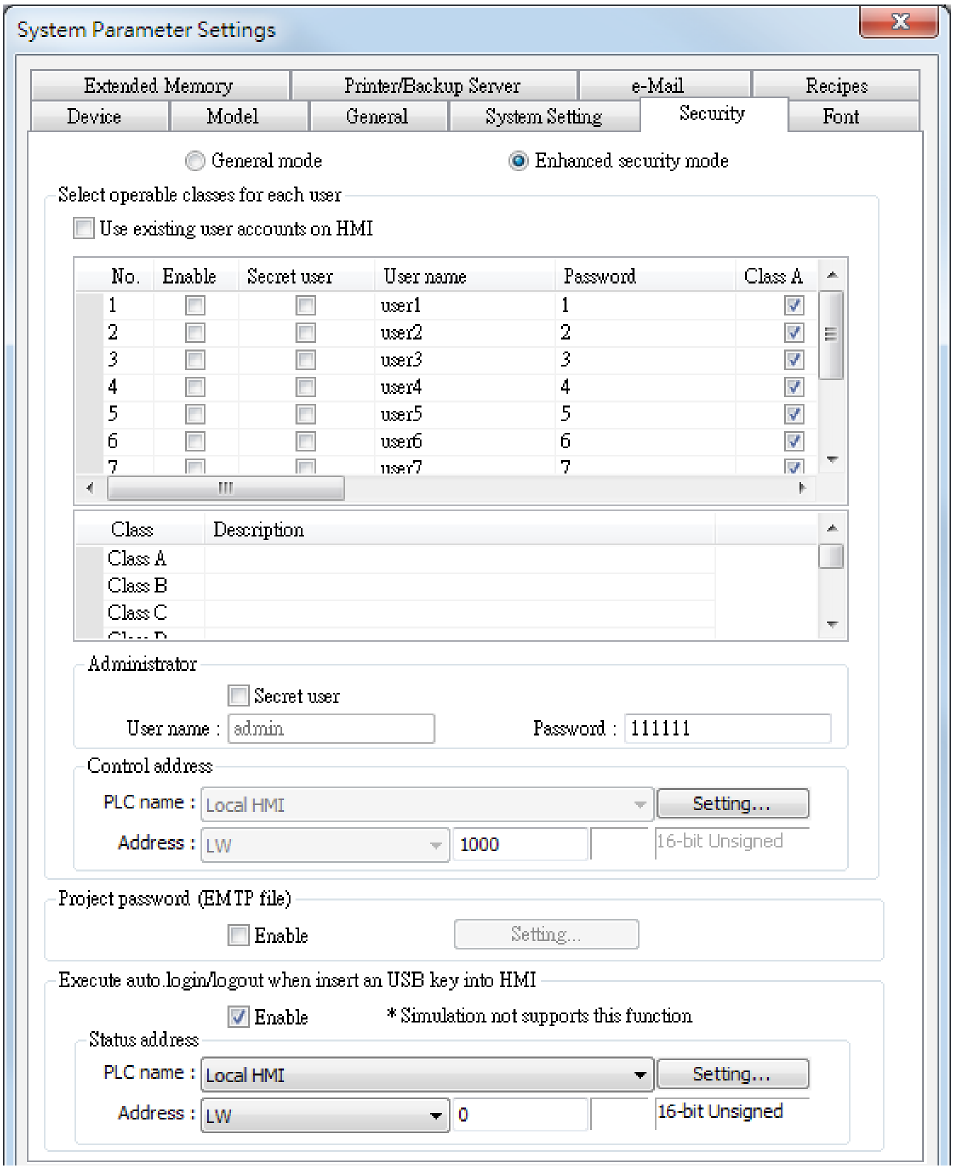
Up to 12 sets of user and password are available. A password should be one non-negative integer. There are six security classes: A to F.
Once the password is entered, the objects that the user can operate are classified. As shown below, “User 1” can only operate objects with class A or class C.

	Note General Mode is not available for cMT-SVR Series.



10.2.2. Enhanced Security Mode

Up to 11 users can be set here. In addition, (Administrator) setting is provided. Administrator has all privileges and can operate all object classes. User passwords must be alphanumeric characters and each user can have up to 12 operable classes: A to L. (Up to 127 users can be set in Administrator Tools. Please see “10.4 Enhanced Security Mode Usage” for more details.)
Enhanced Security Mode provides a (Control address) for users to manage the accounts directly on HMI. Please see “10.3 Enhanced Security Mode and Control Address” for more details. Alternatively, use USB Security Key to log in automatically. Insert the device in which the key is saved to log in. Please see 10.4.3 Login / Logout Automatically with USB Security Key” for more details.



10.3. ENHANCED SECURITY MODE AND CONTROL ADDRESS

The Control Address is used for login and account management. The Control Address can only be assigned to LW register on Local HMI, and 20 consecutive registers will be used. To log in using Control Address, please select to log in by (user name) or (user index).
Please set (user name) and (password) in (System Parameter Settings) » (Security) » (Enhanced security mode) in advance.

10.3.1. Control Address Settings

When control address is set to LW-n, where n is an arbitrary number, the following addresses will be designated:

Address	Tag Name	Description
LW-n (1 word)	command	Commands to be executed: login, logout, add/setting/delete accounts, etc.
LW-n + 1 (1 word)	command execution result	Displays the result of executing commands.
LW-n + 2 (1 word)	user index	The index of accounts (used with Option List Object).
LW-n + 3 (1 word)	user privilege	Binary value. Level A = bit0, Level B = bit1, …
LW-n + 4 (8 words)	user name	Account name (Only alphanumeric characters, "-" or "_", case-sensitive).
LW-n + 12 (8 words)	password	Account password (Only alphanumeric characters, "-" or "_", case-sensitive).

After setting the (Control address), the relevant addresses can be found in (Address Tag Library) » (Customized). For example, setting (Control address) to LW-0: (UAC stands for User Account Control)

LW-0 → (UAC command)
LW-1 → (UAC command execution result)
LW-2 → (UAC user index)
LW-3 → (UAC user privilege)
LW-4 ~ LW-11 → (UAC user name)
LW-12 ~ LW-20 → (UAC password)

Note
In Enhanced Security Mode, if cMT-SVR Series model is used, the Control Address can only be assigned to PLW register of Local HMI.

10.3.2. Commands

Setting different values in LW-n (command) enables different commands:

Set value	Command	Corresponding address
1	Log in by user name	Set (user name) and (password) first. After entering the user name and password, the system will check if they are valid in (System Parameter Settings) » (Security) » (Enhanced security mode).
2	Log in by user index	Set (user index) and (password) first. Please refer to 10.4.4 Enhanced Security Mode with Option List Object.
3	Log out	
4	Change the password of current logged-in user	Set (user name) and (password) first. Please fill in the original password in (user name) and new password in (password).
5	Add an account	Set (user name), (password) and (user privilege) first.
6	Add a temporary account	Set (user name), (password), (user privilege) and (user index) first. The index is for specifying an effective time period (minutes). If 0 is specified, this account is valid until HMI is turned off.
7	Delete an existing account by user name	Set (user name) first.
8	Delete an existing account by user index	Set (user index) first.
9	Setting the privilege of an existing account by user name	Set (user name) and (user privilege) first.
10	Setting the privilege of an existing account by user index	Set (user index) and (user privilege) first.
11	Setting the password of an existing account by user name	Set (user name) and (password) first.
12	Setting the password of an existing account by user index	Set (user index) and (password) first.
13	Read the privilege of an existing account by user name	Set (user name) first. If the command succeeds, (user privilege) can be displayed.
14	Read the privilege of an existing account by user index	Set (user index) first. If the command succeeds, (user privilege) can be displayed.

Note
- Add a temporary account: the difference between general accounts and temporary accounts is that temporary accounts will not be stored in the system; therefore, it is invalid after HMI turned off. The temporary accounts will be automatically deleted when they are expired.
- Delete the existing account: the currently logged in account cannot be deleted.
- Offline/Online Simulation: simulate using the account settings in the program. Any modifications of the account during simulation will not be reserved for next simulation.
- admin: default administrator account, cannot be deleted, has all privileges and cannot be changed.
- System Register LW-10754: displays current user name.
- System Register PLW-10754: displays current user name. (Only available for cMT-SVR)

10.3.3. Command Execution Results

After the command is executed, the system will store the result code to control address LW-n + 1. The listed result codes below are shown in hexadecimal format.

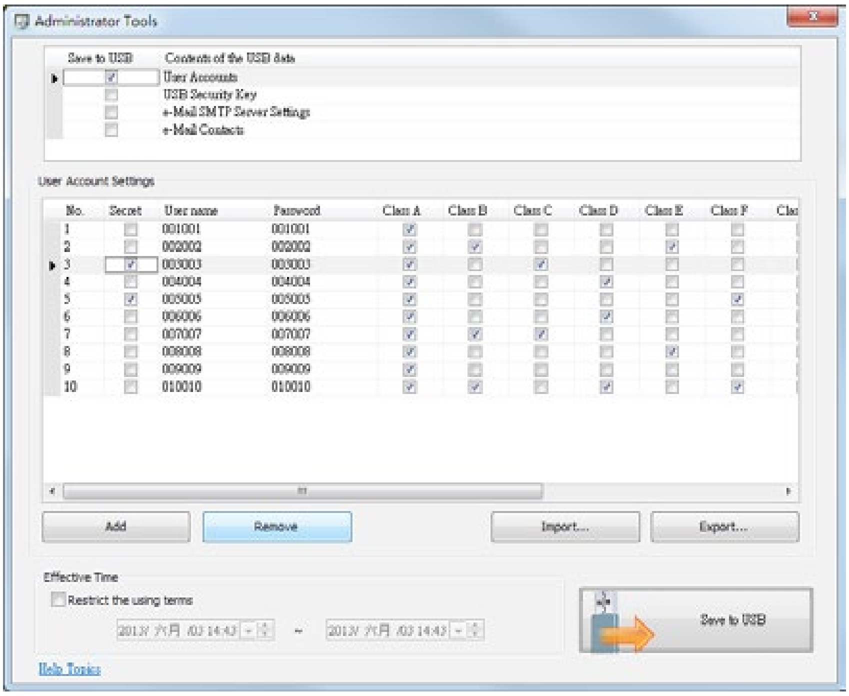
Result Codes	Command execution result
(0x001)	Succeeds
(0x002)	Invalid command
(0x004)	Account exists (when adding a new account)
(0x008)	Account not exists
(0x010)	Password error
(0x020)	Deny command
(0x040)	Invalid name
(0x080)	Invalid password character exists
(0x100)	Invalid import data
(0x200)	Out of validity range (when log in by USB Security Key). The (Effective Time) can be set in administrator tools.

Note
Users can add a new event in Event (Alarm) Log, and designate the (Read address) to LW-n + 1 (command execution result). Open (Message) tab » (Text) » (Content) and specify the message to be displayed in Event Display Object for showing command execution result.

10.4. ENHANCED SECURITY MODE USAGE

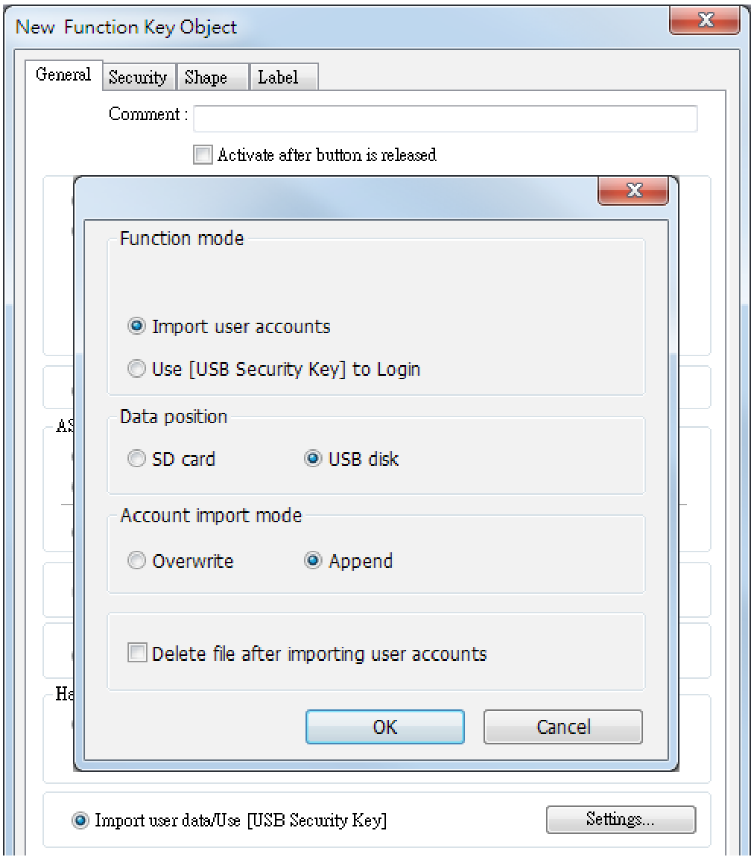
10.4.1. Importing User Accounts

The user accounts can be set using other tools we provide, apart from the settings in (System Parameter Settings) » (Security) tab. Administrator Tools can also be used to set user accounts. Administrator Tools can be found in the installation directory. After the program starts, select the (User Accounts) check box. Up to 127 accounts can be added.



For more information, see “36 Administrator Tools”.

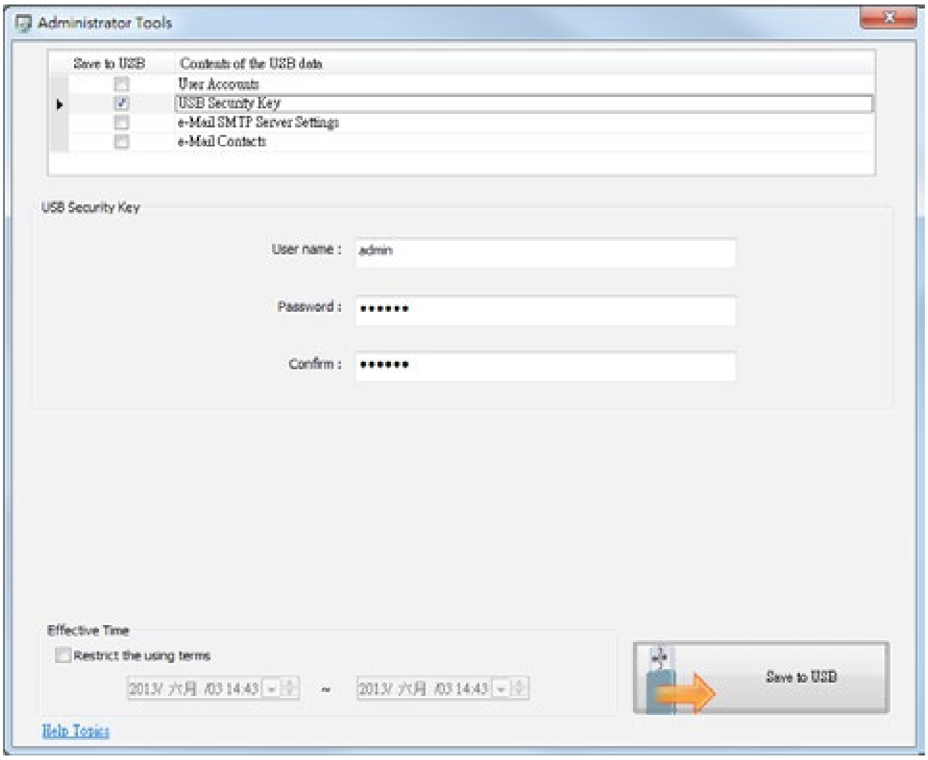
The added accounts can be stored in USB disk or SD card and imported in HMI by a Function Key Object. To do so, create a Function Key Object, and select (Import user accounts).



When finished, insert the external device to HMI, and press Function Key to import accounts. If (Overwrite) is selected, the existing accounts will be overwritten with new accounts and automatically log out after importing. If select (Delete file after importing user accounts) check box, the system will delete the account data saved in the external device after importing. If the (Effective Time) in Administrator Tools is specified, the importing can only be done in the time limit specified. The imported accounts will not be deleted by system when the effective time ends.

10.4.2. Login with USB Security Key

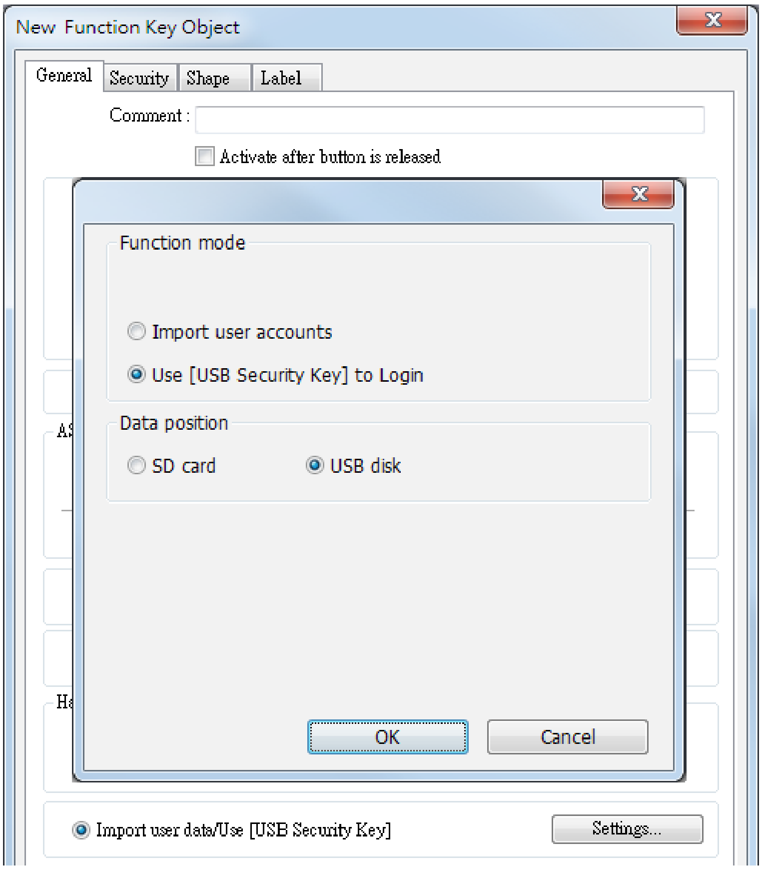
Instead of entering user name and password to login, a key can be used to do so. In EasyBuilder Pro installation directory, launch Administrator Tools, select (USB Security Key) check box. The account information uses the predefined data in (System Parameter Settings) » (Security).



	Note Please note that the user accounts used for USB Security Key must already exist in HMI.

☞ For more information, see “36 Administrator Tools”.

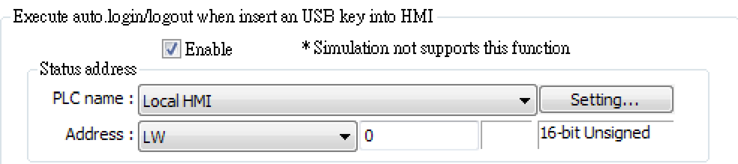
USB Security Key can be stored in USB disk or SD card, and create a Function Key to log in by USB Security Key as shown below:



When finished, insert the external device to HMI, and press Function Key to log in using USB Security Key. If the (Effective Time) in Administrator Tools is specified, the login can only be done in the time limit specified. The system will log out automatically when the key expires.

10.4.3. Login / Logout Automatically with USB Security Key

As shown below, in (System Parameter Settings), select (Enable) check box for (Execute auto. Login / logout when insert an USB key into HMI).



This function allows automatic login / logout using an USB security key. Insert the USB disk in which the key is saved to HMI to log in, and remove the USB disk to log out. The login / logout status will be written into a designated address, the result codes of login / logout:

- 0x00: No Action
- 0x01: Login Succeeds
- 0x04: Login Fails
- 0x08: Login Succeeds
- 0x10: Logout Fails

☞ For more information about USB Security Key, see “36 Administrator Tools”.

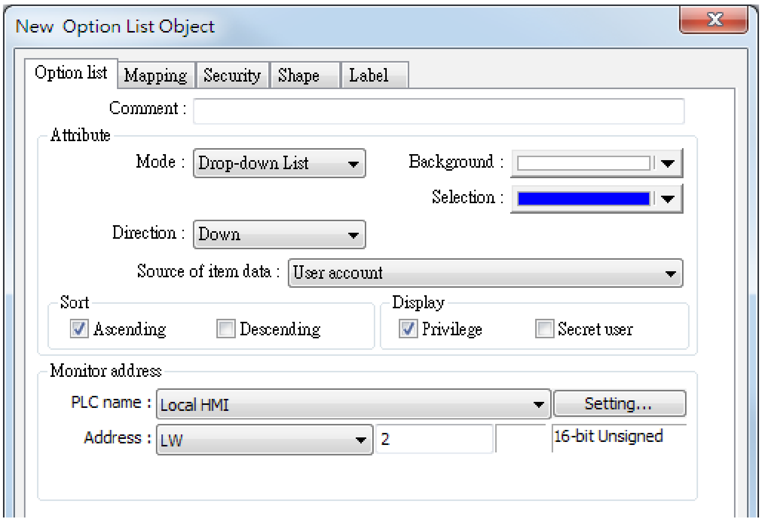
Note

- When Auto Login / Logout is enabled, log in by (Function Key) object is not possible, but it is still possible to log in / out with a designated control address.
- This function does not support On-line / Off-line simulation.
- Only the USB Security Key saved in USB disk is valid.

Download
Click the icon to download the demo project that explains how to use USB Security Key to log in / out.
Please confirm your internet connection before downloading the demo project.

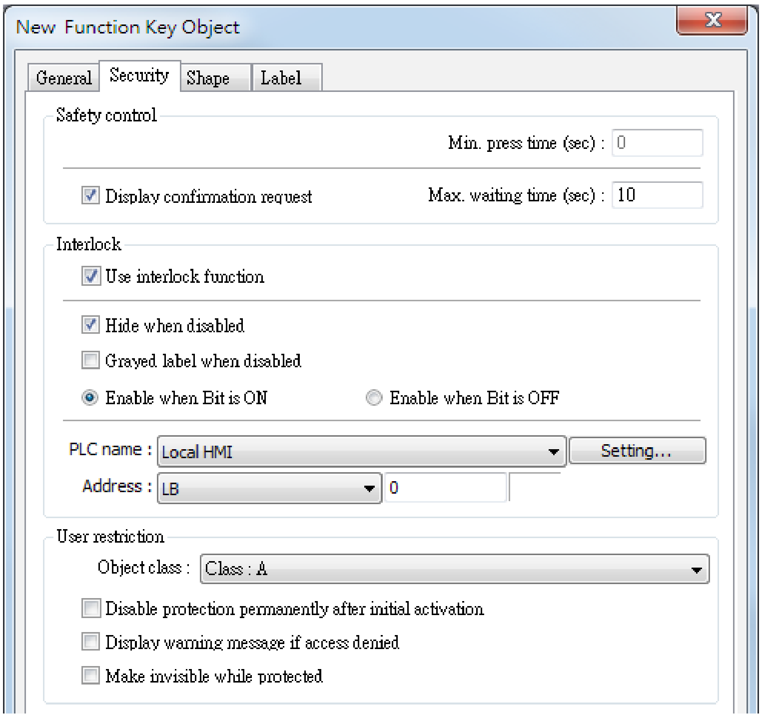
10.4.4. Enhanced Security Mode with Option List Object

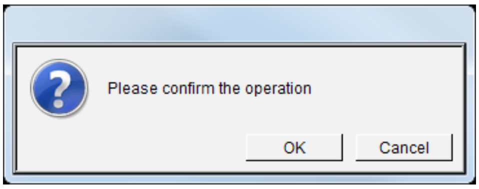
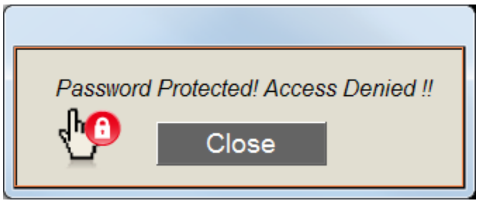
Enhanced Security Mode uses Control Address LW-n + 2 as account index. With Option List Object, account names and privileges can be displayed. Users can select whether or not to display the account privileges and secret users in Option List. Secret users are set to be hidden in (System Parameter Settings) » (Security) » (Enhanced Security Mode); their account names will be hidden in Option List if (Secret user) check box is not selected. If the control address is set to LW-0, the monitor address for index of Option List is designated to LW-2.



Download
Click the icon to download the demo project that explains more about Enhanced Security Mode.
Please confirm your internet connection before downloading the demo project.

10.5. OBJECT SECURITY SETTINGS



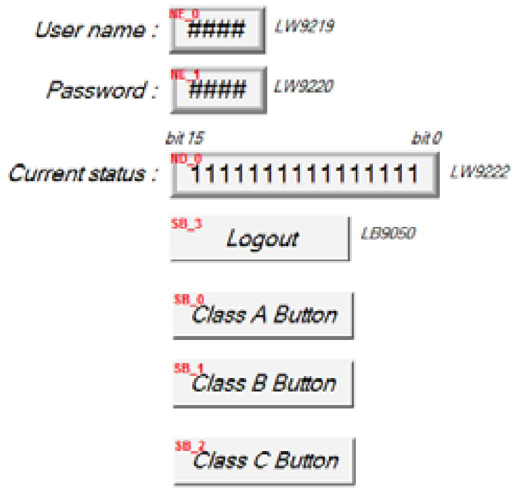
Setting	Description
Safety control	**(Min. press time (sec))** Press and hold the object for longer than the (Min. press time) set here to activate the object. **(Display confirmation request)** After pressing the object, a dialog appears for operation confirmation. If the response to this dialog comes later than the set **(Max. waiting time (sec))**, this dialog disappears automatically and the operation will be canceled. 
Interlock	When this check box is selected, the specified Bit address is used to enable or disable the object. As shown, if LB-0 is ON, the object is enabled. **Hide when disabled:** when the specified Bit is OFF, hide the object. **Grayed label when disabled:** when the specified Bit is OFF, the label of the object turns gray. 
User restriction	Set the security class of the object to be operated by an authorized user. **Object class** “None” means any user can operate this object. Only account “admin” can operate “Administrator” object class. **Disable protection permanently after initial activation** Once the permitted class of the user matches that of the object, the system will stop checking the security class permanently, that means, any user can operate this object freely after it is unlocked. **Display warning message if access denied** When an unauthorized user attempts to operate the object, a warning dialog (Window no. 7) appears. The content of the message in the dialog can be modified.  **Make invisible while protected** When the user’s privilege does not match the object class, hide the object.

10.6. EXAMPLE OF OBJECT SECURITY SETTINGS

The following shows an example of setting object security class:

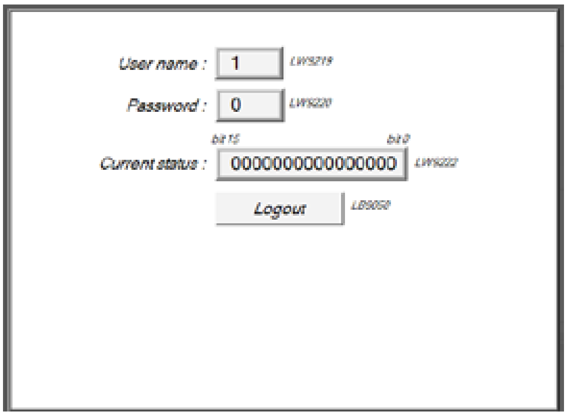
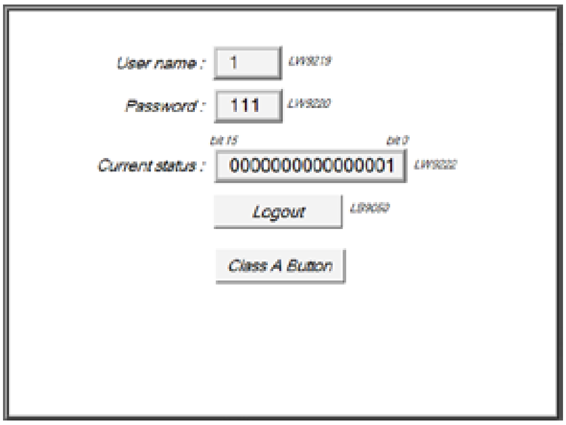
1. Create a project, go to (System Parameter Settings) » (Security) » (General) to enable 3 users:
   - User 1 = Operable class: A
   - User 2 = Operable class: A, B
   - User 3 = Operable class: A, B, C

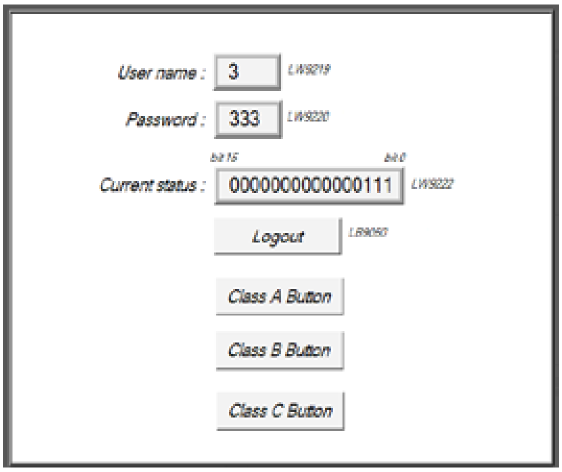
2. Design Window no. 10 as shown:



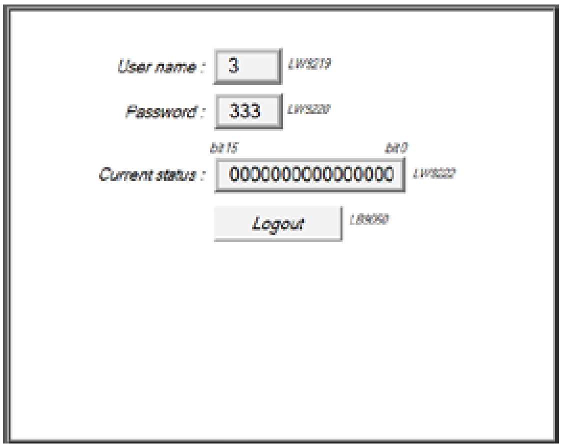
- Create two (Numeric Input) objects:
  - (LW-9219) User no. (1~12), Length = 1word
  - (LW-9220) For entering user password. Length = 2 words
- Create a (Numeric Display) object:
  - (LW-9222) Displays the status of current user. (16-bit Binary)
  - Create a (Set Bit) object
  - (LB-9050) logout
- Create three (Set Bit) objects:
  - Each set to different classes but all select (Made invisible while protected)

3. After setting, please save and compile the project and execute off-line simulation. The below shows how it works when simulating.

	Before entering the password, it displays “0000000000000000”, which means that the user operable object class is “None”. (Class A Button) ~ (Class C Button) objects are classified from “A” to “C” and selected (Made invisible while protected); therefore they are hidden at this moment.
	Enter User 1 password “111”. Since User 1 is only allowed to operate class A objects, (Class A Button) object appears for operating. (LW-9222) bit 0 turns to “1” means that user can operate class A objects.



Enter User 3 password “333”.
Since User 3 is allowed to operate class A, B, C objects, (LW-9222) bit 0 ~ bit 2 turns to “1”, means that user can operate class A ~ C objects.



Click (Logout) button to log out, the system will return to the initial state, and current user can only operate class “None” objects.

Note

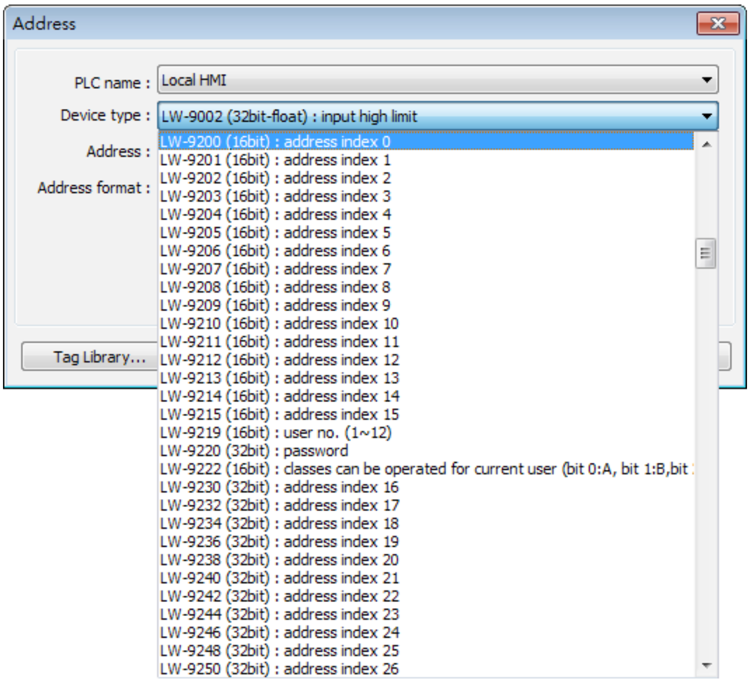
- Password input: if the password is incorrect, (LB-9060) will be ON; if the password is correct, (LB-9060) will be OFF. All user passwords (User 1 to User 12) can be obtained from system registers (LW-9500) ~ (LW-9522), 24 words in total.
- Changing password directly on HMI: when (LB-9061) is set ON, the system will read data in (LW-9500) ~ (LW-9522) to update user password. The new password will be used in future operations. Please note that the user operable object classes will not be changed due to the change of password.

11. INDEX REGISTER

This chapter explains how to use Index Register.

11.1. OVERVIEW

EasyBuilder Pro provides Index Registers for changing addresses flexibly. With Index Registers, user can change the object's read/write address directly on HMI without changing its settings. There are 32 Index Registers, divided into 16-bit and 32-bit.

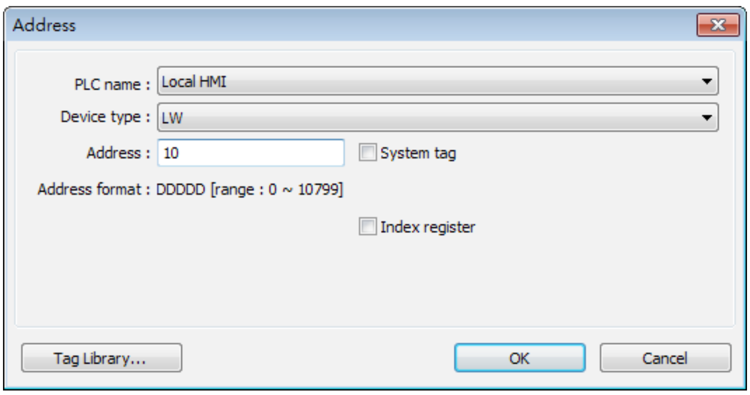


The corresponding address of 16-bit Index Register 0 to 15: LW-9200 (16bit) to LW-9215 (16bit)
The maximum offset range is 65,536 words.
The corresponding address of 32-bit Index Register 16 to 31: LW-9230 (32bit) to LW-9260 (32bit)
The maximum offset range is 4294967296 words.
When using (Index register), the address is designated by the following equation:
The constant set in (Address) + the value in the chosen Index Register.

Note
Index Registers work for the **Word** registers. For **Bit** registers, adding 1 to the value in the Index Register, the offset is 16 bits.

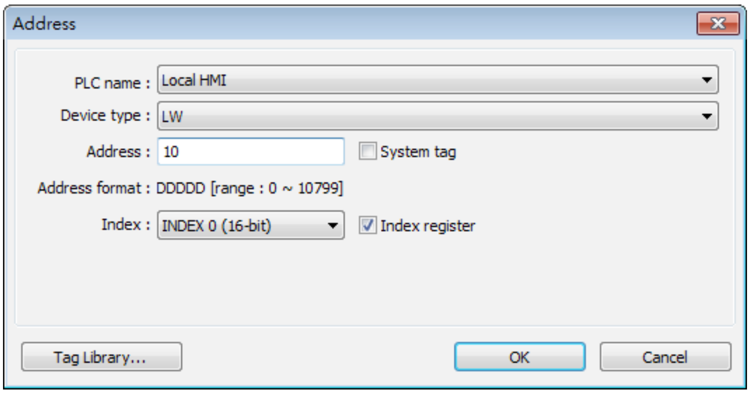
11.2. EXAMPLES OF INDEX REGISTER

The following explains the way to designate the register while Index Register is used.
If not selecting (Index register) check box and set address to (LW-10). The system will directly read / write LW-10.

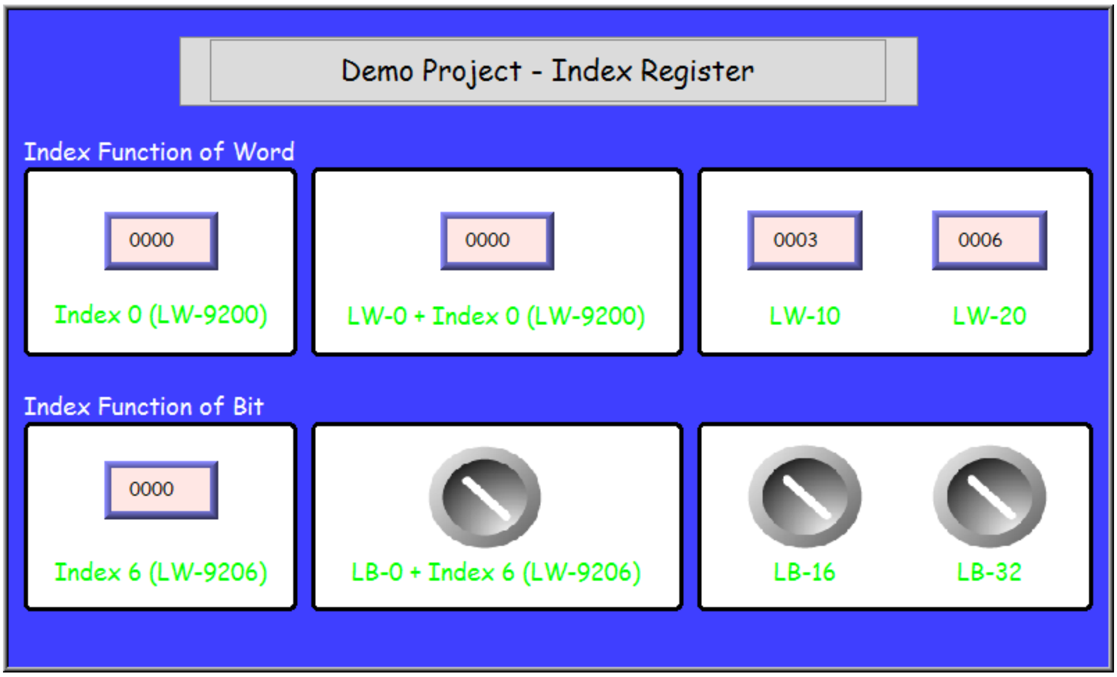


If select (Index register) check box and set (Index) to (INDEX 0 (16-bit)). The system will read / write (LW(10 + value in Index Register 0)).

If the data in (LW-9200) is “5”, the designated address is (LW(10+5)) = (LW-15).

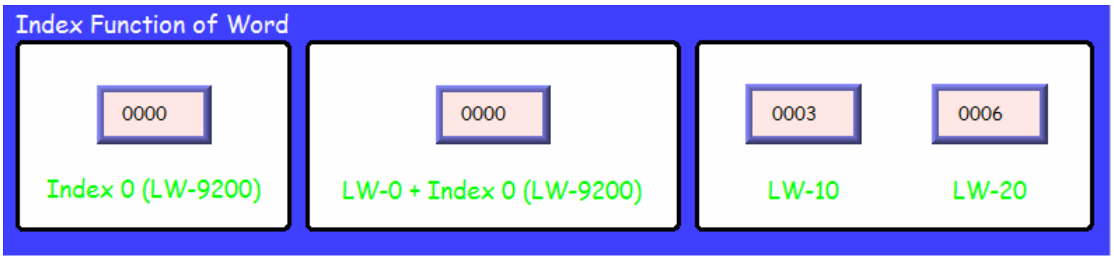


Here’s a demo project shown as an example:

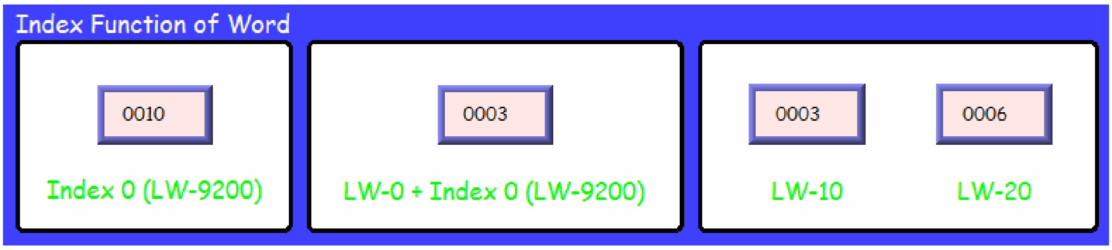


11.2.1. Example 1

The following shows an example of using a Word register and select (Index register). If the value in (LW-0) is 0, in (LW-10) is 3, and in (LW-20) is 6, the result is:



If the value in Index 0 (LW-9200) is 0, then (LW0 + Index 0) = read (LW-0).



If the value in Index 0 (LW-9200) is 10, then (LW0 + Index 0) = read (LW-10) = 3.

11.2.2. Example 2

The following shows an example of using a Bit register and select (Index register).
If the state of (LB-16) is ON, and the state of (LB-32) is OFF.
Since 1 Word = 16 Bit, adding 1 in Index Register, the offset is 16 bits.

If Index 6 (LW-9206) is set to 1, then switch (LB-0 + Index6) reads LB-16 which is in ON state.

If Index 6 (LW-9206) is set to 2, then switch (LB-0 + Index6) reads LB-32 which is in OFF state.

	Note When using Index Registers for Bit register, the offset is 16 bits. For example, if the Bit register is LB-0, and set the value in Index Register to 1, then LB-16 will be activated. If set the value in Index Register to 2, then LB-32 will be activated.

	Download Click the icon to download the demo project. Please confirm your internet connection.

12. KEYBOARD DESIGN AND USAGE

This chapter explains how to design and use keyboard in EasyBuilder Pro.

12.1. OVERVIEW

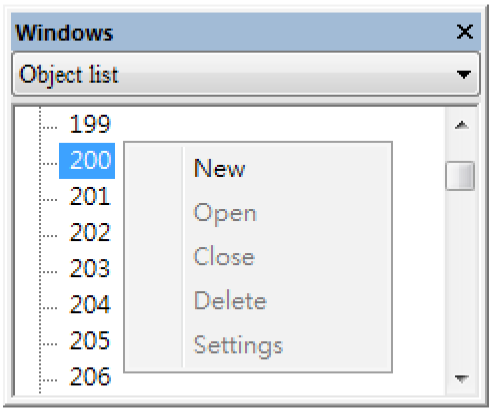
Numeric Input and ASCII Input objects need keyboard as an input tool. Both numeric keyboard and ASCII keyboard are created with Function Key object. Apart from the keyboards provided by EasyBuilder Pro, you can create the keyboard if needed.
The types of the keyboards are:

- Popup Keyboard (with or without title bar)
- Fixed Keyboard
- UNICODE Keyboard

	Note cMT Series uses the built-in keyboard of iPad which cannot be customized. Please skip this chapter if cMT Series is used.

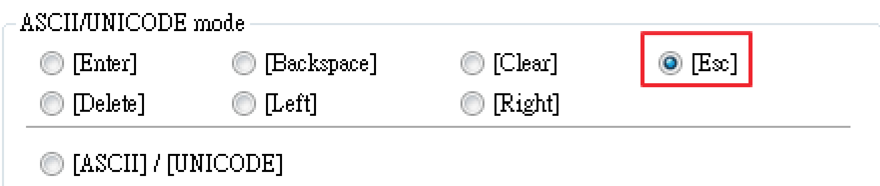
12.2. STEPS TO DESIGN A POPUP KEYBOARD

1. Create and open a window for the new keyboard. For example, set to "window no. 200".

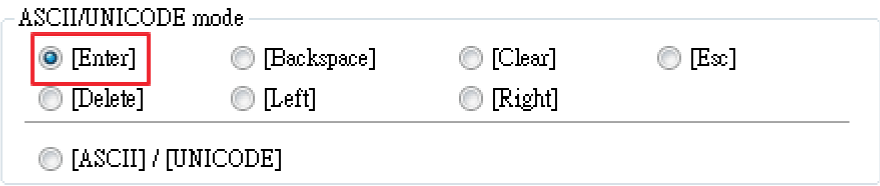


2. Adjust the height and width of "window no. 200" and create a variety of Function Key objects in (ASCII/UNICODE mode).

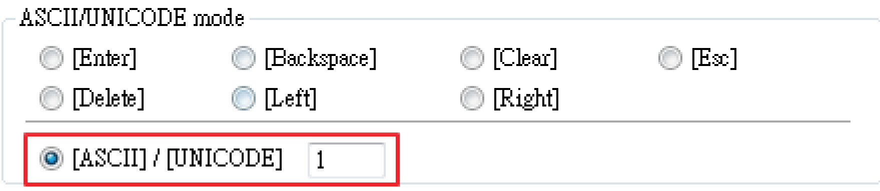
Set one of the Function Key objects as the (Esc) key.



Set another Function Key object as the (Enter) key.

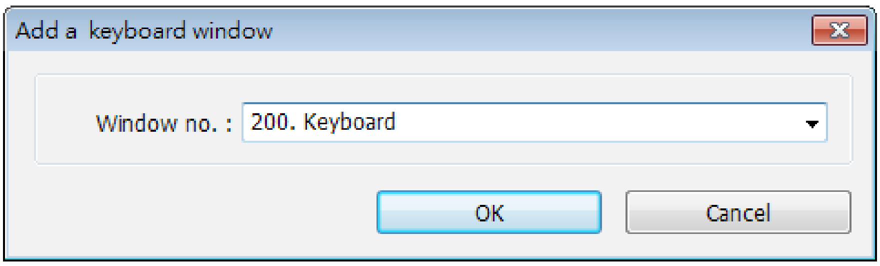


The rest are mostly used to enter numbers.

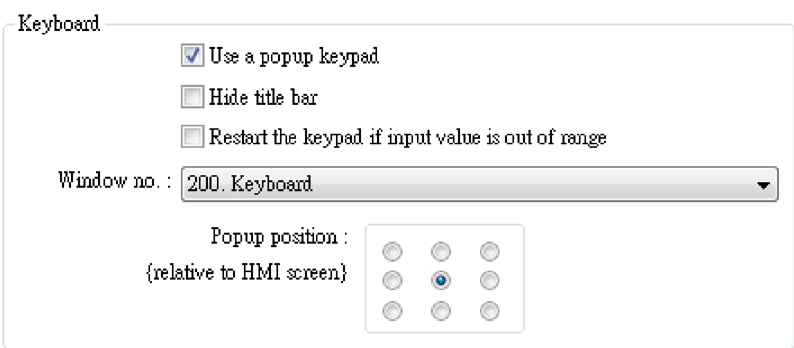


3. Select a suitable picture for each Function Key object and place a picture object at the bottom layer as a background.

4. Select (System Parameter Settings) » (General) » (Keyboard) » (Add) to add “window no. 200”. Up to 32 keyboards can be added.



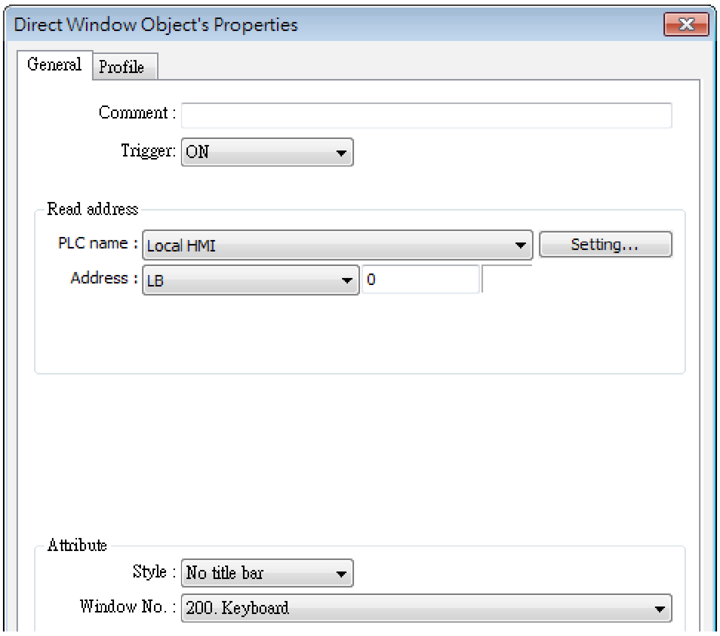
5. After the keyboard window is added, when you create Numerical Input and ASCII Input objects, “200. Keyboard” can be found in (Data Entry) » (Keyboard) » (Window no.). The (Popup position) is for designating the display position of the keyboard on the HMI screen. The system divides the screen into 9 areas.



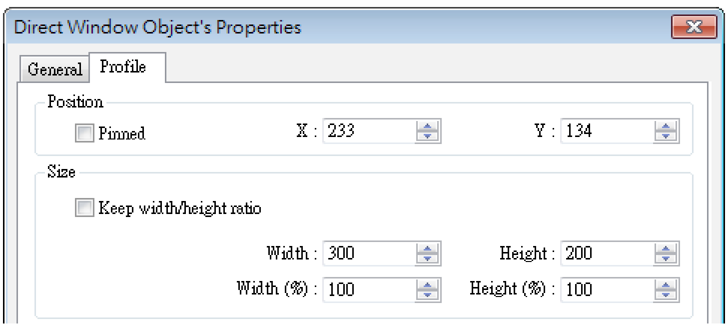
6. Select “200.Keyboard”. When you press Numerical Input or ASCII Input objects on the screen, “window no. 200” will pop up. You can press the keys on the keyboard to enter data.

12.3. STEPS TO DESIGN A KEYBOARD WITH DIRECT WINDOW

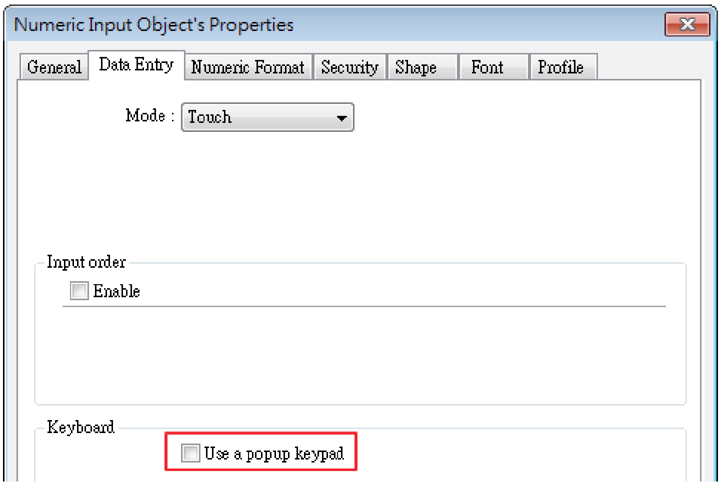
1. Create a Direct Window object and set a read address to activate it.
   In (General) » (Attribute) select (No title bar) and the correct (Window No.).



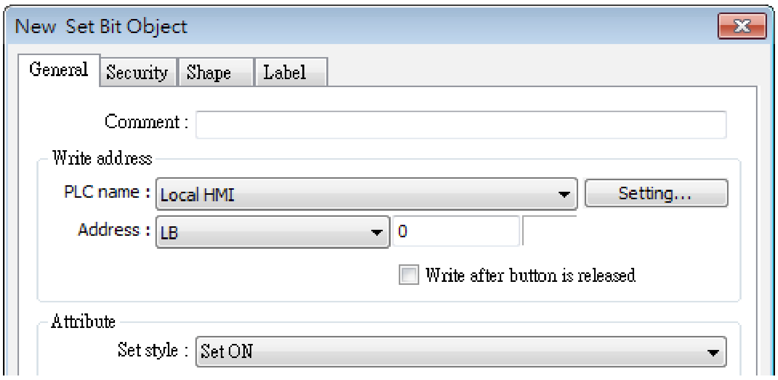
2. Open the (Profile) tab to set the same size as the created keyboard window.



3. Create a Numeric Input object, and don't select (Use a popup keypad) check box.



4. Create a Set Bit object, set address to (LB-0) and set (Set style) to (Set ON). Overlay it on the Numeric Input object. Pressing the Numeric Input object will open the keyboard window.



5. Add Set Bit objects on the (Enter) and (ESC) Function Key objects respectively. Set address to (LB-0) and (Set style) to (Set OFF). In this way when pressing either (Enter) or (ESC) key will close the keyboard window.

12.4. STEPS TO DESIGN A FIXED KEYBOARD ON SCREEN

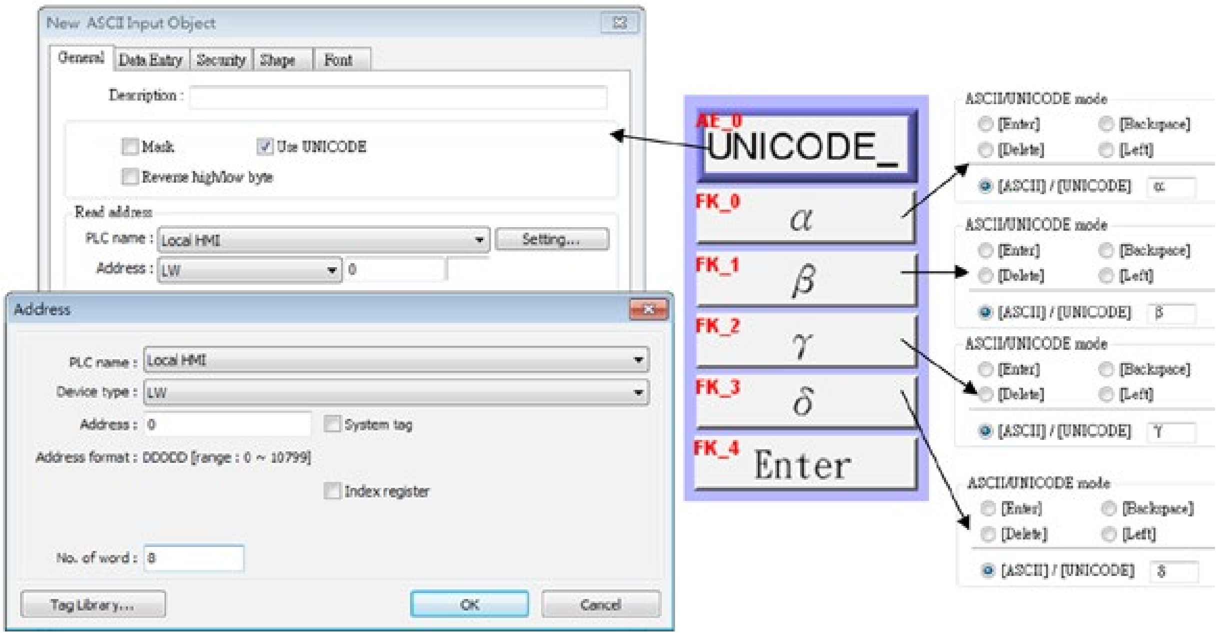
You can also place a fixed keyboard on the screen instead of popup keyboard or Direct Window. This type of keyboard can't be moved or closed.

1. Create a Numeric Input object, in (Data Entry) » (Keyboard) don't select (Use a popup keypad) check box.
2. Use Function Key objects to design the keyboard and place it on the screen.
3. Press the Numeric Input object and enter a value with Function Key objects directly.

12.5. STEPS TO DESIGN A UNICODE KEYBOARD

The following steps explain how to create a UNICODE keyboard with Function Key objects.

1. Place an ASCII Input object on the window and select (Use UNICODE) check box.
2. Create Function Key objects as shown in the following figure, and an (Enter) key. A simple UNICODE keyboard is created.



Note
You can group up the elements of the designed keyboard and save to Group Library for future use.

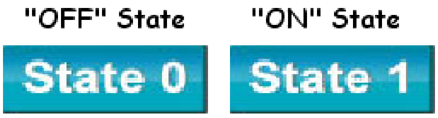
13. OBJECTS

This chapter explains how to use different objects.

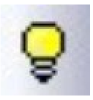
13.1. BIT LAMP

13.1.1. Overview

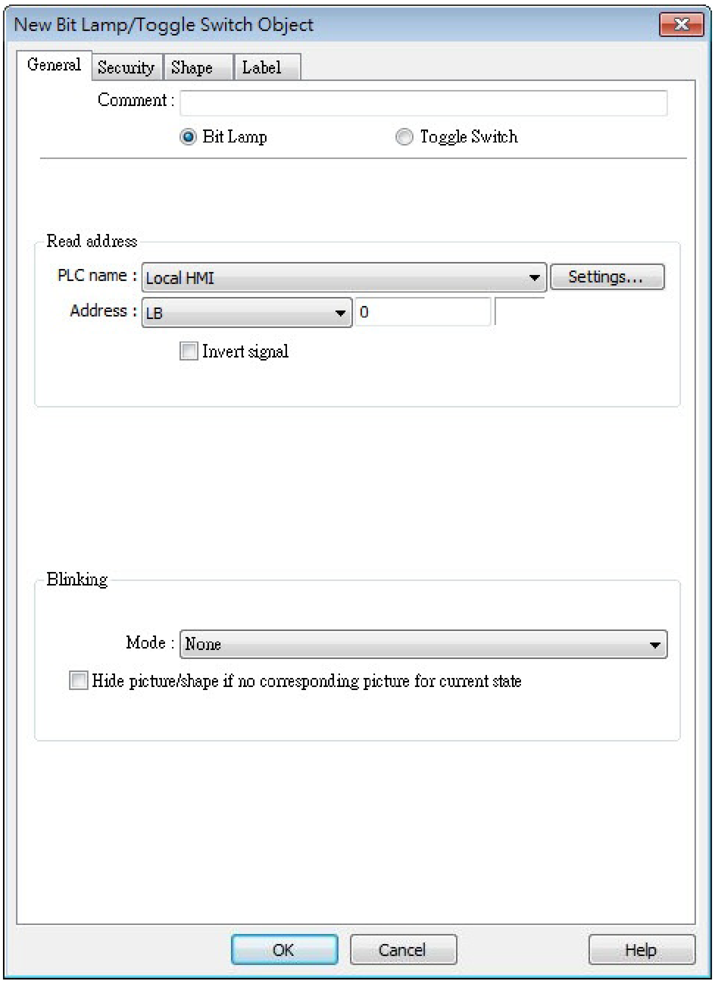
Bit Lamp object displays the state of a designated bit address. If the bit state is OFF, the State 0 shape will be displayed. If the bit state is ON, the State 1 shape will be displayed.



13.1.2. Configuration

	Click the Bit Lamp icon on the toolbar to open a Bit Lamp object property dialog box. Set up the properties, press OK button, and a new Bit Lamp object will be created.

13.1.2.1. General Tab

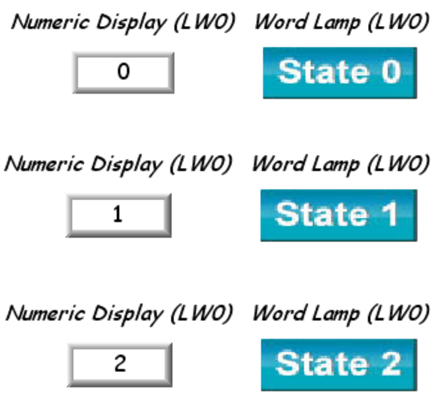


Setting	Description
Comment	User can describe the information of the object. **Bit Lamp / Toggle Switch** Switch between Bit Lamp and Toggle Switch features.
Read address	Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the bit device that controls the (Bit Lamp) object. Users can also set address in (General) tab while adding a new object. **Invert signal** Reverses the display of ON / OFF states. For example, if (Invert signal) check box is selected, when the designated bit is OFF, the object displays ON state.
Blinking	The appearance of the object may alternate between states when the bit is ON or OFF. **Mode:** ▪ None No blinking. ▪ Alternating image on state 0 The appearance of the object alternates between State 0 and 1 when the bit is OFF. ▪ Alternating image on state 1 The appearance of the object alternates between State 0 and 1 when the bit is ON. ▪ Blinking on state 0 The State 0 appearance of the object will blink when the bit is OFF. ▪ Blinking on state 1 The State 1 appearance of the object will blink when the bit is ON. ▪ Hide picture/shape if no corresponding picture for current state If selected, when there are not enough pictures to represent all the states, hides the picture. Otherwise, displays the last state.

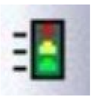
13.2. WORD LAMP

13.2.1. Overview

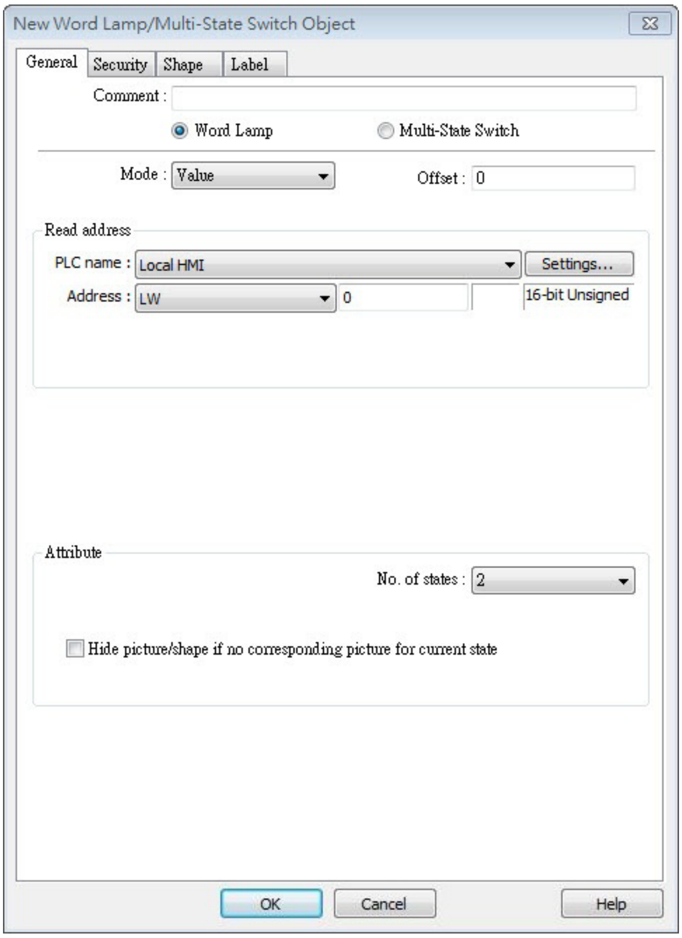
Word Lamp object displays the state according to the value of a designated word register. Up to 256 states are available. When the value of the register is 0, State 0 appearance of the object is displayed, and with the register value being 1 the object displays State 1, and so on.

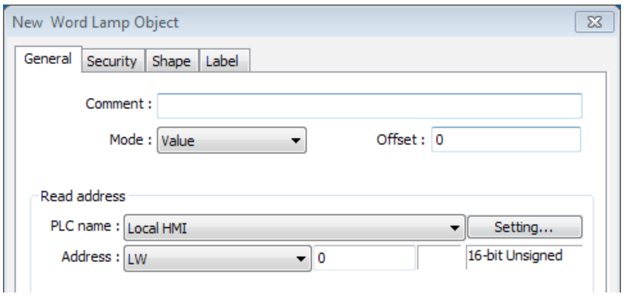


13.2.2. Configuration

	Click the Word Lamp icon on the toolbar to open a Word Lamp object property dialog box. Set up the properties, press OK button, and a new Word Lamp object will be created.

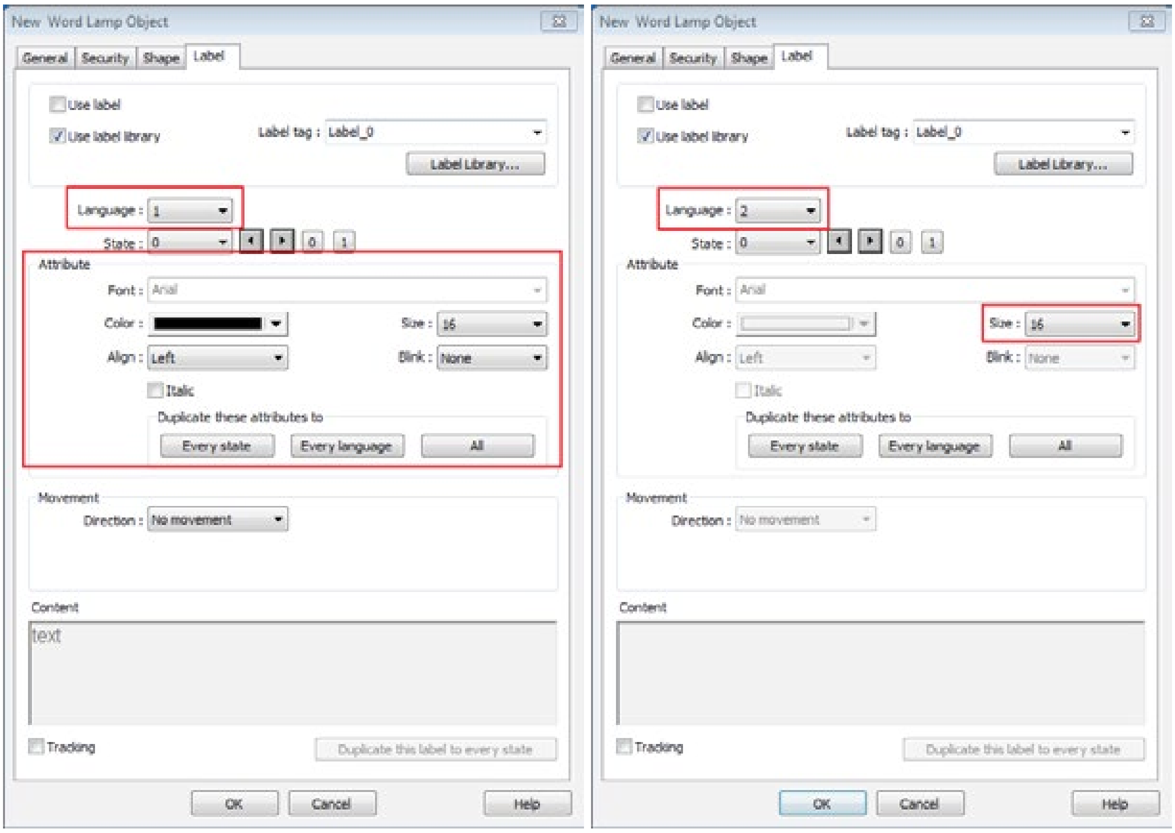
13.2.2.1. General Tab



<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Comment</td><td>User can describe the information of the object.
Word Lamp / Multi-State Switch
Switch between Word Lamp and Multi-State Switch features.</td></tr>
<tr><td>Mode / offset</td><td>Word Lamp object offers the following three modes:
Value
The state is displayed according to the value in the designated word address and plus the (Offset).
As shown below, if the value within LW-200 is 3, since the offset is set to 3, the shape of state 6 is displayed. (value 3 + offset 3)


LSB
Convert the value from decimal to binary. The least significant active bit in a binary data word selects the state displayed.
<table>
<tr><th>Decimal</th><th>Binary</th><th>Displayed state</th></tr>
<tr><td>0</td><td>0000</td><td>State 0 displayed. All the bits are 0.</td></tr>
<tr><td>1</td><td>0001</td><td>State 1 displayed. The least significant active bit is bit 0.</td></tr>
<tr><td>2</td><td>0010</td><td>State 2 displayed. The least significant active bit is bit 1.</td></tr>
<tr><td>3</td><td>0011</td><td>State 1 displayed. The least significant active bit is bit 0.</td></tr>
<tr><td>4</td><td>0100</td><td>State 3 displayed. The least significant active bit is bit 2.</td></tr>
<tr><td>5</td><td>0101</td><td>State 1 displayed. The least significant active bit is bit 0.</td></tr>
<tr><td>6</td><td>0110</td><td>State 2 displayed. The least significant active bit is bit 1.</td></tr>
<tr><td>7</td><td>0111</td><td>State 1 displayed. The least significant active bit is bit 0.</td></tr>
<tr><td>8</td><td>1,000</td><td>State 4 displayed. The least significant active bit is bit 3.</td></tr>
</table>
Change state by time
The state displayed changes on a time basis. The frequency can be set.</td></tr>
<tr><td>Read address</td><td>Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the word device that controls the (Word Lamp) object. Users can also set address in (General) tab while adding a new object.</td></tr>
<tr><td>Attribute</td><td>No. of states
The number of states is utilized by the object. The state is numbered from 0, so the number of states minus 1 will be the state number. If the value within the word register is ≥ (No. of states) defined in Attribute, the highest state will be displayed.
If the number of states is set to 8, the valid states will be 0, 1, 2, …, 7. In this case if the word value is 8 or higher, the system will display the state 7 shape.
Hide picture/shape if no corresponding picture for current state
If selected, when there are not enough pictures to represent all the states, hides the picture. Otherwise, displays the last state.</td></tr>
</table>

	Note In (Label) tab, Language 1 determines the relevant settings of the font. For Language 2~8, only the font size can be changed and other settings follow Language 1.

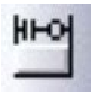


13.3. SET BIT

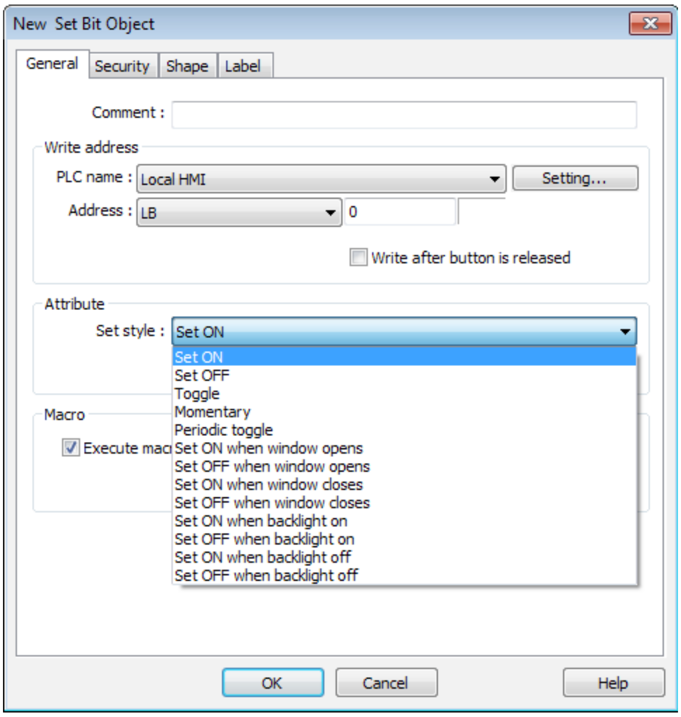
13.3.1. Overview

The Set Bit object provides two operation modes: manual or automatic. Manual mode can trigger a designated bit address to change the state between ON and OFF when the object is touched. In automatic mode, the bit is automatically activated when a pre-defined condition occurs; touching the button will not be effective.

13.3.2. Configuration

	Click the Set Bit icon on the toolbar to open a Set Bit object property dialog box. Set up the properties, press OK button, and a new Set Bit object will be created.

13.3.2.1. General Tab



<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Write address</td><td>Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the bit device that controls the Set Bit object. Users can also set address in (General) tab while adding a new object.
Write after button is released
If this function is selected, the action is delayed till button is released; otherwise, the action is executed once the button is pressed. This function does not work with momentary buttons.</td></tr>
<tr><td>Mode / offset</td><td>
<table>
<tr><th>Set style</th><th>Description</th></tr>
<tr><td>Set ON</td><td>Set ON the designated bit of the device.</td></tr>
<tr><td>Set OFF</td><td>Set OFF the designated bit of the device.</td></tr>
<tr><td>Toggle</td><td>Alternates the bit state each time pressed.</td></tr>
<tr><td>Momentary</td><td>Holds the bit ON only while button is pressed.</td></tr>
<tr><td>Periodical toggle</td><td>Set a designated bit ON and OFF at a set time interval. Time interval can be selected; the range is from 0.1 to 25.5 seconds.</td></tr>
<tr><td>Set ON when window opens</td><td>Set ON the bit within the window when the window opens.</td></tr>
<tr><td>Set OFF when window opens</td><td>Set OFF the bit within the window when the window opens.</td></tr>
<tr><td>Set ON when window closes</td><td>Set ON the bit within the window when the window closes.</td></tr>
<tr><td>Set OFF when window closes</td><td>Set OFF the bit within the window when the window closes.</td></tr>
<tr><td>Set ON when backlight on
(N/A for cMT)</td><td>Set the bit ON when the backlight is turned ON.</td></tr>
<tr><td>Set OFF when back-light on
(N/A for cMT)</td><td>Set the bit OFF when the backlight is turned ON.</td></tr>
<tr><td>Set ON when backlight off
(N/A for cMT)</td><td>Set the bit ON when the backlight is turned OFF.</td></tr>
<tr><td>Set OFF when back-light off
(N/A for cMT)</td><td>Set the bit OFF when the backlight is turned OFF.</td></tr>
</table>
</td></tr>
<tr><td>Macro</td><td>Set Bit object can trigger the start of a Macro routine when the Macro has been created in advance.
For more information, see “18 Macro Reference”.</td></tr>
<tr><td>Trigger mode</td><td>If (Set style) is set to (Toggle), there is a further selection to make of whether the macro operates after Off to ON, ON to OFF transition, or at both of the changes of state.</td></tr>
</table>

13.4. SET WORD

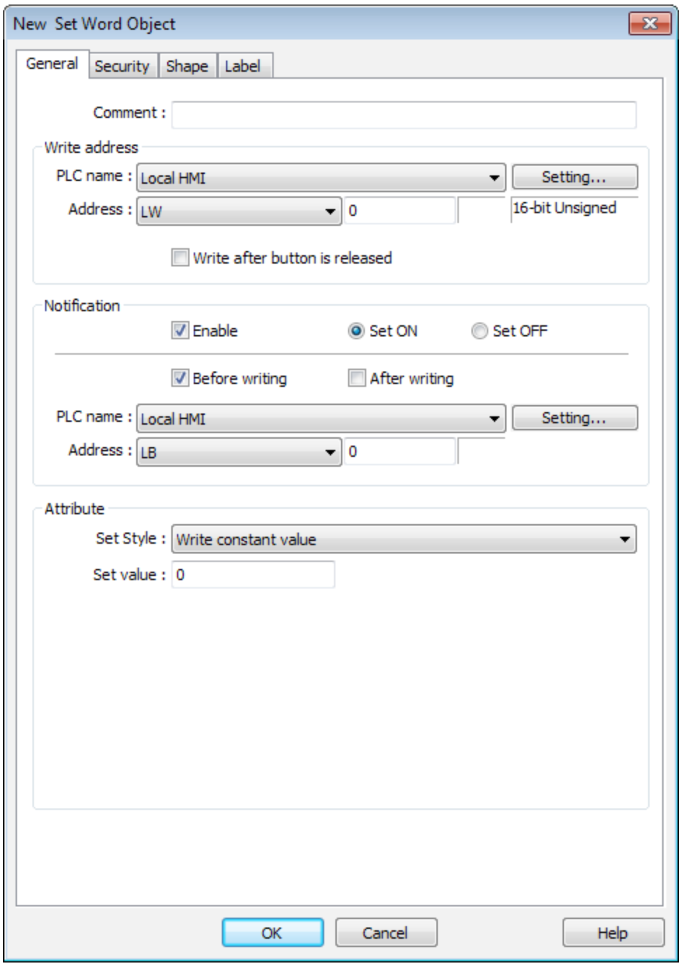
13.4.1. Overview

The Set Word object provides two operation modes: manual or automatic. Manual mode can change the value in a designated word address when the object is touched. In automatic mode, the word register is automatically activated when a pre-defined condition occurs; touching the button will not be effective.

13.4.2. Configuration

	Click the Set Word icon on the toolbar to open a Set Word object property dialog box. Set up the properties, press OK button, and a new Set Word object will be created.

13.4.2.1. General Tab



Setting	Description
Write address	Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the word device that controls the Set Word object. Users can also set address in (General) tab while adding a new object. **Write after button is released** If this function is selected, the action is delayed till button is released; otherwise, the action is executed once the button is pressed.
Notification	If this check box is selected, it will notify a designated bit address (setting ON or OFF). **Before writing / After writing** Set the state of the designated bit address before or after the manual operation.
Attribute	**Set Style** Select the button action from the drop down list, see Example 2. **Dynamic limits** Set the (Bottom limit) and (Upper limit) by a designated register, see Example 1.

Example 1

Set the (Bottom limit) and (Upper limit) by a designated register. When Dynamic Address is LW-n, where n is an arbitrary number, the rule of setting Upper / Bottom limit is:

Content	16-bit	32-bit
Dynamic address	LW-n	LW-n
Bottom limit	LW-n	LW-n
Upper limit	LW-n+1	LW-n+2

When Dynamic Address is LW-100, the rule of setting Upper / Bottom limit is:

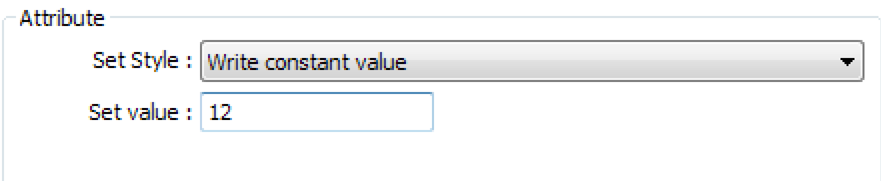
Content	16-bit	32-bit
Dynamic address	LW-100	LW-100
Bottom limit	LW-100	LW-100
Upper limit	LW-101	LW-102

Example 2

The available button actions are:

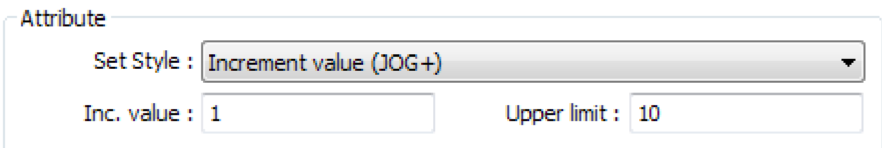
- Write constant value

  Preset a register with the value entered. Each time when the button is pressed, it writes the (Set value) to the designated register. Data format is as set by the (Write address); it can be 16-bit BCD, 32-bit BCD, …32-bit float. As shown below, when the button is pressed, preset the register with 12.



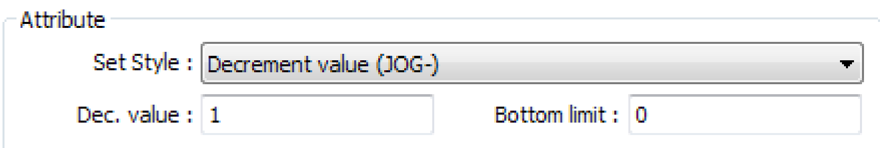
- Increment value (JOG+)

  Increase value in register by a set amount in (Inc. value), each time when the button is pressed, up to the (Upper limit). As shown below, each button press increases the value in the register by 1 until the value is 10.



- Decrement Value (JOG-)

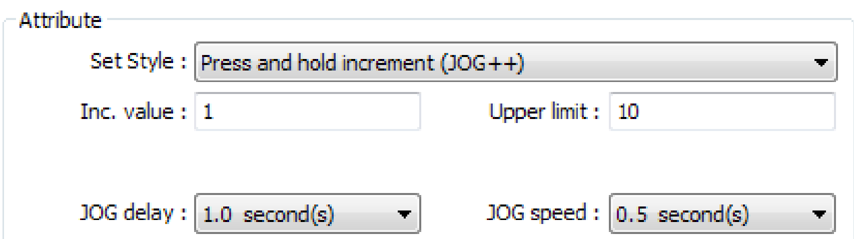
  Decrease value in register by a set amount in (Dec. value), each time when the button is pressed, down to the (Bottom limit). As shown below, each button press decreases the value in the register by 1 until the value is 0.



- Press and hold increment (JOG++)

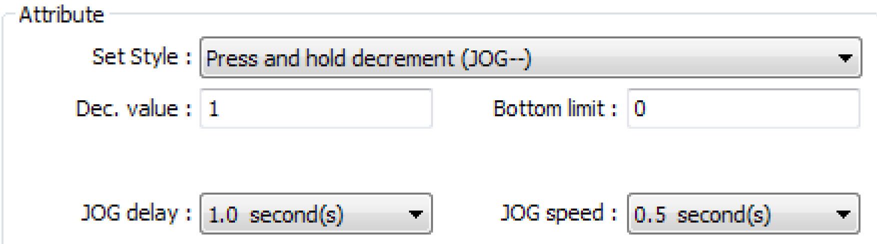
  When the button is held longer than a set time in (JOG delay), it will increase the value in a register by a set amount: (Inc. value) at a set rate: (JOG speed), to the (Upper limit).

  As shown below, when the button is pressed, it increases the value in the designated register by 1. When the button is held longer than 1 second, it increases the value in register by 1 every 0.5 second, till the value is 10.



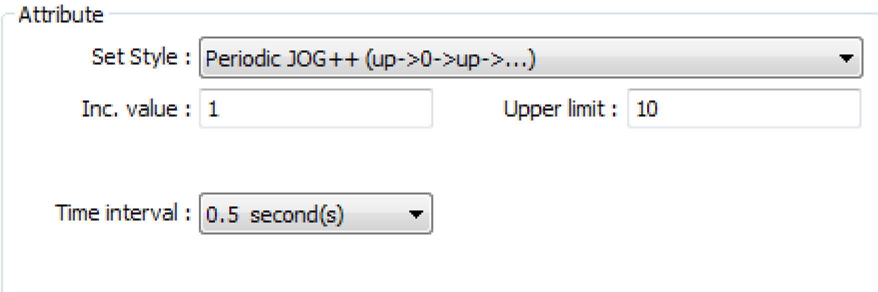
- Press and hold increment (JOG--)

When the button is held longer than a set time in (JOG delay), it will decrease the value in a register by a set amount: (Dec. value) at a set rate: (JOG speed), to the (Bottom limit).
As shown below, when the button is pressed, it decreases the value in the designated register by 1. When the button is held longer than 1 second, it decreases the value in register by 1 every 0.5 second, till the value is 0.



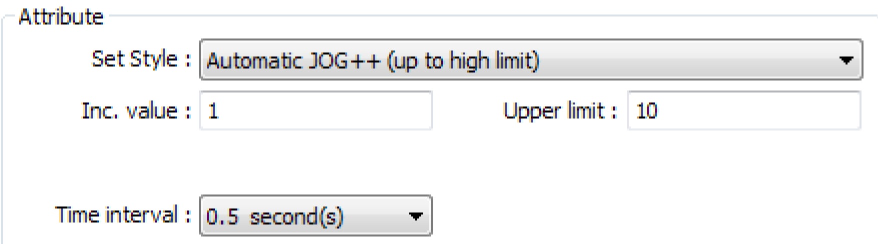
- Periodic JOG++

This automatic function increases the value in the register by a set amount: (Inc. value), at a set rate: (Time interval), to the (Upper limit). As shown below, the system will automatically increase the value in the register by 1 every 0.5 second, till the value is 10. Then the value returns to 0 and add 1 every 0.5 second again.



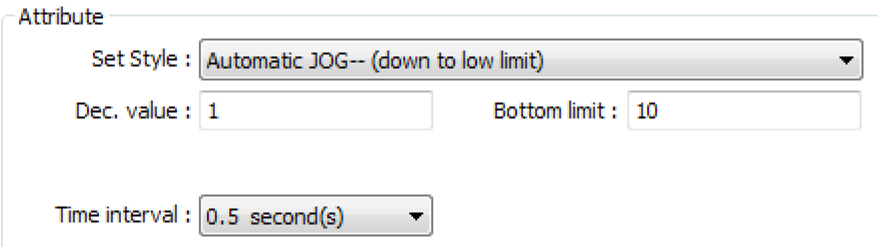
- Automatic JOG++

This automatic function increases the value in the register by a set amount: (Inc. value), at a set rate: (Time interval), to the (Upper limit).then holds this value. As shown below, the system will automatically increase the value in the register by 1 every 0.5 second, till the value is 10, and then stop.

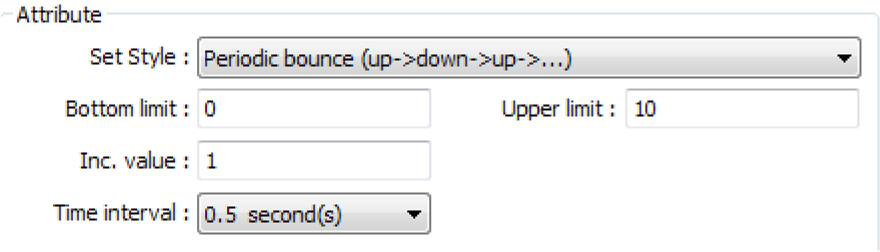


- Automatic JOG--

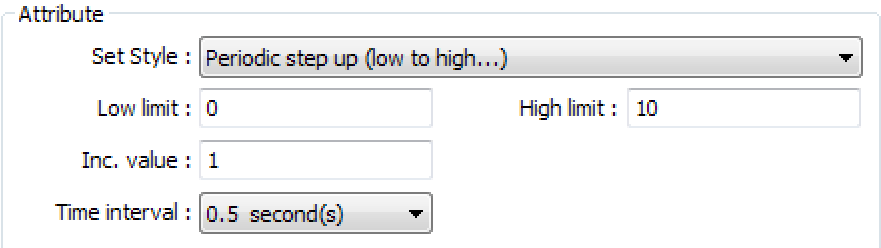
This automatic function decreases the value in the register by a set amount: (Dec. value), at a set rate: (Time interval), to the (Bottom limit).then holds this value. As shown below, the system will automatically increase the value in the register by 1 every 0.5 second, till the value is 10, and then stop.



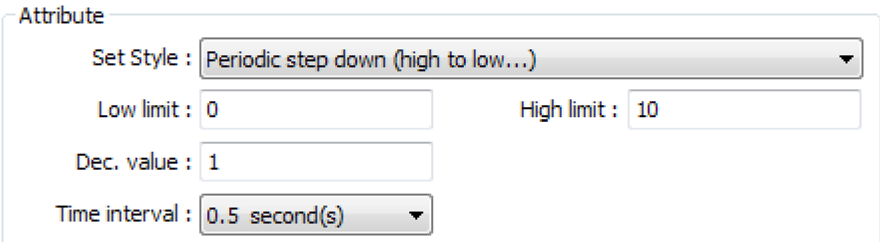
- Periodic bounce
  Increases the word address value to the (Upper limit) by a (Inc. value) at a set rate in (Time interval), then decreases to the (Bottom limit) by the same value at the same rate. As shown below, the system will increase the value in the designated register by 1 every 0.5 second, till the value is 10, and then decrease the value by 1 every 0.5 second till the value is 0 whenever the screen is active.



- Periodic step up
  Step up to the (High limit) by (Inc. value) at a set rate in (Time interval), then reset immediately to the (Low limit). The action repeats whenever the screen is active. As shown below, the system will increase the value in the designated register by 1 every 0.5 second, till the value is 10, and then reset to 0 and increase again, and the action repeats.

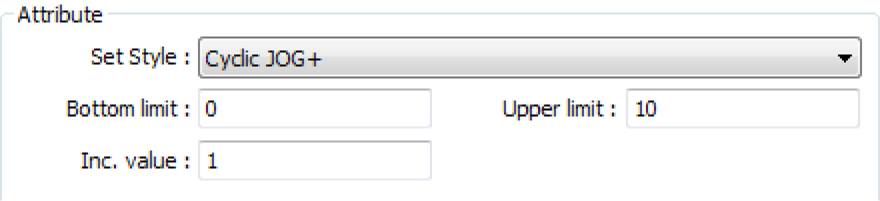


- Periodic step down
  Step down to the (Low limit) by (Dec. value) at a set rate in (Time interval), then reset immediately to the (High limit). The action repeats whenever the screen is active. As shown below, the system will decrease the value in the designated register by 1 every 0.5 second, till the value is 0, and then reset to 10 and decrease again, and the action repeats.

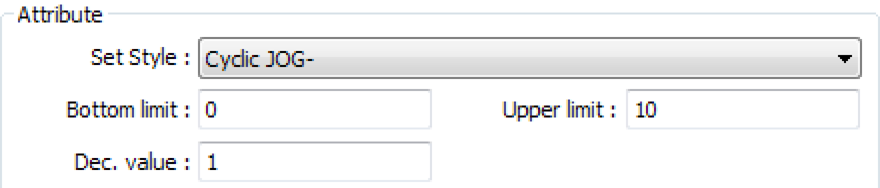


- Set when window opens / Set when window closes
  Automatic function occurs whenever the screen is active. The value entered in (Set value) is set into the word address when the action occurs. If (Set value) is set to 5, when the window opens / closes, the system enters 5 into the designated register
- Set when backlight on / Set when backlight off.
  Automatic function occurs whenever the backlight is active. The value entered in (Set value) is set into the word address when the action occurs. If (Set value) is set to 5, when the backlight turns ON / OFF, the system sets 5 into the designated register

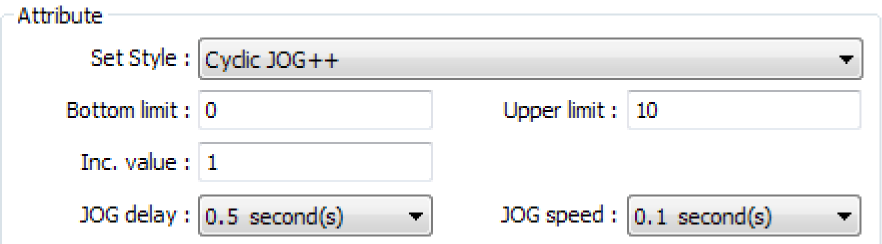
- Cyclic JOG+
Each time when the button is pressed, increases the word address value to the (Upper limit) by (Inc. value) then reset to the (Bottom limit). As shown below, each time when pressing the button, the system will increase the value in the designated register by 1, till the value is 10, and then reset to 0 and increase again by pressing the button



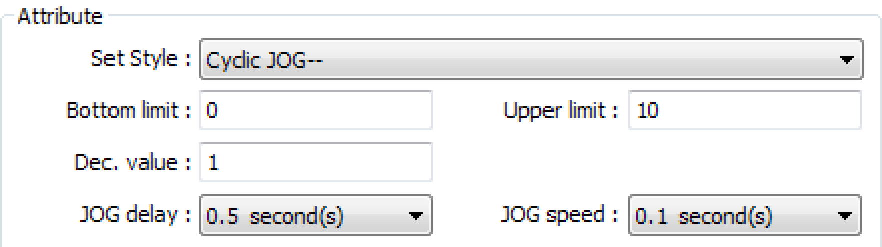
- Cyclic JOG-
Each time when the button is pressed, decrease the word address value to the (Bottom limit) by (Dec. value) then reset to the (Upper limit). As shown below, each time when pressing the button, the system will decrease the value in the designated register by 1, till the value is 0, and then reset to 10 and decrease again by pressing the button.



- Cyclic JOG++
When the button is held longer than a set time in (JOG delay), it increases the value in a register by a set amount in (Inc. value) at a set rate in (JOG speed), to the (Upper limit), then reset to the (Bottom limit). As shown below, when the button is held longer than 0.5 second, increase the value in the designated register by 1 every 0.1 second, till the value is 10, and then reset to 0 and increase again by holding the button.



- Cyclic JOG--
When the button is held longer than a set time in (JOG delay), decrease the value in a register by a set amount in (Dec. value) at a set rate in (JOG speed), to the (Bottom limit), then reset to the (Upper limit). As shown below, when the button is held longer than 0.5 second, decrease the value in the designated register by 1 every 0.1 second, till the value is 0, and then reset to 10 and decrease again by holding the button.

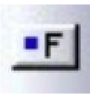


13.5. FUNCTION KEY

13.5.1. Overview

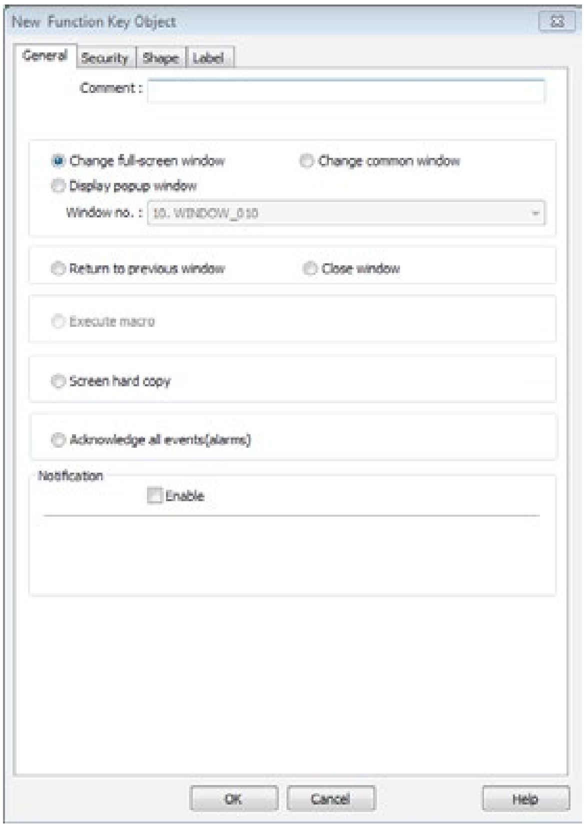
The Function Key object can be used for several tasks, such as switching between windows, keypad design, Macro execution, screen hardcopy and setting USB security key.

13.5.2. Configuration

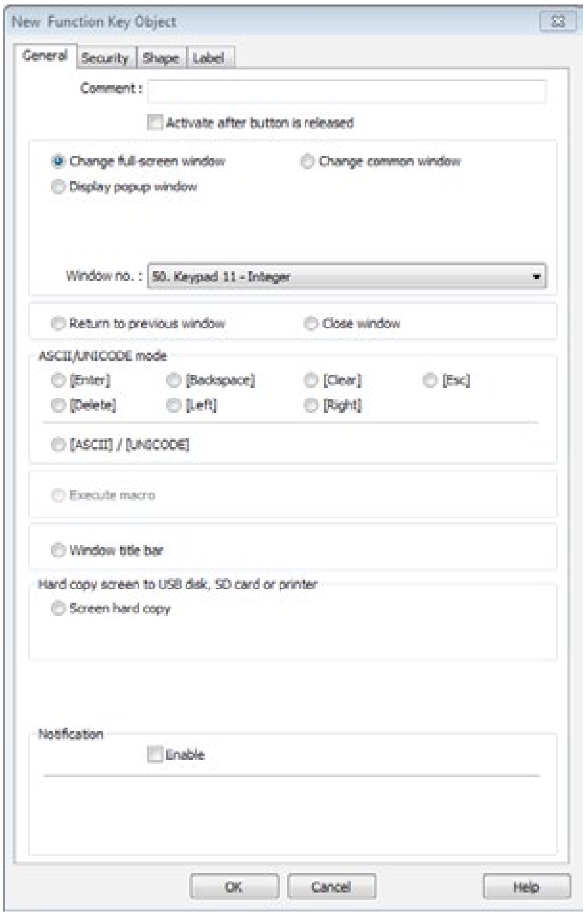
	Click the Function Key icon on the toolbar to open a Function Key object property dialog box. Set up the properties, press OK button, and a new Function Key object will be created.

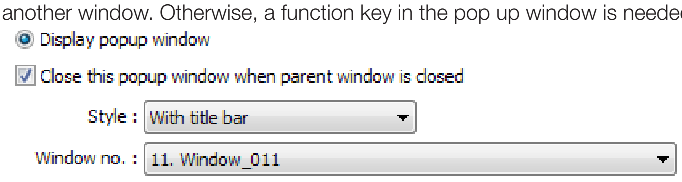
13.5.2.1. General Tab

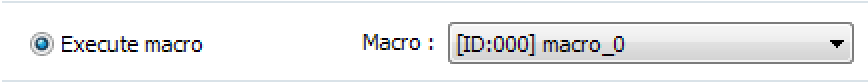
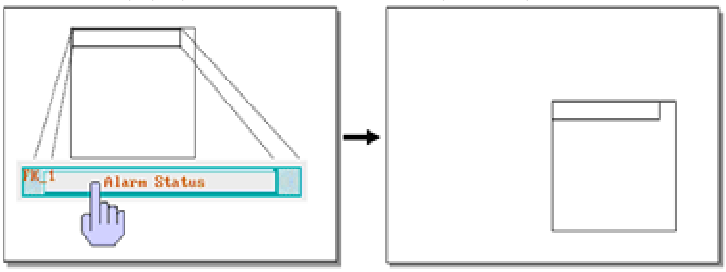
- cMT Series

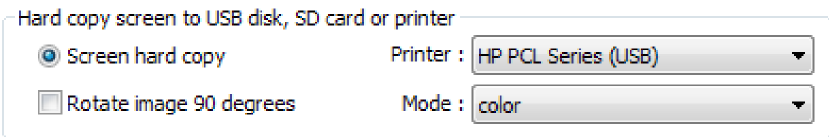


▪ eMT, iE, XE, mTV Series



Setting	Description
Activate after button is released	If this function is selected, the action is delayed till button is released; otherwise, the action is executed once the button is pressed.
Change window	**Change full-screen window:** change to another base window. **Change common window:** change common window. **Display popup window:** a pop-up window displays in the base window. If (Close this popup window when parent window is closed) check box is selected, the pop up window will be closed when change the base window to another window. Otherwise, a function key in the pop up window is needed to close it.  **Return to previous window:** if this is selected, the Function Key will change from the current screen to the previous one displayed. For example, when window no. 10 is changed to window no. 20, press the function key to return to window no. 10. This function is only available for base window. **Close window:** close any active pop-up windows, message windows included.
ASCII/ UNICODE mode	Configures the button as a keypad key, and the character it enters, via (Numeric) or (ASCII) objects. **Enter:** same as the keyboard's "Enter" function. **Backspace:** same as the keyboard's "Backspace" function. **Clear:** clear the value in the word register. **Esc:** same as the (Close window) function; it is used to close the keyboard window. **Delete:** same as the keyboard's "Delete" function, deletes the number or character on the right side of the text cursor. **Left:** same as the keyboard's "←" key moves the text cursor to the left side of the previous number or character. **Right:** same as the keyboard's "→" key moves the text cursor to the left side of the next number or character. **ASCII/UNICODE:** specify the character to be entered by this key.

<table>
<tr><td>Execute macro</td><td>Select this check box to execute one of the Macros from the drop down list that has already been configured by users.
☞ For more information, see “18 Macro Reference”.

</td></tr>
<tr><td>Window title bar</td><td>Function Key defined can be used to move a pop-up window which has no (window title bar) to a preferred position on screen. Select the pop-up window and then click on a preferred position, the window will be moved.


Select the window title bar firstly.
Touching the screen for the new position the popup window will be moved.</td></tr>
<tr><td>Hard copy screen to USB disk, SD card or printer</td><td>Print the current window. Before using this function, choose a printer model in (System Parameter Settings) » (Model) » (Printer).
If a monochrome printer is used, selecting (grayscale) can provide a better print result, but the text may not be clearly printed. To improve text printing, avoid using (grayscale).

</td></tr>
<tr><td>Screen hard copy (cMT Series)</td><td>Output current screen to iPad Photos.</td></tr>
<tr><td>Acknowledge all events (alarms) (cMT Series)</td><td>Acknowledge all events once by pressing the Function Key.</td></tr>
<tr><td>Import user data / Use (USB Security Key)</td><td>A Function Key can be used to import the e-mail contacts or user accounts set, also, to log in using USB Security Key.


Data Position
Select the external device to store data from (SD card) or (USB disk).
Account import mode
If (Overwrite) is selected, the existing accounts will be overwritten with new accounts in the external device after importing. If (Append) is selected, HMI will append more accounts while the old accounts still exist.
Delete file after importing user accounts
If select this check box, the system will delete the account data saved in the external device after importing, this can prevent the account data from leaking out.</td></tr>
<tr><td>Notification</td><td>If this selection is enabled, it will notify a designated bit address to set ON or OFF, each time the button is pressed.</td></tr>
</table>

Note

(Overwrite) is the only option when importing the e-mail contacts. This means that all existing contacts will be removed first, and then the new contacts are added.

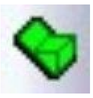
☞ For more information, see “6 Window Operations”, “12 Keyboard Design and Usage”, “36 Administrator Tools”.

13.6. TOGGLE SWITCH

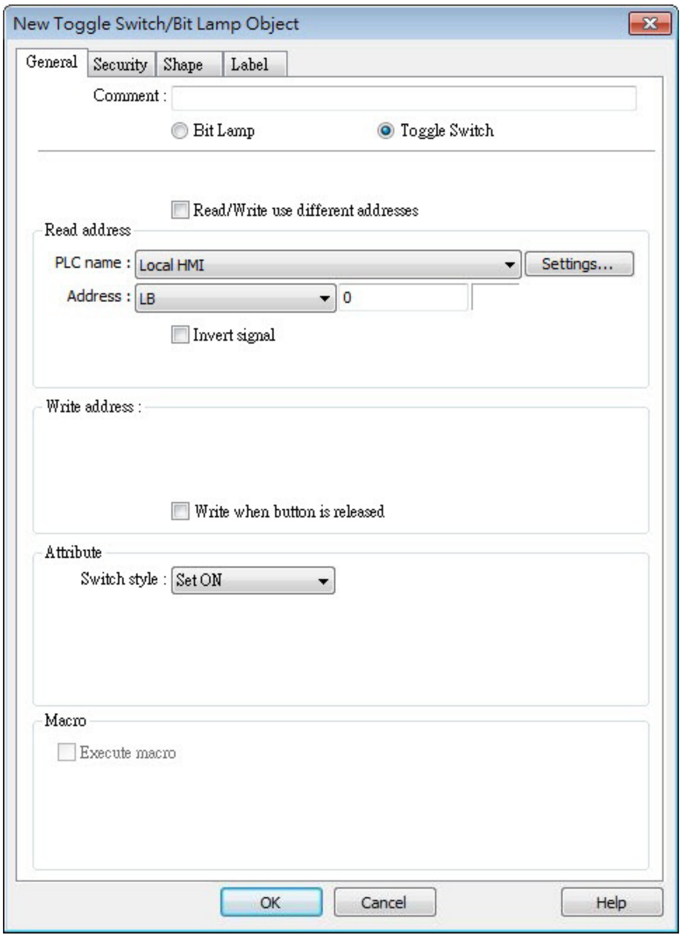
13.6.1. Overview

Toggle Switch object is a combination of Bit Lamp object and Set Bit object. The appearance of the object is controlled by the ON / OFF state of the read bit address. As well, pressing the button sets the value in the bit address according to the settings.

13.6.2. Configuration

	Click the Toggle Switch icon on the toolbar to open a Toggle Switch object property dialog. Set up the properties, press OK button and a new Toggle Switch object will be created.

13.6.2.1. General Tab



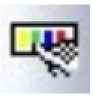
<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Comment</td><td>User can describe the information of the object.
Bit Lamp / Toggle Switch
Switch between Bit Lamp and Toggle Switch features.</td></tr>
<tr><td>Read address</td><td>Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the bit device that controls the (Toggle Switch) object. Users can also set address in (General) tab while adding a new object.
Invert signal
Reverses the display of ON / OFF states. For example, if (Invert signal) check box is selected, when the designated bit is OFF, the object displays ON state.</td></tr>
<tr><td>Write address</td><td>Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the bit device that controls the (Toggle Switch) object. Users can also set address in (General) tab while adding a new object. The address can be the same or different from (Read address).
Write after button is released
If this function is selected, the action is delayed till button is released, otherwise, the action is executed once the button is pressed. This function does not work with momentary buttons.</td></tr>
<tr><td>Attribute</td><td>
<table>
<tr><th>Set style</th><th>Description</th></tr>
<tr><td>Set ON</td><td>Set ON the designated bit of the device.</td></tr>
<tr><td>Set OFF</td><td>Set OFF the designated bit of the device.</td></tr>
<tr><td>Toggle</td><td>Alternates the bit state each time pressed.</td></tr>
<tr><td>Momentary</td><td>Holds the bit ON only while button is pressed.</td></tr>
</table>
</td></tr>
<tr><td>Macro</td><td>Toggle Switch object can trigger the start of a Macro routine when the Macro has been created in advance.
For more information, see “18 Macro Reference”.</td></tr>
</table>

13.7. MULTI-STATE SWITCH

13.7.1. Overview

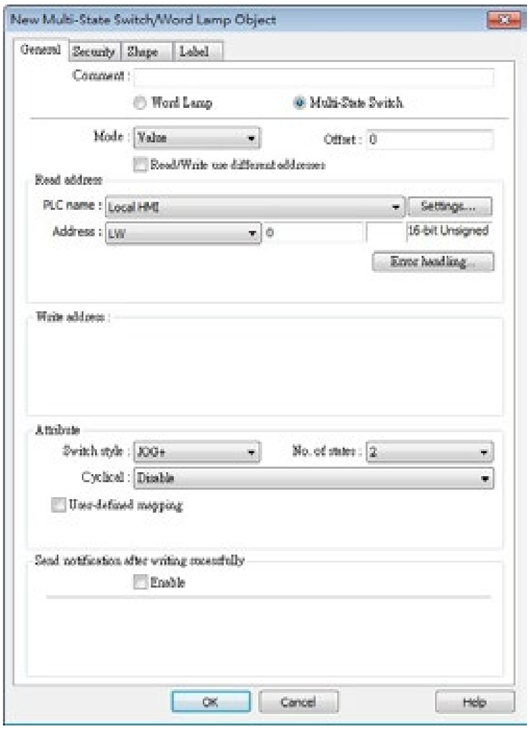
Multi-state Switch object is a combination of Word Lamp object and Set Word object. The appearance of the object is controlled by the value of the read word address. As well, pressing the button sets the value in the word address according to the settings.

13.7.2. Configuration

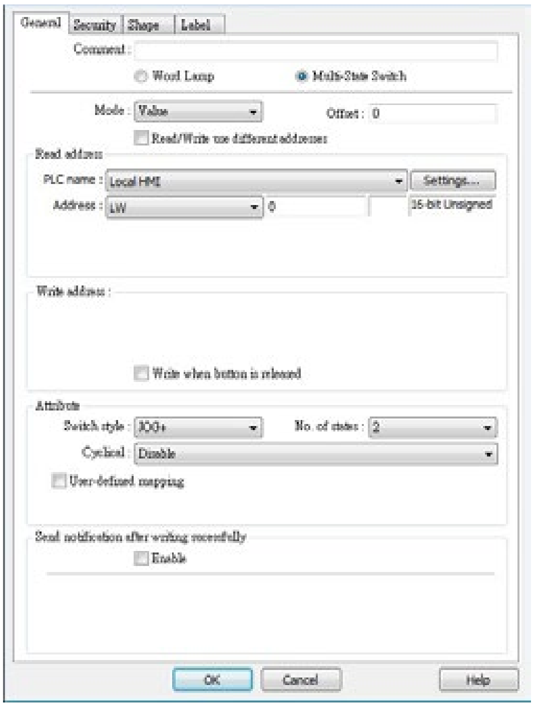
	Click the Multi-State Switch icon on the toolbar to open a Multi-State Switch object property dialog box. Set up the properties, press OK button, and a new Multi-State Switch object will be created.

13.7.2.1. General Tab

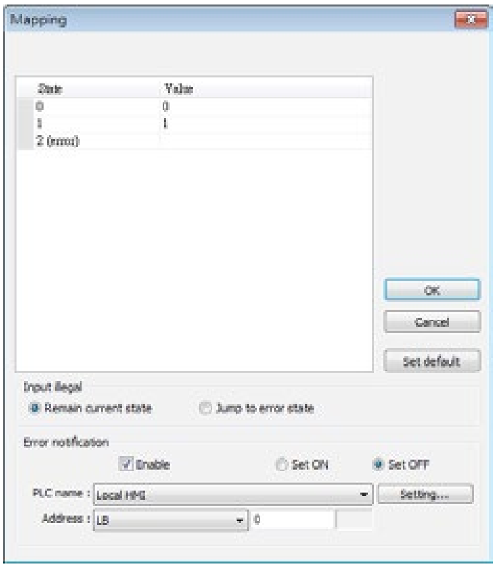
▪ cMT Series



▪ eMT, iE, XE, mTV Series



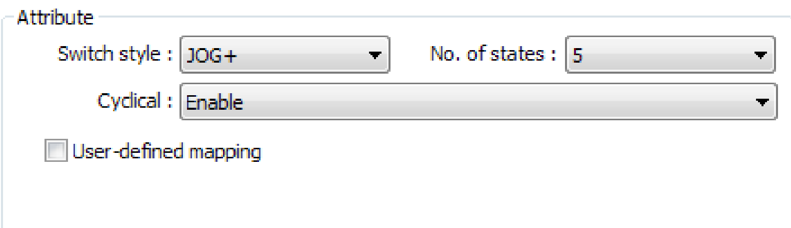
Setting	Description
Comment	User can describe the information of the object. **Word Lamp / Multi-State Switch** Switch between Word Lamp and Multi-State Switch features.
Model / offset	Different modes can be selected: (Value), (LSB). ☞ For more information, see “13.2 Word Lamp”.
Read address	Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the word device that controls the Multi-state Switch object. Users can also set address in (General) tab while adding a new object.
Write address	Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the word device that controls the Multi-state Switch object. Users can also set address in (General) tab while adding a new object. **Write after button is released** If this function is selected, the action is delayed till button is released; otherwise, the action is executed once the button is pressed.

Attribute	**Switch style** Select the object's operation mode, see Example 1. **User-defined mapping** The value placed in the write register of each selection, the action taken when an illegal value is entered, and error notification to a designated bit address can be set.  **Remain current state** If an illegal value is entered, Multi-state Switch will remain at the current state. **Jump to error state** If an illegal value is entered, Multi-state Switch will jump to the error state. **Error notification** If an illegal value is entered, automatically set the value placed in the designated register.
Send notification after writing successfully	After the system successfully writes data to PLC, the designated bit address will be set On/Off.
Error handling (cMT-SVR Series)	The action taken when an illegal value is entered or notify a designated bit address. This is similar to (User-defined mapping); the difference is the value corresponding to each state need not to be preset.

Example 1

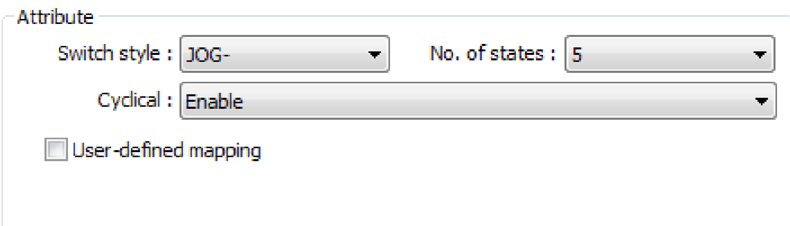
- JOG+

  Increase the value of a designated register by 1 each time when pressing the button, till the value equals to (No. of states). A cyclic action can be enabled. As shown below, each time when pressing the button, the state number will add 1 start from state 0, till state 4 ((no. of state)-1), and returns to 0 and step up again.



- JOG-

  Decrease the value of the designated register by 1 each time when pressing the button, till the value equals to 0. A cyclic action can be enabled. As shown below, each time when pressing the button, the state number will minus 1 start from state 4 ((no. of state)-1), till state 0, and returns to state 4 and step down again.



13.8. SLIDER

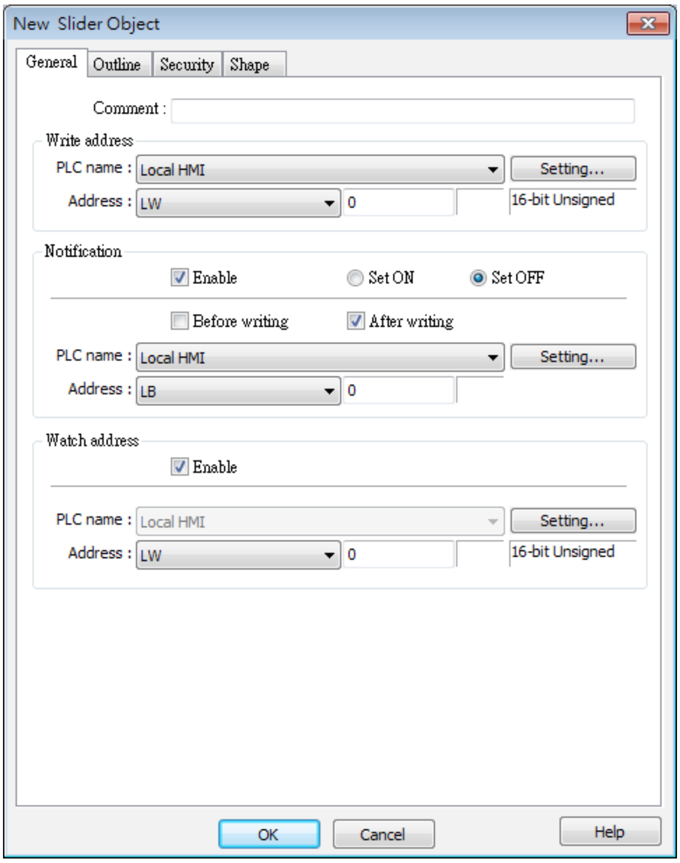
13.8.1. Overview

Slider object is used to change the value in a designated word register address by moving the slide on the screen.

13.8.2. Configuration

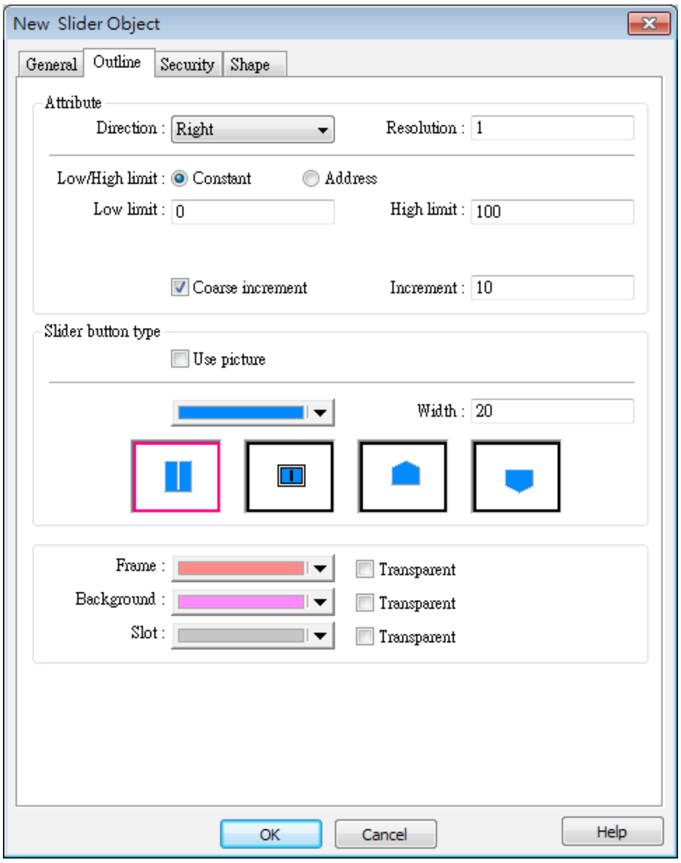
	Click the Slider icon on the toolbar to open a Slider object property dialog box. Set up the properties, press OK button and a new Slider object will be created.

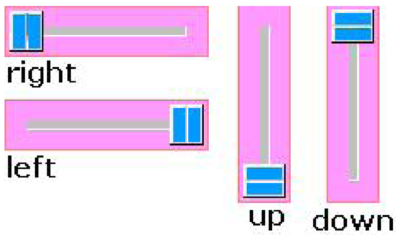
13.8.2.1. General Tab

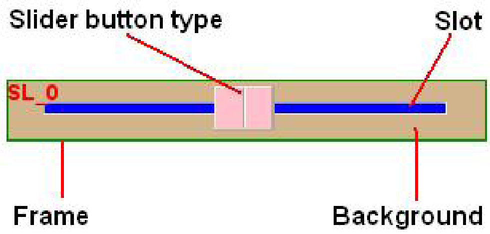


Setting	Description
Write address	Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the word device that controls the Slider object. Users can also set address in (General) tab while adding a new object.
Notification	If enabled, the state of a designated bit address will be set to ON or OFF. Click **(Setting)** to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the bit device that controls the notification settings. Users can also set address in (General) tab while adding a new object. **(Before writing) / (After writing)** Change the state of a designated bit register before, or after the slider is slid.
Watch address	When moving the roller, the new value written to the word register address can be displayed in real time.

13.8.2.2. Outline Tab



Setting	Description
Attribute	**Direction** Select the direction of the slider. (Right, Up, Left, Down)  **Resolution** Sets the value change in the word register for each step of the Slider. For example, if set to 10, the register value changes by 10 points for each increment or decrement on the Slider. **Constant** Sets the range of the Slider. For example, If set (Low limit) to 5, and (High limit) to 100, the Slider will enter values between 5 and 100. **Address** Set the (Low/High limit) by a designated register, see Example 1. **Coarse increment** Apart from moving the roller to change the value as in (Resolution), if this option is selected, the word value will increase / decrease by the (Increment) value each time the object is touched.

Slider button type	**Use picture** (cMT Series only) If this checkbox is selected, a picture can be selected from the Picture Library to be the slider button. Four default styles are offered here, and the width of the Slider may be set. **Color** Colors of the frame, background, and slot may be chosen. 

Example 1
Set the low or high limit by a designated register. When write address is LW-n, where n is an arbitrary number, the rule of setting limits is:

Content	16-bit	32-bit
Address	LW-n	LW-n
Low limit	LW-n	LW-n
High limit	LW-n+1	LW-n+2

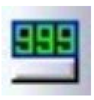
When address is LW-100, the rule of setting limits is:

Content	16-bit	32-bit
Address	LW-100	LW-100
Low limit	LW-100	LW-100
High limit	LW-101	LW-102

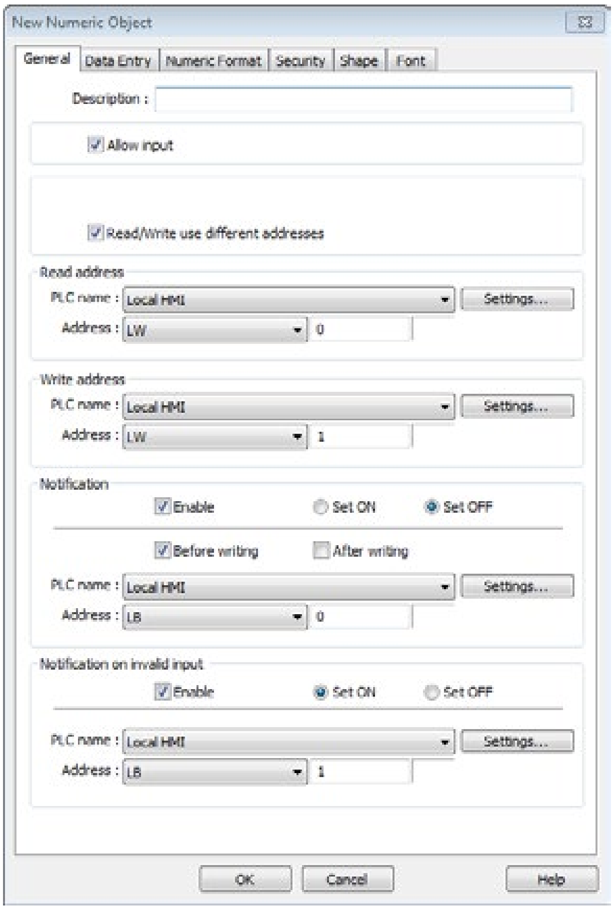
13.9. NUMERIC

13.9.1. Overview
Numeric object can be used to input or display the value of a designated word register.

13.9.2. Configuration

	Click the Numeric icon on the toolbar to open a Numeric object property dialog box. Set up the properties, press OK button, and a new Numeric object will be created.

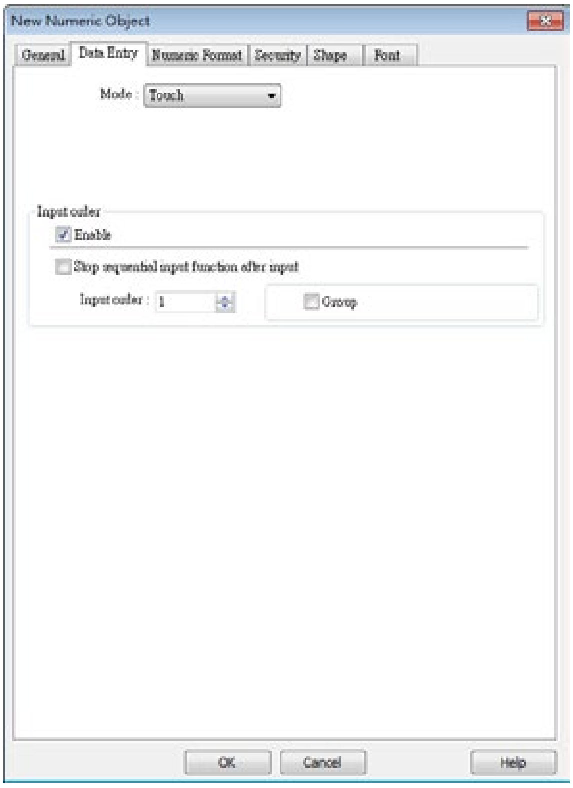
13.9.2.1. General Tab



Setting	Description
Allow input	If selected, the input features and relevant settings are enabled.
Read / write use different address	Set (Read address) and (Write address) differently.
Read address	Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the word device that displays the value. Users can also select a tag defined in Address Tag Library.
Write address	Select the (PLC name), (Device type), (Address) of the word device that system writes to.
Notification	If this check box is selected, it will notify a designated bit address (setting ON or OFF). **Before writing / After writing** Set the state of the designated bit address before or after the manual operation.
Notification on invalid input	If an illegal value is entered, automatically set the state of a designated register.

13.9.2.2. Data Entry Tab

▪ cMT Series



▪ eMT, iE, XE, mTV Series

Setting	Description
Mode	**Touch** Used when data entry is initiated by touching the screen object. **Bit control** Used when data entry is enabled by turning ON a designated bit, and entry ends when the bit goes OFF.
Allow input bit address	Specify a bit address that enables or ends data entry. The order of data entry is specified in (Input order) and an external USB keyboard is needed for data entry. For cMT-SVR, use iPad's keyboard.
Input order	Perform continuous input by setting (Input order) and (Group). The criterion of searching the next input object: ▪ The range of (Input order): 1 ~ 511, range of (Group): 1 ~ 15 ▪ If (Group) is not selected, its input order is 0 ▪ The system only searches for the objects within the same Group ▪ The lower number of order is entered before the higher number of order ▪ For multiple objects within the same group and with the same input order, the object placed in the lower layer is entered first
Keyboard (For eMT, iE, XE, mTV Series)	**Use a popup keypad** If selected: a pre-designed pop-up keypad can be chosen by selecting a check box, and selecting the relative position on the HMI screen. When data entry is enabled, the pop-up keypad displays in the selected position, and closed when data entry ends. If not selected: when data entry is enabled, the pop-up keypad is not displayed. Users may: ▪ Create a custom design on the same screen window ▪ Use a USB keyboard **Hide title bar** Use a keypad without the title bar. **Restart the keypad if input value is out of range** When entering data, if the value entered is not within the valid range, the system will automatically restart the keypad.

Note
To enter data for cMT-SVR, the iPad's keyboard is used.

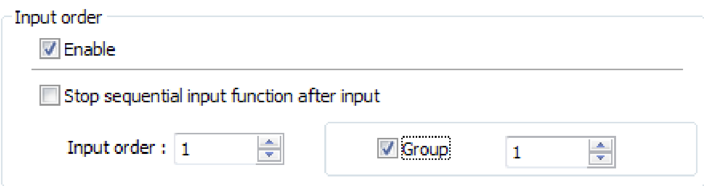
To create a keyboard in current window, see "12 Keyboard Design and Usage".

Example 1

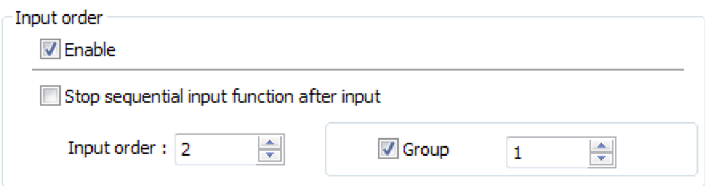
This example demonstrates how to use (Input Order) and (Group) to perform continuous input in several Numeric objects. After entering data in one object, entry will be passed to the next input order object which is in the same group.

1. Create three Numeric objects, and set (Input order) to 1, 2, and 3 respectively. Include the three objects in (Group 1) as shown in the following figure.

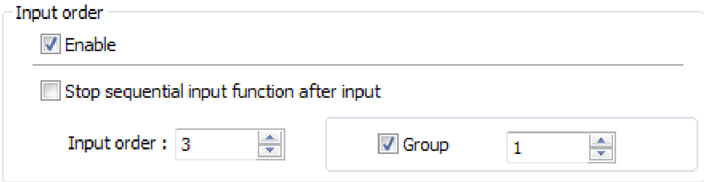
   ▪ LW-0

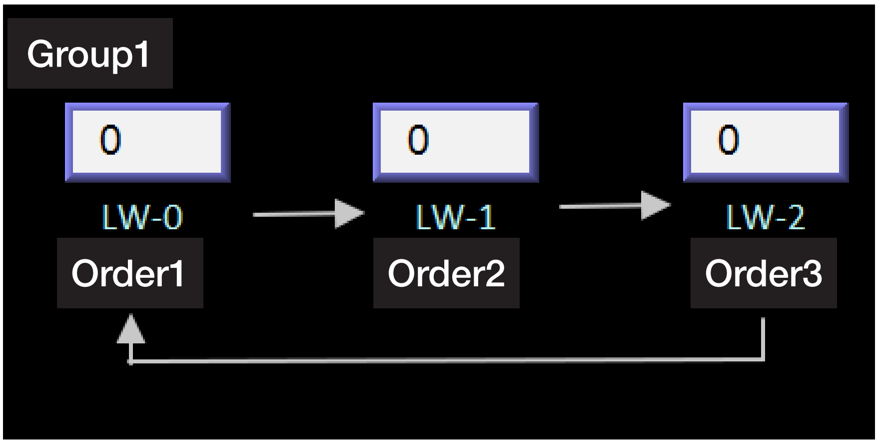


   ▪ LW-1

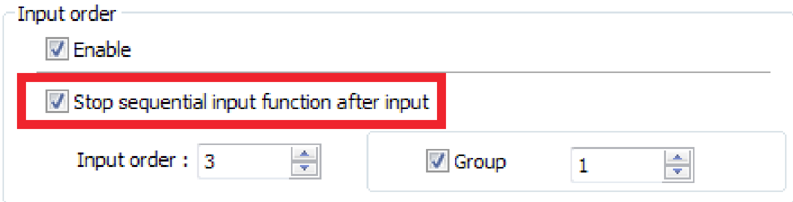


- LW-2



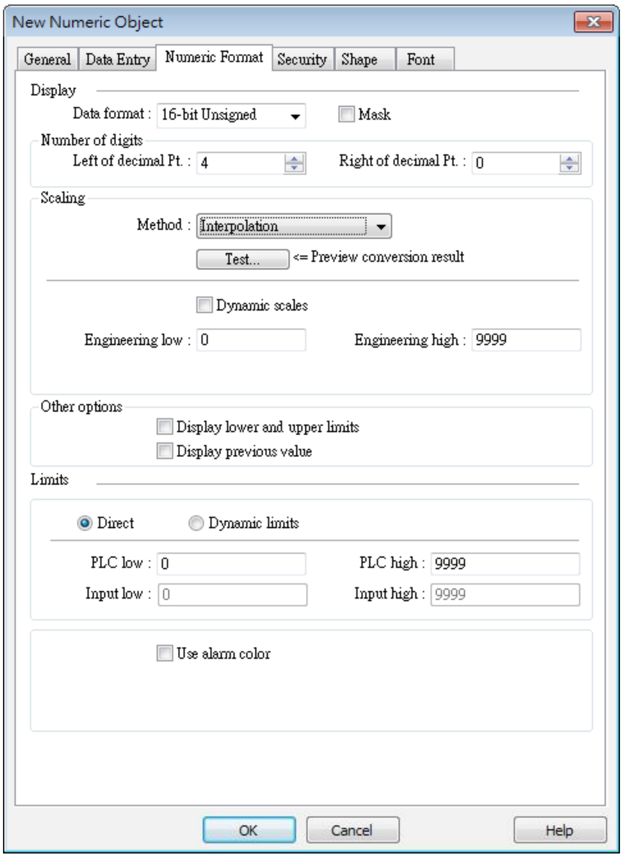


2. When finish entering data in the last object, to end data entry of all objects, please select (Stop sequential input function after input) check box.

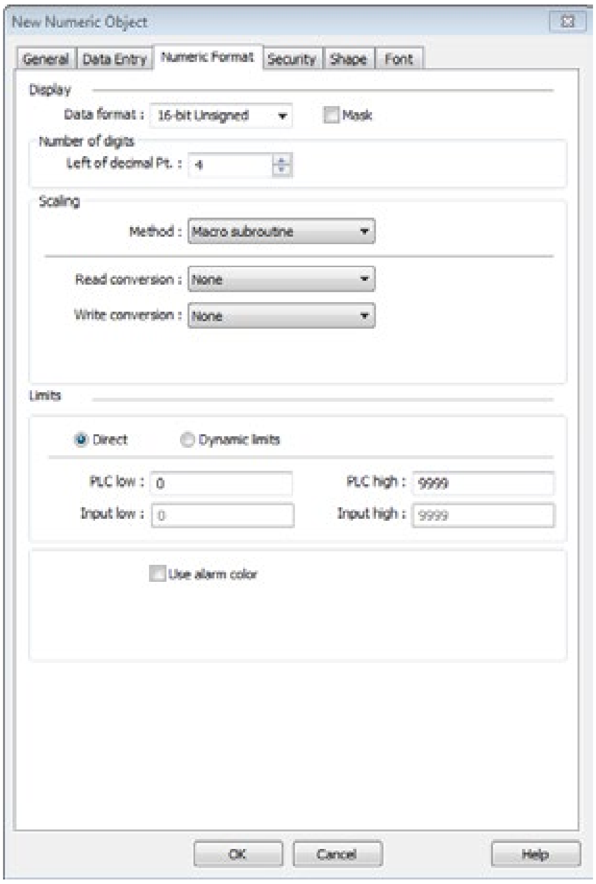


13.9.2.3. Numeric Format Tab

▪ cMT Series



▪ eMT, iE, XE, mTV Series



Setting	Description
Display	**Data Format** Set the data format of a designated word register. The selections include: BCD, HEX, Binary, Unsigned, Signed, Float. 16-bit uses 1 word where 32-bit uses two words. **Mask** If selected, any values entered will be hidden by displaying them as ****.
Number of digits	**Left of decimal Pt.** The number of digits before the decimal point. **Right of decimal Pt.** The number of digits after the decimal point.
Scaling	**Interpolation** If this check box is selected, (Engineering low) and (Engineering high) boxes appear. Values entered in these boxes correspond to the display range required. The setting also requires (Input low) and (Input high) in the limits section. See Example 2. Test: preview the result of Interpolation. See Example 2. Dynamic scales: set the (Engineering low) and (Engineering high) by a designated register. See Example 4. **Macro subroutine** (Not available for cMT Series) The value read from or written to the register can be computed by macro subroutines selected in (Read conversion) and (Write conversion). The macro subroutines should be diefined in Macro Function Library. To use this feature, see “13.9.2.1 The rule of using Macro subroutine”.
Other options (For cMT Series)	**Display lower and upper limits** If selected, when entering a value, the range is displayed near the object. **Display previous value** If selected, when entering a value, the value before update is displayed near the object. 
Limits	This section allows users to apply display limits to the values held in the input register. The color when the register value is outside limits can be set. **Direct** Sets the limits by entering values in (Input low) and (Input high). If the value entered is outside the limits, the value in the register cannot be changed. **Dynamic limits** Set the limits by a designated register, see Example 5.
Use alarm color	**Low limit** When the value in the register is outside the (Low limit), display digits by the color set. **High limit** When the value in the register is outside the (High limit), display digits by the color set. **Blink** When the value in the register is outside either limit, the digits flash.

13.9.2.4. The Rule of Using Macro Subroutine

- There must be a return value and exactly one parameter
  - Examples:
    - sub char test (short a) // (Correct)
    - sub test (char a) // (Incorrect, no return value.)
    - sub char test (char a, char b) // (Incorrect, two parameters.)

- Use the Macro data type that corresponds to the object’s data format
  - The mapping is as follows:

Macro data type	Numeric object data format
short	16-bit Signed
Int	32-bit Signed
unsigned short	16-bit BCD, 16-bit HEX, 16-bit Binary, 16-bit Unsigned
unsigned int	32-bit BCD, 32-bit HEX, 32-bit Binary, 32-bit Unsigned
float	32-bit Float

- For example, if the data format of the numeric object is 16-bit Unsigned, only the corresponding Macro data type: unsigned short, is available
  - Examples:
    - sub char test(unsigned short a) // (Correct)
    - sub char test(char a) // (Incorrect)

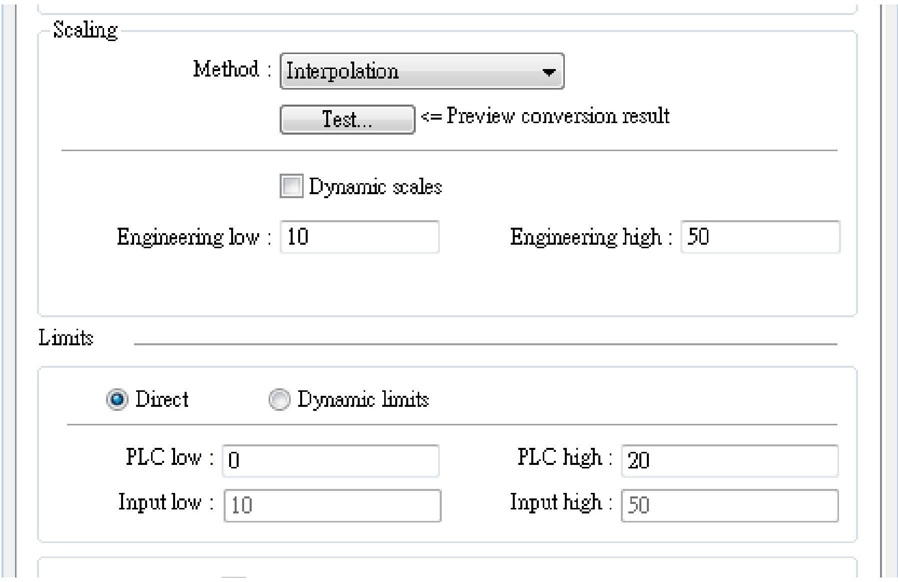
- Supports only the local HMI address
  - Examples:
    - GetData(var, “Local HMI”, LB, 0, 1) // (Correct)
    - GetData(var, “Modbus RTU”, 0x, 0, 1) // (Incorrect)

- The following system defined functions are unable to be invoked:
  - ASYNC_TRIG_MACRO, SYNC_TRIG_MACRO, DELAY, FindDataSamplingDate, FindDataSamplingIndex, FindEventLogDate, FindEventLogIndex, INPORT, INPORT2, OUTPORT, PURGE, TRACE

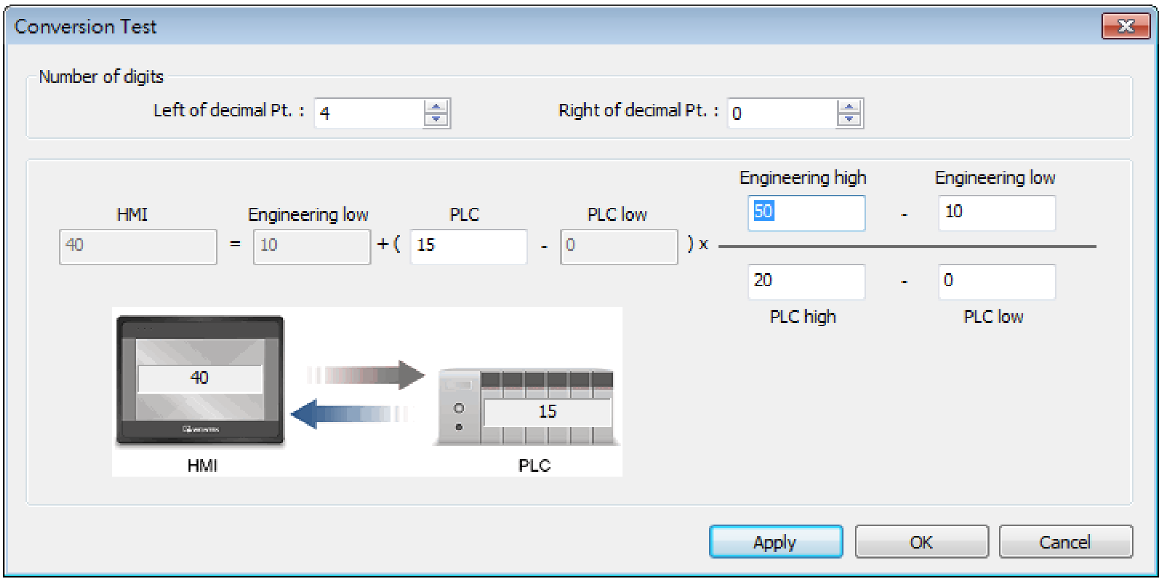
- The following statements are not supported:
  - For-Next, While-Wend

Example 2

If (Interpolation) is selected, the scaling equation is as the following:
If A indicates the original data and B indicates the displayed data:
B = [Engineering low] + (A - [PLC low]) × ratio
where, ratio = ([Engineering high] - (Engineering low)) / ([PLC high] - [PLC low])
As shown below, the original data is 15, after conversion, 40 will be displayed.



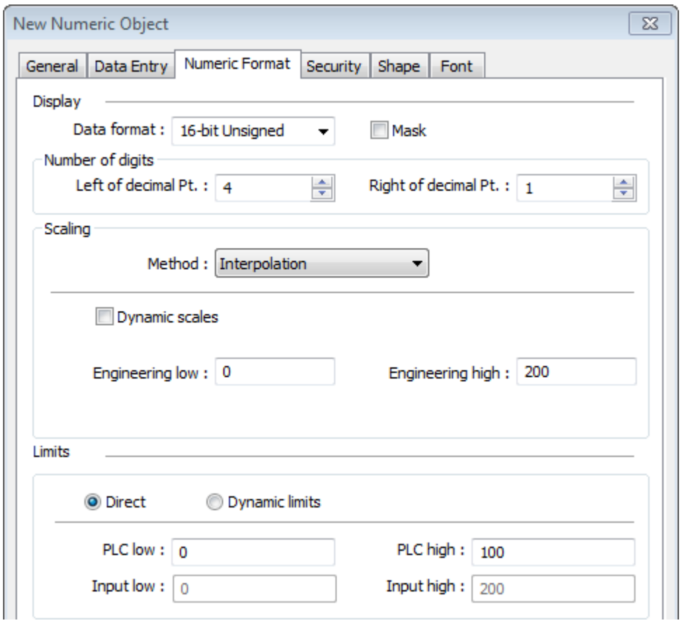
Click (Test) button to preview the result of Interpolation. Enter a value in (PLC) field as shown in the following figure, for example, enter value 15, and the result, which is 40, will be displayed.



Example 3

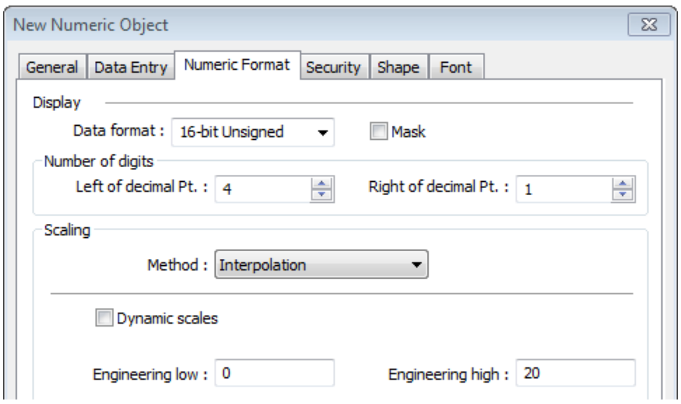
If the numeric format selected is not Float and decimal point is used, the decimal place of the converted result will not be adjusted automatically, please adjust (Engineering high) to correctly place the decimal point of the result gained in (Interpolation) mode. Please see the illustration below.

1. Create two Numeric objects, set (Right of decimal Pt.) to 1 and select (Interpolation) method for one of the objects as shown in the following figure.



2. Enter value “123”, the object set to (Interpolation) displays “246.0” instead of “24.6”.

3. To move the decimal point one place to the left, adjust (Engineering high) as shown in the following figure.



Example 4
If (Interpolation) is selected, set the (Engineering low) and (Engineering high) by a designated register. When Dynamic Address is LW-n, where n is an arbitrary number, the rule of setting (Engineering low) and (Engineering high) is:

Content	16-bit	32-bit
Dynamic address	LW-n	LW-n
Engineering low	LW-n	LW-n
Engineering high	LW-n+1	LW-n+2

When address is LW-100, the rule of setting limits is:

Content	16-bit	32-bit
Dynamic address	LW-100	LW-100
Engineering low	LW-100	LW-100
Engineering high	LW-101	LW-102

Example 5
Set the limits by a designated register. When (Address) is LW-n, where n is an arbitrary number, the rule of setting limits is:

Content	16-bit	32-bit
Address	LW-n	LW-n
Low limit	LW-n	LW-n
High limit	LW-n+1	LW-n+2

When address is LW-100, the rule of setting limits is:

Content	16-bit	32-bit
Address	LW-100	LW-100
Low limit	LW-100	LW-100
High limit	LW-101	LW-102

Example 6
The following demonstrates how to use (Macro subroutine) for scaling when configuring Numeric object.
The following two macros are used, one for (Read conversion) and one for (Write conversion).

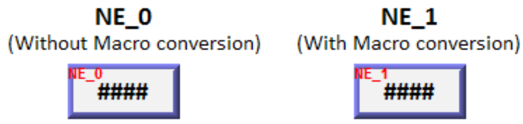
Read Conversion

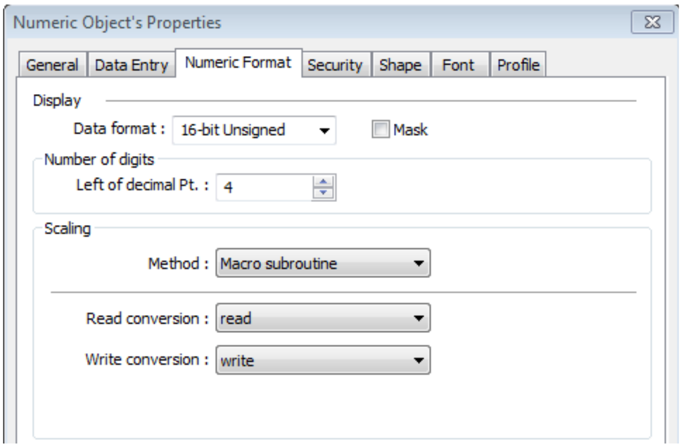
```
sub short read(unsigned short a)

short b
b = a + 10
return b

end sub
```

Write Conversion

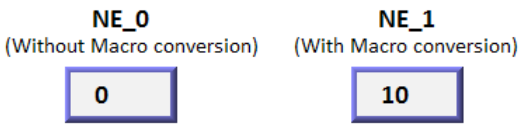
```
sub short write(unsigned short a)

short b
b = a - 10
return b

end sub
```

1. Create two Numeric objects: NE_0 and NE_1 and use the same control address. Select (Macro subroutine) for NE_1.





2. Enter 0 in NE_0 then NE_1 will execute (Read conversion). The value gained will be 10.



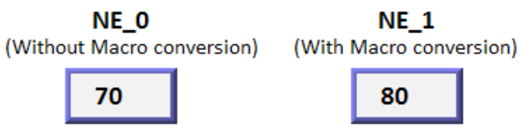
Read Conversion

```
sub short read(unsigned short a)

short b
b = a + 10
return b

end sub
```

Write Conversion

```
sub short write(unsigned short a)

short b
b = a - 10
return b

end sub
```

3. Enter 80 in NE_1, (Write conversion) is executed and the value gained will be 70. NE_0 displays 70.



Read Conversion

```
sub short read(unsigned short a)

short b
b = a + 10
return b

end sub
```

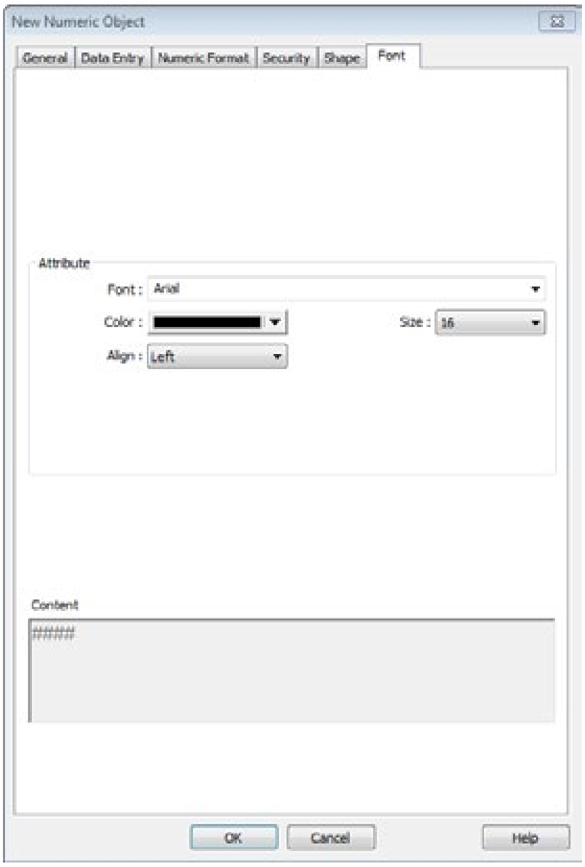
Write Conversion

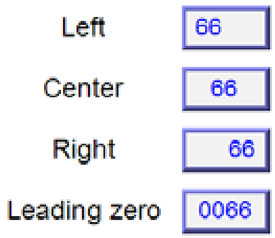
```
sub short write(unsigned short a)

short b
b = a - 10
return b

end sub
```

Note

If executing (Read conversion) and (Write conversion) by the same numeric object, the value entered in this object is computed by the Macro subroutine of (Write conversion) first, and then the result is computed by the Macro subroutine of (Read conversion). In Example 5, if the subroutine of (Write conversion) is set to b=a-20, then entering 80 in NE_1 will get 60 after (Write conversion) and then the object displays 70 after (Read conversion).

13.9.2.5. Font Tab



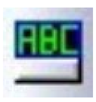
Setting	Description
Color	When the value is within the limits, display digits using color set in this tab.
Align	**Left:** align the number to the left. **Center:** align the number to the center. **Right:** align the number to the right. **Leading zero:** the number is preceded with leading zeros when the number of digits is less than that set. 
Size	Set the font size.

13.10. ASCII

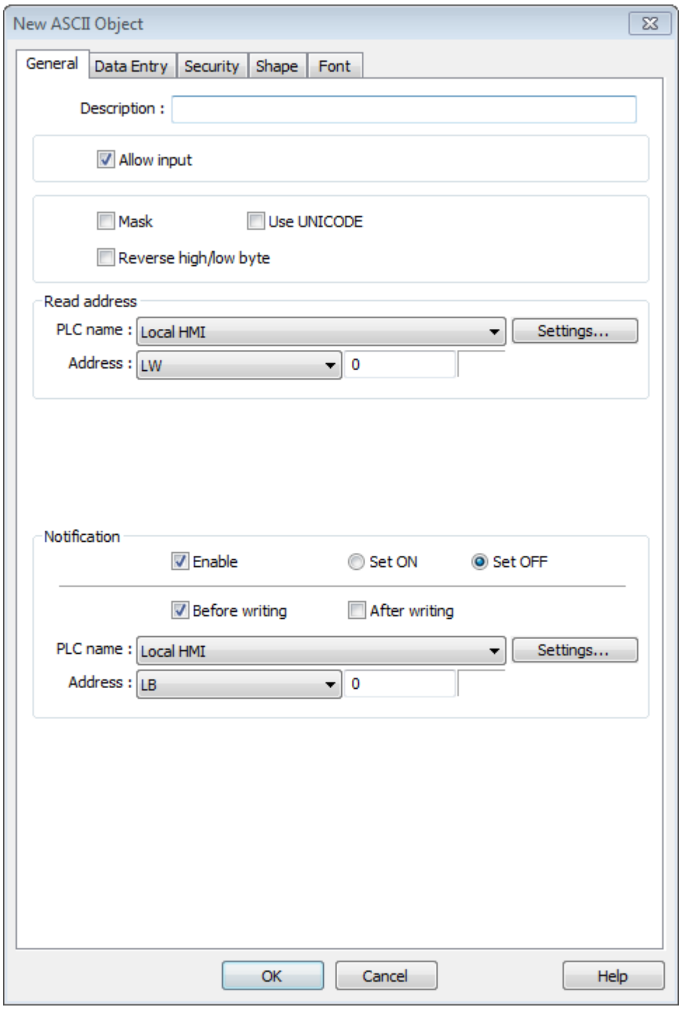
13.10.1. Overview

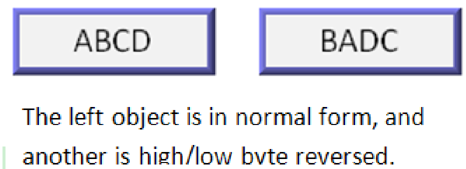
ASCII object can be used to input or display ASCII or UNICODE characters held in designated word registers.

13.10.2. Configuration

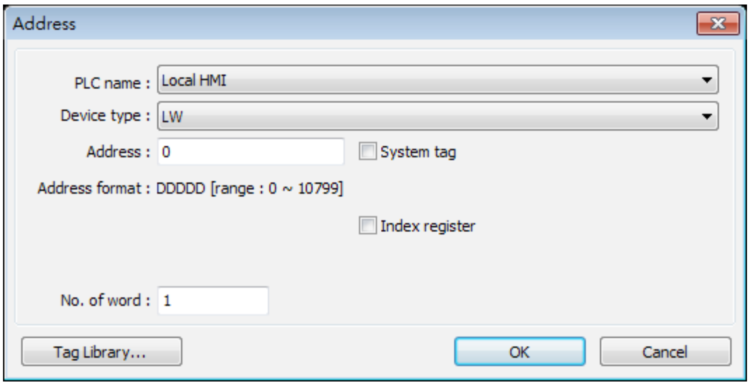
	Click the ASCII icon on the toolbar to open an ASCII object property dialog box. Set up the properties, press OK button, and a new ASCII object will be created.

13.10.2.1. General Tab



Setting	Description
Allow input	If selected, the input features and relevant settings are enabled.
Mask	If selected, any values entered will be hidden by displaying them as ****.
Use UNICODE	Select this check box to display data in UNICODE format. If not selected, the characters are displayed in ASCII format. This feature can be used with the (Function Key) object that uses (ASCII/UNICODE).
Reverse high/low byte	Normally an ASCII code is displayed in “high byte”, “low byte” order. Reverse selection makes the system display ASCII characters in “low byte”, “high byte” order. 

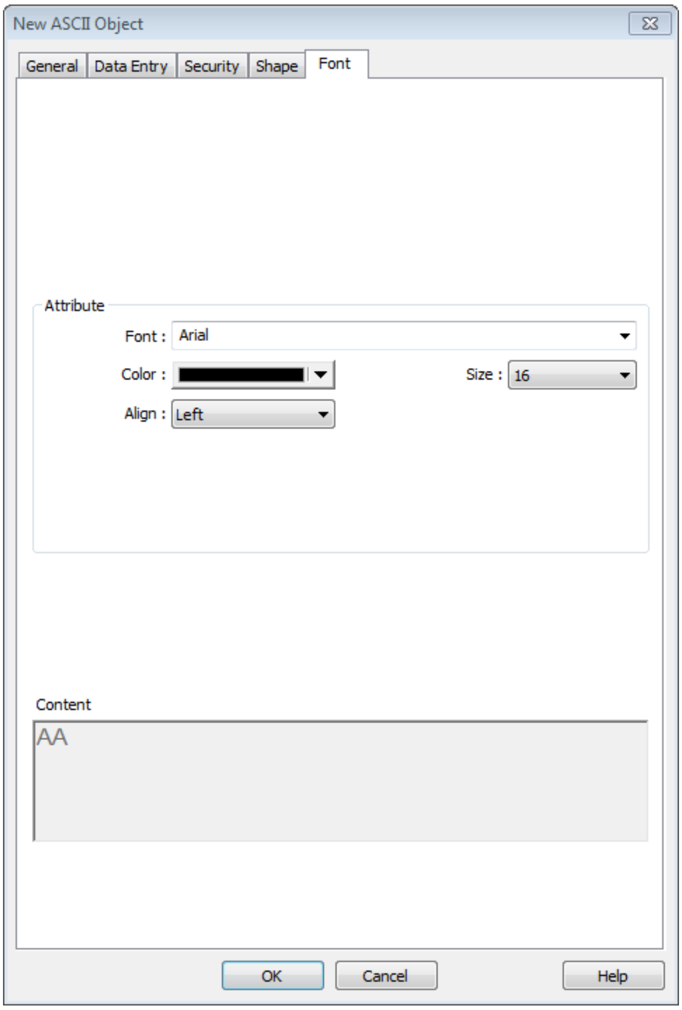
<table>
<tr>
<td>Read address</td>
<td>Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the word device that displays characters. Users can select a defined address tag from Address Tag Library, or set address in (General) tab while adding a new object.



No. of words
Select the maximum number of words to be displayed.</td>
</tr>
</table>

Note
An UNICODE character uses 1 word, and an ASCII character uses 1 byte. Therefore 1 word can be used as 1 UNICODE character or 2 ASCII characters. (1 word equals to 2 bytes)

13.10.2.2. Font Tab



Setting	Description
Attribute	The font, size, color, and alignment can be set. **Align** Left: align the text to the left. Center: align the text to the center. Right: align the text to the right.

13.11. INDIRECT WINDOW

13.11.1. Overview

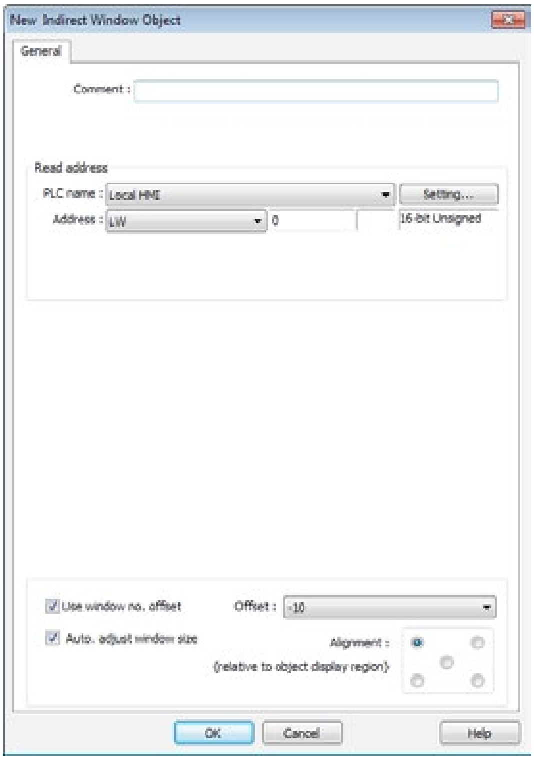
Indirect Window object opens the pop-up window assigned by a designated word register. There are two ways to use Indirect Window object: the first is to use the profile of Indirect Window object, and let the pop-up window be resized and displayed in the defined profile; the second is to automatically resize the window according to the size of the pop-up window to be displayed. To close the pop-up window, assign 0 to the designated word register. The difference between Direct Window and Indirect Window is that Direct Window is controlled by a bit register, while Indirect Window is controlled by a word register.

13.11.2. Configuration

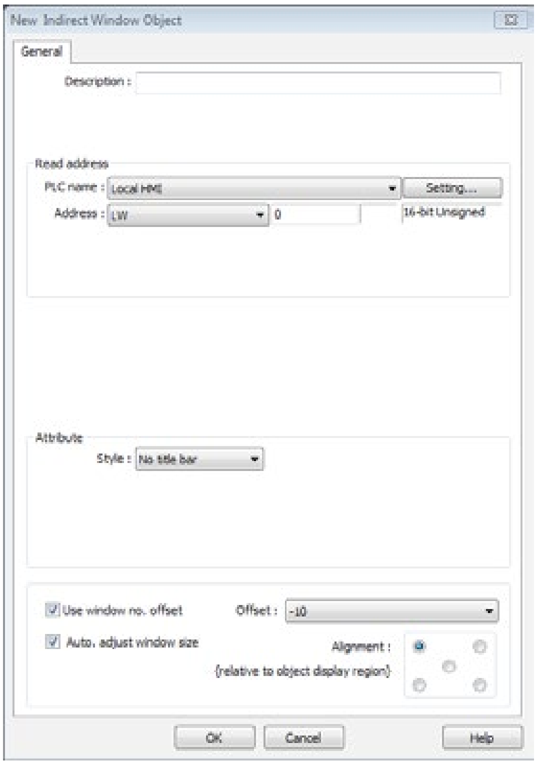
	Click the Indirect Window icon on the toolbar to open the object property dialog box. Set up the properties, press OK button, and a new Indirect Window object will be created.

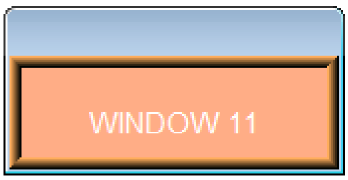
13.11.2.1. General Tab

- cMT Series



▪ eMT, iE, XE, mTV Series

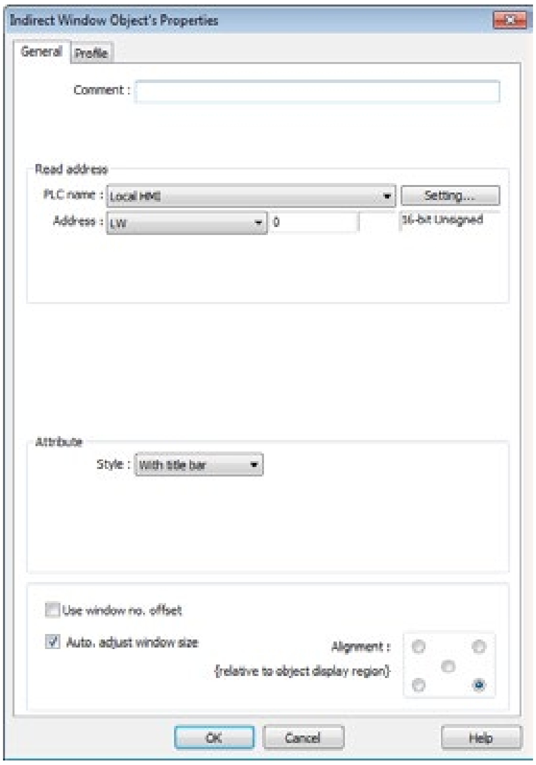


Setting	Description
Read address	Click (Setting) to select the (PLC name), (Address), (Device type), (System tag), (Index register) of the word device that controls the pop-up window. Users can also set address in (General) tab while adding a new object.
Attribute	**Style** Set the display style of the pop-up window. There are two styles: ▪ No title bar The pop-up window has no title bar and cannot be dragged.  ▪ With title bar The pop-up window has a title bar that can be dragged to move the window. 
Use window no. offset	Sets the offset of the window number for selecting the pop-up window. The window number of the pop-up window is calculated by the value in the word register added to the offset. For example, assume the value in the register is 20 and offset is 5, the pop-up window number will be 25.
Auto. adjust window size	Automatically resize the Indirect Window and align the pop-up window to the preset region. **Alignment** Sets a reference point of the pop-up window from one of the five positions on the screen. For example, if the lower-right region is selected, the lower-right corner of the pop-up window is aligned to the lower-right region of the Indirect Window. See Example 1.

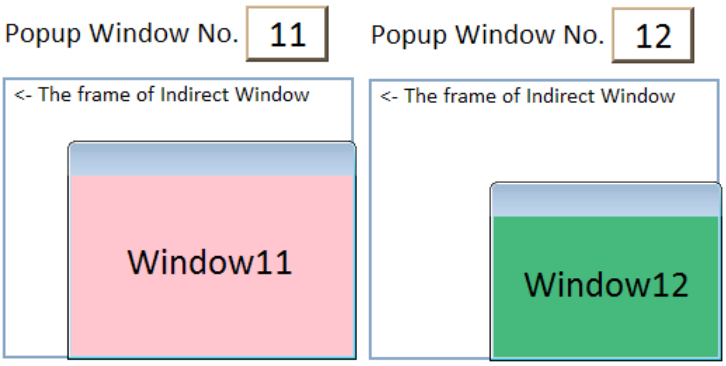
Example 1

Here is an example of using Indirect Window. The setting is shown in the following figure, set the address to LW-0 which assigns the window number. Create window no. 11 and 12 first.

1. Create an Indirect Window object, set address to LW-0, and select (Auto. adjust window size).
2. Select the region where the window is to be displayed.



3. Enter value 11 in LW-0, the pop-up window displayed is window no. 11.
4. Enter value 12 in LW-0, the pop-up window displayed is window no. 12.
5. Enter value 0 in LW-0, the pop-up window is closed.



To close the pop-up window, apart from entering 0 in the designated word register, another way is to place a Function Key object in the pop-up window, and set the key to (Close window).

	Note ▪ At most 24 windows can be displayed simultaneously at run time. ▪ The system does not allow opening the same window with two Direct (or Indirect) windows in one base window. ▪ If the pop up window has monopoly property enabled, then when the window pops up, all background windows may not be operated until the monopolizing window has been closed.

13.12. DIRECT WINDOW

13.12.1. Overview

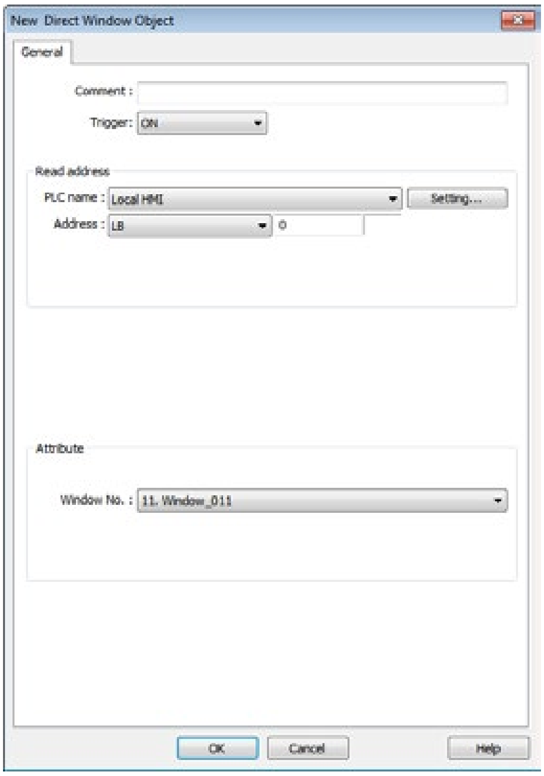
Direct Window object defines the position and size of a pop-up window location on a window. When the content of the bit register is changed, the window will pop up at the predefined location. The display area for the pop-up window is limited by the size of predefined location. Restoring the value of the bit register closes the pop-up window. The difference between the Direct Window and the Indirect Window is that Direct Window object has a predefined window number, and is controlled by a bit register, while Indirect window is controlled by a word register, and the value of the word register determines the window displayed.

13.12.2. Configuration

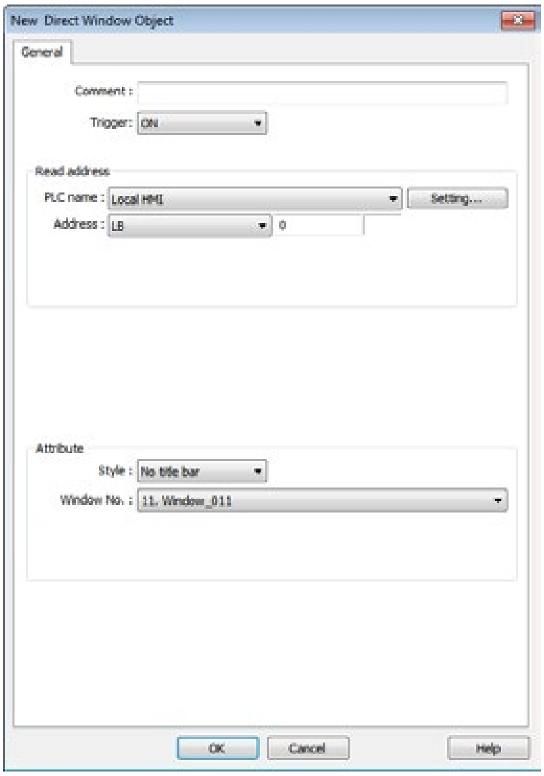
	Click the Direct Window icon on the toolbar to open a Direct Window object property dialog box. Set up the properties, press OK button, and a new Direct Window object will be created.

13.12.2.1. General Tab

▪ cMT Series



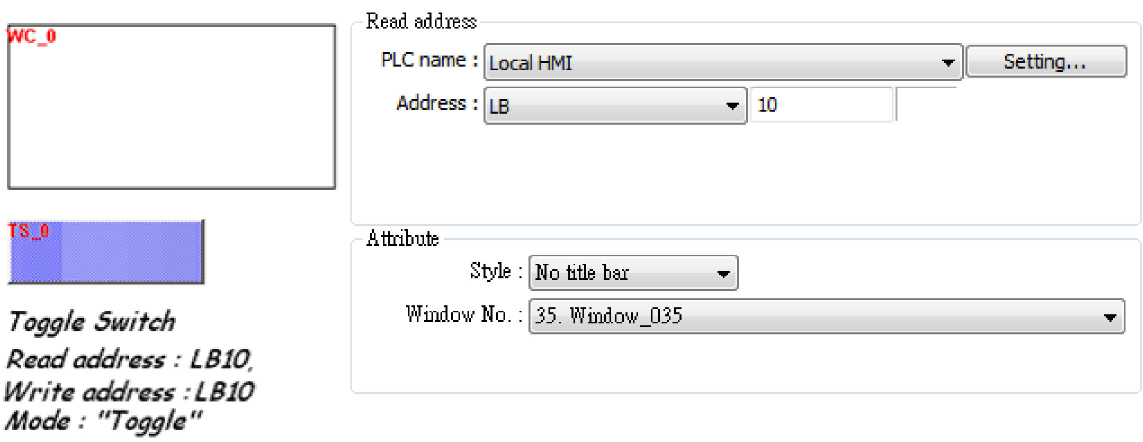
▪ eMT, iE, XE, mTV Series



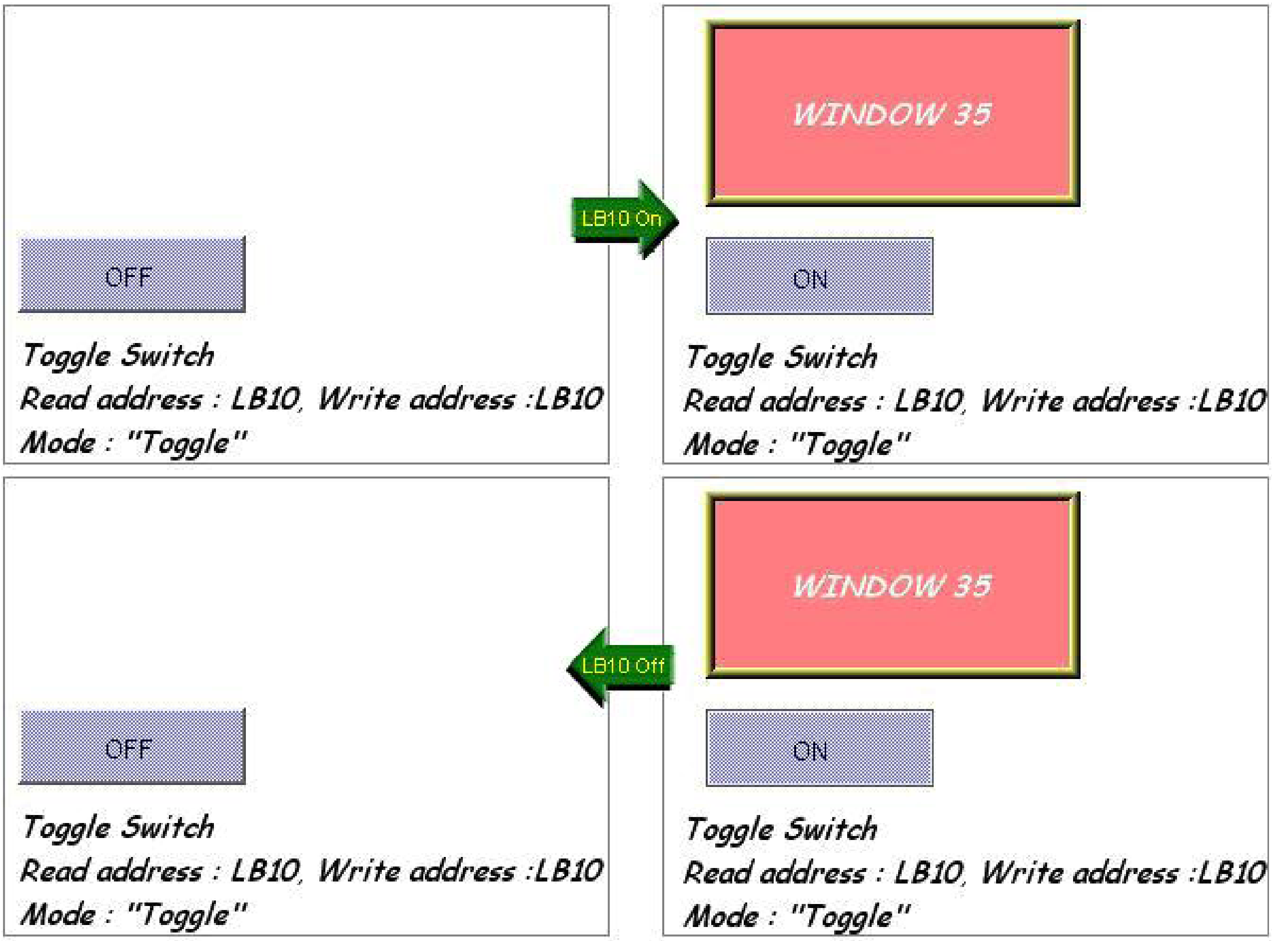
Setting	Description
Read address	Click (Setting) to select the (PLC name), (Device type), (Address), (System tag), (Index register) of the bit device that control the window popup. Users can select a defined address tag from Address Tag Library, or set the address in (General) tab while adding a new object.
Attribute	**Style** Define the pop-up window style. Two styles are available, (No title bar) and (With title bar). **Window no.** Set the pop-up window number.

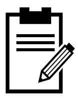
Example 1

Here is an example to explain how to use the Direct Window object. The following figure shows the settings of the Direct Window object. In the example, use LB-10 to call up window no. 35.



If the state of LB-10 turns to ON, window no. 35 will pop up; if the state of LB-10 turns to OFF, window no. 35 will be closed, as shown in the following figure.



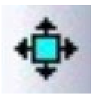
	**Note** ▪ A screen can simultaneously display up to 24 pop-up windows including System Message Window, Direct Window and Indirect Window. ▪ The system does not allow opening the same window with two Direct (or Indirect) Windows in one base window. ▪ If the pop up window has monopoly property enabled, then when the window pops up, all background windows may not be operated until the monopolizing window has been closed.

13.13. MOVING SHAPE

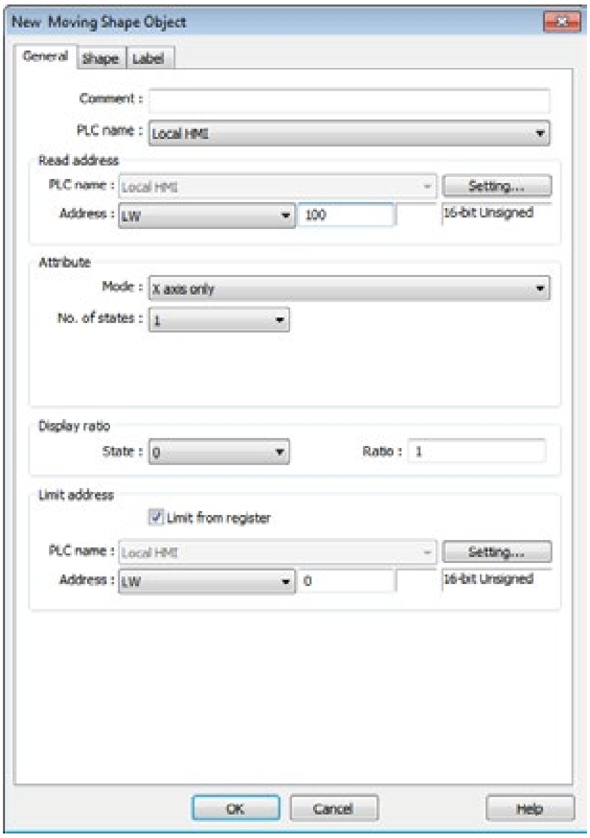
13.13.1. Overview

Moving Shape object defines the states and moving distance of an object. The state and the location of the object depend on consecutive registers.

13.13.2. Configuration

	Click the Moving Shape icon on the toolbar to create a Moving Shape object. Set up the properties, press OK button and a new Moving Shape object will be created.

13.13.2.1. General Tab



Setting	Description
Read address	Click (Setting) to configure the (PLC name), (Device type), (Address), (System tag), or (Index register) of the word devices that control the display of object's state and moving distance. Users can also set the address in (General) tab while adding a new object.
Attribute	Select the object's movement mode and range. See "13.13.2.1 Illustration of Modes" in the following part.
Display ratio	The size of shape in different states can be set individually as shown in the following figure. 
Limit address	The object's moving range can be set by adjusting the data in the designated register, see Example 1.

Example 1

Supposed that the object's moving range is limited by register LW-n, the addresses in the following table are used to limit the moving range.

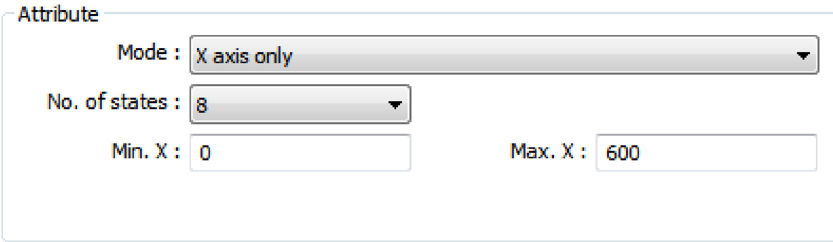
Data format	16-bit	32-bit
(Min. X) address	LW-n	LW-n
(Max. X) address	LW-n+1	LW-n+2
(Min. Y) address	LW-n+2	LW-n+4
(Mas. Y) address	LW-n+3	LW-n+6

13.13.2.2. Illustration of Modes

Available modes are: (Assume Read Address is LW-n)

- X axis only

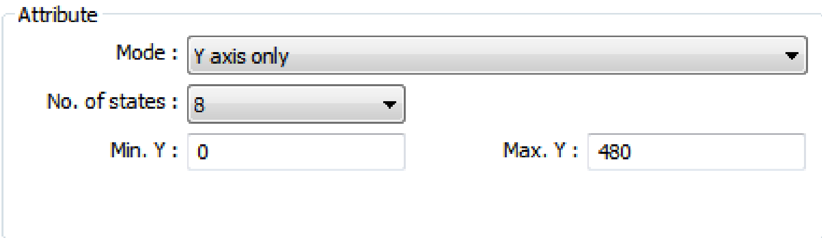
The object is only allowed to move along the X-axis. The moving distance ranges from (Min. X) to (Max. X).



Data format	16-bit	32-bit
Object state	LW-n	LW-n
Moving distance on X-axis	LW-n+1	LW-n+2

- Y axis only

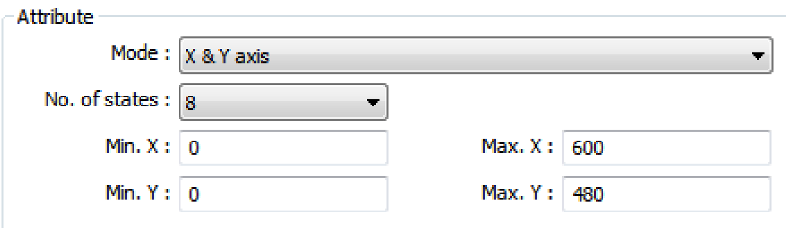
The object is only allowed to move along the Y-axis. The moving distance ranges from (Min. Y) to (Max. Y).



Data format	16-bit	32-bit
Object state	LW-n	LW-n
Moving distance on Y-axis	LW-n+1	LW-n+2

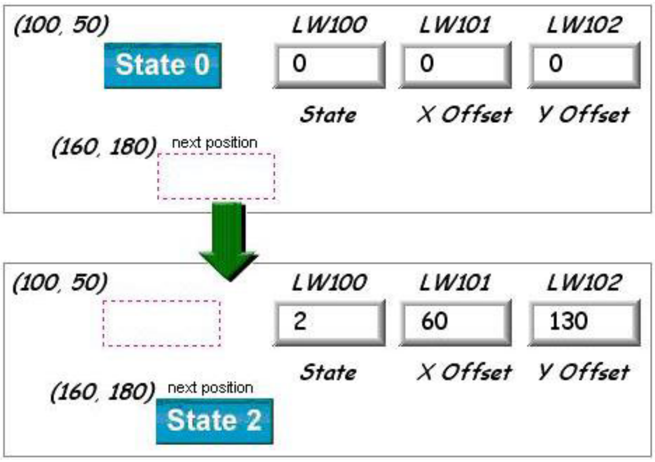
- X & Y axis

The object is allowed to move along the X-axis and Y-axis. The moving range in X and Y directions is defined by (Min. X), (Max. X) and (Min. Y), (Max. Y) respectively.



Data format	16-bit	32-bit
Object state	LW-n	LW-n
Moving distance on X-axis	LW-n+1	LW-n+2
Moving distance on Y-axis	LW-n+2	LW-n+4

For example, if the object's read address is LW-100 and the data format is (16-bit Unsigned), LW-100 is used to control the object's state, LW-101 is used to control the object's moving distance on the X-axis, and LW-102 is used to control the object's moving distance on the Y-axis. The following figure shows that the object's read address is LW-100 and initial position is (100, 50). To move the object to the position (160,180) and change its state to State 2, assign 2 to LW-100, 160-100 = 60 to LW-101, 180-50 = 130 to (LW102).



- X axis w/ scaling

The object moves in X-axis only with scaling. Suppose that the value of the designated register is DATA, the system uses the following equation to calculate the moving distance on the X-axis.

Data format	16-bit	32-bit
Object state	LW-n	LW-n
Moving distance on X-axis	LW-n+1	LW-n+2

- Y axis w/ scaling

The object is for Y axis movement with scale, and the equation to calculate the moving distance on the Y-axis is the same as the one in (X axis w/ scaling).

Data format	16-bit	32-bit
Object state	LW-n	LW-n
Moving distance on Y-axis	LW-n+1	LW-n+2

- X axis w/ reverse scaling

This works in the way as (X axis w/ scaling), but the moving direction is in reverse.
- Y axis w/ reverse scaling

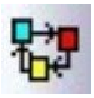
This works in the way as (Y axis w/ scaling), but the moving direction is in reverse.

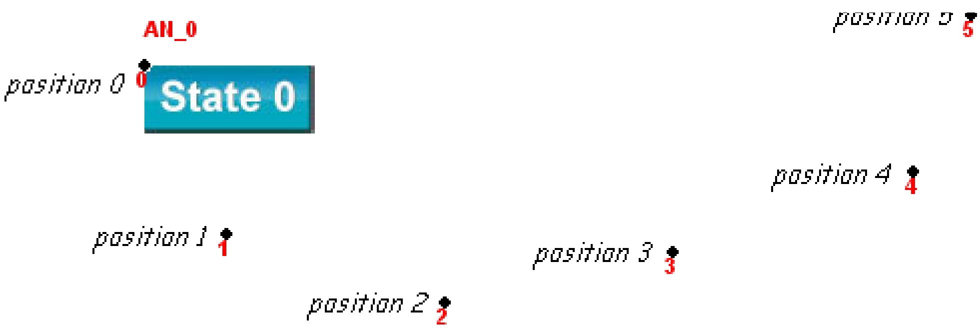
13.14. ANIMATION

13.14.1. Overview

Animation object is defined by a pre-defined point set and states. Animation object will then move to a given point in a given state defined by designated registers. The object state and position depend on current value of two consecutive registers. The first register controls the state of the object and the second register controls the position along the predefined path.

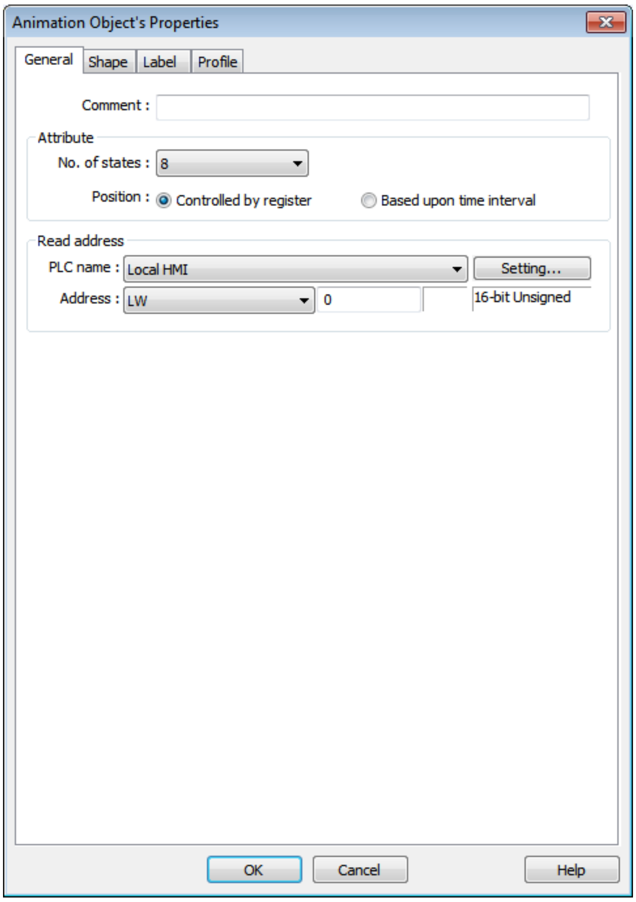
13.14.2. Configuration

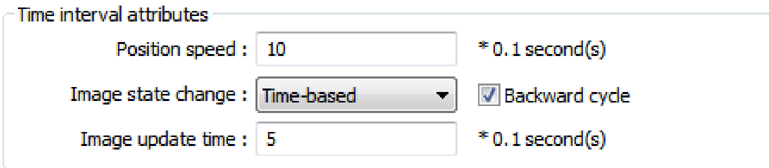
	Click the Animation icon on the toolbar. First, create the pre-defined path. Move the mouse to each moving position, and click the left button to define positions one by one. When it is done, right click on the screen, set up the properties, press OK button, and a new Animation object will be created.



To change the object’s attributes, double click on the object to open Animation Object’s Properties dialog box.

13.14.2.1. General Tab



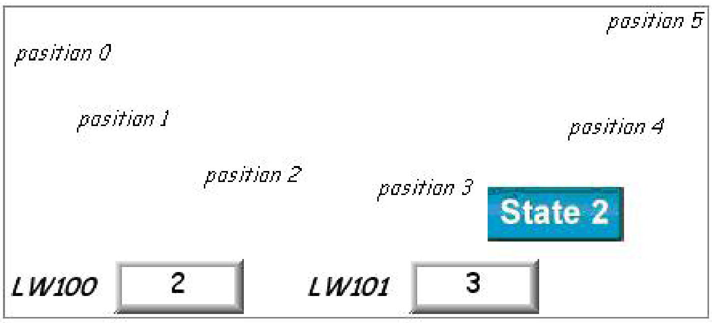
Setting	Description
Attribute	**No. of states** Configure the number of states for this object. **Controlled by register** Use the designated registers to control the object's state and position. See Example 1. **Based upon time interval** The object's state and position will change from time to time. (Time interval attributes) is used to set the time interval for states and positions.  **Position speed:** the speed of movement. The unit is 0.1 second. Supposed that (Speed) is set to 10, the object's position will change each second. **Image state change:** determines how state changes, either (Position dependent) or (Time-based). If (Position dependent) is selected, the object state will change when position changes. If (Time-based) is selected, the object position will change based on (Position speed) and the object state will change based on (Image update time). **Backward cycle:** assumed the object has four positions: position 0, position 1, position 2, and position 3, and (Backward cycle) is not selected. When the object moves to the last position (position 3), the next position will be back to the initial position 0, and repeat. The moving path is shown as follows: position 0 → position 1 → position 2 → position 3 → position 0 → position 1 → position 2… If (Backward cycle) is selected, when the object moves to the last position (position 3), it will move backwards to position 2, position 1 and then the initial position 0, and start over again. The moving path is shown as follows. position 0 → position 1 → position 2 → position 3 → position 2 → position 1 → position 0…

Example 1

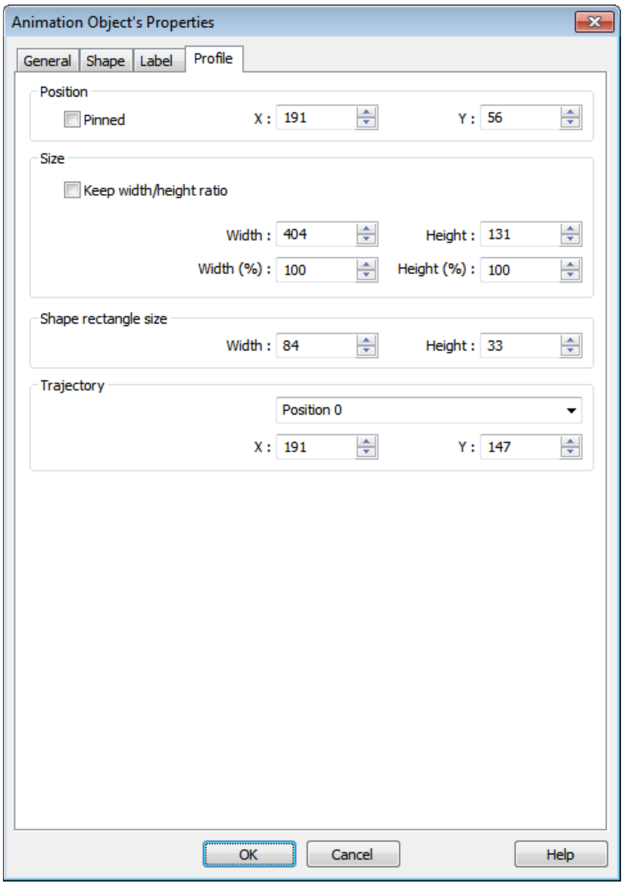
The object's state and position are determined by the registers, and the addresses must be configured correctly, as in the following table:

Data format	16-bit	32-bit
Object state	LW-n	LW-n
Object position	LW-n+1	LW-n+2

For example, if the designated register is LW-100 and the data format is (16-bit Unsigned), then LW-100 represents object's state, LW-101 represents position. In the picture below, LW-100 = 2, LW-101 = 3, so the object's state is 2 and position is 3.



13.14.2.2. Profile Tab



Setting	Description
Shape rectangle size	Set the size of the shape.
Trajectory	Set the position of each point on the moving path.

	**Note** Since multiple pictures might be used by an (Animation) object, (Set to original dimension) will not return all pictures to the original size.

13.15. BAR GRAPH

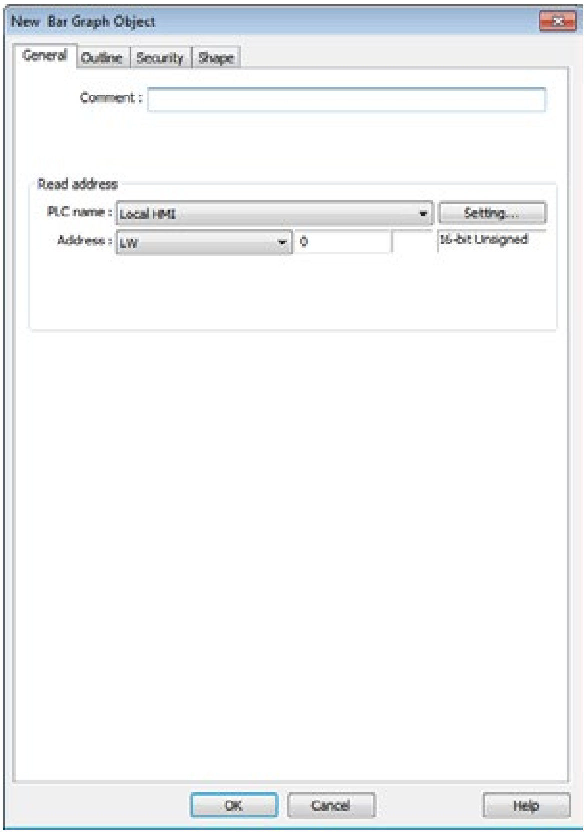
13.15.1. Overview

Bar Graph object displays data as a bar graph for visualization.

13.15.2. Configuration

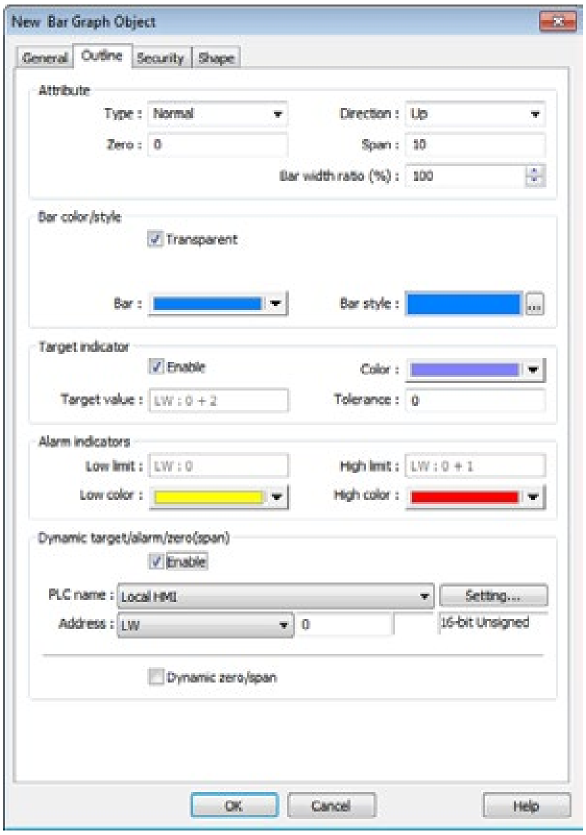
	Click Bar Graph icon on the toolbar to open Bar Graph dialog box. Select properties, click OK button, a new Bar Graph object is created.

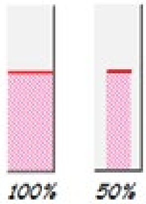
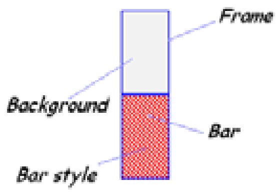
13.15.2.1. General Tab



Setting	Description
Read address	Click (Setting) to Select the (PLC name), (Device type), (Address), (System tag), and (Index register) of the word devices that controls how the bar graph displays.

13.15.2.2. Outline Tab



Setting	Description
Attribute	**Type** Choose either (Normal) or (Offset). If (Offset) is selected, an original value (Origin) must be entered for reference. **Direction** Determine the bar graph direction. Available options are (Up), (Down), (Right), and (Left). **Zero / Span** The percentage of filling can be calculated by the formula, see Example 1. **Bar width ratio (%)** It is the ratio of bar to object width. The figure below shows two ratios, 100% and 50%. 
Bar color/style	Set the bar's frame and background color, bar style, and bar color. See the picture below. 
Target indicator	When the register value meets the condition, the color of filled area will change to the target color, see Example 2.
Alarm indicators	If the register value is larger than (High limit), the color of filled area will change to (High color). If the register value is smaller than (Low limit), the color will change to (Low color).
Dynamic taget/alarm /zero(span)	When (Enable) is selected, the (Low limit) and (High limit) of (Alarm indicator) and the (Target Value) of (Target indicator) will use designated registers, which is shown in their respective fields see Example 3.

Example 1

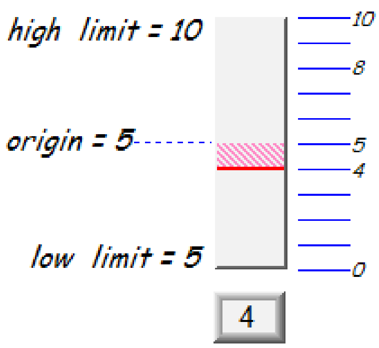
The percentage of filling can be calculated by the following formula:

$$\text{Percentage of filling} = \frac{\text{Register value} - [\text{Zero}]}{[\text{Span}] - [\text{Zero}]} \times 100\%$$

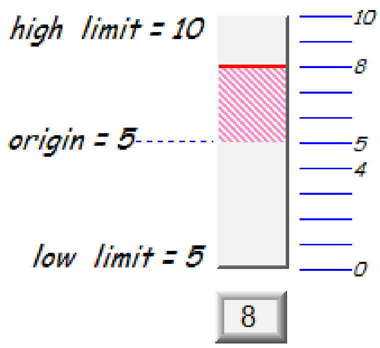
Assume (Offset) is selected. If (Register value – (Zero)) is greater than 0, the bar will fill up from (Origin). If (Register value – Zero) is less than 0, the bar will be drawn below (Origin).

For example, (Origin) is 5, (Span) is 10, and (Zero) is 0.
For different value in read address, it will display as below:

If the value at read address is 4:

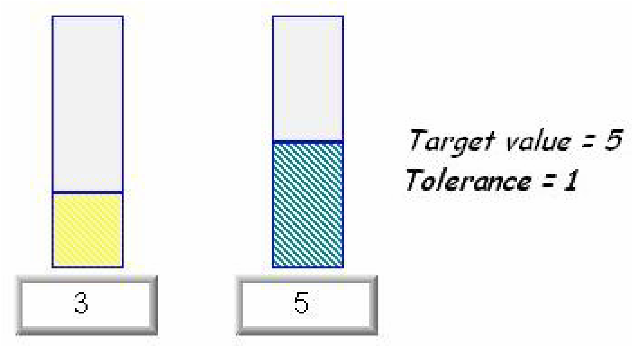


If the value at read address is 8:



Example 2
When the register value meets the following condition, the color of filled area will change to the target color.
[Target Value] - [Tolerance] ≤ Register value ≤ [Target Value] + [Tolerance]

Assume (Target Value) is 5 and (Tolerance) is 1. As shown below, if the register value is equal to or larger than 4 (=5-1) and equal to or less than 6 (=5+1), the filled area's color of the bar will change to the target color.



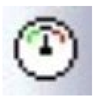
Example 3
If (Dynamic target/alarm) is enabled, (Low limit) and (High limit) of (Alarm indicator) are defined by designated registers as shown in the following table. Furthermore, if (Dynamic zero/span) is used, (Zero), (Span) and (Origin) will be defined by designated registers. Assume the address is LW-n, the limits are:

Data format	16-bit	32-bit
Alarm low limit	LW-n	LW-n
Alarm high limit	LW-n+1	LW-n+2
Target	LW-n+2	LW-n+4
Zero	LW-n+3	LW-n+6
Span	LW-n+4	LW-n+8
Origin	LW-n+5	LW-n+10

13.16. METER DISPLAY

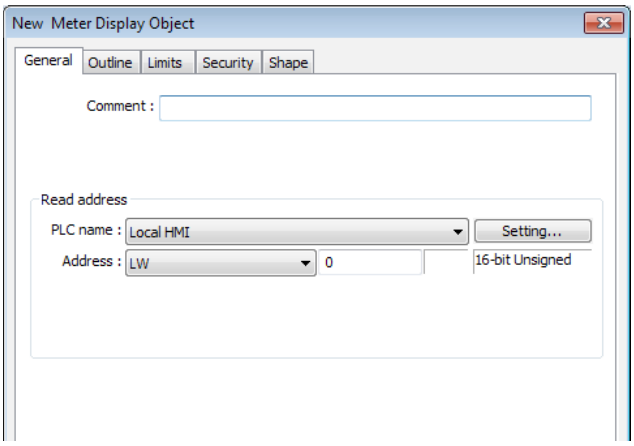
13.16.1. Overview
Meter Display object displays the value of word register with a meter.

13.16.2. Configuration

	Click the Meter Display icon on the toolbar to open the Meter Display dialog box. Set the object's attributes and then click OK to create a new Meter Display object.

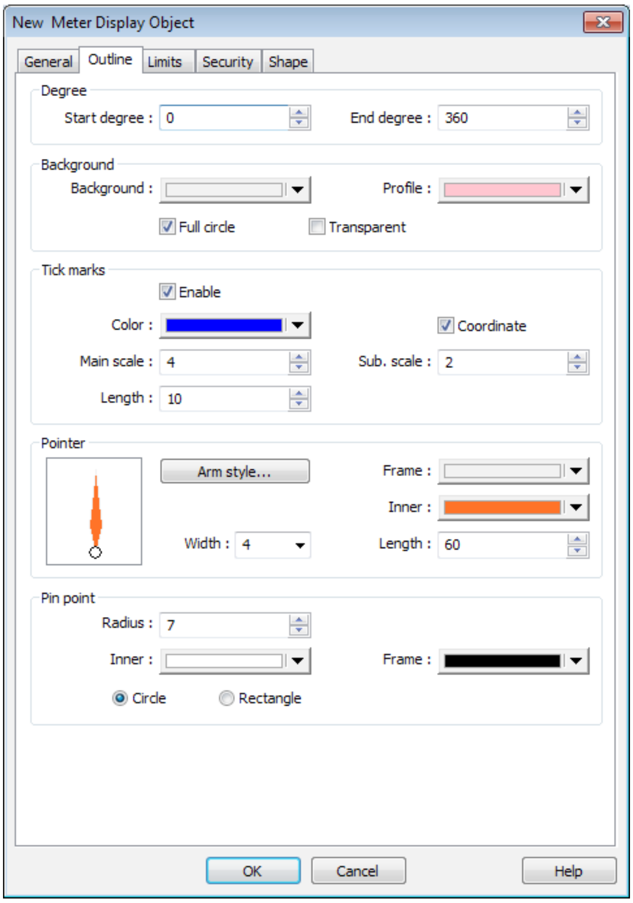
13.16.2.1. eMT, iE, XE, mTV Series

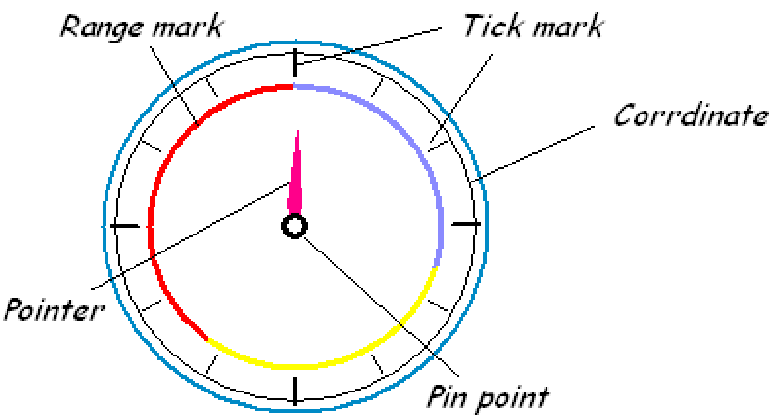
- General Tab

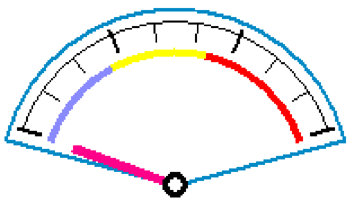
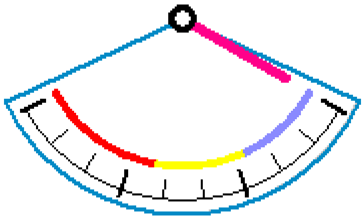
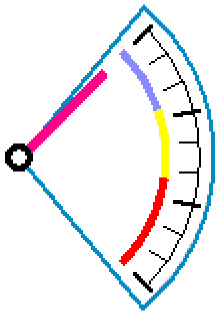
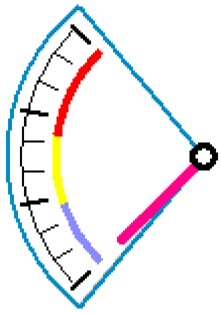
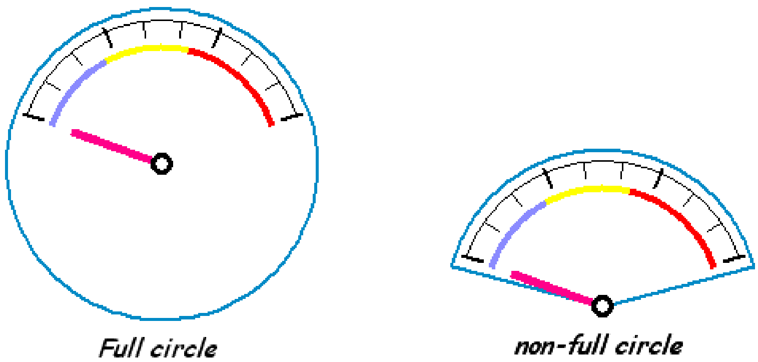


Setting	Description
Read address	Click (Setting) to select the (PLC name), (Device type), (Address), (System tag), and (Index register) of the word devices that controls the Meter Display object.

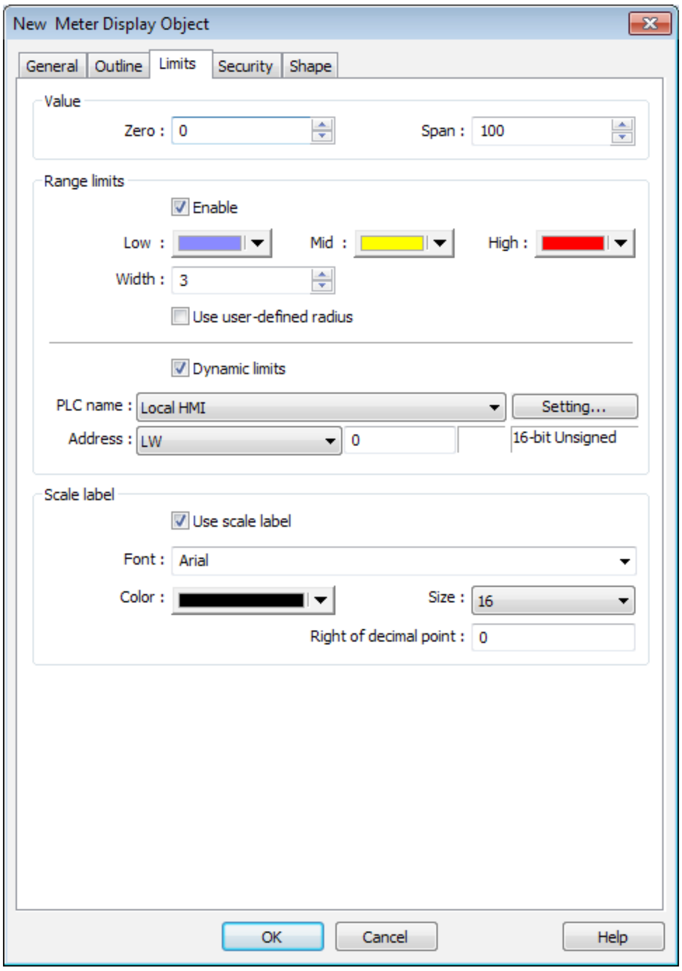
▪ Outline Tab

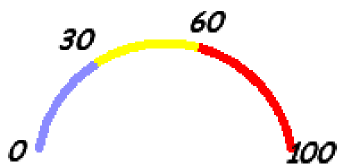


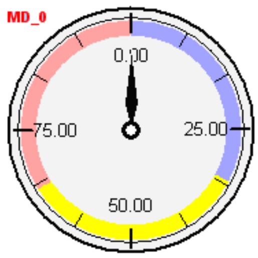


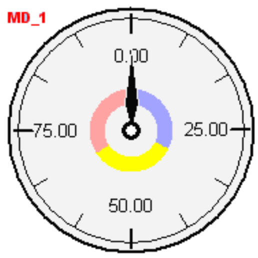
Setting	Description
Degree	Sets the object's start degree and end degree measured clockwise from the 12 o'clock position. The angle range is 0 to 360 degrees. The following shows meters of different settings. ▪ (Start degree) =290º (End degree) = 70º  ▪ (Start degree) = 120º (End degree) = 240º  ▪ (Start degree) = 40º (End degree) = 140º  ▪ (Start degree) = 225º (End degree) = 315º 
Background	Sets the object's background color and profile color. **Full circle** When selected, the object will display the whole circle. Otherwise, the object will only display a partial circle in the defined degree range, as shown in the following figure.  **Transparent** When selected, the object will not display the background and profile color.
Tick marks	Configures the number of tick mark and color.
Pointer	Configures pointer's style, length, width, and color.
Pin point	Configures the style, radius, and color of the pin point.

13.16.2.2. Limits Tab

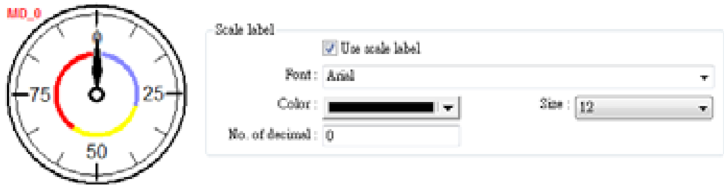


<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Value</td><td>Sets the object’s display range. Meter Display object will use the value of (Zero) and (Span) and the value of register to calculate the pointer’s position. See Example 1.</td></tr>
<tr><td>Range limits</td><td>Configures the values of (Low limit), (High limit), their corresponding display colors, and the width.


Use user-defined radius
Configures the radius to display range limits.
For example:
Set to 80:


Set to 30:


Dynamic Limits
The low limit and high limit are set by registers. See Example 2.</td></tr>
</table>

Scale label	Select the attribute of scale label on Meter Display. 

Example 1: pointer position calculation
Set object's display range. Meter Display object will use the value of (Zero) and (Span) and the value of register to calculate the pointer's position. For example, supposed that (Zero) is 0, (Span) is 100, when the value of register is 30, (Start degree) is 0, and (End degree) is 360, then the degree indicated by the pointer is:

{ (30 – [Zero]) / ([Span] – [Zero]) } * ([End degree] - [Start degree]) = {(30 – 0) / (100 - 0)} * (360 – 0) = 108

Pointer will be pointing at 108 degrees.

Example 2: dynamic Limits
The low limit and high limit are set by the register.
Suppose the address is LW-n, the following table shows the read address of low limit and high limit:

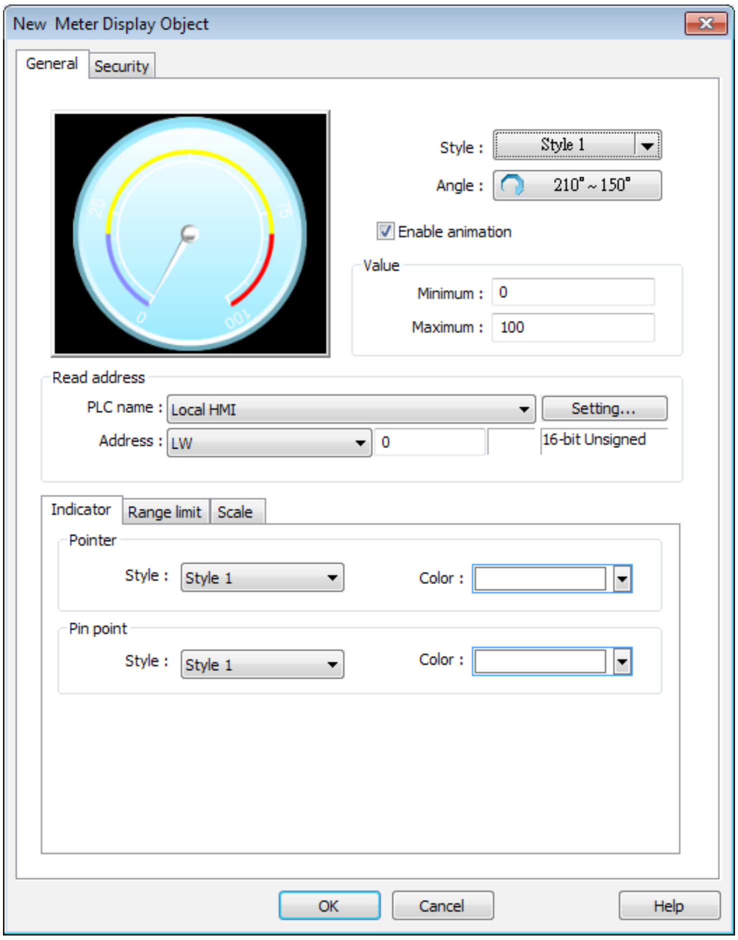
Content	16-bit	32-bit
Low limit	LW-n	LW-n
High limit	LW-n+1	LW-n+2

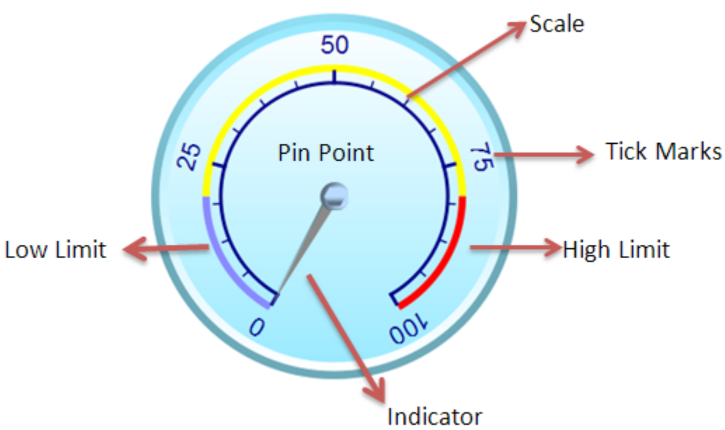
For instance, when address is LW-100, the rule of setting limits is:

Content	16-bit	32-bit
Low limit	LW-100	LW-100
High limit	LW-101	LW-102

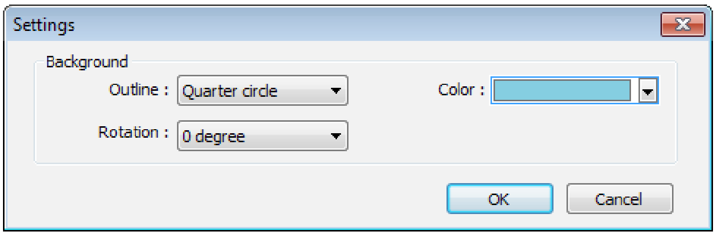
13.16.2.3. cMT Series

▪ General Tab

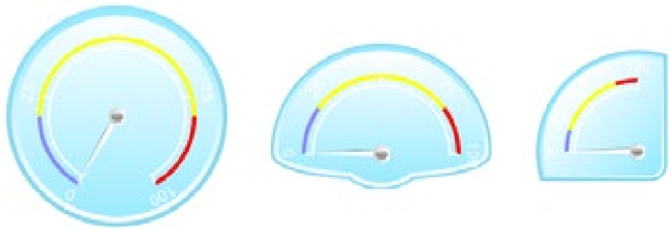




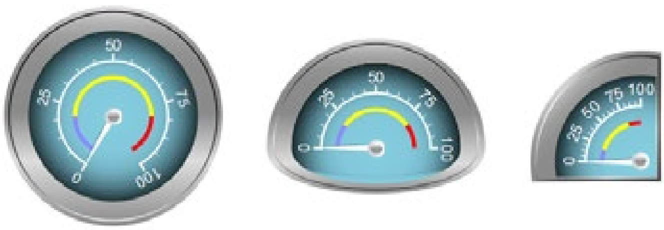
<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Style</td><td>Three options are available: (Custom), (Style 1) and (Style 2). Click on the text to configure the properties. If (Custom) is selected, set the properties such as (Pointer), (Pin point), (Background picture), etc.</td></tr>
<tr><td>Setting</td><td>
Click the button of (Style 1) or (Style 2) to set the style of meter.



Outline
The following is the outline of Style 1 when (Full circle), (Half circle), or (Quarter circle) is selected.



The following is the outline of Style 2 when (Full circle), (Half circle) or (Quarter circle) is selected.



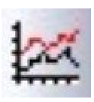
Rotation
Rotates the background picture clockwise according to the angles set.
Color
Sets the color of the background picture of meter.
</td></tr>
<tr><td>Angle</td><td>Sets the range to label the scale, 0º to 360º clockwise or counterclockwise from the twelve o’clock position.
Full circle
If selected, the full circle is drawn according to the selected direction and the start angle. The limits are determined by the value set in (Minimum) and (Maximum) field under (Value).</td></tr>
<tr><td>Enable animation</td><td>If selected, the pointer slides to the designated position when the read value changes; if not selected, the pointer directly points to the designated position when the read value changes.</td></tr>
<tr><td>Value</td><td>Sets the lower and upper limits of the meter.</td></tr>
<tr><td>Read address</td><td>Displays the value in meter according to the value in the designated word register.</td></tr>
<tr><td>Indicator</td><td>Sets the style of pointer and pin point. If (Custom) is selected, the direction of the pointer must points upward to correctly display.</td></tr>
<tr><td>Range limit</td><td>Sets the colors to indicate different ranges.
Dynamic limits
The low limit and high limit are decided by the register. See Example 2 above.</td></tr>
<tr><td>Scale</td><td>Sets the number of main and sub scale, the color of tick marks and scale label.</td></tr>
</table>

13.17. TREND DISPLAY

13.17.1. Overview

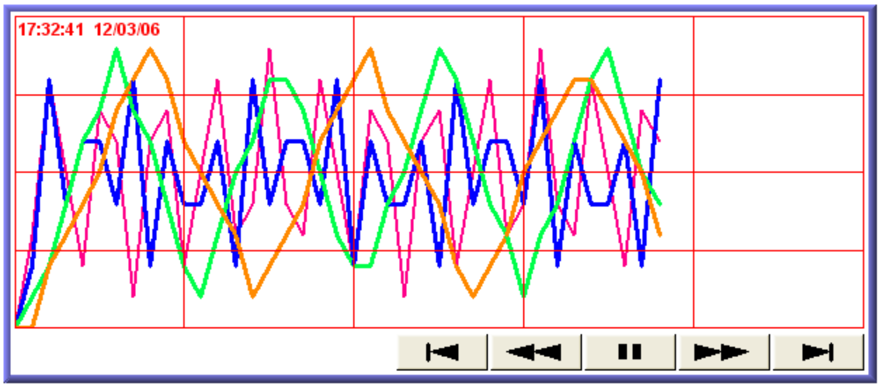
Trend display objects draw curves of the data recorded by Data Sampling object.

13.17.2. Configuration

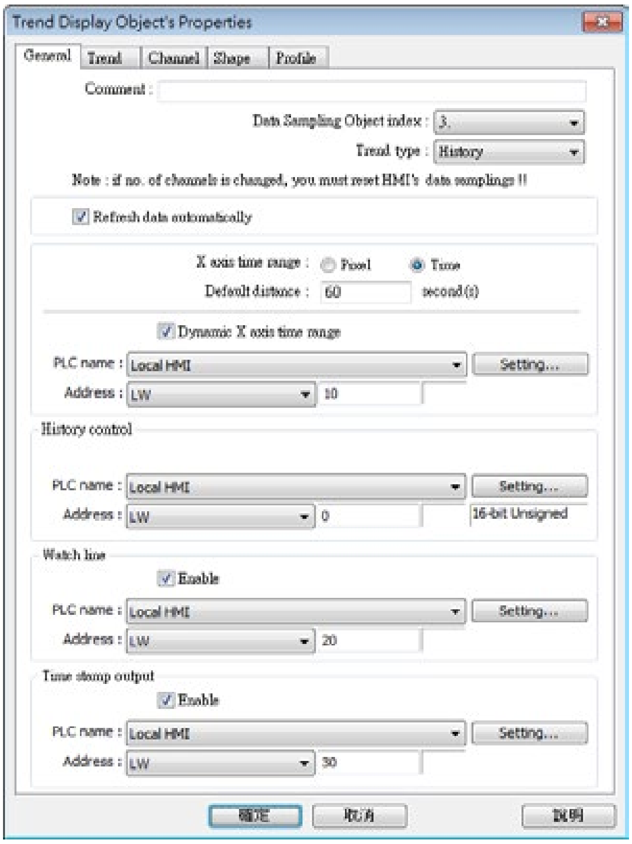
	Click the Trend Display icon on the toolbar to open a Trend Display object property dialog box. Set up the properties, press OK button, and a new Trend Display object will be created.

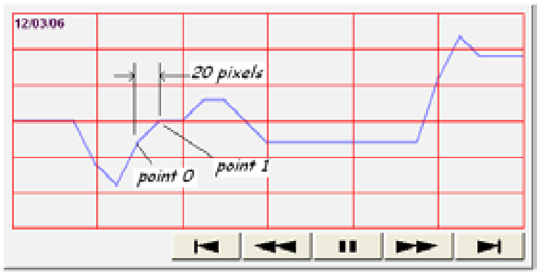
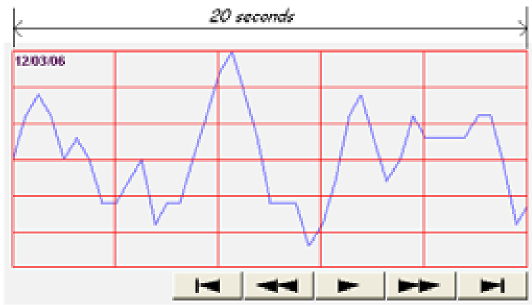
13.17.2.1. eMT, iE, XE, mTV Series

▪ General Tab



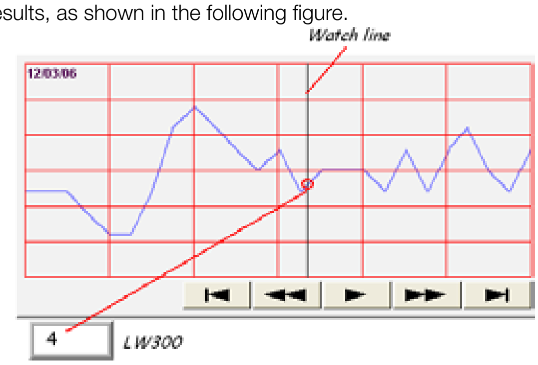
Button	Description
	Go to the earliest sampling data.
	Go to the previous time interval.
	Click to stop auto-scrolling. When the new sampling data is generated, the display does not scroll, nor is the new data outside the display range displayed.
	Click to start auto-scrolling. The display scrolls as the new sampling data is generated.
	Go to the next time interval.
	Go to the latest sampling data.



<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Data Sampling Object index</td><td>Select a Data Sampling object as the source data.</td></tr>
<tr><td>Trend type</td><td>Select the mode of data source, either (Real-time) or (History).
Real-time
In this mode, the display object shows all sampled data since the HMI started. The maximum number of records that can be sampled is set in (Max.data records) (Real-time mode) of the Data Sampling object. When the sampling data exceed this setting, the earlier data will be deleted.To show older data, use (History) mode.
(Hold control): suspends the update of Trend Display. However, It does not stop the sampling process of Data Sampling object.
History
In this mode, the data comes from the history data files stored on HMI. . The history data files are sorted by dates and each is given an index. The system uses (History control) to select the history data files that are created on different dates.
The system sorts the history data of sampling data by date; the latest file is record 0 (typically the data sampled today), the second latest file is record 1, and so on. If the value of designated register in (History control) is n, the Trend Display object will display data record n.
Here is an example to explain (History control). If the designated register is LW-0, and the sampling data files available are pressure_20061120.dtl, pressure_20061123.dtl, pressure_20061127.dtl, and pressure_20061203.dtl, and it is 2006/12/3 today, based on the value of LW-0, the sampling data file which will be selected by (Trend Display) is shown as follows:
<table>
<tr><th>Value of LW-0</th><th>Selected sampling history data</th></tr>
<tr><td>0</td><td>pressure_20061203.dtl</td></tr>
<tr><td>1</td><td>pressure_20061127.dtl</td></tr>
<tr><td>2</td><td>pressure_20061123.dtl</td></tr>
<tr><td>3</td><td>pressure_20061120.dtl</td></tr>
</table>
If use with Option List object and select data source as (Dates of historical data), the history data will be sorted by date and displayed in Option List object, see “13.29 Option List”.</td></tr>
<tr><td>Pixel</td><td>(Distance) is used to set the distance between two sampling points, as shown in the following figure.

</td></tr>
<tr><td>Time</td><td>(Distance) is used to set the X-axis in unit of time, as shown in the following figure.


Select (Time) for (X axis time range) and go to (Trend) » (Grid) and enable (Time scale). Please refer to (Time scale) in the later section.</td></tr>
<tr><td>Dynamic distance between data samples/ Dynamic X axis time range</td><td>Designate a 32-bit word register for adjusting the distance between two sampling points (select Pixel), or the time unit represented by X-axis (select Time). If no value is entered, the default value will be used.</td></tr>
</table>

<table>
<tr><td>Refresh data automatically</td><td>If selected, every time when opening the window on which the Trend Display object in history mode is placed, the display is automatically refreshed once per second. Please note that:
▪ The refresh status can be observed from the control button of Trend Display object.
▪ Showing button: the automatic refresh feature is enabled.
▪ Showing button: the automatic refresh feature is disabled.
▪ When scrolling to the previous data, the automatic refresh feature is disabled, the button is shown.
▪ If (Refresh data automatically) check box is selected, when change back to the window, the display is refreshed, ignoring the control buttons. For example, select (Refresh data automatically), and scroll to the previous data, the automatic refresh feature is disabled. In this case, changing to another window and then change back will still refresh the display.
▪ If (Refresh data automatically) check box is not selected when building the project, the feature can still be enabled by pressing button on HMI. In this case, the automatic refresh feature is disabled, that is, even when change back to the current window, the display will not be refreshed.</td></tr>
<tr><td>Hold control</td><td>When the register is set ON, suspend the update of Trend Display. It does not stop the sampling process of Data Sampling object. This setting is available only in Real-time mode.</td></tr>
<tr><td>Watch line</td><td>Use the (Watch line) function to display a “watch line” when user touches the Trend Display object. It will also export the sampling data at the position of watch line to the designated word device and use Numeric objects to display the results, as shown in the following figure.



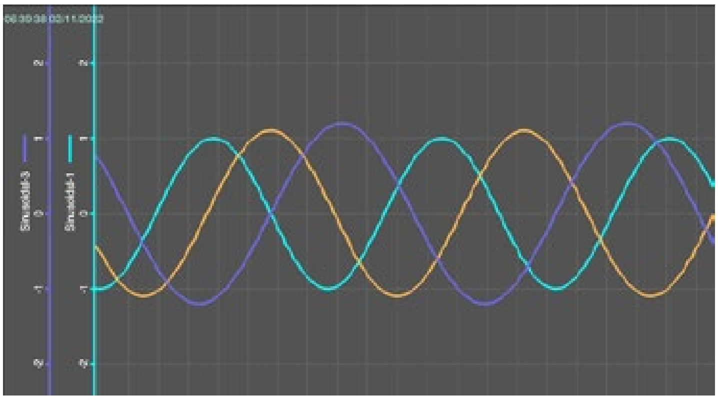
(Watch line) can also export sampling data with multiple channels. The system will consecutively write each channel to the specified address and the following addresses, in the same order as in (Data Sampling) object. The address assigned to (Watch line) is the start address, and sampling data for each channel will be exported to the word devices starting from “start address.” If the data format of each channel is different, the corresponding address of each channel is arranged from the first to the last. If the watch register is LW-300, watch function will export each channel’s data to the following addresses:
<table>
<tr><th>Register</th><th>Channel</th><th>Data format</th></tr>
<tr><td>LW-300</td><td>0</td><td>16-bit Unsigned (1 word)</td></tr>
<tr><td>LW-301</td><td>1</td><td>32-bit Unsigned (2 words)</td></tr>
<tr><td>LW-303</td><td>2</td><td>32-bit float (2 words)</td></tr>
<tr><td>LW-305</td><td>3</td><td>16-bit Signed (1 word)</td></tr>
</table></td></tr>
<tr><td>Time stamp output</td><td>Time stamp output
Suppose the address is set to LW-n, then:
If enabled, the system will use the time of the first sampling data as “time origin”, and write the time stamp of the most recent sampled data (relative to “time origin”) to (LW-n+2).
When clicking on the curve, the time stamp of the closest sampled point will be written to (LW-n).
Clear real-time data address (Data Sampling object) will clear the time origin as well.
Time stamp is recorded in seconds.</td></tr>
</table>

Note

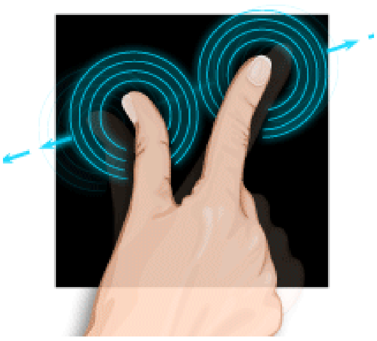
- LW-n and LW-n+2 are both in 32-bit format.
- LW-n is for both real-time and history mode, whereas LW-n+2 only applies to real-time mode.
- This function is available upon enabling (relative time mode) in (Trend) tab.

13.17.2.2. cMT Series

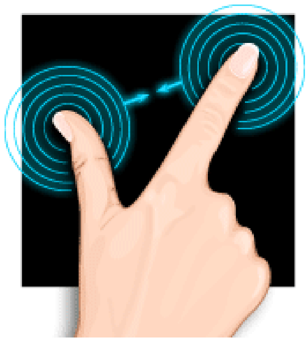
- General Tab



The Trend Display on cMT Series combines Real-time mode and History mode. Drag left to scroll the Trend Display to view history data and drag right to view the latest sampling data. Pinch two fingers together to zoom out Trend Display or spread them apart to zoom in.

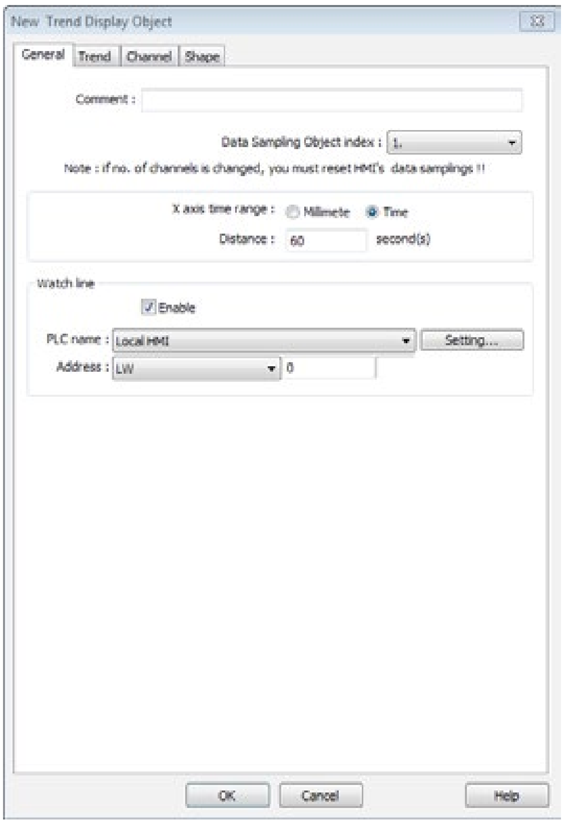


Zoom In



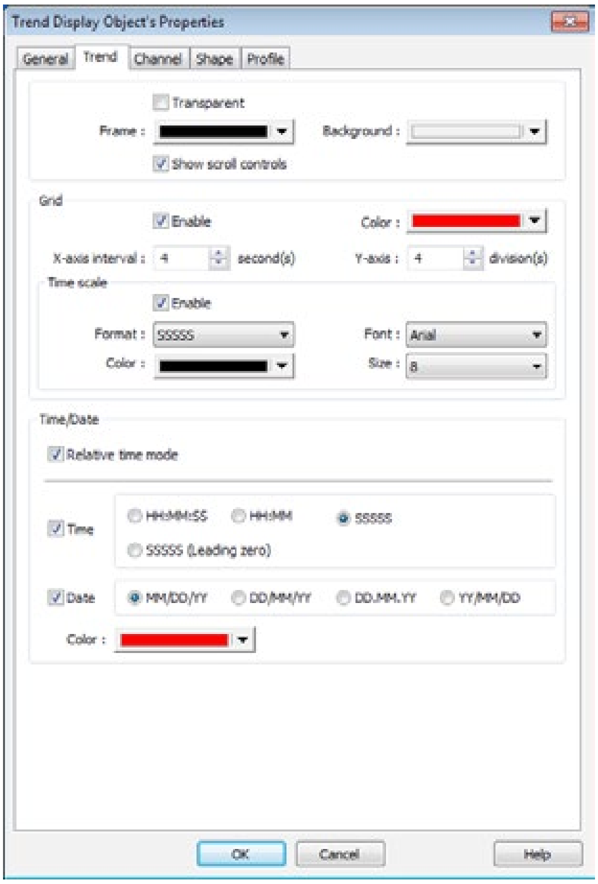
Zoom Out

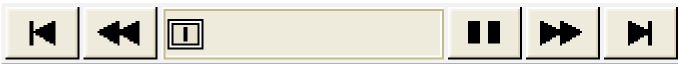
☞ For more information about how sampling data is saved, see “8 Data Sampling”.

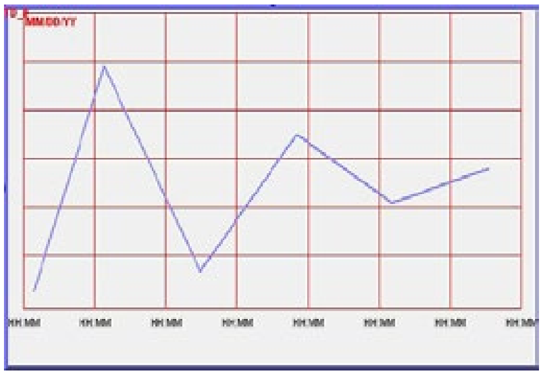


Setting	Description
Data sampling object index	Select a (Data Sampling) object as the source data.
Millimeter	See 13.1.2.1.
Time	See 13.1.2.1.
Watch line	See 13.1.2.1.

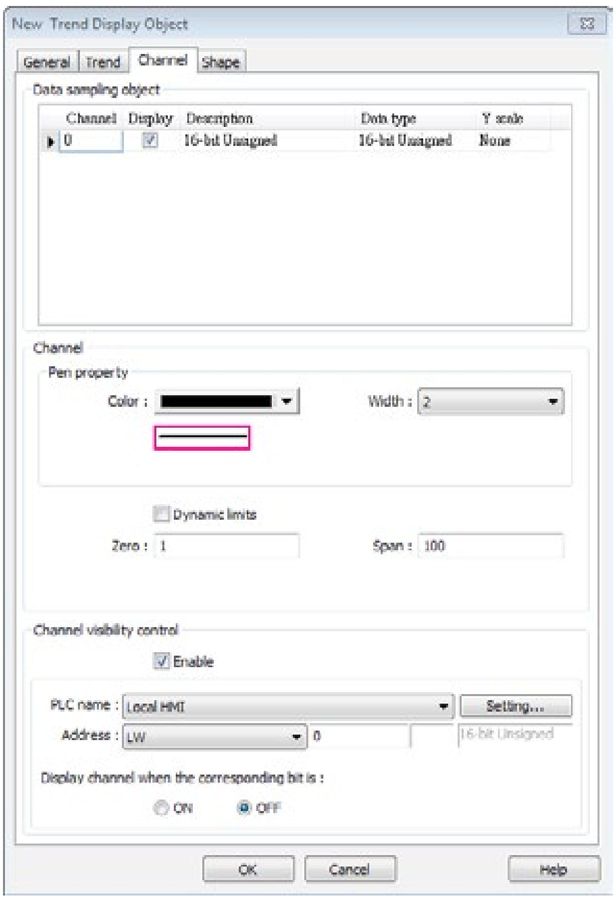
- Trend Tab



Setting	Description
Frame / background	Select the color of frame and background.
Show scroll controls	Enable or disable the scroll control as shown in the following figure. 
Grid	Set the number of dividing lines and the line color. The number of divisions depends on the setting in General tab. **X-axis interval** The number of vertical grid lines. ▪ Select (Pixel) / (Millimeter) in General tab: ▪ Select how many sampling point will be included between two vertical grid lines ▪ Select (Time) in General tab: ▪ Select the time range between two vertical grid lines **Y-axis interval** The number of horizontal grid lines.

Time scale	Select (Enable) check box to display the time scale along the x axis.  **Format** The available time scale formats: HH:MM or HH:MM:SS. **Font / Color / Size** Select the font, font color, and font size of the time scale. The default font size is 8.
Time / date	The time of latest sampling data will be marked on the top left corner of the object. This group box is used to set the time display format and font color.

▪ Channel Tab

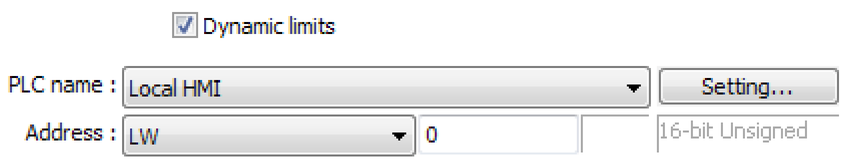


<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Y scale</td><td>Available only for cMT Series, see “13.17.2.3 Y Scale Usage”.</td></tr>
<tr><td>Channel</td><td>Configure each sampling line’s format and color. At most 64 channels could be configured.
Dynamic limits
▪ Not selected:
(Zero) and (Span) are used to set the low limit and high limit of sampling data. If the low limit is 50 and the high limit is 100 for one sampling line, (Zero) and (Span) must be set as (50) and (100), so that all the sampling data can be displayed in the trend display object.
▪ Selected:
The low limit and the high limit are read from the designated word devices, as shown below. When address is LW-n , the register’s address:
<table>
<tr><th>Data format</th><th>16-bit</th><th>32-bit</th></tr>
<tr><td>Low limit</td><td>LW-n</td><td>LW-n</td></tr>
<tr><td>High limit</td><td>LW-n+1</td><td>LW-n+2</td></tr>
</table>
For example, if LW-100 is used here, the low limit and the high limit will be read from:
<table>
<tr><th>Data format</th><th>16-bit</th><th>32-bit</th></tr>
<tr><td>Low limit</td><td>LW-100</td><td>LW-100</td></tr>
<tr><td>High limit</td><td>LW-101</td><td>LW-102</td></tr>
</table>
A typical usage of this is to zoom in and zoom out of Trend Display. (Not available for cMT Series). See Example 1.</td></tr>
<tr><td>Channel visibility control</td><td>If (Enable) is selected, the bits of the assigned word register will be used to show/hide each channel. The first bit controls the first channel, and the second bit controls the second channel, and so on. For example, suppose there are 5 channels and LW-0 is used, channels which will be shown given the states of the control bits are:
<table>
<tr><th>Channel</th><th>Control bit</th><th>State</th><th>Displayed</th></tr>
<tr><td>1</td><td>LW_bit-000</td><td>OFF</td><td>YES</td></tr>
<tr><td>2</td><td>LW_bit-001</td><td>ON</td><td>NO</td></tr>
<tr><td>3</td><td>LW_bit-002</td><td>ON</td><td>NO</td></tr>
<tr><td>4</td><td>LW_bit-003</td><td>OFF</td><td>YES</td></tr>
<tr><td>5</td><td>LW_bit-004</td><td>OFF</td><td>YES</td></tr>
</table>
Note on using this feature: each control bits are not reserved for the channel. If a particular channel is not displayed, the control bit is assigned to the next displayed channel. For example, if the third channel of the 5 channels is not displayed, only 4 channels will be displayed in Trend Display, and the used control bits will only be: LW_bit-000~003.</td></tr>
</table>

Example 1

The example explains how to zoom in or zoom out Trend Display. The feature described is not available for cMT Series.

In Channel tab select (Dynamic limits) check box. If the (Address) is set to LW-n, then LW-n controls the low limit where LW-n+1 controls the high limit.



Set (Address) to LW-0 and create two Numeric objects for entering the low / high limit. The address that controls the low limit is LW-0; the address that controls the high limit is LW-1.
Let's suppose the data is between 0 and 30; set the (Low limit) to 0 and the (High limit) to 30, the trend curve is displayed as shown in the following figure.

To zoom out the Trend Display, enter a value greater than 30 in (High limit) as shown in the following figure.

To zoom in the Trend Display, enter a value less than 30 in (High limit) as shown in the following figure.

13.17.2.3. Y Scale Usage
This feature is only available for cMT Series. The scale along the Y axis of a specific channel can be displayed. To enable Y Scale, (Grid) should first be enabled in (Trend) tab. Y Scale can be configured on the iPad as shown in the following steps.

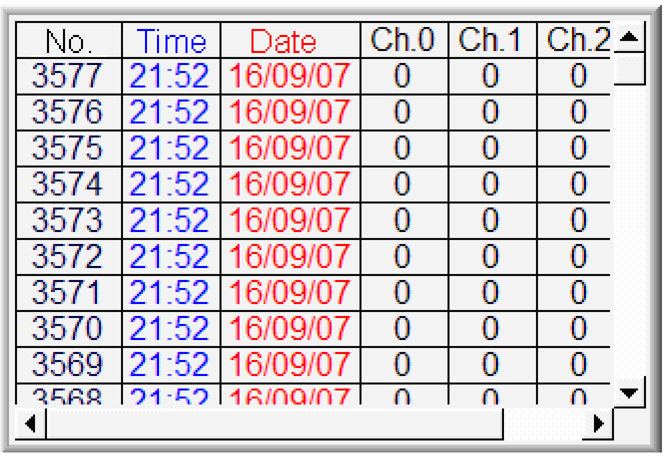
1. Tap the button on the upper right corner of Trend Display object.
2. Tap (Trend Display Setting) » (Y Scale).

3. Select the channels.

13.18. HISTORY DATA DISPLAY

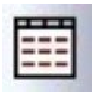
13.18.1. Overview

History Data Display object displays data stored by Data Sampling object. It differs from Trend Display in that History Data Display object uses a table to display data. In case when the trend display shows history data from today, the display will refresh once per second. The following is an example of a history data display object.

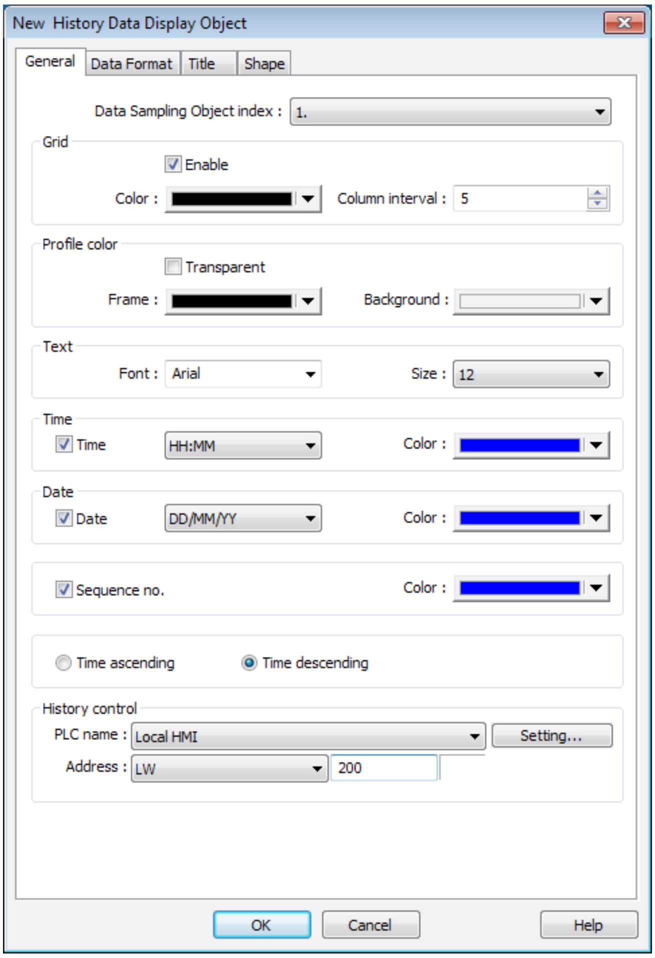


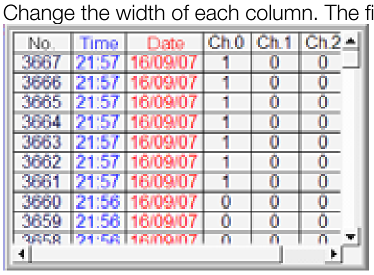
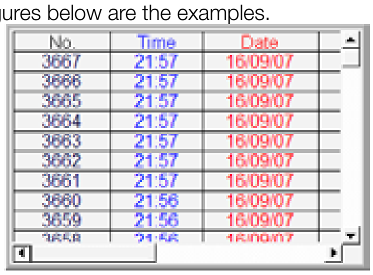
No.	Time	Date	Ch.0	Ch.1	Ch.2
3577	21:52	16/09/07	0	0	0
3576	21:52	16/09/07	0	0	0
3575	21:52	16/09/07	0	0	0
3574	21:52	16/09/07	0	0	0
3573	21:52	16/09/07	0	0	0
3572	21:52	16/09/07	0	0	0
3571	21:52	16/09/07	0	0	0
3570	21:52	16/09/07	0	0	0
3569	21:52	16/09/07	0	0	0
3568	21:52	16/09/07	0	0	0

13.18.2. Configuration

	Click the History Data Display icon on the toolbar to open a History Data Display object property dialog box. Set up the properties, press OK button, and a new History Data Display object will be created.

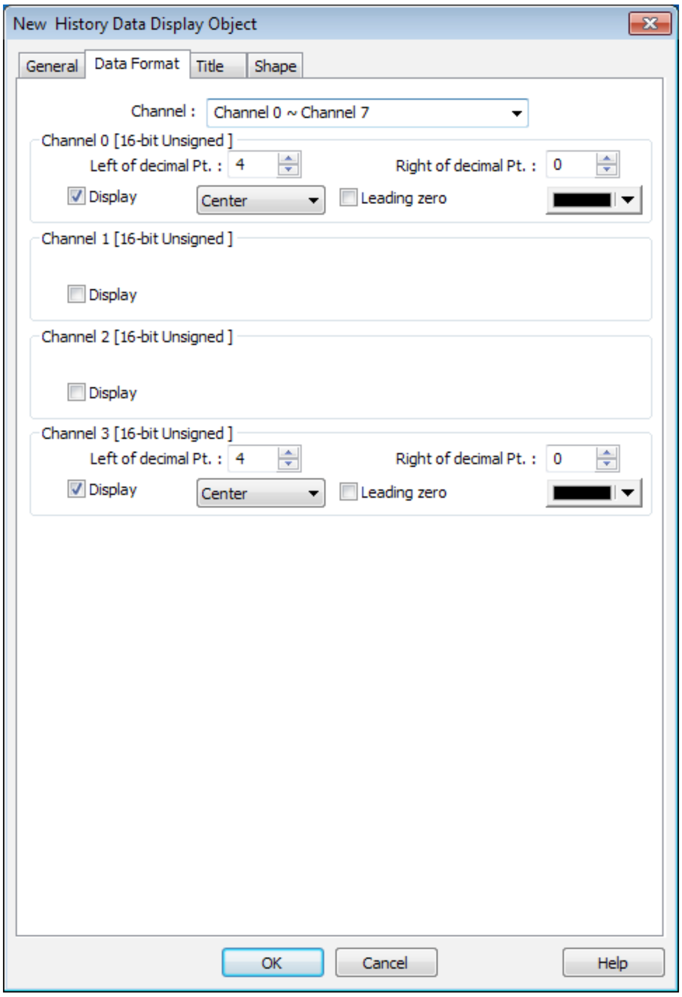
13.18.2.1. General Tab

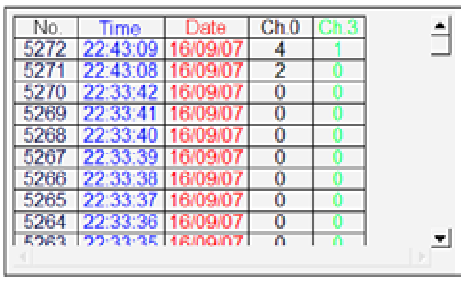
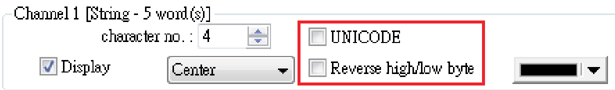


Setting	Description
Data sampling object index	Select a Data Sampling object as the source data.
Grid	Shows grids between rows and columns. **Color** Change the color of grids. **Column interval** Change the width of each column. The figures below are the examples.  
Profile color	Change the color of frame and background. Use (Transparent) to hide frames and background.
Time / date	Enable or disable showing the time and date and configure its format. **Time ascending** Put earlier data at the top and the latest data at the bottom. **Time descending** Put the latest data at the top and the earlier data at the bottom.
History control (eMT, iE, XE, mTV Series)	The history files are sorted by date and each file is given an index. The latest one is assigned index 0 (in most cases: today), the second latest file is assigned index 1, and so on. (History Control) is used to specify the history data to be shown.

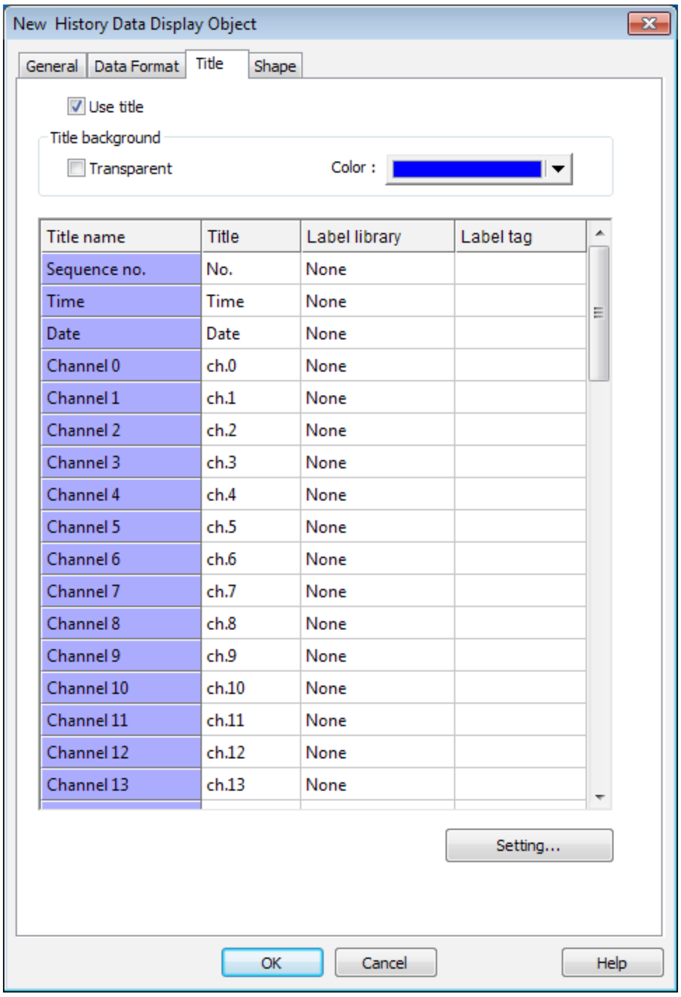
Note
When using cMT-SVR, use the filter icon in the upper-right corner of History Data Display object on iPad to select the date and display the data.

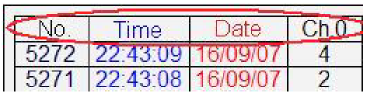
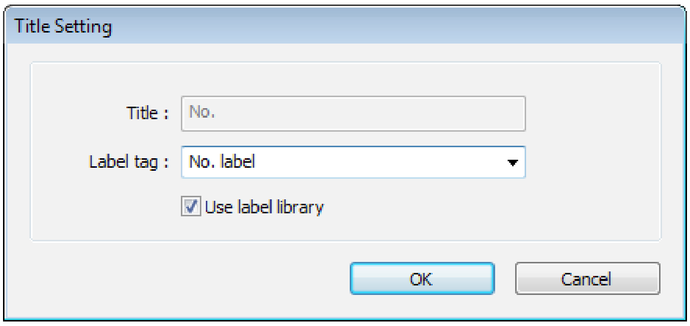
13.18.2.2. Data Format Tab



Setting	Description
Channel	Each History Data Display object can display up to 64 channels. Check (Display) to select the channels to be shown on the screen. In the figure above, there are 4 channels (channel 0 to channel 3) in the Data Sampling object, and only Ch.0 and Ch.3 are selected. The data formats are shown next to channel name. The data format of each channel is decided by the corresponding Data Sampling objects. The result is shown below:  When display (String) format in History Data Display object, there are two additional options:  ▪ Display in (UNICODE) mode ▪ Reverse high byte and low byte data and then display.

13.18.2.3. Title Tab



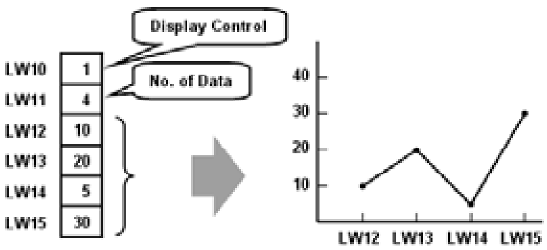
Setting	Description
Use title	Enable or disable title, which is marked as shown below: 
Title background	**Transparent** When selected, hide the background for title area. **Color** Set the background color of title.
Setting	Defines the text to be shown on the title. Label Tag Library can be used for title in order to display the title in multiple languages. Click (Setting) and select (Use label library). 

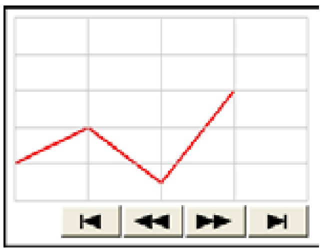
Note
During off-line simulation, if the format of sampling data has been changed, please delete previous data records in C:\(EasyBuilder Pro directory)\HMI_memory\datalog to prevent the system from misinterpreting the old data records.

13.19. DATA BLOCK DISPLAY

13.19.1. Overview

Data Block is a combination of several word devices with continuous address, where the X axis of Data Block Display object represents the address and the numbers on the Y axis represent the data values in the corresponding address. Data Block Display object can display multiple data blocks. For example, it can display two data blocks LW-12~LW-15 and RW-12~RW-15 in trend curves simultaneously. It is very useful to observe and compare the difference of trend curves.



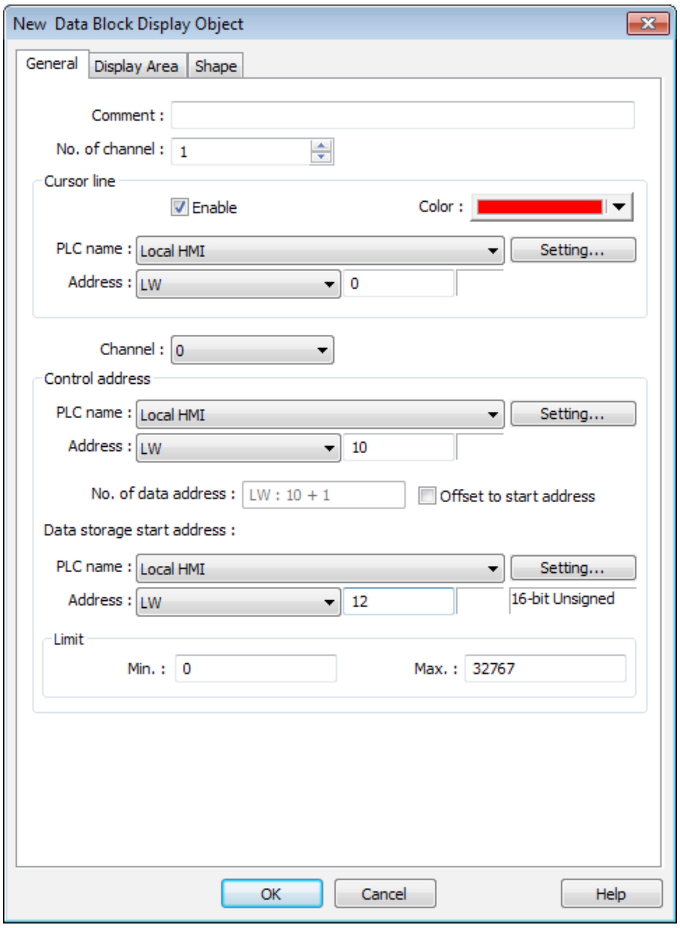


The display result

13.19.2. Configuration

Click the Data Block Display icon on the toolbar to open the property dialog box.
Set up the properties, press OK button, and a new Data Block Display object will be created.

13.19.2.1. General Tab



Setting	Description
Comment	Description of the object.
No. of channel	Set the no of channel for this object. Each channel represents one data block. The maximal number of channels is 12.
Cursor line	If enabled, when user touches the (Data Block Display) object, it will display a vertical cursor line on it, and store the data on the line to the designated registers. See Example 1.
Channel	Select the channel to be configured.
Control address	Specify the control address also the data source. Control address is used to control and clear the drawn curve. After executing the operation below, the system will reset the control word to zero. Enter “0” = No action (default) Enter “1” = Draw (Without clear first) Enter “2” = Clear Enter “3” = Redraw **No. of data address** If control address is LW-n, then LW-n+1 stores the number of word devices in each data block, i.e. the number of data. The maximum value is 1,024. **Data storage start address** If (Offset to start address) is enabled, the (Offset value storage address) will be set as (Control address) + 2. If select 16-bit data format, the address for each data will be start address, start address + 1, start address + 2 and so on. If select 32-bit data format, the address for each data will be start address, start address + 2, start address + 4 and so on. For more information about control address, see Example 2 to 5.
Limit	Set the minimum and maximum limit for the curve.

Note
The system can draw at most N curves, where N = 32 divided by the number of channel.

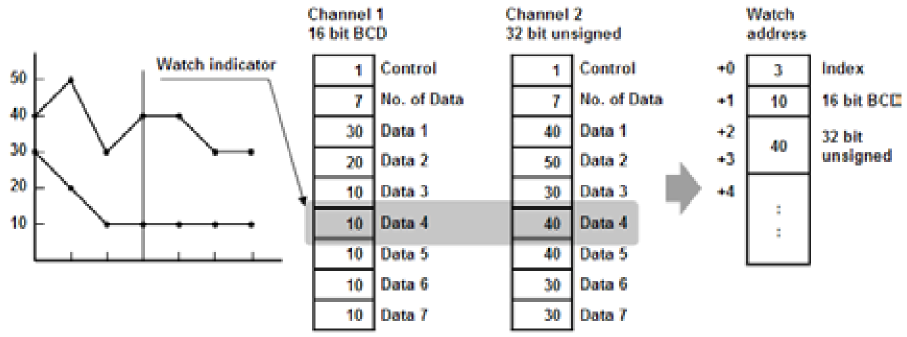
Example 1

- How to use watch (Cursor Line) feature

Use “Watch” function to check the value of any point of the curve. When the user touches (Data Block) object, it will display a “cursor line”, and the system will write the index and value of that data on the cursor line to the designated address.

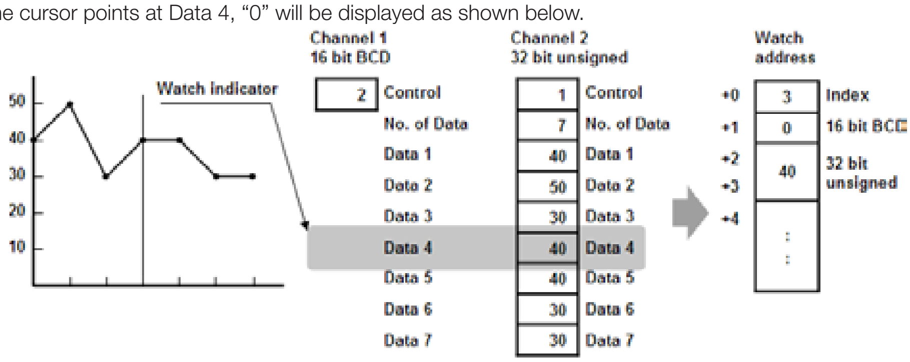
Data format	Index value	Channel 1 value	Channel 2 value
16-bit	Address	Address + 1	Address + 2
32-bit	Address	Address + 2	Address + 4

When watch address is set to LW-n, the value written into LW-n represents the channel index number to be called up. (Start form 0)

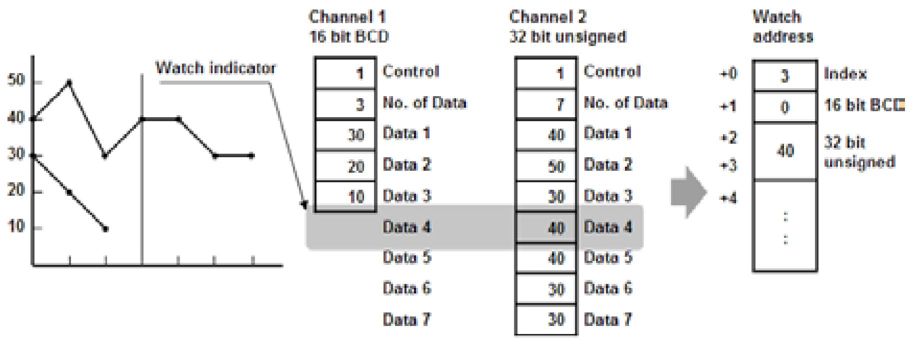


Note

- (Data Index) is a 16 bit unsigned integer. When the designated register of cursor line is 32 bit device, it will be stored in the bit 0-15.
- If the channel to be viewed has no data, “0” will be displayed, as shown below. In the example, there is no data in channel 1, when the cursor points at Data 4, “0” will be displayed as shown below.



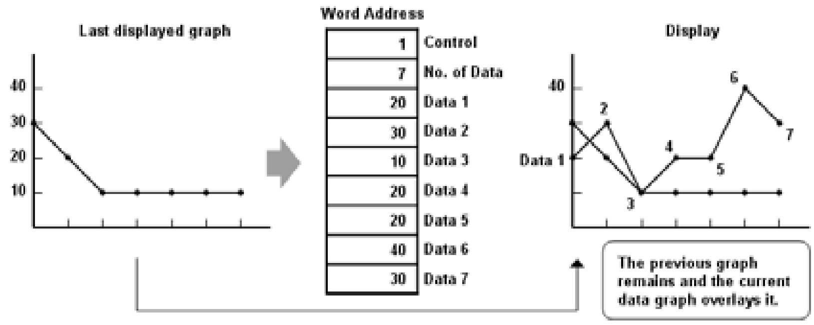
- If there is less data in Channel 1, “0” will be displayed, as shown below.



Example 2

▪ How to show a data block

1. Write the number of data to (No. of data address), i.e. "Control address+1"
2. Store the data consecutively beginning at (Data storage start address).
3. Write "1" to (Control address) to draw the curve without cleaning the plot. All previous curves will not be erased.
4. The system will write "0" to (Control address) after marking the plot.

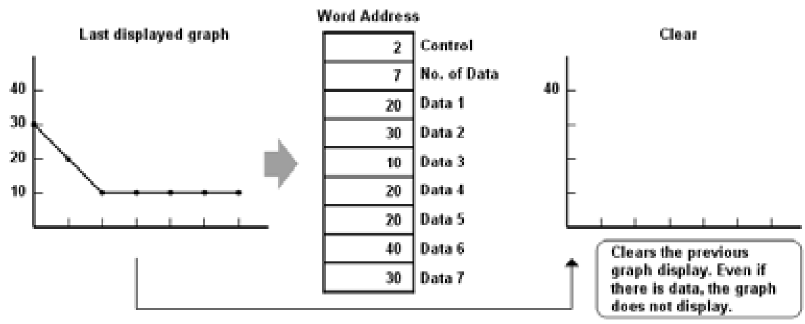


Note
Do not change the content of (Control address), (No. of data address) and (Data storage start address) between step 3 and step 4 above as doing so might cause error for the trend curve plot.

Example 3

▪ How to clear the graph

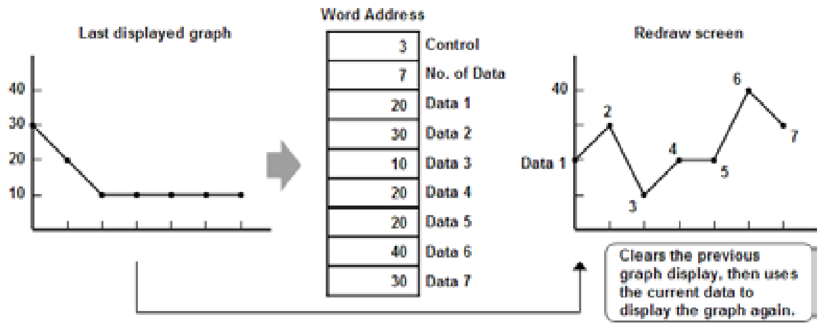
1. Write "2" to (Control address), all the trend curves will be cleared.
2. The system will write "0" to (Control address) after the trend curve is cleared.



Example 4

▪ How to clear the previous trend curve and display new one

1. Write the number of data to (No. of data address), i.e. "control address+1".
2. Store the data consecutively beginning at (Data storage start address).
3. Write "3" to (Control address), the previous trend curves will be cleared and the new content in data block will be plotted on the screen.
4. The system will write "0" to (Control address) after the trend curve has been plotted.



Example 5

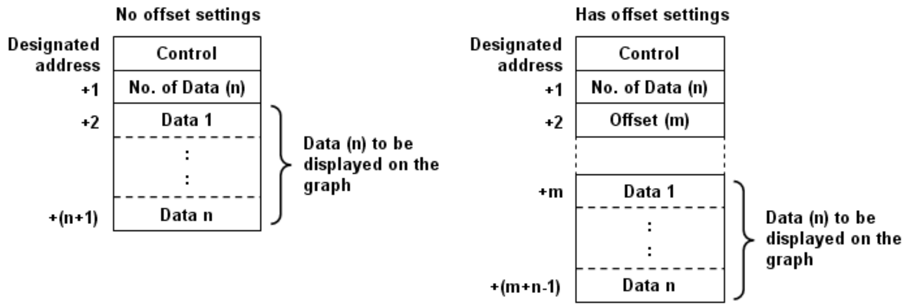
▪ How to use offset mode

If (Offset to start address) is selected, (Control address), (No. of data address), and (Offset value storage address) will use 3 consecutive addresses.

For example, assume the total number of channels is 3 (start from 0 to 2), and the (Control address) are LW-0, LW-100, and LW-200, respectively. Then, the other addresses will be set as follows: (In the example, format 16-bit Unsigned is used and (Offset value storage address) are all m).

Item	Channel 0	Channel 1	Channel 2
Control address	LW-0	LW-100	LW-200
No. of data address	LW-1	LW-101	LW-201
Offset value storage address	LW-2 (=m)	LW-102 (=m)	LW-202 (=m)
Data 1	LW-0+m	LW-100+m	LW-200+m
Data 2	LW-1+m	LW-101+m	LW-201+m
...	...	...	...

The following figure on the left shows the result when offset mode is not used while the figure on the right shows the result when offset mode is used.

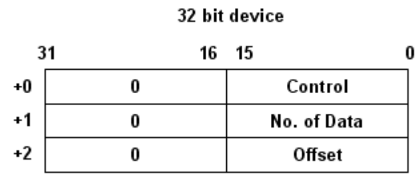


Note

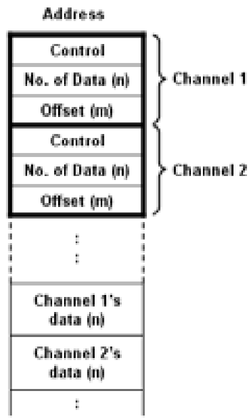
- When (Control address) is set to LW-n, (No. of data address) and (Offset value storage address) are as follows:

Data type	16-bit	32-bit
Control address	LW-n	LW-n
No. of data address	LW-n+1	LW-n+2
Offset value storage address	LW-n+2	LW-n+4

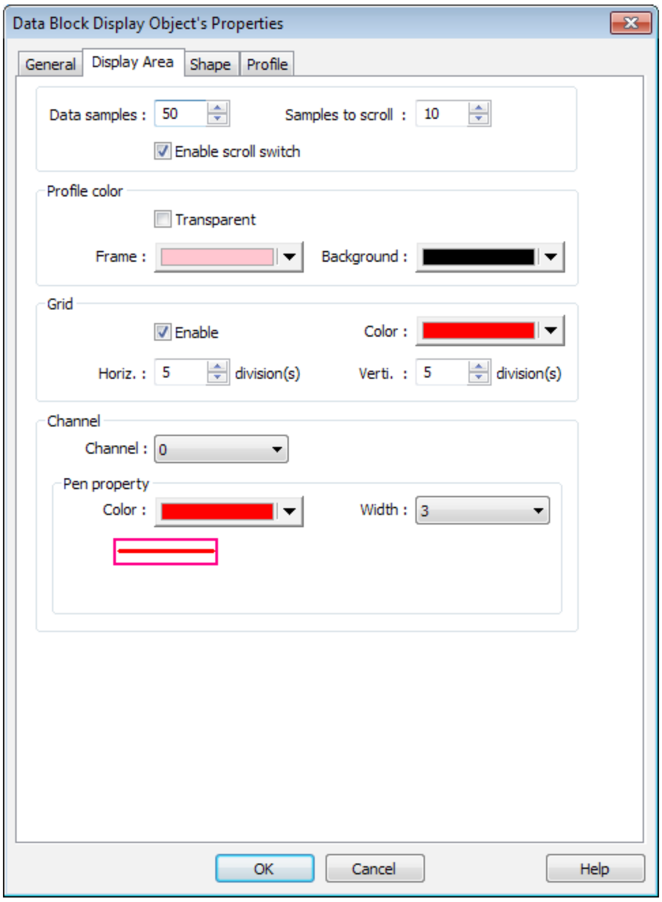
- If the control registers are 32-bit devices, only bit 0-15 will be used for control purpose, bit 16-31 will be ignored. (as illustrated below)

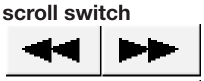


- When the value in (Control address) is not zero, the system will read (No. on data address) and (Offset value storage address).
- It is recommended to use (Offset to start address) for data block display with multiple channels and the same device type. As shown in the following figure, The control words of channel 1 is located from LW-n, the control words of channel 2 is located from LW-n+3, and so on.



13.19.2.2. Display Area



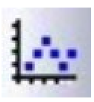
Setting	Description
Description	**Data samples** Configure the maximal number of data samples (points) to be displayed. **Samples to scroll** Configure the number of data samples being scrolled. **Enable scroll switch** Clicking  displays the previous or next data point. Clicking  displays the first or the last data point.
Profile	Set the color of the frame and background of the object. **Transparent** Hides the background. Color selection will not be available.
Grid	Set the number of horizontal and vertical divisions shown by grid.
Channel	Set the color, width and style of each curve.

13.20. XY PLOT

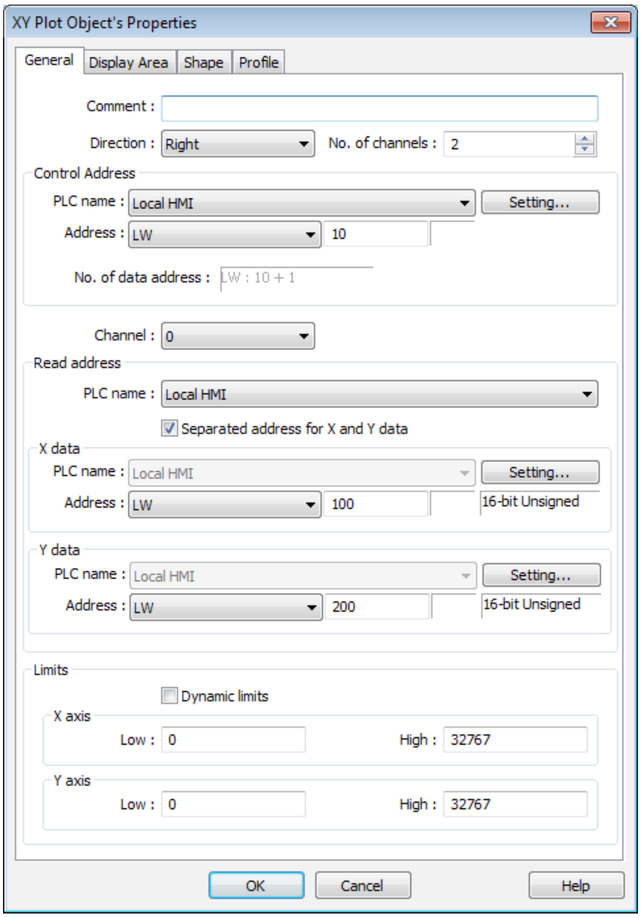
13.20.1. Overview

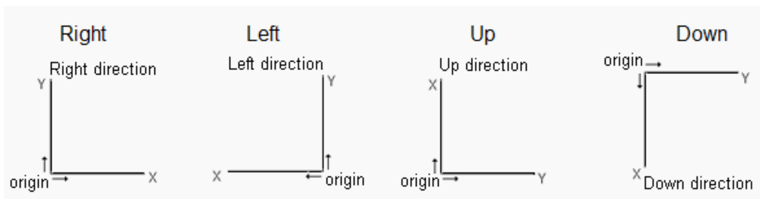
XY Plot object is used to display values for two variables (x, y) for a set of data, where the data comes from word registers. Up to 32 channels can be displayed simultaneously. This object facilitates data observation and analysis. Additionally, negative numbers can be displayed as well.

13.20.2. Configuration

	Click the XY Plot icon on the toolbar to open a (XY Plot) object property dialog box.

13.20.2.1. General Tab



<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Direction</td><td>There are four selections, right, left, up or down.

</td></tr>
<tr><td>No. of channels</td><td>Set the number of channels for observation.</td></tr>
<tr><td>Control address</td><td>Controls the operation of all channels simultaneously. When the (Control address) is LW-n, assigning values to LW-n will issue commands to XY plot according to the table below. Meanwhile, LW-n+1 I controls the number of data points plotted. After operation, the (Control address) will be reset to 0.

<table>
<tr><th>Control address</th><th>Value</th><th>Result</th></tr>
<tr><td rowspan="3">LW-n</td><td>1</td><td>Plots point on XY curve.
(The plotted points are kept.)</td></tr>
<tr><td>2</td><td>Clears all XY curves.</td></tr>
<tr><td>3</td><td>Clears then plots new XY curve.</td></tr>
<tr><td>LW-n+1</td><td>Any number</td><td>Controls the number of data points plotted.</td></tr>
</table>

No. of data address
Controls the number of data points. Each channel can plot up to 1,023 points.</td></tr>
<tr><td>Channel</td><td>Select a channel to configure.</td></tr>
<tr><td>Read address</td><td>PLC name
Select a PLC which will be the source of (X data) and (Y data) and designate a read address.
The format of the data register blocks used for the display channels depends on whether (Separated address for X and Y data) and/or (Dynamic limits) has been selected. See Example 1.</td></tr>
<tr><td>Dynamic limits</td><td>■ When not selected (See Example 2)
The Low and High limits can be set by entering constants. The Low and High limits are used for calculating X and Y range in percentage.
■ When selected (See Example 3)
A zoom effect can be created by changing the Low / High Limits.</td></tr>
</table>

Example 1

The format of the data register blocks used for the display channels depends on whether (Separated address for X and Y data) has been selected, and if (Dynamic limits) has been selected. The following explains the situations where 16-bit register is used:

- If (Separated address for X and Y data) is not selected, and set (Read address) to LW-n:

	Select (dynamic limits)		Not select (dynamic limits)	
	X data	Y data	X data	Y data
Low limit	LW-n	LW-n+2	Constant	Constant
High limit	LW-n+1	LW-n+3	Constant	Constant
1st data	LW-n+4	LW-n+5	LW-n+0	LW-n+1
2nd data	LW-n+6	LW-n+7	LW-n+2	LW-n+3
3rd data	LW-n+8	LW-n+9	LW-n+4	LW-n+5
4th data	LW-n+10	LW-n+11	LW-n+6	LW-n+7

- If (Separated address for X and Y data) is selected, and set (X data) to LW-m, (Y data) to LW-n:

	Select (dynamic limits)		Not select (dynamic limits)	
	X data	Y data	X data	Y data
Low limit	LW-m+0	LW-n+0	Constant	Constant
High limit	LW-m+1	LW-n+1	Constant	Constant
1st data	LW-m+2	LW-n+2	LW-m+0	LW-n+0
2nd data	LW-m+3	LW-n+3	LW-m+1	LW-n+1
3rd data	LW-m+4	LW-n+4	LW-m+2	LW-n+2
4th data	LW-m+5	LW-n+5	LW-m+3	LW-n+3

Example 2

When (Dynamic limits) is not selected, the Low and High limits can be set. The Low and High limits are used for calculating X and Y range in percentage.

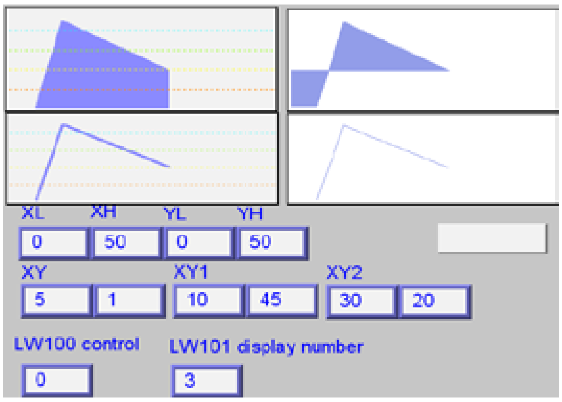
$$\text{Scale (\%)} = \frac{\text{Read Address Value} - \text{Low Limit}}{\text{High Limit} - \text{Low Lmit}}$$

If (Separated address for X and Y data) is not selected and the address is LW-n, the corresponding limits are retrieved from the addresses as shown in the following table.

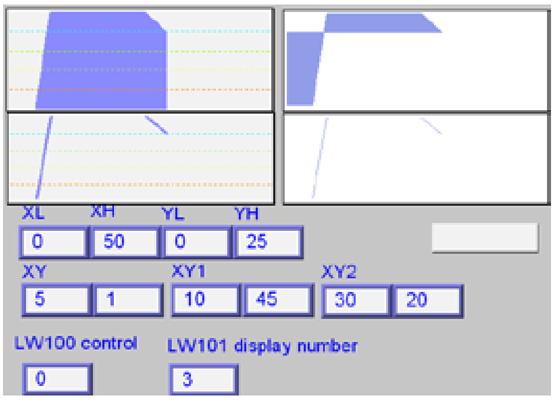
Data format	16-bit	32-bit
X axis low limit	LW-n	LW-n
X axis high limit	LW-n+1	LW-n+2
Y axis low limit	LW-n+2	LW-n+4
Y axis high limit	LW-n+3	LW-n+6

Example 3

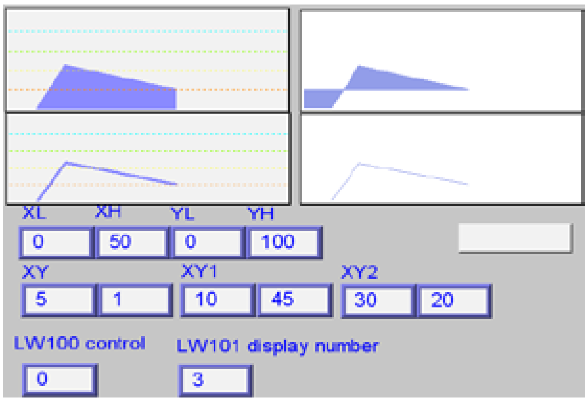
If (Dynamic limits) is selected, a zoom effect can be created by changing the setting of Low / High Limits.
In the following example, XL=X low limit, XH=X high limit, YL=Y low limit, YH=Y high limit, and XY, XY1, XY2 are three XY data. When changing the high limits of X and Y axis, the result is shown below:



Original



Change the high limit of Y axis to 25. (zoom in)

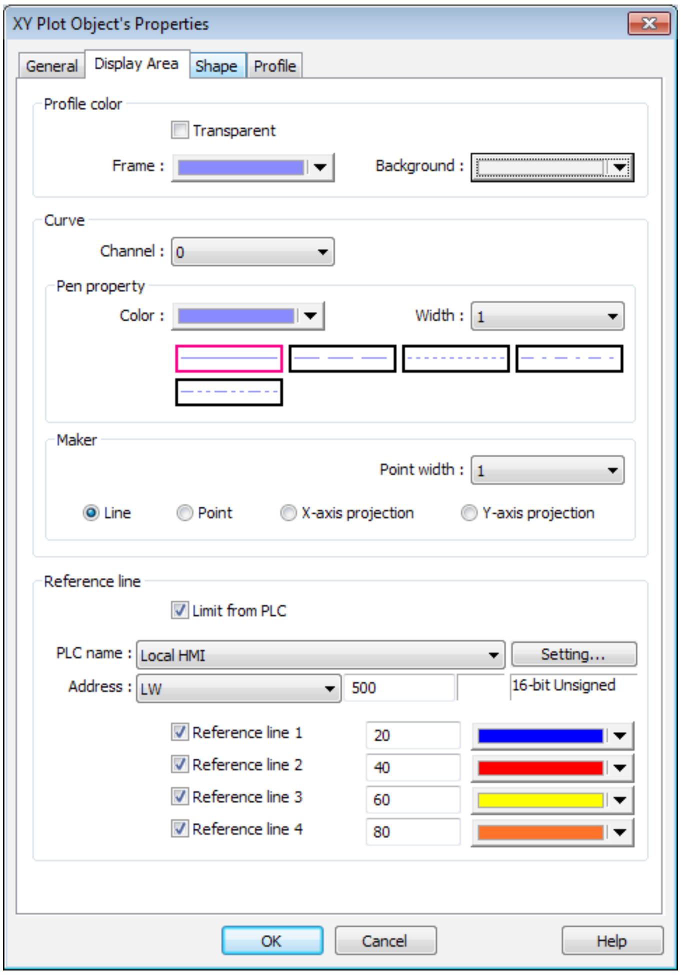


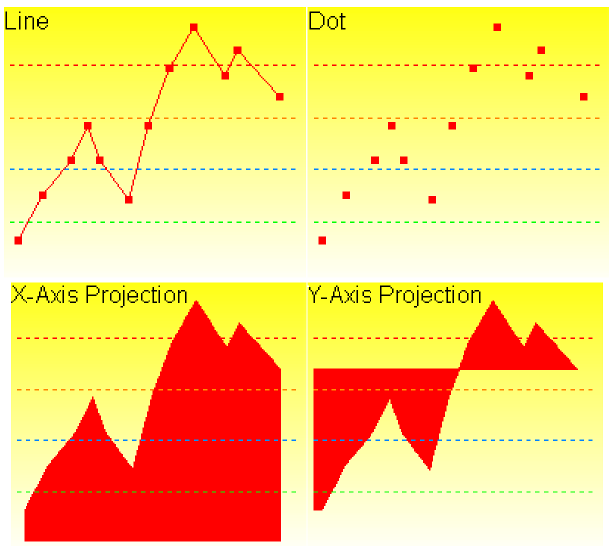
Change the high limit of Y axis to 100 (zoom out)

For more information, see "13.17 Trend Display".

	Note ▪ For cMT Series, on the screen of the visualization device directly pinch two fingers together to zoom out or spread them apart to zoom in. ▪ X and Y data can be set to different formats. For example: If X data uses 16-bit unsigned, Y data uses 32-bit signed, please note the address setting. ▪ When using a Tag PLC, such as AB tag PLC, X and Y must be in the same format. When using different formats a warning will be shown.

13.20.2.2. Display Area Tab

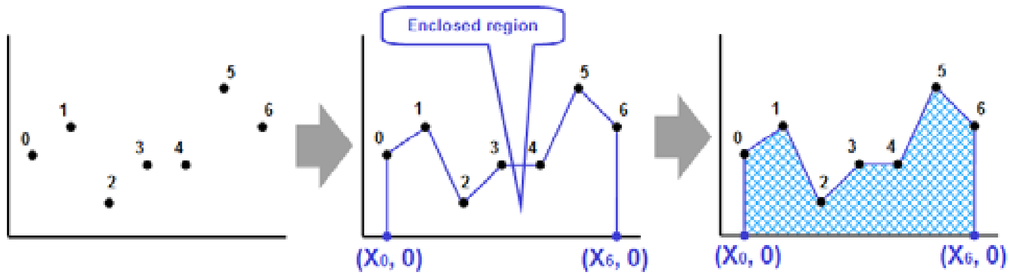


Setting	Description
Profile color	Select the color of the frame and the background, or select (Transparent) check box to hide the frame and background.
Curve	For each channel select the properties of color, width, and line style.
Maker	There are four different types of XY plot. The result is shown below:  See Example 4.
Reference line	Up to 4 horizontal reference lines can be shown on the graph. Fill in high, low limits and Y axis percentage values. Different colors can be selected for each reference line. If (Limit from PLC) is selected, designate a register to be the read address of reference line.

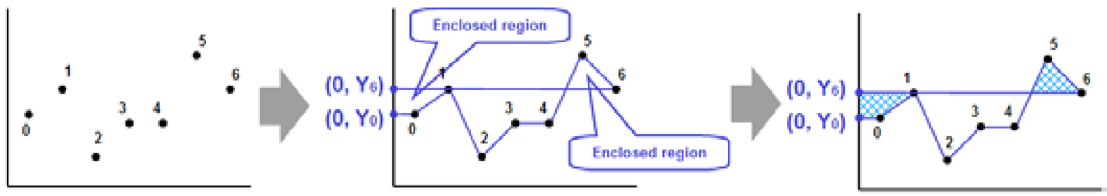
Example 4

The curve shown in the following figure is drawn with 7 points numbered from P0 to P6.
The steps the system draws the X-axis Projection are:

1. Calculates the two points in X-axis – (X0, 0) and (X6, 0).
2. Link all the points in the order of (X0, 0), P0… P6, (X6, 0) and returns to (X0, 0) at last.
3. Fill out all enclosed areas.



Similarly for Y-axis projection:



	Note
	▪ XY Plot can be drawn repeatedly up to 32 times: ▪ 1 channel → 32 times ▪ 2 channels → 16 times ▪ The way to calculate: 32 divided by the number of channels.

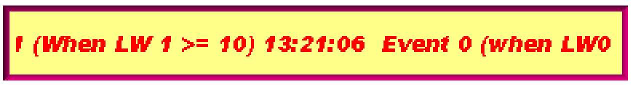
13.21. ALARM BAR AND ALARM DISPLAY

13.21.1. Overview

Alarm Bar and Alarm Display objects are used to display alarm messages which are defined in Event (Alarm) Log objects. When the trigger conditions are met, events or alarms will be displayed as they occur in chronological order in Alarm Bar or Alarm Display object.

Alarm Bar scrolls all alarm messages in one single display line, whereas Alarm Display shows alarm messages in multiple lines.

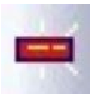
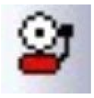
For more information, see “7 Event Log”.



Alarm Bar - Displays alarm messages in one scrolling line.

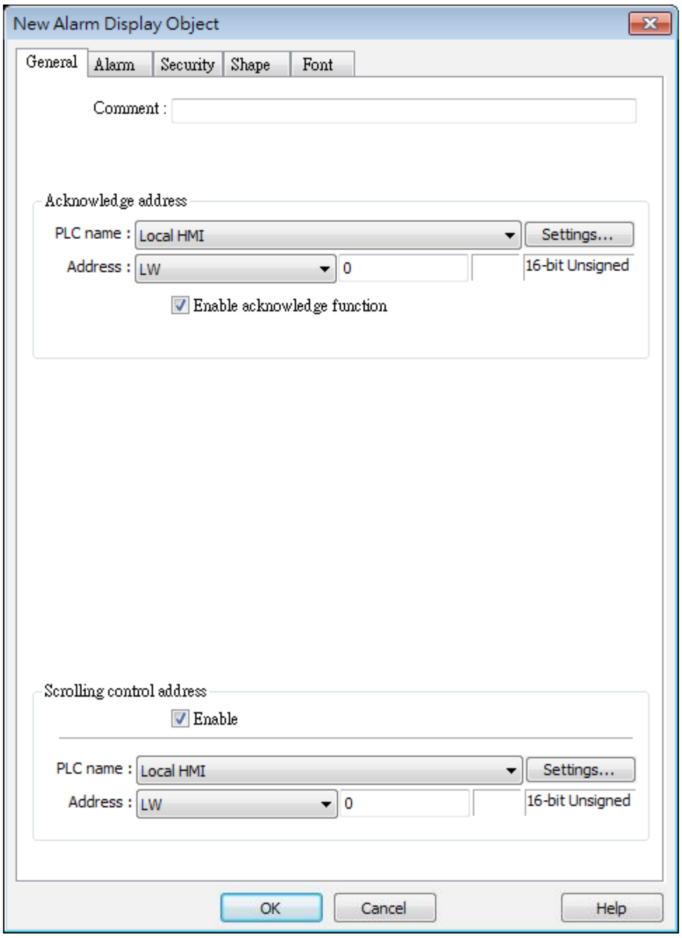
Alarm Display – Displays alarm messages in multiple lines.

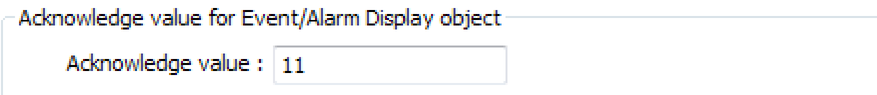
13.21.2. Configuration

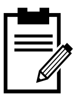
		Click the Alarm Display or Alarm Bar icon on the toolbar to open the object property dialog box. Set up the properties, press OK button, and a new object will be created.

13.21.2.1. General Tab

The difference between these two objects is that Alarm Display allows an (Acknowledge address) and a (Scrolling control address) to be set.

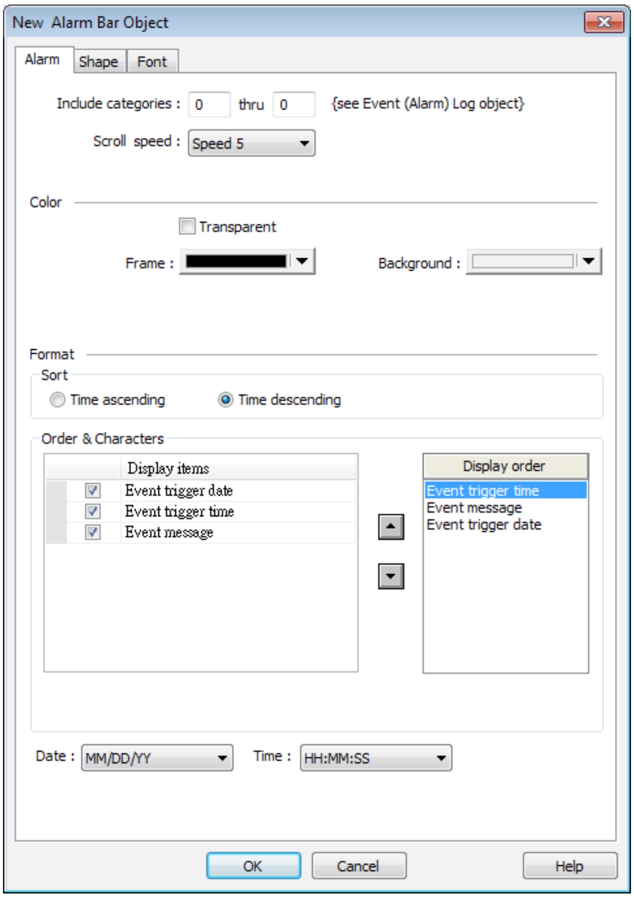


Setting	Description
Enable acknowledge function	If selected, the (Acknowledge value) selected for the associated event, specified in Alarm (Event) Log » Message tab will be written to the (Acknowledge address) designated in Alarm Display. For more information, see “7 Event Log”. 
Scrolling control address	If enabled, the value in the designated control address indicates the number of lines to be scrolled down. The minimal value is 0, which indicates the first line.

	**Note** When using cMT-SVR, press and hold the event on the screen to acknowledge an event; drag a finger on the screen to scroll.

The following are general settings of these two objects:

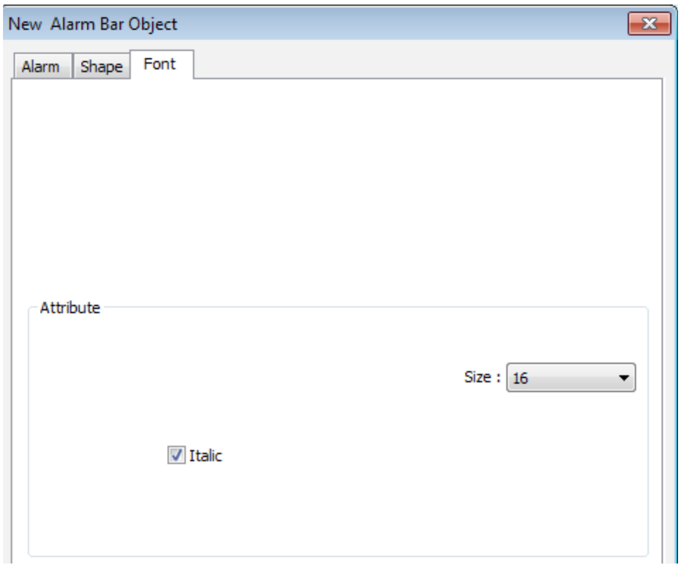
13.21.2.2. Alarm Tab



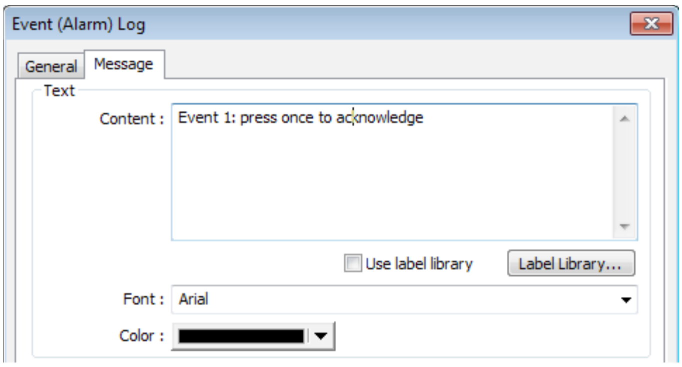
Setting	Description
Include categories	Events in the selected category will be displayed. The categories are set in Event (Alarm) Log object. For example, if the category is set to "2 to 4" here, only events in categories 2, 3, 4 will be displayed. For more information, see "7 Event Log".
Scroll speed	This selection is only available for Alarm Bar. Select one of the speed settings at which the messages scroll.
Format	**Time ascending** Latest alarm is placed last in the list (the bottom). **Time descending** Latest alarm is placed first in the list (the top). **Display order** Select the items to be displayed and use the up and down arrow buttons to adjust the display order of the alarms. **Date** Displays the date tag with each alarm message. The four formats of date tag: MM/DD/YY / DD/MM/YY / DD/MM/YY / YY/MM/DD **Time** Displays the time tag with each alarm message. The four formats of time tag: HH:MM:SS / HH:MM / DD:HH:MM / HH

13.21.2.3. Font Tab

Set the font size or select (Italic).



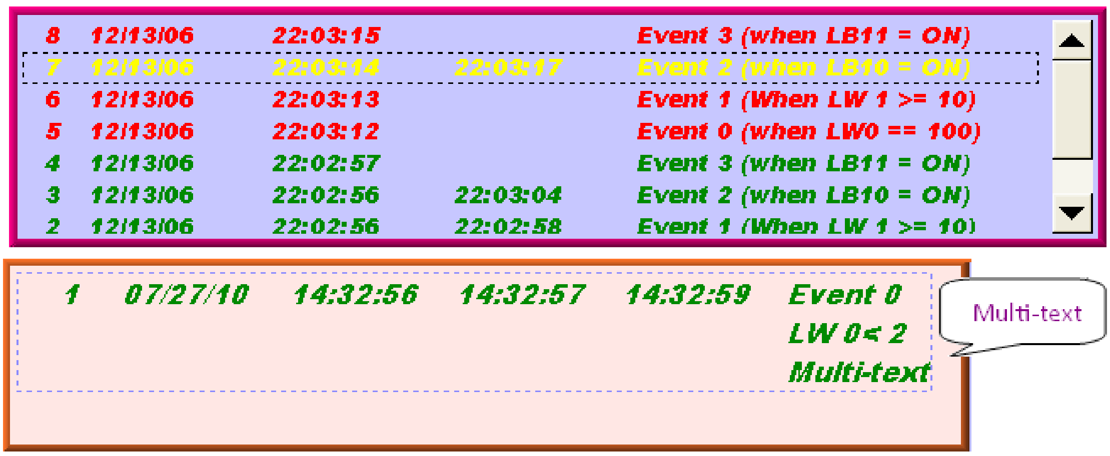
The font, color, and content of the alarm messages displayed in Alarm Bar and Alarm Display objects are set in Alarm (Event) Log object:



13.22. EVENT DISPLAY

13.22.1. Overview

Event Display object is used to display event messages which are defined in Event (Alarm) Log and have met a trigger condition. The triggered events are displayed in the chronological order. Event Display object displays not only the date and time the event occurs, but also the time the event is acknowledged, the time the event returns to normal, and the event message. Multi-lined messages can also be displayed.

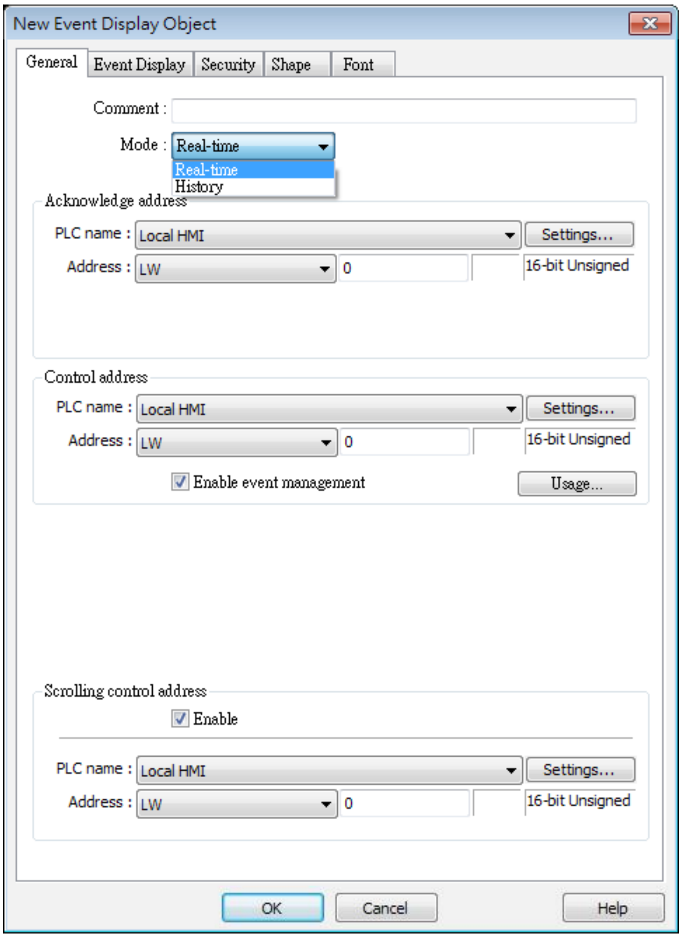


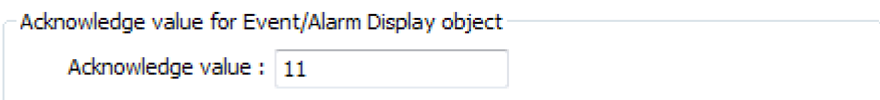
13.22.2. Configuration

	Click the Event Display icon on the toolbar to open an Event Display object property dialog box. Set up the properties, press OK button, and a new Event Display object will be created.

13.22.2.1. General Tab

- eMT, iE, XE, mTV Series



Setting	Description
Mode	The available modes are: (Real-time) and (History). ▪ Real-time: All the events triggered since HMI starts up are displayed. ▪ History: The system reads the event log in HMI memory and displays them. The content can be updated by changing window. In case when the trend display shows history data from today, the display will refresh once per second.
Acknowledge address	When in Real-time mode, and an event is acknowledged by touching an active display line, the (Acknowledge value) specified in Event (Alarm) Log object, Message tab, is output to the (Acknowledge address) of Event Display object. For more information, see "7 Event Log". 

History Control

When in History mode, and if:

- (Enable reading multiple histories) is not selected

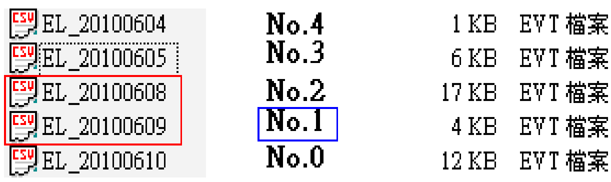
  Daily event log files can be displayed. A history control address can be designated. The value in the designated register is used as an index to select historical files.
  Index value 0 displays the latest file.
  Index value 1 displays the second latest file.
  Index value 2 displays the third latest file, and so on.
  If control address to LW-100, and four data log exist with dates: EL_20100720.evt, EL_20100723.evt, EL_20100727.evt, and EL_20100803.evt.
  Each index value in the control word corresponds to a record according to the table below:

Value in LW-100	The corresponding record
0	EL_20100803.evt
1	EL_20100727.evt
2	EL_20100723.evt
3	EL_20100720.evt

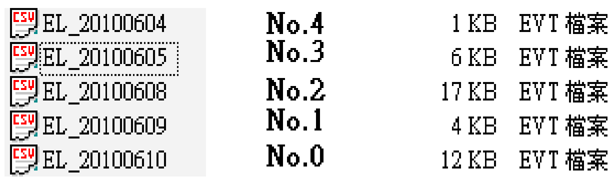
- (Enable reading multiple histories) is selected

  Displays a list of events triggered in multiple days. If (History control) address is set to LW-n, the LW-n to LW-n+1 form a range of log selection.

Number of days
The data range starts from the number in LW-n. The value in LW-n+1 represents how many days to be included from the start to days before.
Example: as illustrated below, if LW-n = 1, and LW-n+1 = 3, then the range of data will start from 20100609, and include the data of 2 days before (20100609 included). However, since the data of 20100607 does not exist in this example, the data displayed will only include 20100609 and 20100608.



Index of the last history
The range of data will start from the number in LW-n and end in LW-n+1. If LW-n = 1, and LW-n+1 = 3, the data displayed will include data No.1, No.2, No.3. If the number entered in LW-n+1 is greater than the number of event log files, LW-n+1 will not be effective and only the data specified by LW-n is displayed.

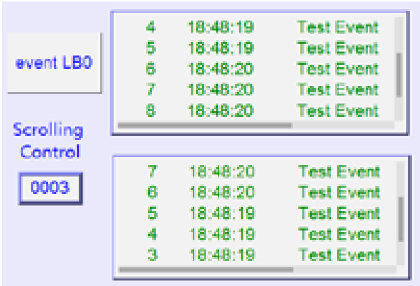


The maximum size of data that can be displayed is 4MB; the exceeding part will be ignored.
The following shows how data will be stored when the data size is too big.
5 history data, each 0.5MB → Data displayed: 8 x 0.5MB
5 history data, each 1MB → Data displayed: 4 x 1MB
5 history data, each 1.5MB → Data displayed:
2 x 1.5MB+1 x 1MB (partial)

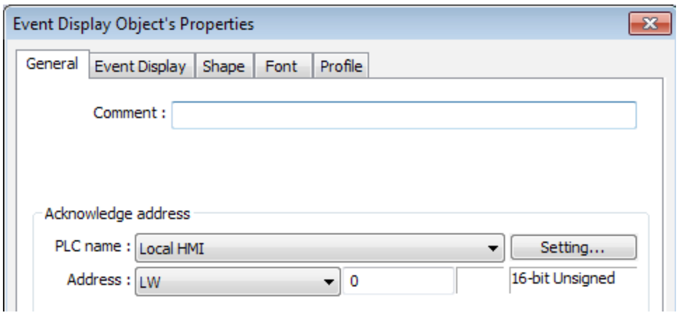
Control address

Enable event management
If this check box is selected, writing a specific value into register LW-n and LW-n+1, where n is an arbitrary number, will control (Event Display) object with different commands as shown below:

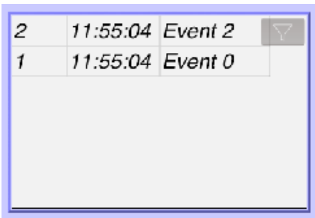
Address	Value	Command
LW-n	0	Display all events.
	1	Hide (Confirmed) events.
	2	Hide (Recovered) events.
	3	Hide (Confirmed) or (Recovered) events.
	4	Hide (Confirmed) and (Recovered) events.
LW-n+1	1	Delete a single selected event.

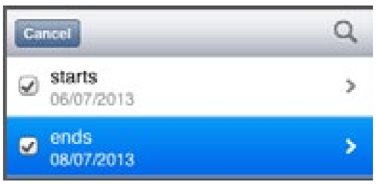
Scrolling control address	If enabled, the value in the designated control address indicates the number of lines to be scrolled down. The minimal value is 0, which indicates the first line. In the following figure, there are 10 events recorded in the object, and value 3 is set at the control address. The upper event object displays the events in time ascending order, and begins at the 4th event; on the other hand, the lower one displays the events in time descending order, and begins at the 7th event.  If (Scrolling control address) is enabled, the scroll bar cannot be used for scrolling, but still shows the relative position of the content. If the control address holds a value that is larger than the total number of lines, the display will stroll to the end.

13.22.2.2. cMT-SVR Series



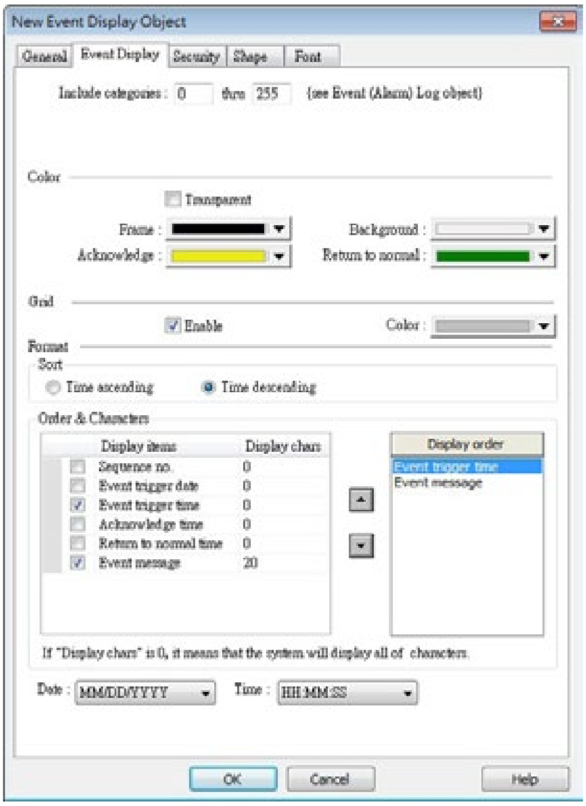
For cMT-SVR, all the events occur are displayed and updated in real-time.
Press the filter icon in the upper-right corner of the object and set the start and end date. If the dates are not set, all the events are displayed.



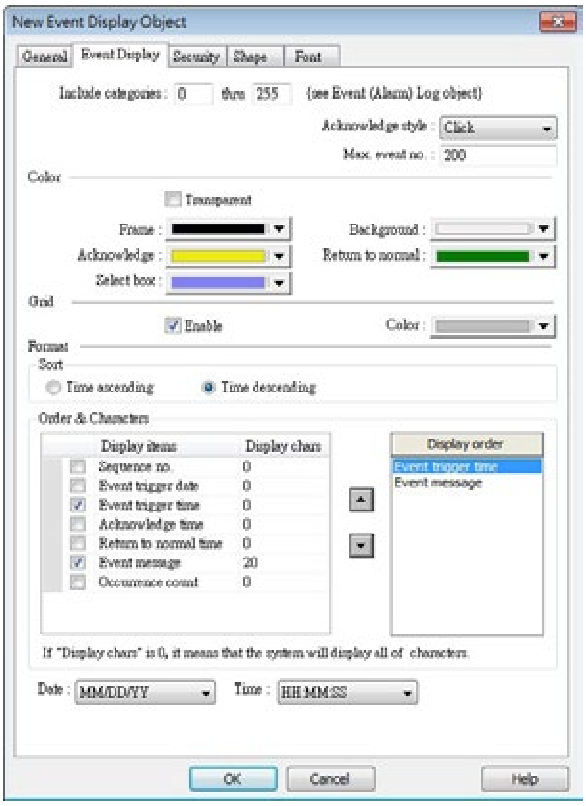


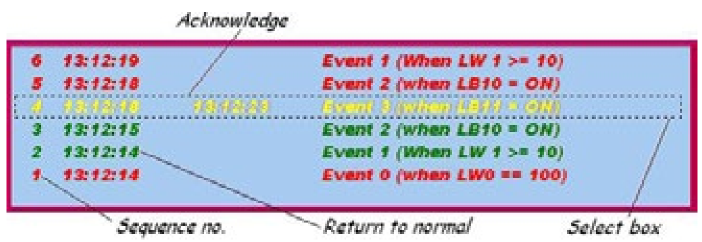
13.22.2.3. Event Display Tab

▪ cMT Series



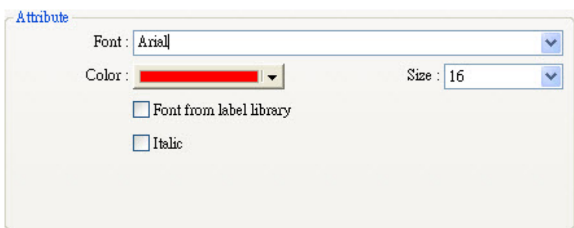
▪ eMT, iE, XE, mTV Series



Setting	Description
Include categories	Events in the selected category will be displayed. The categories are set in Event (Alarm) Log object. For example, if the category is set to 2 to 4 here, only events in categories 2, 3, 4 will be displayed. For more information, see “7 Event Log”.
Acknowledge style	Select (Click) or (Double Click) to acknowledge each single event. When an event occurs the user can tap the event line once or twice to acknowledge the new event. When acknowledged, the text color of the event will change to the selected color, and the acknowledge value associated with that event will be sent to the register designated in (Acknowledge address). If the address is set to LW-100, and the acknowledge value is set to 31, when user acknowledges the event, value 31 is written to LW-100. This can be used in conjunction with Indirect Window object so that when an event is acknowledged, the corresponding message window is displayed.
Max. event no.	The maximum number of events to be displayed in this Event Display object. When the number of the displayed events equals to the set maximum number, the new coming event will overwrite the latest event.
Color	Different colors indicate different event states, such as acknowledged, returns to normal, or selected. The system draws a highlight box around the latest selected event. 
Grid	Displays a grid of rows and columns in the object. The color of the grid lines can be selected.
Format	 **Time ascending** Latest event is placed last in the list (the bottom). **Time descending** Latest event is placed first in the list (the top). **Order & Characters** Select the items to be displayed and use the up and down arrow buttons to adjust the display order of the events. **Date** Displays the date tag with each event message. The four formats of date tag: MM/DD/YY / DD/MM/YY / DD/MM/YY / YY/MM/DD **Time** Displays the time tag with each event message. The four formats of time tag: HH:MM:SS / HH:MM / DD:HH:MM / HH

13.22.2.4. Font Tab

In Real-time mode: users may select Italic font and set the font size. The font is displayed according to the setting in Event Log object.
In History mode: users may select Italic font and set the font size, font and color, or tick the (Font from label library) check box.



13.23. DATA TRANSFER (TRIGGER-BASED)

13.23.1. Overview

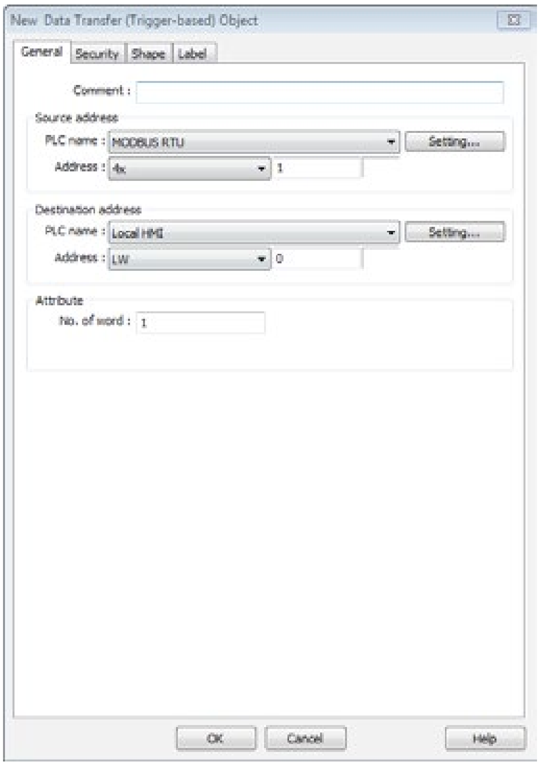
Data Transfer (Trigger-based) object can transfer values from the source register to the destination register. The data transfer operation can be activated by changing the state of the designated bit register, or by manually pressing the object.
For cMT Series, only touch trigger mode is available.

13.23.2. Configuration

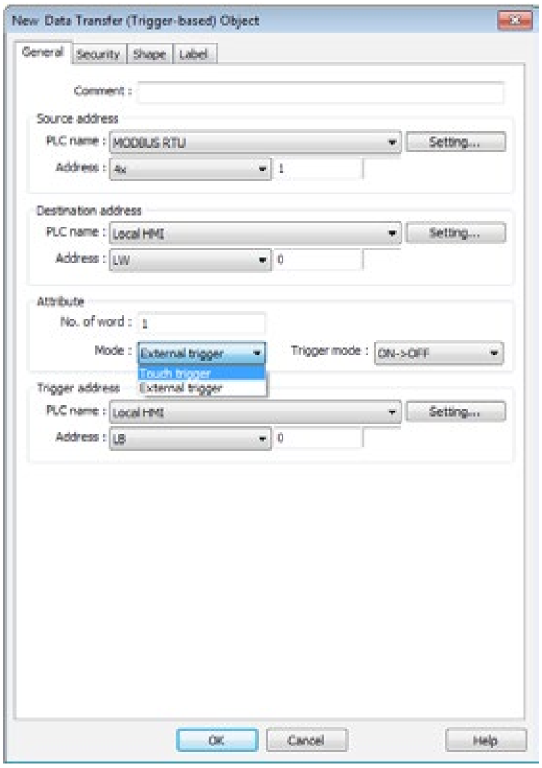
	Click the Data Transfer (Trigger-based) icon on the toolbar to open the property dialog box. Set up the properties, press OK button, and a new Data Transfer (Trigger-based) object will be created.

13.23.2.1. General Tab

- cMT Series



- eMT, iE, XE, mTV Series



Setting	Description
Source address	Data transfer object reads the data from (source address).
Destination address	Data transfer object writes the data to (destination address).
Attribute	**No. of words** The number of words to be transferred from (source address) to (destination address). The unit is word. **Mode** **Touch trigger** Press the object to activate data transfer operation. **External trigger** The data transfer operation is activated when the state of the designated bit address changes. There is a further selection to make of whether the data transfer operation is activated after Off to ON, ON to OFF transition, or at both of the changes of state.
Trigger address	Specify a bit address for (External trigger) mode.

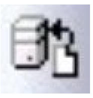
Note
When using Data Transfer Trigger Based object, place the control bit addresses in the same window in order to trigger Data Transfer. If the Data Transfer Trigger Based object is placed in the common window, when the state of the control bit addresses placed in any window changes, Data Transfer is triggered.

13.24. BACKUP

13.24.1. Overview

Backup object can transmit recipe data (RW, RW_A), event log, recipe database, sampling data, and operation log to an external device (SD card, USB disk), in a specified time range or format. For example, when the event log is saved in a SD card, a USB disk can be inserted when HMI power is still ON, and use Backup object to copy the data into USB disk from SD card, and then remove USB disk without turning off HMI power. The data saved in USB disk can be used on PC for analyzing. When the system is backing up, the state of system register (LB-9039) is set ON. With (e-Mail) option, information can be sent to configured email contacts.

13.24.2. Configuration

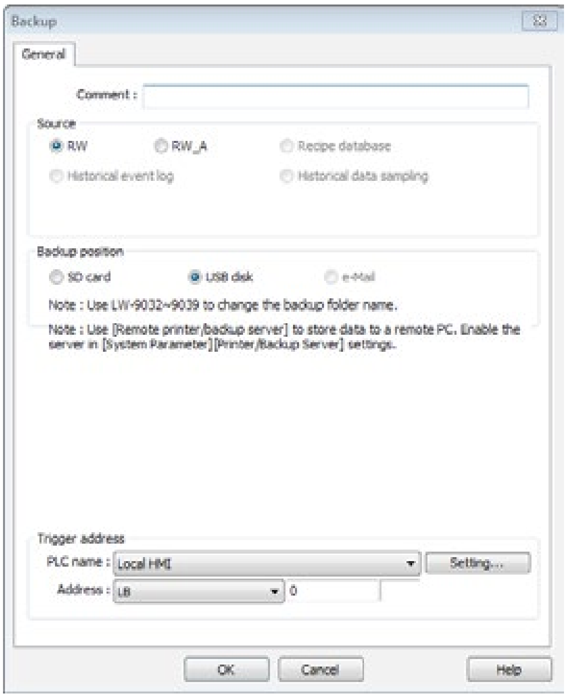


For eMT, iE, XE, mTV Series, click Backup icon on the toolbar to open a (Backup) object property dialog box. Set up the properties, press OK button, and a new Backup object will be created.

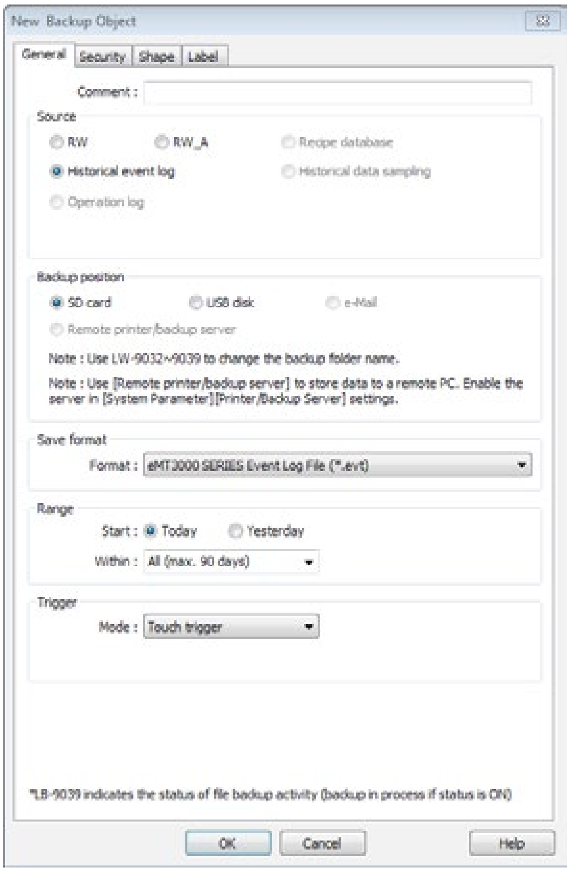
For cMT Series, click Backup icon on the toolbar to open a (Backup) object managing dialog box, click (New) to open a (Backup) object property dialog box. Set up the properties, press OK button, and a new Backup object will be created.

13.24.2.1. General Tab

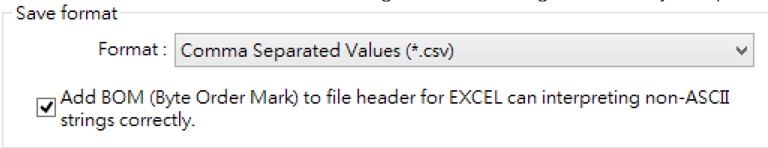
- cMT Series

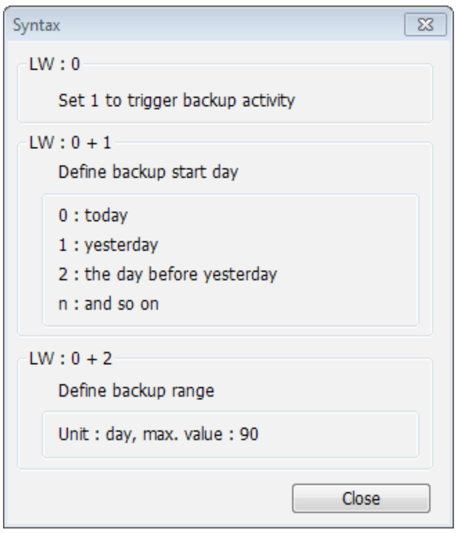


▪ eMT, iE, XE, mTV Series



Setting	Description
Source	**(RW), (RW_A), (Recipe database), (Historical event log), (Historical data sampling), (Operation log)** Select one from the above for the source. When backing up (Historical data log), use (Data Sampling object index) to select the one to back up.
Backup position	Select the destination where the source files will be saved to. **SD card / USB disk** The external device connected to HMI. If using cMT series, SD card and USB disk can only save (RW), (RW_A), and (Recipe database). **Remote printer/backup server** (eMT, iE, XE, mTV Series) To select this, enable MT remote printer/backup server at: (Menu) » (Edit) » (System Parameters) » (Printer/Backup Server). Please note that (Operation log) backup can only be saved to Remote printer/backup server. To save into a SD card or USB drive, please use the control address of Operation Log object. For more information, see “26 EasyPrinter”. **E-mail** To use e-mail, go to (System Parameters) » (e-Mail) tab to configure first. And then go to Backup object » (e-Mail) tab to configure the recipient address, subject, and message.

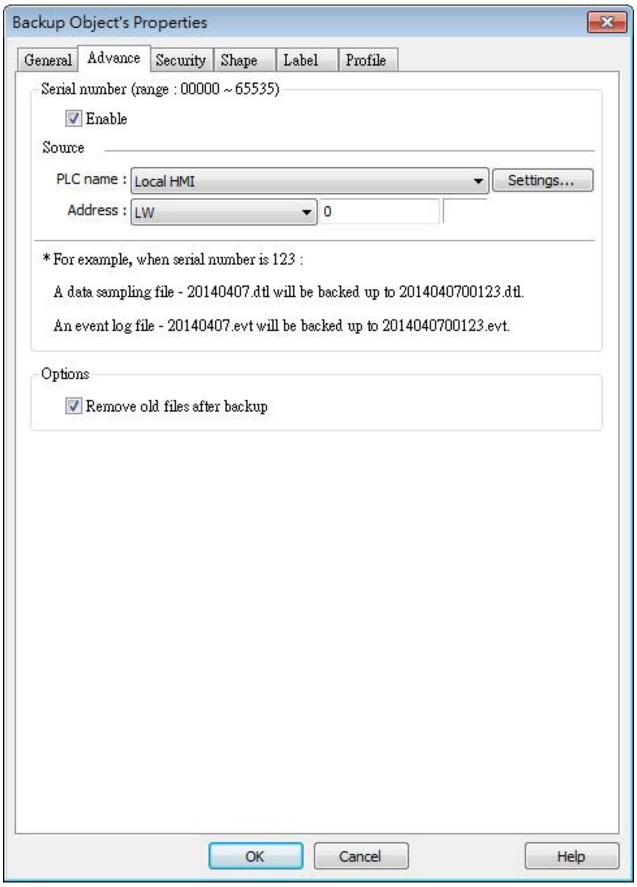
<table>
<tr><td>Save format</td><td>Select the desired format to back up the file.
eMT, iE, XE, mTV Series:
▪ HMI Event Log File (.evt) / HMI Data Log File (.dtl)
▪ Comma Separated Values (.csv)
When back up event log in .csv format, open the csv file in Excel. The BOM (Byte Order Mark) can be added to the file header so that the .csv file containg non-ASCII strings can directly be opened in Excel.


The (Event) column is included in the backup file to indicate the type of the event.
<table>
<tr><th></th><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th></tr>
<tr><td>1</td><td>Event</td><td>Category</td><td>Date</td><td>Time</td><td>Message</td></tr>
<tr><td>2</td><td>0</td><td>1</td><td>2013/7/4</td><td>16:12:11</td><td>Event A</td></tr>
<tr><td>3</td><td>2</td><td>1</td><td>2013/7/4</td><td>16:12:12</td><td>Event A</td></tr>
<tr><td>4</td><td>0</td><td>0</td><td>2013/7/4</td><td>16:12:33</td><td>Event B</td></tr>
<tr><td>5</td><td>2</td><td>0</td><td>2013/7/4</td><td>16:12:36</td><td>Event B</td></tr>
<tr><td>6</td><td>0</td><td>0</td><td>2013/7/4</td><td>16:12:37</td><td>Event B</td></tr>
<tr><td>7</td><td>1</td><td>0</td><td>2013/7/4</td><td>16:12:37</td><td>Event B</td></tr>
<tr><td>8</td><td>2</td><td>0</td><td>2013/7/4</td><td>16:12:39</td><td>Event B</td></tr>
<tr><td>9</td><td>0</td><td>0</td><td>2013/7/4</td><td>16:12:40</td><td>Event B</td></tr>
</table>
0 = Event is triggered
1 = Event is acknowledged
2 = Event returns to normal
EasyConverter can be used to easily convert HMI Event Log File (.evt) and HMI Data Log File (.dtl) to .xls or .csv format.
▪ SQLite Database File (.db)
cMT Series:
▪ SQLite Database File (.db)
▪ Comma Separated Values (.csv)</td></tr>
<tr><td>Range</td><td>Within
Select the number of days. For example, (Yesterday) is selected at (Start), and (2 day(s)) is selected here, which means the files obtained yesterday and the day before yesterday will be backed up. Select (All) to save all files in the system, the maximum is 90 days.</td></tr>
<tr><td>Trigger
(eMT, iE, XE, mTV Series)</td><td>Mode
There are three ways to activate Backup function.
Touch trigger
Touch the object to activate backup operation.
External trigger (bit)
Register a bit device to trigger the backup operation.
Select whether the backup operation is activated after Off to ON, ON to OFF transition, or at both of the changes of state.
External trigger (word)
Users can specify the number of days to backup data using (Trigger address). (Trigger address) usage (suppose LW-n is used):
LW-n: will start to back up when the value changes from 0 to 1.
LW-n+1: the start date of backup.
LW-n+2: the number of days for backup. (The maximum: 90 days).

</td></tr>
<tr><td>Trigger address
(cMT Series)</td><td>When the state of the designated register is set ON, the backup operation is activated. When the backup operation is done, the state of the designated register is set OFF.</td></tr>
</table>

Note
- All history files should have been saved in memory, either HMI memory, USB disk or SD card. Otherwise, the Backup object will not work.
- The maximum number of days for backup is 90 days. (Not including cMT Series).

For cMT Series, see “7 Event Log” and “8 Data Sampling” that explain the mechanism of synchronizing data to external device.

13.24.2.2. Advance Tab



Setting	Description
Serial number	If enabled, when backing up history files, a user-defined, 5-digit serial number can be appended to the end of the file name of the history data backup. The serial number is determined by the value in the designated source address. After backup, the value of this LW address will automatically increment by 1. The range of the serial number is 0~65535 For example, if the serial number is 123, the appended 5 digits will be 00123. A data sampling file -20140407.dtl will be backed up as 2014040700123.dtl. An event log file -20140407.evt will be backed up as 2014040700123.evt.
Options	**Remove old files after backup** If selected, the old history files will be removed after backup.

Note
cMT Series does not support Advance settings.

13.25. MEDIA PLAYER

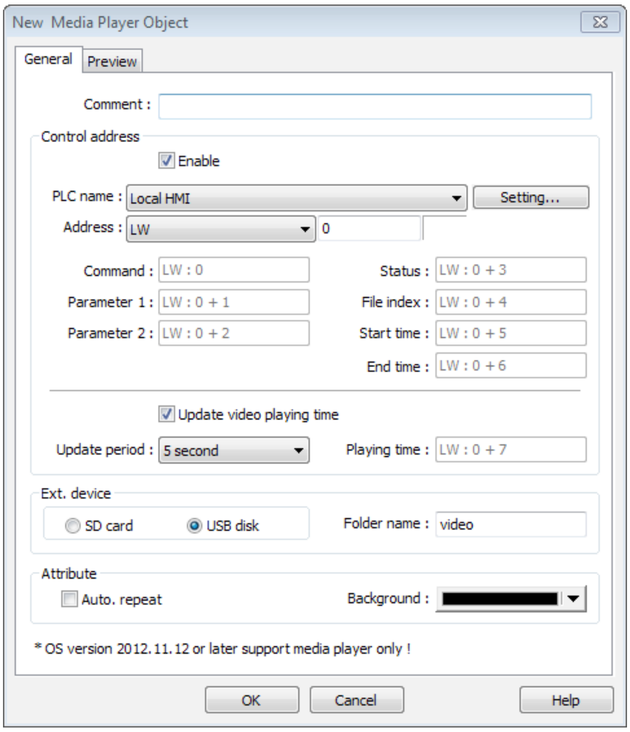
13.25.1. Overview

At the first time using Media Player object in the project, download the project to HMI via Ethernet. EasyBuilder Pro installs Media Player driver automatically.
Media Player object plays video files with controls such as seek, zoom, and volume adjustment to provide maintenance instructions or procedures on video so as to enable on-site operators to perform tasks efficiently.

13.25.2. Configuration

	Click the Media Player icon on the toolbar to open a Media Player object property dialog box. Set up the properties, press OK button, and a new Media Player object will be created.

13.25.2.1. General Tab



Setting	Description
Control address	▪ Selected Designate a word register to control the object operations. ▪ Not selected No manual control. Video will be played automatically when the designated window opens. **Command** (control address + 0) Enter a value in the Command register to designate which action is executed. **Parameter 1** (control address + 1) Enter a value in Parameter 1 associated with each command action. **Parameter 2** (control address + 2) Enter a value in Parameter 2 associated with each command action. **Status** (control address + 3) Indicates the status or errors. **File index** (control address + 4) The file number in the designated folder. It is recommended to file the video name with a number. **Start time** (control address + 5) The start time of the video (second). 0, normally. **End time** (control address + 6) The end time of the video (second). (The time length of the video) **Update video playing time** If enabled, the elapsed playing time of video will be written into (Playing time) register at a rate set by (Update period) in seconds. **Update period** Update period of (Playing time), range from 1 to 60 (second). **Playing time** (control address + 7) The elapsed playing time of video (Second). Normally between start time and end time.
Ext. device	Play video files in SD card / USB disk. **Folder name** The folder name of video files stored in SD card or USB disk. Files must be stored in root directory. Subdirectories won't be accepted. (For example, "example\ex" is an invalid directory.) (Folder name) cannot be empty, must be alpha-numeric, and all in ASCII character.
Attribute	**Auto. repeat** When finish playing all the video files, replay from the first file. Ex: video 1 > video 2 > video 1 > video 2 **Background** The background color of the object.

Note
The data format for control address is 16-bit Unsigned or 16-bit Signed. If using 32-bit Unsigned or 32-bit Signed, only the previous 16 bits will be effective.

13.25.2.2. Control command

The following are the settings of different commands.

- Play index file
  - [Command] = 1
  - [Parameter 1] = file index
  - [Parameter 2] = ignore (set 0)

Note
- Files are stored with file names in ascending order.
- If the file cannot be found, (Status) bit 8 is set ON.
- Please stop the playing video before switching to another.

- Play previous file
  - [Command] = 2
  - [Parameter 1] = ignore (set 0)
  - [Parameter 2] = ignore (set 0)

Note
- If (File index) is zero, the same file is replayed.
- If the file cannot be found, (Status) bit 8 is set ON.

- Play next file
  - [Command] = 3
  - [Parameter 1] = ignore (set 0)
  - [Parameter 2] = ignore (set 0)
    - If there are no more files, the index 0 file is played
    - If the file cannot be found, [Status] bit 8 is set ON

- Pause / Play Switch
  - [Command] = 4
  - [Parameter 1] = ignore (set 0)
  - [Parameter 2] = ignore (set 0)

- Stop playing and close file
  - [Command] = 5
  - [Parameter 1] = ignore (set 0)
  - [Parameter 2] = ignore (set 0)

- Start playing from the designated time
  - [Command] = 6
  - [Parameter 1] = target time (second)
  - [Parameter 2] = ignore (set 0)

Note
Parameter 1 (target time) must be less than the ending of time or it plays the last second.

- Forward
  - [Command] = 7
  - [Parameter 1] = target time (second)
  - [Parameter 2] = ignore (set 0)

Note
- Going forward to the designated second in [Parameter 1]. If the video is paused, the forwarding action will be started by playing.
- When the designed time is later than the end time, it plays the last second.

- Backward
  - [Command] = 8
  - [Parameter 1] = target time (second)
  - [Parameter 2] = ignore (set 0)

Note
- Going Backward to the designated second in [Parameter 1], if the video is paused, the backward action will be started by playing.
- When the designed time is earlier than the beginning time, it plays from beginning.

- Adjust volume
  - [Command] = 9
  - [Parameter 1] = volume (0 ~ 128)
  - [Parameter 2] = ignore (set 0)

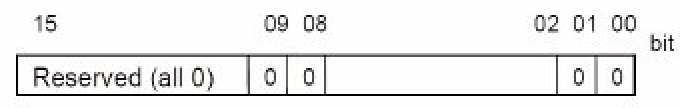
Note
Default volume is 128.

- Set video display size
  - [Command] = 10
  - [Parameter 1] = display size (0 ~ 16)
  - [Parameter 2] = ignore (set 0)

Note
- [Parameter 1 = 0] : fit video image to object size.
- [Parameter 1 = 1 ~ 16] : magnification from 25% ~ 400% in 25% increments where 1 = 25%, 2 = 50%, 3 = 75% and so on.

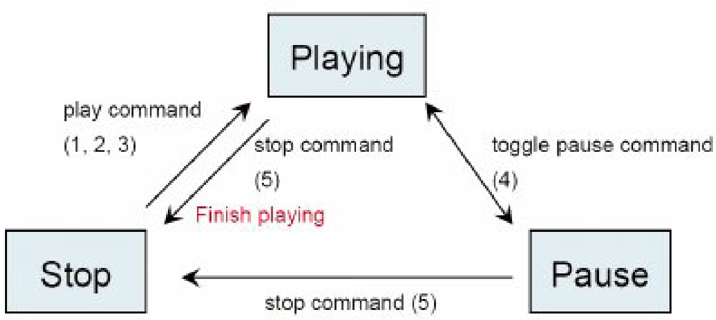
Status (control address + 3)

When playing a video the system sets (File Open (bit00)) and (File Playing (bit01)) to ON. If the file cannot be found, or an invalid command is entered, the Command Error bit 08 is set ON. If the file format is not supported, or a disk I/O error occurs, during playback (for example, USB disk unplugged), the File Error bit 09 is set ON.



00: File Opened / Closed (0 = closed, 1 = opened)
01: File Playing (0 = not playing, 1 = playing)
08: Command Error (0 = accepted, 1 = incorrect)
09: File Error (0 = accepted, 1 = incorrect)

Note
- The figure shows the status value associated with each state:
- Stop = 0, Pause = 1, Playing = 3



- (Command), (Parameter 1), and (Parameter 2) are write addresses. All others are read only.

13.25.2.3. Preview Tab

Users can test whether the video format is supported by using the preview function.

Setting	Description
Forward << / Backward >>	Go forward or backward of the video. (in minutes)
Play / Pause	Select to start playing video or pausing.
Stop	Stop playing and close the video file. To test another video, please stop playing the current video first.
Load	Select a video to preview.

Note

- Only one video file can be played at one time.
- If (control address) is not enabled and (Auto. repeat) is not selected, after finish playing the first file, the system will stop playing and close the video file.
- If (control address) is not enabled, the system will find the first file in the designated folder and start to play (in ascending order of the file name).
- If the file can be previewed, the format is supported. If the video image quality is poor, please adjust the resolution.
- The supported formats: mpeg4, xvid, flv…etc.

Download

Click the icon to download the demo project. Please confirm your internet connection before downloading the demo project.

13.26. DATA TRANSFER

13.26.1. Overview

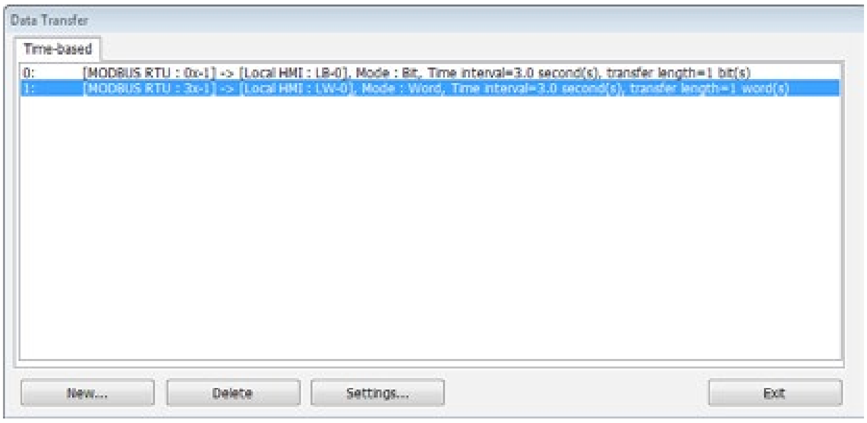
Data Transfer object is similar to Trigger-based Data Transfer object. They transfer the data from source to destination register. The difference is that Data Transfer object transfers data based on time schedule, and is able to transfer data in bits.

When using cMT Series, Data Transfer object is divided into two modes: (Time-based) and (Bit trigger). In these two modes, the system automatically detects the state of the designated bit register and executes data transfer. (Time-based) mode is the same as described earlier, where (Bit trigger) mode transfers data when the state of the designated bit register changes. For the detail of (Bit trigger) mode, see "13.26.2.2 Data Transfer Bit Trigger".

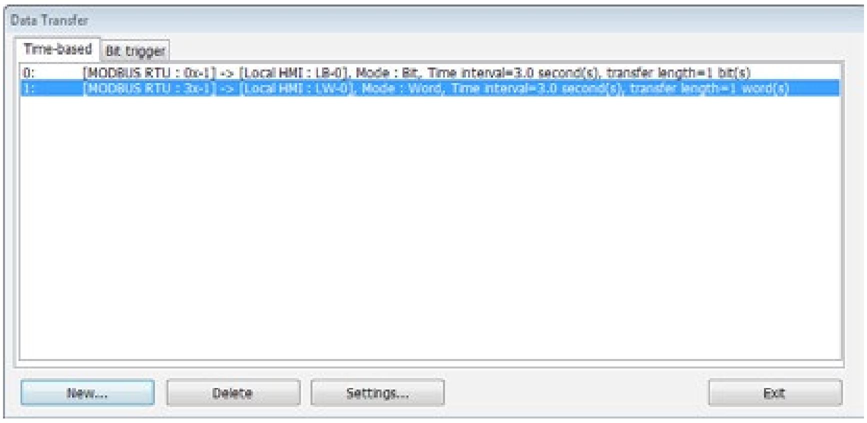
13.26.2. Configuration

	Click Data Transfer icon on the toolbar to open the Data Transfer management dialog box. Click (New) and configure the properties. All the defined Data Transfer can be viewed from the dialog box as shown in the following figure.

- eMT, iE, XE, mTV Series



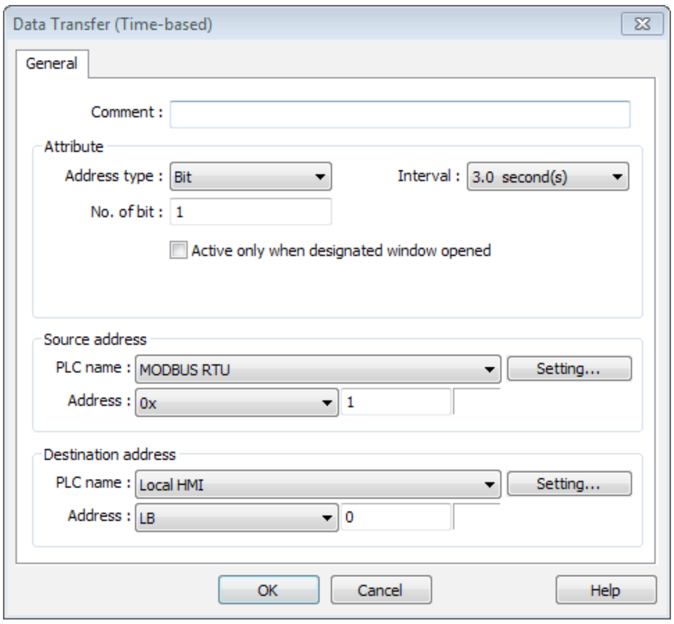
- cMT Series



13.26.2.1. Data Transfer Time-Based

- General Tab

Click the (New) button in the Data Transfer management dialog box.

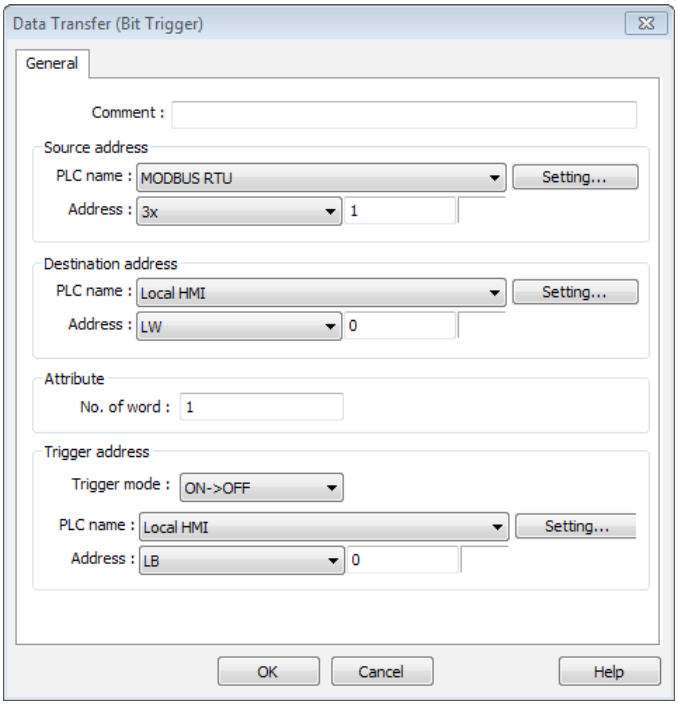


Setting	Description
Attribute	**Address type** Select the data type, either (Bit) or (Word). **No. of bits /No. of words** When (Bit) is selected in (Address type), set the number of bits transferred each time when data transfer is triggered. When (Word) is selected in (Address type), set the number of words transferred each time when data transfer is triggered. **Interval** Select the time interval of data transfer, for example, when 3 seconds is set, the system will transfer data every 3 seconds. Specifying a short time interval or a big number of data to transfer may cause an overall performance of system decrease. Therefore, it is recommended that users choose a longer time interval and a smaller amount of data to transfer. When a short interval is inevitable, be aware of the interval must be longer than the data transfer operation. For example, if the data transfer operation takes 2 seconds, set the interval longer than 2 seconds.
Source address	Data Transfer object reads the data from (Source Address).
Destination address	Data Transfer object writes the data to (Destination Address).

13.26.2.2. Data Transfer Bit Trigger

▪ General Tab

Click the (New) button in the Data Transfer management dialog box, and open (Bit trigger) tab.



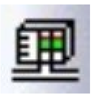
Setting	Description
Source address	Data Transfer object reads the data from (Source Address).
Destination address	Data Transfer object writes the data to (Destination Address).
No. of word	Set the number of words transferred each time when data transfer is triggered.
Trigger address	Set the register that controls data transfer and select the trigger mode. **Trigger mode** Trigger data transfer when the state of the designated register changes from Off to ON, ON to OFF, or at both of the changes of state.

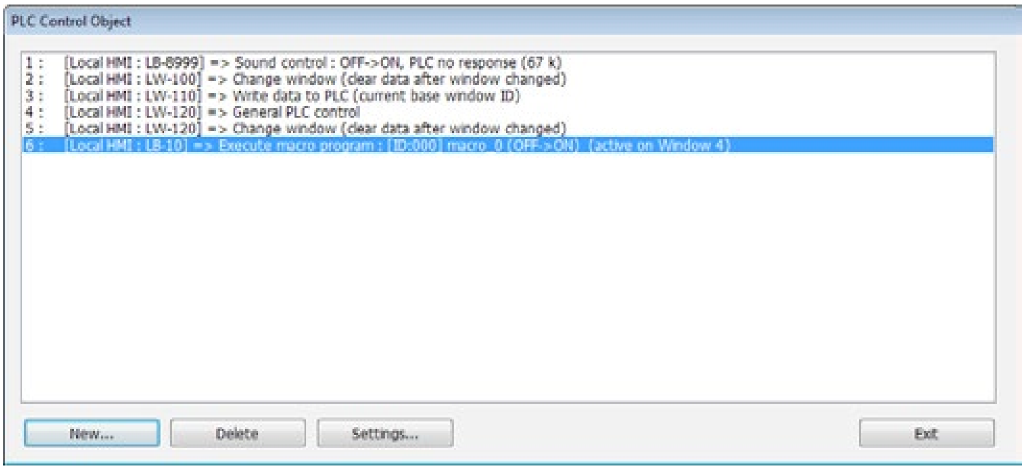
13.27. PLC CONTROL

13.27.1. Overview

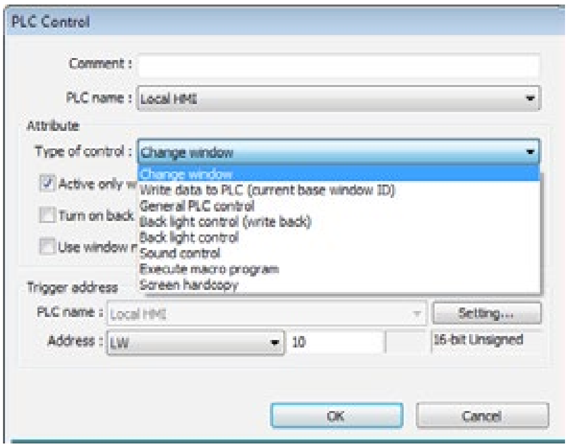
PLC Control object can execute commands when it is triggered.

13.27.2. Configuration

	Click the PLC Control icon on the toolbar to open the PLC Control Object management dialog box. To add a PLC Control object, click (New), set up the properties, press OK button and a new PLC Control object will be created.



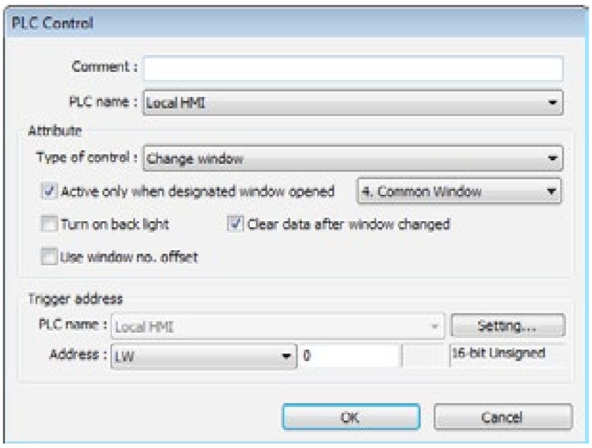
Click (New) and the following dialog box appears. See “13.27.2.1 Type of Control”.



	Note The (PLC Control) and (Backlight Control) options are not available for cMT Series.

13.27.2.1. Type of Control

- Change window



Setting	Description
Active only when designated window opened	Allow this operation only if a particular screen is displayed.
Turn on back light	The backlight is turned ON when the window object is changed. (Not available for cMT Series)
Clear data after window changed	Reset the value at trigger address to zero when the window object is changed.
Use window no. offset	Select the check box and select a window offset, the new window no. to change to will be the value in (Trigger address) plus the offset. For example, if (Trigger address) is LW-0 and offset is set to -10. When the value in LW-0 is 25, the system will change to window no. 15 (25-10=15). The range of the offset is -1,024 to 1,024. The (Clear data after window changed) check box is not available if (Use window no. offset) is selected.

Note
If (LB-9017) is set ON, the write-back function will be disabled, the new window number is not written back into a designated address.

Place a valid window number in the designated trigger address to change the base screen to the new window number. The new window number is written back into the designated address.
For example, if current window is window no. 10, and (Trigger address) is set to LW-0,
When LW-0 is changed to 11, the system will change the current window to window no. 11, and then write 11 to LW-1.
When the window is changed, the new window number is written back into the address that is calculated by (Trigger address) and the data format, as shown in following table.

Data format	Trigger address	Write address
16-bit BCD	Address	Address + 1
32-bit BCD	Address	Address + 2
16-bit Unsigned	Address	Address + 1
16-bit Signed	Address	Address + 1
32-bit Unsigned	Address	Address + 2
32-bit Signed	Address	Address + 2

- Write data to PLC (current base window)
  Each time the base window is changed, the new window number will be written into the (Trigger address).
- General PLC Control (eMT, iE, XE, mTV)
  Transfer word data blocks from PLC to HMI, and vise-versa, and the transfer direction is controlled by the value in the (Trigger address).

Value in (trigger address)	Action
1	Transfer data from PLC register → HMI RW register
2	Transfer data from PLC register → HMI LW register
3	Transfer data from HMI RW register → PLC register
4	Transfer data from HMI LW register → PLC register

Four consecutive word registers are used as described in the following table:

Address	Purpose	Description
(Trigger address)	Determine the direction of data transfer	The valid values are listed in the above table. When a new control code is written into the register, HMI will start to transfer. After data transfer is finished, the value will be set to 0.
(Trigger address) +1	The size of data to transfer.	The unit is word.
(Trigger address) +2	Offset to the start address of PLC register	Assume the value is “n”, where n is an arbitrary number, the start address of PLC register is (Trigger address + 4 + n). Take an OMRON PLC as an example: If (Trigger address) uses DM-100, (Trigger address + 2) will be DM-102. If the value in DM-102 is 5, the start address of data source would be DM-109 (100 + 4 + 5 = 109).
(Trigger address) +3	Offset to the start address of LW or RW memory in HMI	Take OMRON PLC as an example: If set (Trigger address) to DM-100, (Trigger address + 3) will be DM-103. If the value in DM-103 is 100, the start address of memory in HMI is RW-100 or LW-100.

Example 1
To use PLC Control object to transfer 16 words data in OMRON PLC, starting from address DM-100, to the HMI address, starting from RW-200. The setting is shown below:

1. Firstly, create a PLC Control object, set (Type of control) to (General PLC control), and set (Trigger address) to DM-10, that is, to use the four sequential registers start from DM-10 to control data transfer.
2. Confirm the data size and the offset addresses.
   Set DM-11 to 16, since the number of words to transfer is 16 words.
   Set DM-12 to 86, which indicates the address of data source is DM-100 (100=10+4+86).
   Set DM-13 to 200, which indicates the destination address is RW-200.
3. Set DM-10 according to the direction of data transfer.
   If set DM-10 to 1, the data will be transferred from PLC to HMI RW register.
   If set DM-10 to 3, the data will be transferred from HMI RW register to PLC.

- Back light control (write back) (eMT, iE, XE, mTV):
  When (Trigger address) is turned ON, HMI backlight will be turned ON/OFF and (Trigger address) will be turned OFF. Any touch on the screen will turn the backlight on.
- Back light control (eMT, iE, XE, mTV):
  When (Trigger address) is turned ON, HMI backlight will turn ON/OFF and the state of (Trigger address) will not be changed.
- Sound control:
  When the state of the designated (Trigger address) changes, the HMI will play the sound selected from the sound library. There is a further selection determines whether the sound is played after Off to ON, ON to OFF transition, or at both of the changes of state.
- Execute macro program:
  Select a pre-defined Macro from the drop-down list. When the state of the designated (Trigger address) changes, the selected Macro is executed. There is a further selection determines whether the Macro is executed after Off to ON, ON to OFF transition, or at both of the changes of state. If select (Always active when ON), the macro will be executed repeatedly. (The shortest time interval between runs is 0.5 second.)
- Screen hardcopy:
  When the state of the designated (Trigger address) changes, print the selected screen. There is a further selection determines whether the screen is printed after Off to ON, ON to OFF transition, or at both of the changes of state.
- There are three options to specify the source window for hardcopy:
  - Current base window:
    Print the base window at the time the operation is activated
  - Window no. from register:
    Print the window designated by the value in a designated address. If the window number is valid, the screen is printed
  - Designate window no.
    Directly select a base window to be printed

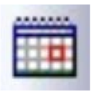
If not specifying any printer, there are other options such as SD card or USB disk.
The printer can be set in (System Parameter Settings) » (Model) tab.

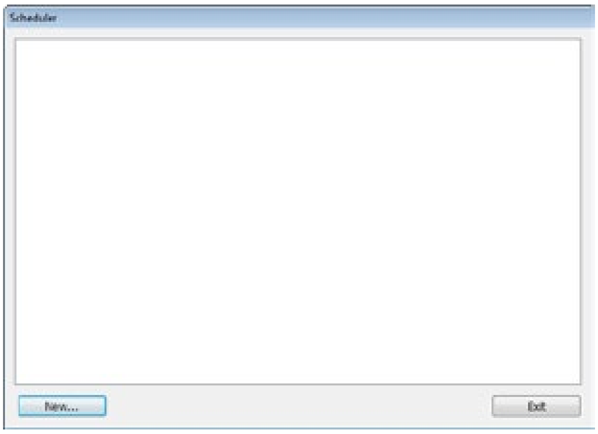
Note
- The (Printer) setting is not available for cMT Series. The window hardcopy file is saved in iPad Photo folder.
- A background printing procedure is performed when the printed window is not the current base window.
- If the hard-copied window is not the current base window, its (Direct Window) and (Indirect Window) objects will not be printed.

13.28. SCHEDULER

13.28.1. Overview
Scheduler object turns bits ON/OFF, or writes values to word registers at designated start times. It works on a weekly basis.

13.28.2. Configuration

	Click the Scheduler icon on the toolbar to open the Scheduler management dialog box, click (New) to open the Scheduler property setting dialog box.

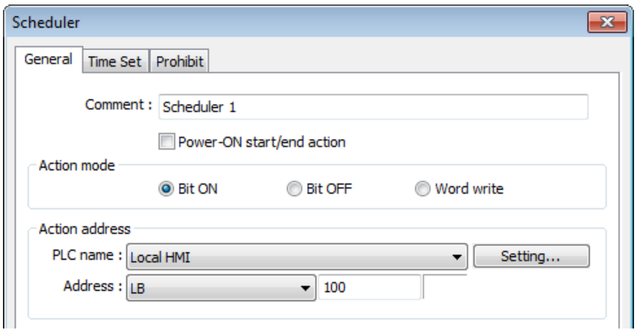


The following two demonstrations explain the usage of Scheduler.

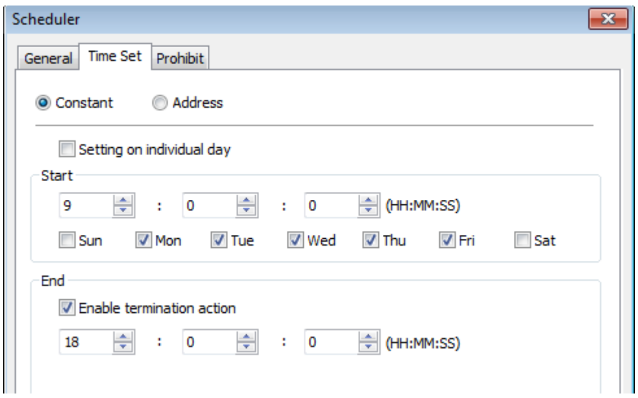
Example 1

A motor is scheduled to power - ON at 9:00 and power – OFF at 18:00, Monday to Friday.
We are using LB-100 to control the motor state. LB-100 will be set ON at 9:00 and OFF at 18:00.

1. Click the Scheduler icon on the toolbar to open the Scheduler management dialog box, click (New).
2. In (General) tab, select (Bit ON) in (Action mode) and set (Action address) to LB-100.



3. In (Time Set) tab, select (Constant).

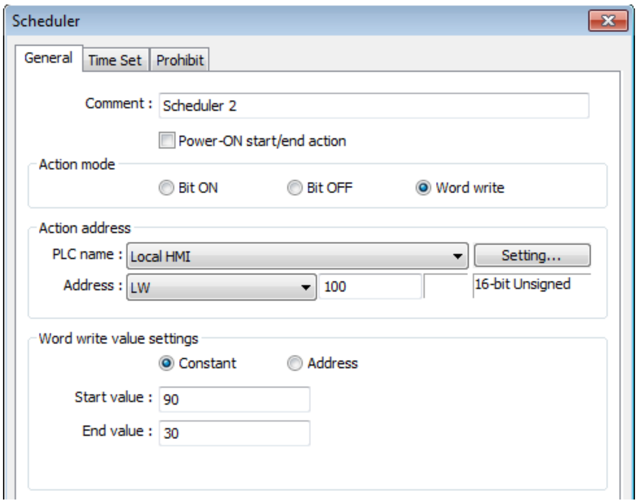


4. Enter (Start) time as 9:00:00 and select Monday to Friday. Do not select (Setting on individual day).
5. Enter (End) time as 18:00:00 and select (Enable termination action) check box.
6. Click (OK), a new Scheduler object will be created on the (Scheduler) list.

Example 2

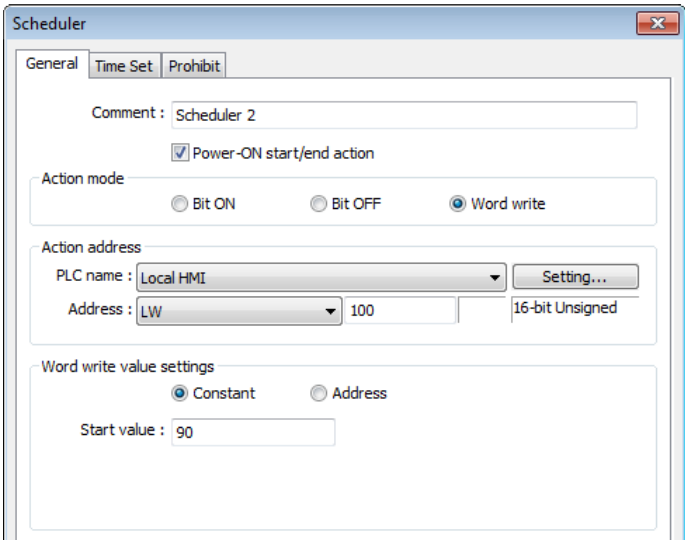
A thermal heater is scheduled to heat up to 90 °C at 08:00 and cool down to 30 °C at 17:00, Monday to Friday. LW-100 is used to store the set point value.

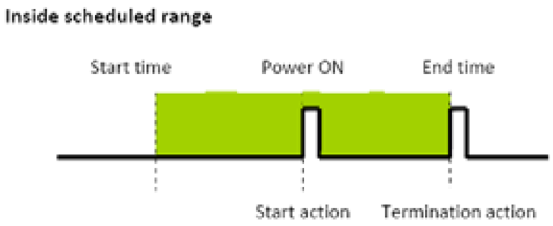
1. Click the Scheduler icon on the toolbar to open the Scheduler management dialog box, click (New).
2. In (General) tab, select (Word write) in (Action mode) and set (Action address) to LW-100.
3. Select (Constant) for (Word write value settings) and enter 90 in (Start value).

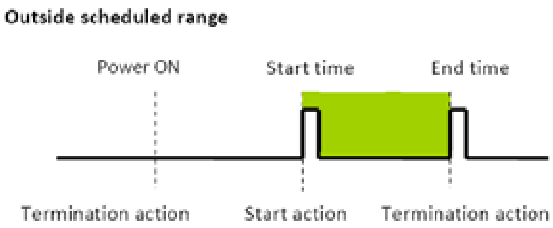


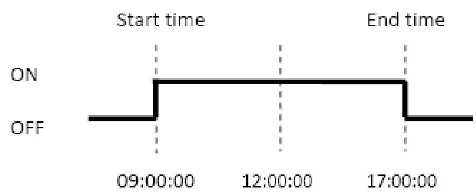
4. In (Time set) tab select (Constant).
5. Enter (Start) time as 8:00:00 and select Monday to Friday. Do not select (Setting on individual day).
6. Enter (End) time as 17:00:00 and select (Enable termination action) check box.
7. Return to (General) tab and enter 30 in (End value).
8. Click (OK), a new schedule object will be created on the (Scheduler) list.

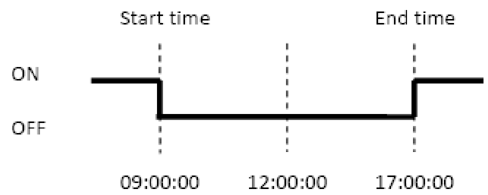
13.28.2.1. General Tab

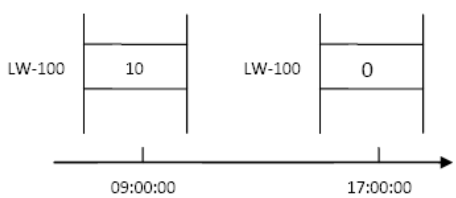


<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Power ON start/end action</td><td>
Execute the defined action when the HMI is powered ON.
▪ Enabled
When HMI is powered ON within the scheduled time range, the start action will be performed automatically. When HMI is powered ON outside the scheduled time range, the termination action will be executed.




▪ Disabled
When the HMI is powered ON at a time later than the start time, the start action will not be performed, but the termination action will be performed. When the termination action is not defined, the scheduled range is not recognized and no action is performed.
</td></tr>
<tr><td>Action mode</td><td>
Choose the action to do at the given time.
Bit ON
At the start time, set the designated bit ON. At the end time, set the designated bit OFF.
Example: start time: 09:00:00 End time: 17:00:00


Bit OFF
At the start time, set the designated bit OFF. At the end time, set the designated bit ON.
Example: start time: 09:00:00 end time: 17:00:00


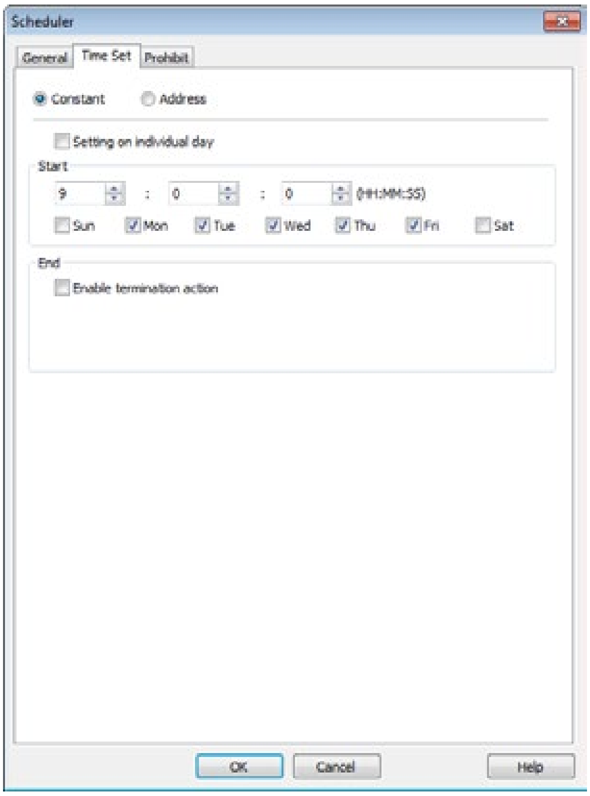
Word write
The (Write start value) entered here is transferred to the designated (Action address) word register at the start time. At end time, the (Write end value) entered here is written to the (Action address). The valued can be entered manually or be set by using (Address) mode. In (Address) mode, the value in the specified address is the start value where the value in (Address + 1) is the end value.
Example: device address: LW-100
Start time: 09:00:00 End time: 17:00:00
Write start value: 10 Write end value: 0
Use register: if control address is LW-n, then enter 10 in LW-n and enter 0 in LW-(n+1).


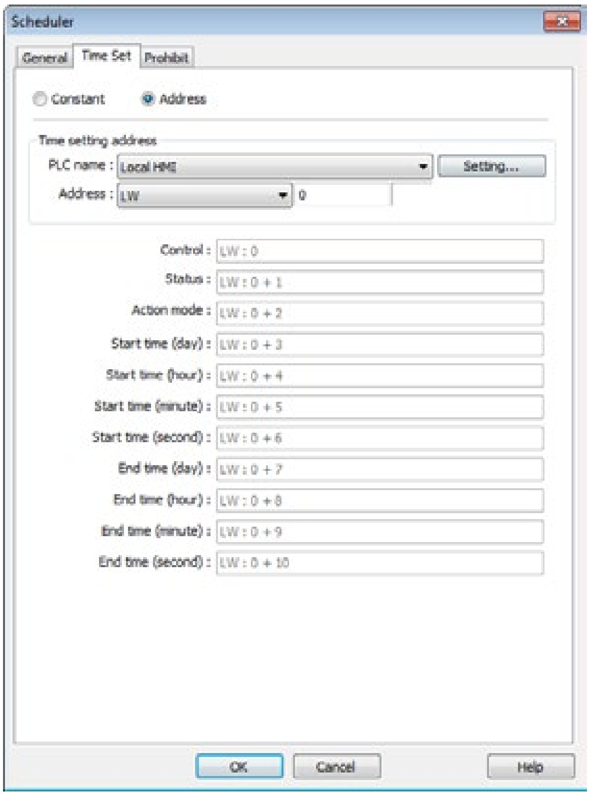
</td></tr>
</table>

Note
Only is an (End time) is set in the (Time set) tab will the (Write end value) box appear.

13.28.2.2. Time Set

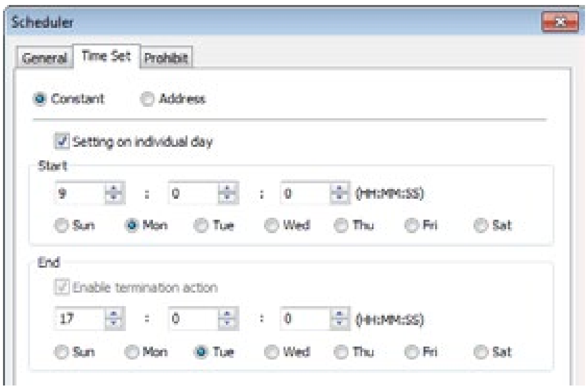
Specify start time and end time. (Constant) allows specifying a date or period and time. (Address) allows controlling the time by the designated address.

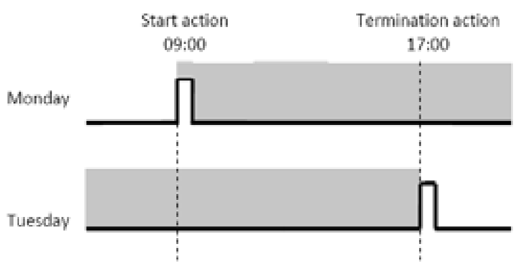




Constant

- Setting on individual day
- If (Setting on individual day) is selected
- The same start time and end time can be assigned to different days of the week

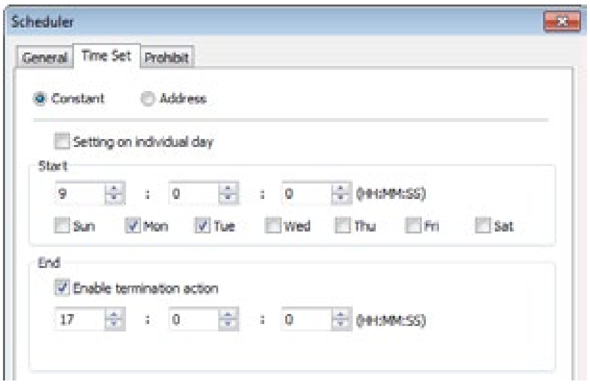


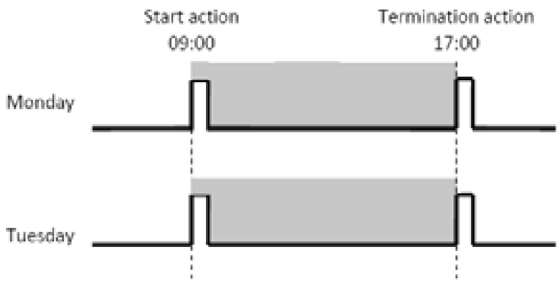


Note

- Start and end time must be entered.
- Start and end time must be on a different time, or same time but different day.

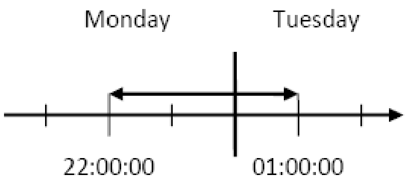
If (Setting on individual day) is not selected. Start time and end time entered must start and end within 24-hours.





Note
Start time and end time must be on a different time, different day.
If an end time is earlier than a start time, the end action will occur in the next day.

Start date: Monday
Start time: 22:00
End time: 01:00



Address
The scheduler object retrieves the start/end time and day of week information from word registers, enabling all parameters to be set and changed under PLC or user control.
Designated as the top address in a block of 11 sequential registers which are used to store time setting data.
The format of the 11 word registers should normally be 16-unsigned integer. If a 32-bit word address is chosen, only bits 0-15 are effective, and bits 16-31 should be written as zero.
The following describes each register.

Control (Time setting address + 0)
When (Control) bit is ON, the HMI will read and update (Action mode), (Start time), and (End time) values.

15	0	Bit
Reserved (0 fixed)	0	

Bit 0: no action 1: read times/action mode

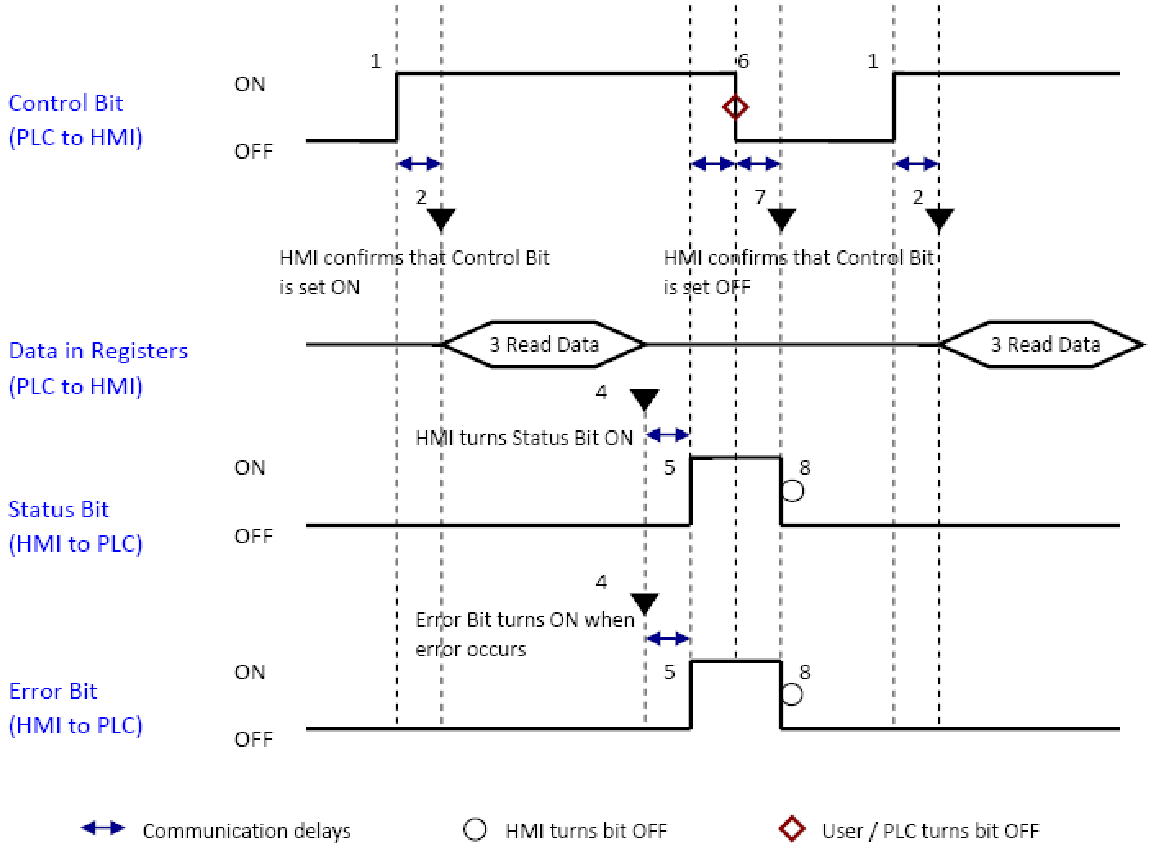
Note
HMI will not regularly read the data from (Action mode) (address + 2) to (End time) (address + 10). Please turn (Control) ON when the settings are changed.

Status (Time setting address + 1)
When the read operation is completed, Bit00 of this register turns ON. If time data read is out of range or incorrect in any way Bit01 turns ON.

15	02	01	00	Bit
Reserved (0 fixed)		0	0	

Bit 00: Status bit: Read operation completed. (0: reading or reading not started. 1: reading completed.)
Bit 01: Error bit: Start or end time format incorrect. (0: corrected 1: error)

Note
After the scheduler reads the data and the status is turned ON (The value in (Address + 1) = 01), the control bit must be turned OFF (address = 0). The status bit and error bit will be turned OFF (1→0) at the same time.



Action mode (Time setting address + 2)

Enable/disable (Enable termination action) and (Setting on individual day). Whatever the (Enable termination action) bit is, all the time data, from (Control) to (End time (second)), will be read.

15	02	01	00	Bit
Reserved (0 fixed)		0	0	

Bit 00 Enable termination action (0: Disabled 1: Enabled)

Bit 01 Setting on individual day (0: Disabled 1: Enabled)

Note
- If (Enable termination action) is OFF, all 11 registers are still read but end time is ignored.
- If (Setting on individual day) is ON, make sure that all start end times are entered. If more than one start / end day bit is ON, and error will occur.

Start/End Day (Start Day: time setting address + 3, End Day: time setting address + 7)
Designates which day of week is used to trigger the start or end action.

15 – 07	06	05	04	03	02	01	00	Bit
Reserved (0 fixed)	Sat	Fri	Thu	Wed	Tue	Mon	Sun	

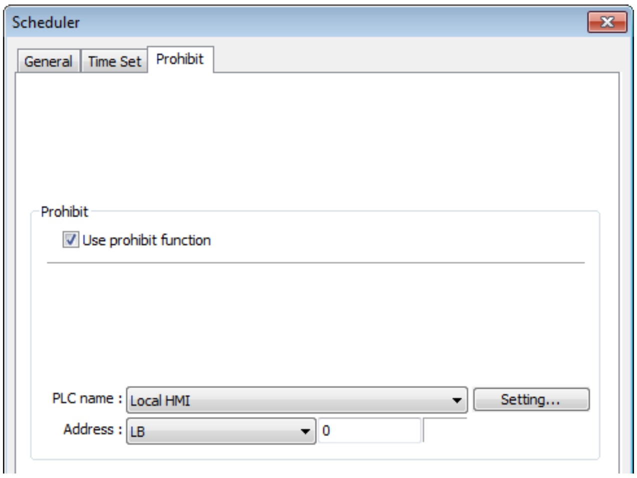
Bit 00 Sunday (0: not used 1: used)
Bit 01 Monday (0: not used 1: used)
Bit 02 Tuesday (0: not used 1: used)
Bit 03 Wednesday (0: not used 1: used)
Bit 04 Thursday (0: not used 1: used)
Bit 05 Friday (0: not used 1: used)
Bit 06 Saturday (0: not used 1: used)

Start/End Time (Start Time: time setting address + 4 to + 6, End Time: time setting address + 8 to + 10)
Hour: 0 – 23 Minute: 0 – 59 Second: 0 - 59
Values outside these ranges will cause error.

Note
- 16-bit unsigned integer format must be used; BCD format is not supported here.
- End time depends on (Action mode) (address + 2). (Enable termination action) (Bit 00) and (Setting individual day) (Bit 01) are related:

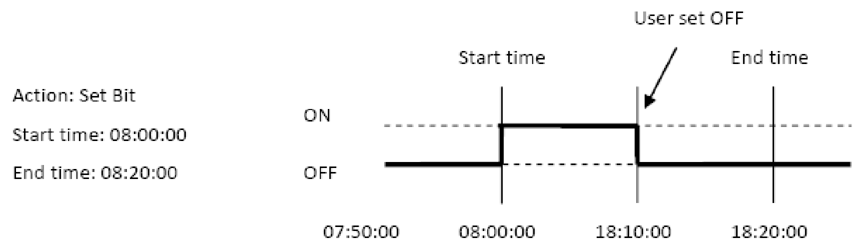
Setting individual day	Enabled	Disabled	
Enable termination action	Enabled	Enabled	Disabled

13.28.2.3. Prohibit Tab



Before the scheduled action is performed, the HMI will read the specified bit state. If it is ON, the scheduled start/end action will be skipped. Otherwise, it will be performed normally.

Note
- Up to 32 scheduler objects are allowed.
- A time schedule applies one action only when the start time is reached.

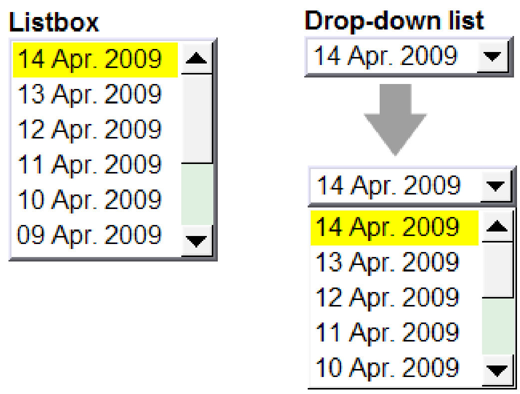


- (Write start/end value) and (Prohibit) bit is read only once before start action. After that, even to change the state of (Prohibit) bit or (Write start/end value), the end action and the value written will not be affected. Also, to read data of (Write start/end value) and (Prohibit) bit, there is a delay of start action due to the communication
- Each time RTC data is changed, scheduler list entries that possess both start and end times will be checked for in-range or out-range conditions. For in-range, the start action will occur. If the end action is not set, the new range is not recognized, the action will not occur
- If several Scheduler objects are set to the same start time or end time, the action is performed in ascending order of the schedule number
- In (Time Set) » (Address) mode, the system will read (Control) word regularly. The length of the period depends on the system
- In (Time Set) » (Address) mode, when start time and end time is out- range, error occurs in the set action time. (Note: BCD is not an acceptable format)
- In (Time Set) » (Address) mode, the action will not start up until the first time the time data is successfully updated

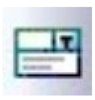
13.29. OPTION LIST

13.29.1. Overview

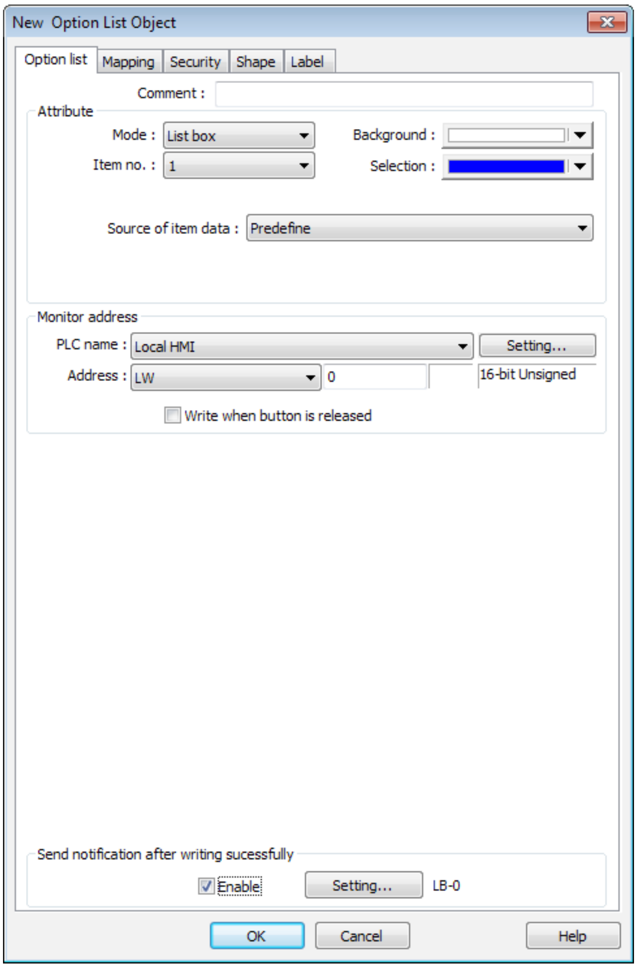
Option List object displays a list of items that the user can view and select. Once the user selects an item, the corresponding data will be written to a word register.
There are two forms of this object – (List box) and (Drop-down list). The (List box) lists all items and highlights the selected one. The (Drop-down list) normally displays only the selected item. Once the object is pressed, the system will display a list (which is similar to list box) as shown in the following figure.



13.29.2. Configuration

	Click the Option List icon on the toolbar to open an Option List object property dialog box. Set up the properties, press OK button, and a new Option List object will be created.

13.29.2.1. General Tab



Setting	Description
Attribute	**Mode:** the list style, either (List box) or (Drop-down list). **Item no.:** set the number of items for the object. Each item represents a state displayed in the list and the corresponding value will be written to the (Monitor address). **Background:** set background color. **Selection:** set background color for the selected item. **Source of item data:** there are 4 sources available: (Predefine), (Dates of historical data), (Item address), and (User account). See 13.29.2.1.
Monitor address	The corresponding value of the selected item will be written to (Monitor address). **Write when button is released** If this check box is selected, the selected item value will be written to (Monitor address) after the button is released.
Send notification after writing successfully	Set On/Off the designated bit address after successfully writing data to PLC.

	Note For cMT Series, the (Dates of historical data) and the (write when button is released) selections are not available.

13.29.2.2. Source of Item Data

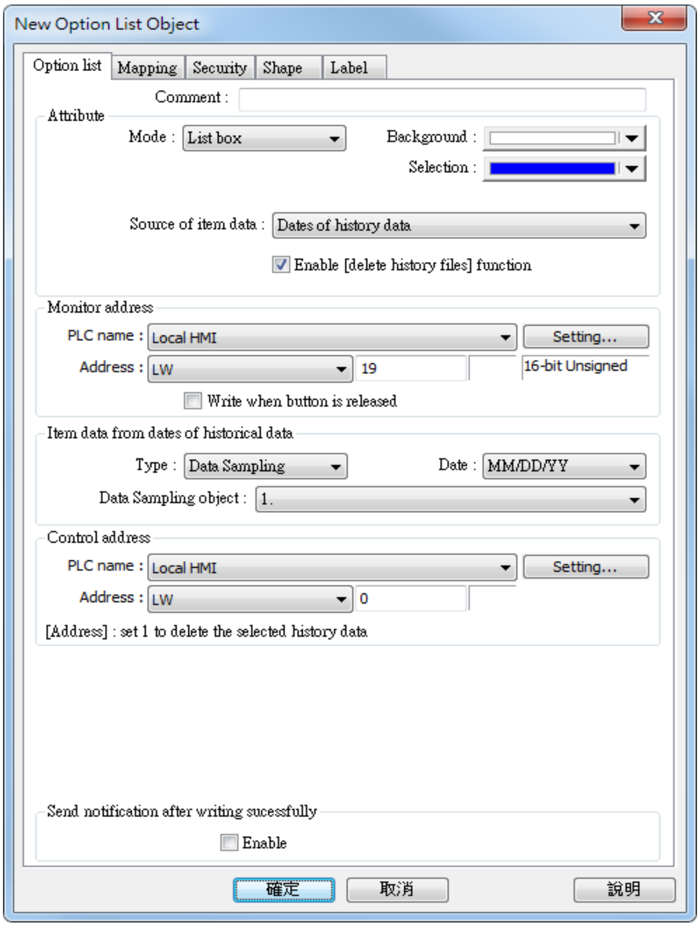
Predefine

The list is manually defined in (Mapping) tab.

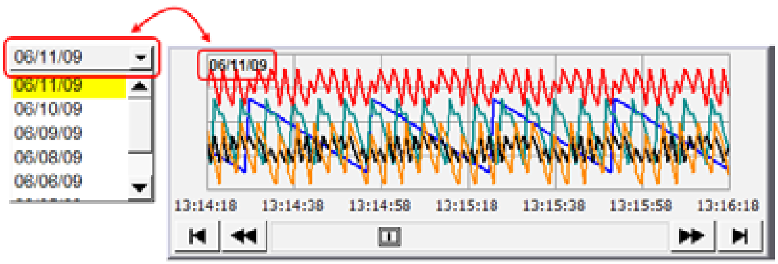
The number of items can be adjusted by (Item no.), and each item represents one state. Each item has a corresponding value which will be written to (Monitor address).

Dates of historical data

This selection is not available for cMT Series.



Option List object can be used with historical data display objects, such as Trend Display object, History Data Display object and Event Display object to control which history file should be shown. The figure below is an example of Option List used with Trend Display.



Setting	Description
Type	Two options are available: (Event (Alarm) log) and (Data sampling).
Date	Set the date format. YYYY means a four digits year (EX: 2,012), YY means a two digits year (EX: 12), MM means month and DD means day.
Data sampling object	Select which Data Sampling object is displayed when (Type) is (Data Sampling), and it should be the same as the (Data sampling object index) configured in (Trend Display) or (History Data Display).
Enable (delete history data) function	If selected, a control address can be set. Writing “1” to this address will delete the history data of the specified date.

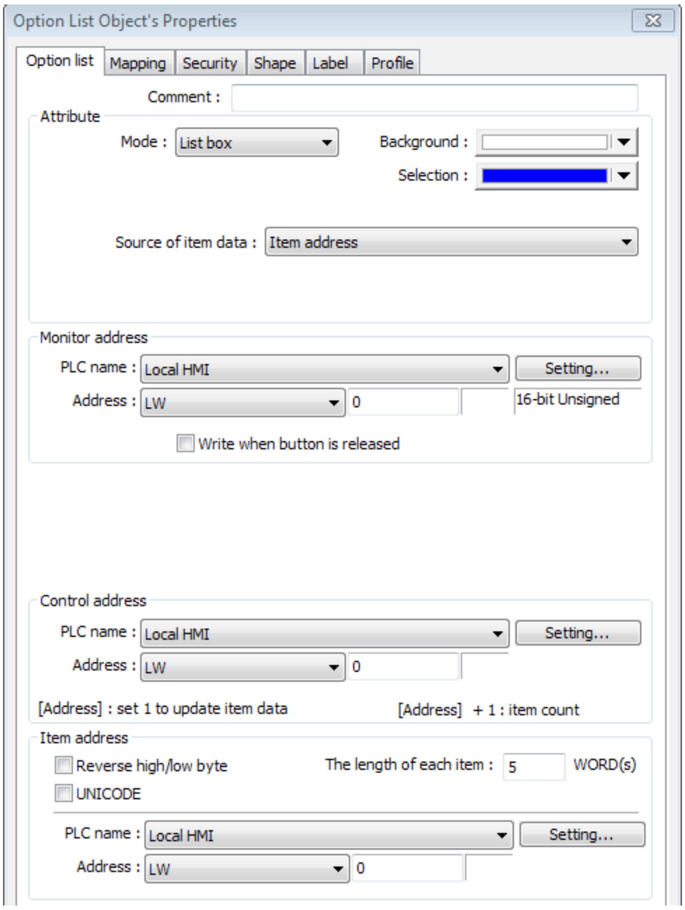
Note

- In (Dates of historical data) mode, since the system automatically reads the historical data and finds the date information, it is not necessary to configure in the (Mapping) tab.
- The error message displayed in Option List can be modified in (Mapping) tab.

Item	Value	Item data
0 (error)		Error!!

Item address

The list will be read from the given (Item address) and controlled by (Control address). The following options will be available:



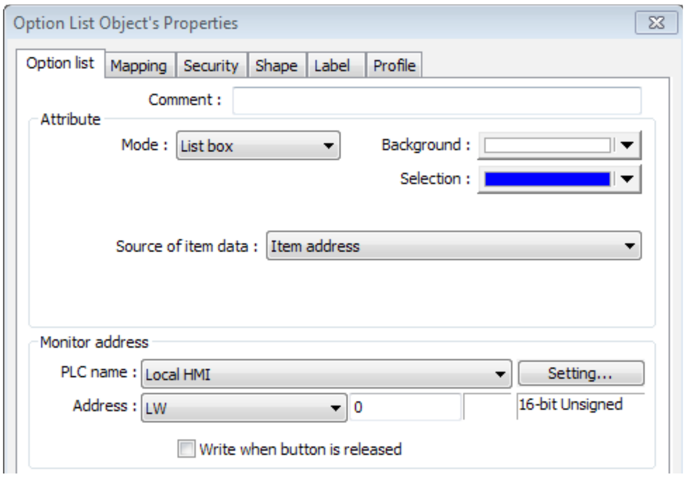
Setting	Description
Control address	▪ (Address): if the value at this address is changed to 1, the option list would be replaced by items defined at (Item address). After updating, the value will be restored to 0. ▪ (Address + 1): define the number of items in (Item address).
Item address	Assign the item address **UNICODE** The item will use UNICODE characters, such as Chinese characters. **The length of each item** Define the number of letters for each item, the unit is Word.

Note

- The UNICODE characters used here should be used by Text object, so that EasyBuilder Pro will compile the needed fonts and download these fonts to HMI, then the UNICODE letters can be correctly displayed.
- (The number of items) multiplied by (The Length of each item) must be less than 1,024 words.
- The system automatically disables (Mapping) tab in (Item address) mode.

User account

If (Enhanced Security) mode is enabled, (User account) would appear in the (Source of item data) and it lists the names of users.



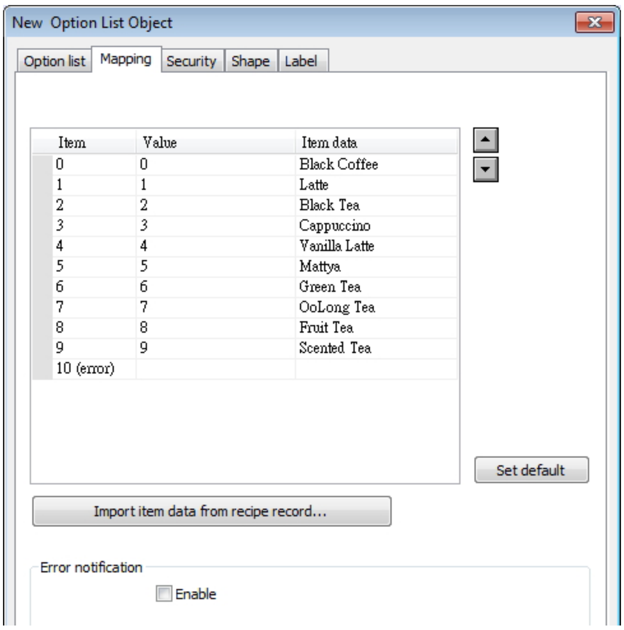
Setting	Description
Sort	Select the sorting method from (Ascending) of (Descending).
Display	If (Privilege) is selected, the privileges for each user will be displayed in option list. If (Secret user) is selected, even though it is defined to be hidden in (System parameter settings) » (Security) » (Enhanced Security), the users will still be displayed in (Option List).

Note

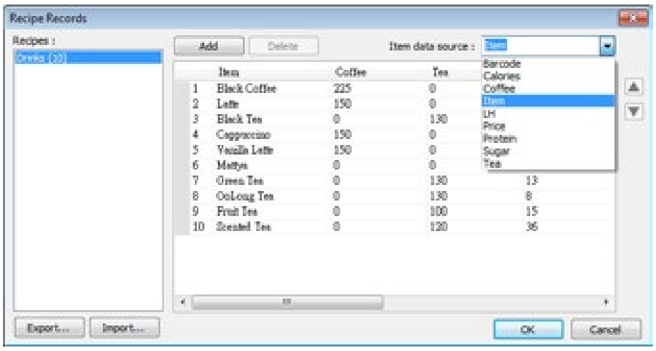
The address that controls user index is (Control Address +2 (LW-n+2)) which is set in (System Parameters) » (Security) » (Enhanced Security).

13.29.2.3. Mapping Tab

This table displays all available states/items, their item data and values. To change the number of available items, please go to (Option list tab) » (Attribute) » (Item no.).



Setting	Description
Item	The system lists all available items. Each item represents a state that will be displayed in the list. This field is read-only.
Value	Here user can assign value for each item, basing on the following two criteria: For reading: if the value in (Monitor address) is changed, the object selects the first-matched item. If no item is matched, the status goes to error state and signals the notification bit register (if requested). For writing: the system writes this value to (Monitor address) when user selects an item.

Item data	Text displayed for each item. The Option List object displays the text of all items in the list for users to review and select.
Import item data from recipe record	This feature is enabled when select (Recipe-Selection) as (Monitor address). Click (Import item data from recipe record) to open the (Recipe Records) setting dialog box. Select (Item data source), the data belonging to the selected column will all be imported to Option List object.  Before importing, the number of items in Option List changes according to the number of items defined in Recipe Records. After importing, modifying Recipe Records will not change the content of Option List.
Error state	On error state, the list box removes the highlight to represent no item is selected and the drop-down list displays the data of error state. Only the drop-down list uses error state, list box is not able to use error state. For example, item number 8 is the error state when specifying 8 in (Item no.). (The first item number is 0).
Set default	Reset all values or states to default. That is, set 0 for item 0, 1 for item 1, and so on.
Error notification	The system will set ON/OFF to the specified bit register when error is detected. The signal of the bit register could be used to trigger a procedure for correcting the error by using objects such as Event Log, Alarm Bar, or pop-up window.

13.30. TIMER

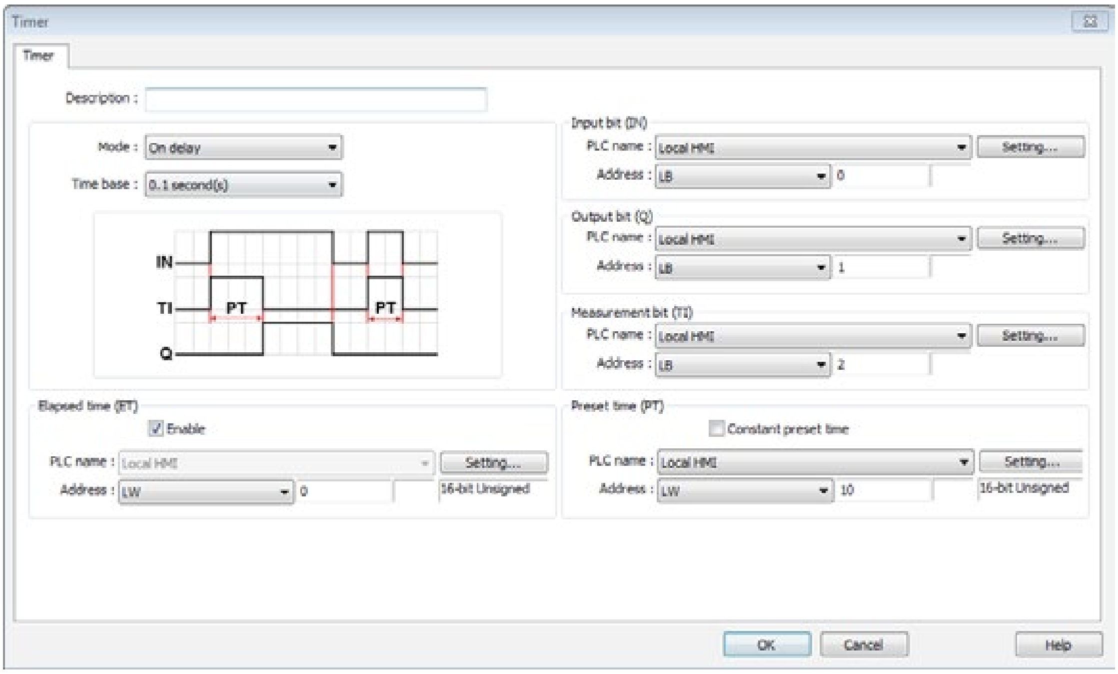
13.30.1. Overview

Timer object is a switch that can be used to control the mode to count time. The modes are explained later.
Timer object uses the following 6 variables:

Timer variable	Type	Description
Input bit (IN)	Bit	The main switch of Timer.
Measurement bit (TI)	Bit	Turns ON when the Timer begins counting time.
Output bit (Q)	Bit	Activated when the Timer finishes counting time.
Preset time (PT)	Word	Presets a time before the Timer begins counting time.
Elapsed time (ET)	Word	Displays the elapsed time.
Reset bit (R)	Bit	Resets the elapsed time (ET) to 0.

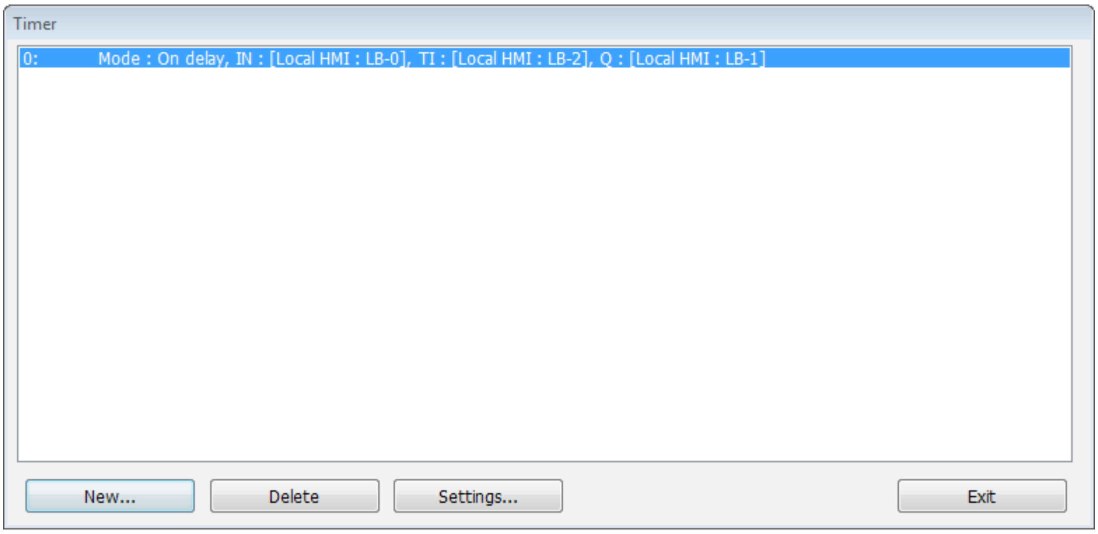
13.30.2. Configuration

	Click the Timer icon on the toolbar to open the property dialog box as shown in the following figure.

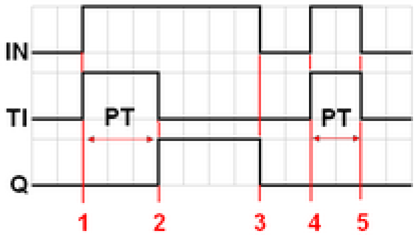


	Note (Constant preset time) is only available for cMT series.

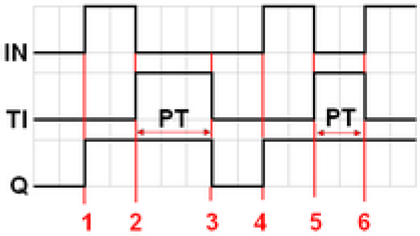
If use cMT Series, clicking the Timer icon on the toolbar will open the Timer managing window, click (New) to configure.



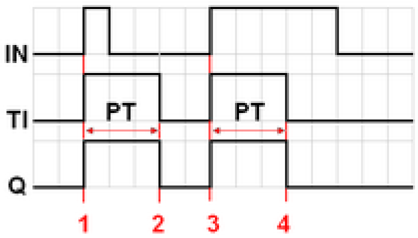
On delay

<table>
<tr><th>Mode</th><th>Register</th></tr>
<tr><td>
</td><td>Input bit (IN): the main switch of Timer.
Measurement bit (TI): turns ON when the Timer begins counting time.
Output bit (Q): turns ON when the Timer finishes counting time.
Preset time (PT): presets a time before the Timer begins counting time.
Elapsed time (ET): displays the elapsed time.</td></tr>
<tr><th colspan="2">Description</th></tr>
<tr><td colspan="2">Period 1: when the IN turns ON, TI turns ON and the ET starts counting. The Q remains OFF.
Period 2: when the ET equals to the PT, the TI turns OFF and the Q turns ON.
Period 3: when the IN turns OFF, the Q turns OFF and the ET is reset to 0.
Period 4: when the IN turns ON, the TI turns ON and the ET starts counting. The Q remains OFF.
Period 5: turns IN OFF before the ET reaches the PT, the TI turns OFF, and the ET is reset to 0. Since the ET doesn't reach the PT, the Q remains OFF.</td></tr>
</table>

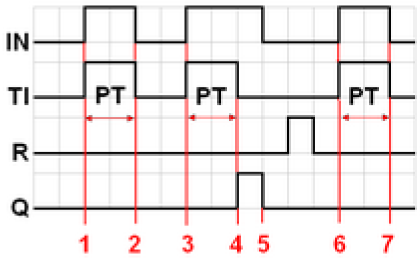
Off delay

<table>
<tr><th>Mode</th><th>Register</th></tr>
<tr><td>
</td><td>Input bit (IN): the main switch of timer.
Measurement bit (TI): turns ON when the timer begins counting time.
Output bit (Q): turns OFF when the timer finishes counting time.
Preset time (PT): presets a time before the timer begins counting time.
Elapsed time (ET): displays the elapsed time.</td></tr>
<tr><th colspan="2">Description</th></tr>
<tr><td colspan="2">Period 1: when the IN turns ON, the TI remains OFF and the Q turns ON, the ET is reset to 0.
Period 2: when the IN turns OFF, the TI turns ON and the Q remains ON, the ET starts counting.
Period 3: when the ET equals to the PT, the Q and TI turn OFF.
Period 4: when the IN turns ON, the TI remains OFF and the Q turns ON, the ET is reset to 0.
Period 5: when the IN turns OFF, the TI turns ON and the Q remains ON, the ET starts counting.
Period 6: turns the IN to ON before the ET reaches the PT, the TI turns OFF, the Q remains ON, and the ET is reset to 0.</td></tr>
</table>

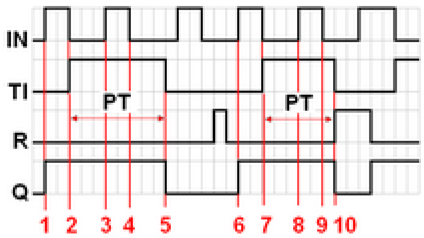
Pulse

<table>
<tr><th>Mode</th><th>Register</th></tr>
<tr><td>
</td><td>Input bit (IN): the main switch of timer.
Measurement bit (TI): turns ON when the timer begins counting time.
Output bit (Q): turns ON when the timer begins counting time and turns OFF when the timer finishes counting time.
Preset time (PT): presets a time before the timer begins counting time.
Elapsed time (ET): displays the elapsed time.</td></tr>
<tr><th colspan="2">Description</th></tr>
<tr><td colspan="2">Period 1: when the IN turns ON, the TI and Q turn ON simultaneously, and the ET starts counting.
Period 2: when the ET equals to PT, the TI and Q turn OFF simultaneously. Since IN is turned OFF when counting time, the ET is reset to 0.
Period 3: when the IN turns ON, the TI and Q turn ON simultaneously, and the ET starts counting.
Period 4: when the ET equals the PT, the TI and Q turn OFF simultaneously.</td></tr>
</table>

Accumulated ON delay

Mode	Register
	**Input bit (IN):** the main switch of timer. **Measurement bit (TI):** turns ON when the timer begins counting time. **Output bit (Q):** turns ON when the timer finishes counting time. **Preset time (PT):** presets a time before the timer begins counting time. **Elapsed time (ET):** displays the elapsed time. **Reset bit (R):** resets ET to 0.
Description	
Period 1: when the IN turns ON, the TI turns ON and the elapsed time ET starts counting, the Q remains OFF. **Period 2:** when the IN turns OFF, if the ET doesn't reach the PT, the TI turns OFF, and at the same time the Q remains OFF. The ET is in the retentive state. **Period 3:** when the IN turns ON, the TI turns ON. The timer measurement starts again and the ET starts counting from the kept value. The Q remains OFF. **Period 4:** when the ET reaches the PT, the TI turns OFF and the Q turns ON. **Period 5:** when the IN turns OFF, the Q turns OFF. Turning ON the reset bit R will reset the ET to 0, and then the reset bit turns OFF.	

Accumulated OFF delay

Mode	Register
	**Input bit (IN):** the main switch of Timer. **Measurement bit (TI):** turns ON when the timer begins counting time. **Output bit (Q):** turns OFF when the timer finishes counting time. **Preset time (PT):** presets a time before the timer begins counting. **Elapsed time (ET):** displays the elapsed time. **Reset bit (R):** resets ET to 0
Description	
Period 1: when the IN turns ON, the TI remains OFF and the Q turns ON. **Period 2:** when the IN turns OFF, the TI turns ON and the Q remains ON. The ET starts counting. **Period 3:** when the IN turns ON, the TI and Q remain ON, and the ET is in the retentive state. **Period 4:** when the IN turns OFF again, the ET starts counting from the kept value. **Period 5:** when the ET equals to the PT, the TI and Q turn OFF simultaneously. Turning ON the reset bit R will reset the ET to 0, and then the reset bit turns OFF.	

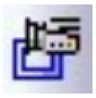
13.31. VIDEO IN AND VIDEO IN (USB CAMERA)

13.31.1. Overview

The eMT, XE, and mTV Series HMI models provide the Video Input feature. By installing a surveillance camera, user can monitor the site on HMI. The video images can be stored in external devices and then analyzed on PC. This feature can be utilized in different places for monitoring, such as vehicles or buildings.

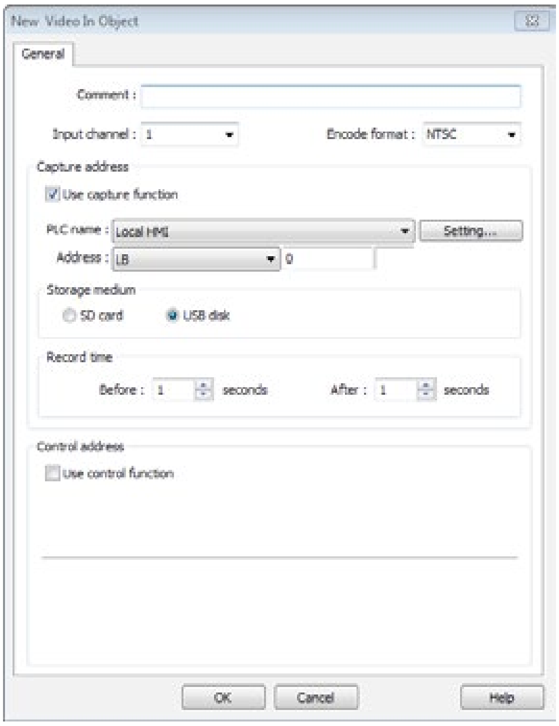
- eMT, XE, and mTV Series: support USB camera video input
- eMT3120A/eMT3150A: not only support USB camera video input but also NTSC and PAL analog video systems. For hardware, HMI provides 2 channels for Video Input. User can freely switch the channel, and capture real-time images even when Video Input is paused

13.31.2. Configuration

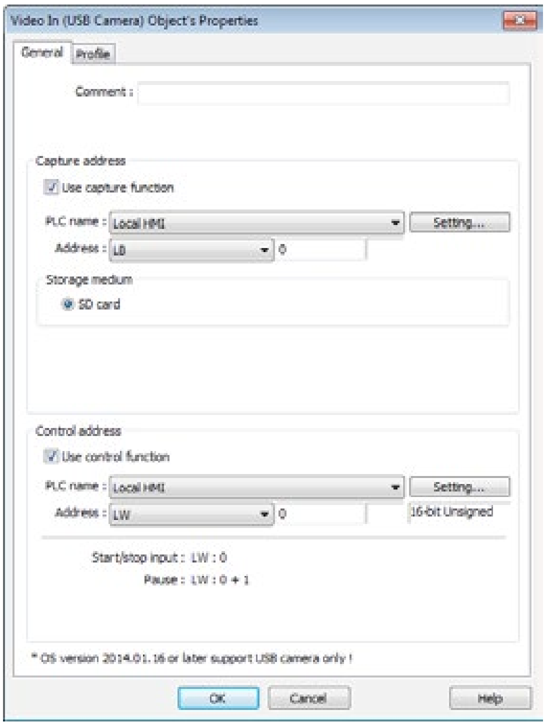
	Click the Video In icon on the toolbar to open the property dialog box. Set up the properties, press OK button, and a new Video In object will be created.

13.31.2.1. General Tab

Analog Video Systems



USB Camera



<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Input channel</td><td>Select the Video Input channel from channel 1 or channel 2. (Analog video systems)</td></tr>
<tr><td>Encode format</td><td>Select the format from NTSC or PAL. (Analog video systems)</td></tr>
<tr><td>Capture address</td><td>Select (Use capture function) check box and configure the settings.
Capture address
Designate the address that triggers image capturing.
Storage medium
Designate the storage device.
<table>
<tr><th>System</th><th>Storage device</th></tr>
<tr><td>Analog video system</td><td>Select SD card or USB disk to save the captured images. The images of channel 1 will be saved in “VIP1” folder in the chosen storage and so on.</td></tr>
<tr><td>USB Camera</td><td>Only saves the captured images in SD card.</td></tr>
</table>
Record time
Set a period of time to capture the images.
<table>
<tr><th>System</th><th>Method</th></tr>
<tr><td>Analog video system</td><td>▪ The longest period can be set from 10 seconds before triggering (Capture address) to 10 seconds after triggering
▪ The time interval of image capturing is once every second
▪ The captured .jpg file will be named in the following format:
  ▪ Before or after (Capture address) is triggered: YYYYMMDDhhmmss.jpg
  ▪ The moment that (Capture address) is triggered: YYYYMMDDhhmmss@.jpg
For example, set (Record time) “Before” and “After” to “5” seconds. When the state of (Capture address) changes from OFF to ON, the system will start capturing one image per second, from 5 seconds before the triggering time to 5 seconds after the triggering time, which is 11 images in total including the one captured at the triggering moment.</td></tr>
<tr><td>USB Camera</td><td>Only the image of the triggering moment is captured. The name format: YYYYMMDDhhmmss.png.</td></tr>
</table>
</td></tr>
<tr><td>Control address</td><td>Use control function
If enabled, enter certain value to the control address and the following addresses can control Video Input object. For example, if the designated control address is LW-n (n is any address), enter certain value to the designated addresses will execute commands as the following table.
<table>
<tr><th>Address</th><th>Value</th><th>Command</th></tr>
<tr><td rowspan="5">LW-n</td><td>0</td><td>Stop displaying image</td></tr>
<tr><td>1</td><td>Open channel 1 and display the image on HMI</td></tr>
<tr><td>2</td><td>Open channel 2 and display the image on HMI</td></tr>
<tr><td>3</td><td>Open channel 1 but don’t display the image on HMI (Capture function operable)</td></tr>
<tr><td>4</td><td>Open channel 2 but don’t display the image on HMI (Capture function operable)</td></tr>
<tr><td>LW-n+1</td><td>1</td><td>Pause / resume the video</td></tr>
<tr><td>LW-n+2</td><td>1~100</td><td>Adjust the contrast ratio (Analog video systems)</td></tr>
<tr><td>LW-n+3</td><td>1~100</td><td>Adjust the brightness (Analog video systems)</td></tr>
</table>
▪ After changing the value in (Control address (LW-n)), the system will keep the new value
▪ After changing the value in (Control address + 1 (LW-n+1)), the system will execute the command and then reset the value to 0
▪ If (Use control function) check box is not selected, the system will play the image of the selected channel
▪ If (Display adjustment) check box is selected, the contrast ratio and brightness can be adjusted. (Analog video systems)
▪ The USB Camera is controlled by LW-n (value 0 or 1) and LW-n+1</td></tr>
</table>

Note

About analog video systems:

- Only one channel can be opened at a time.
- Real-time images can still be captured when Video In is paused.
- Recommended analog video systems and resolutions:

	1:1	50%
NTSC	720 x 480	360 x 240
PAL	720 x 576	360 x 288

About USB Camera:

- If the USB Camera is removed during video playing, the image will not be loaded even though the USB Camera is returned. If (Control address) is selected, please stop and then restart video input. If (Control address) is not selected, please switch to another screen and then return, or restart HMI.
- The maximum size of Video In object of eMT3070A is 340*240, as for eMT3105P, eMT3120A, eMT3150A, XE, and mTV Series, the maximum size of Video in object is 640*480.
- When using USB Camera, the resolution of the run-time video image is determined by the resolution supported by the USB Camera that is closest to the size of Video In object . The resolution supported by the USB Camera may not be identical to the size of the object. The same resolution of Video In object and USB Camera image is recommended.
- When using USB Camera, the right and bottom edge of the Video In object will keep a distance of 50 pixels away from the window edge to prevent the run-time video image from exceeding the window.
- When using USB Camera, the background color of Video In object is black. If the resolution of the run-time video image is smaller than Video In object, the empty area is colored black. The same resolution of Video In object and USB Camera image is recommended.
- The tested and available USB Cameras are: logitech C170, Logitech C310, Logitech C910, LifeCam VX-2000.
- The OS versions that support USB Camera:

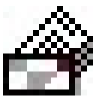
Model	Os version (or later)
eMT3070A	20140116
eMT3105P, eMT3120A, eMT3150A	20140701
XE Series	20140624
mTV	20140807

13.32. SYSTEM MESSAGE

13.32.1. Overview

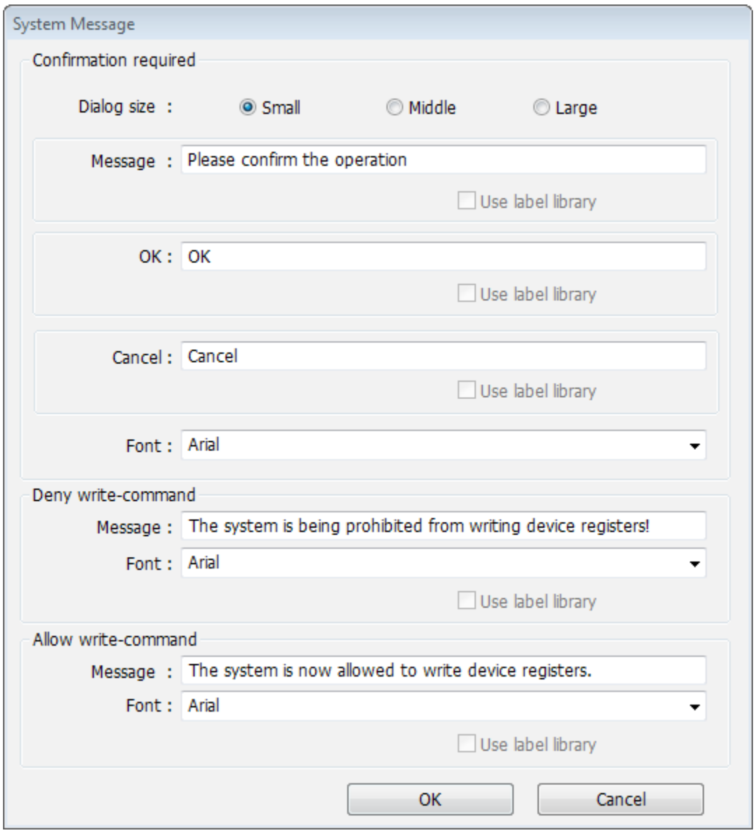
If objects use (Display confirmation request) or (local HMI supports monitor function only) is turned on/off, the corresponding messages configured here will be displayed in pop-up message boxes.

13.32.2. Configuration



Click the System Message icon on the toolbar to open the setting dialog box.

13.32.2.1. System Message



Setting	Description
Dialog size	Select the size for pop-up window and texts.
Confirmation required	If an object uses (Display confirmation request), this message would pop up when the object is used. (Message) shown on confirmation dialog box, and the text label of the 2 buttons, (OK) and (Cancel), can be set. Please use the same font for the labels of (Message), (OK) and (Cancel). Additionally, only when selecting (Label Library) for (Message), the use of Label Library for (OK) and (Cancel) buttons can be enabled.
Deny write-command	Displays when system tag LB-9196 (local HMI supports monitor function only) is turned ON.
Allow write-command	Displays when system tag LB-9196 (local HMI supports monitor function only) is turned OFF.

Note
cMT-SVR does not support adjusting dialog size and using system tag LB-9196.

13.33. RECIPE VIEW

13.33.1. Overview

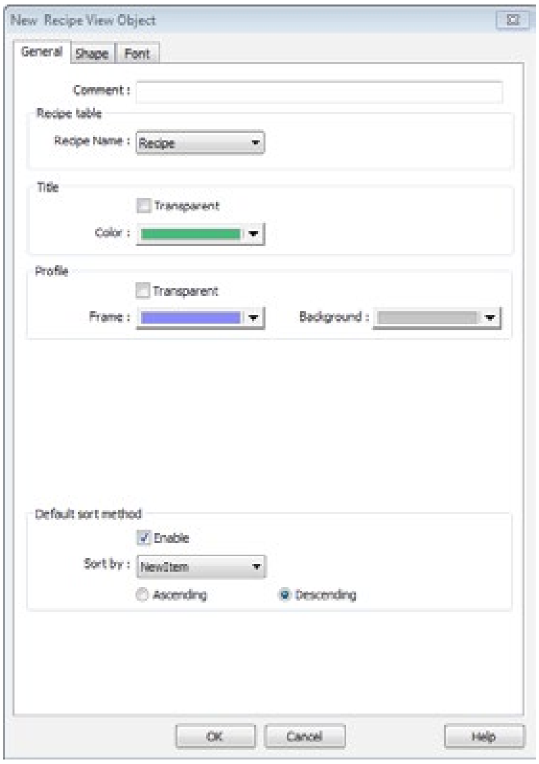
Recipe View object can be used to display a specific recipe. All items and values of the recipe can be viewed by using this object.

13.33.2. Configuration

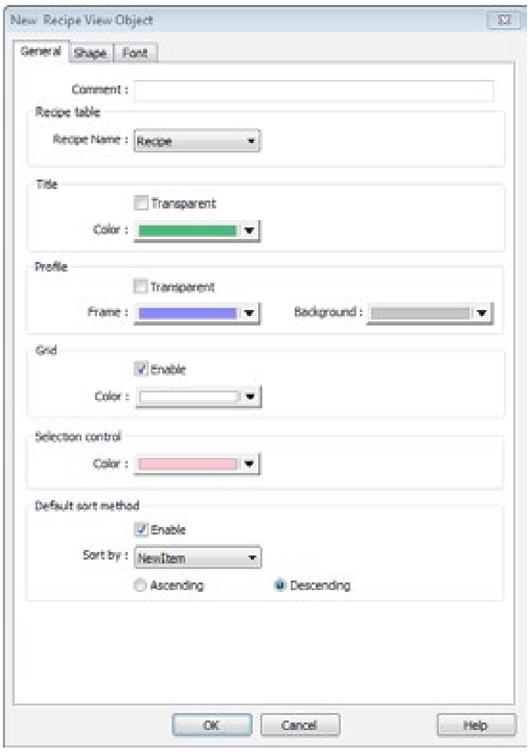
Click the Recipe View icon on the toolbar to open a Recipe View object property dialog box.
Set up the properties, press OK button, and a new Recipe View object will be created.

13.33.2.1. General Tab

- cMT Series



- eMT, iE, XE, mTV Series



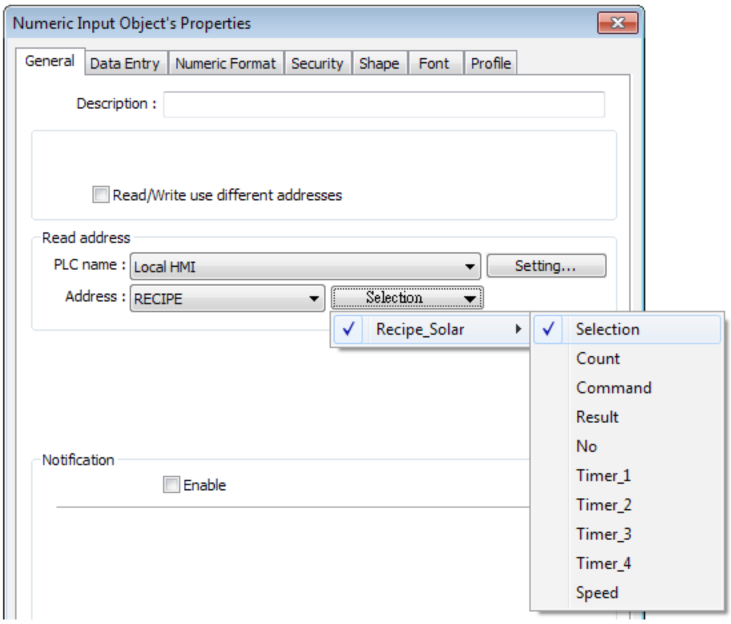
The name of each part of the Recipe View object is shown in the following figure.

Setting	Description
Recipe table	Choose the recipe name or look for other recipes from the drop-down list.
Title	The item name assigned in (System Parameter Setting) » (Recipe). **Transparent** If selected, the title row has no shading; the color selection is not available.
Profile	The frame and background color of the object can be set. **Transparent** Select to hide the background, the color selection is not available.
Grid **(N/A for cMT)**	The dividing lines between columns and rows. **Enable** Select to show the grid.
Selection control **(N/A for cMT)**	Change the shading color of the selected row.
Default sort method	Configure how the records are sorted. (Ascending) and (Descending) can be selected.

Note

There are 4 system registers that can be used to view/update/add/delete recipe database:

- Selection
  Current selection of record in Recipe View object, and it is numbered from zero. If the first record is chosen, the value of Selection will be 0. When the value of Selection is changed, the corresponding values will be updated, such as "No", "Timer_1", "Timer_2", as shown in the following figure.



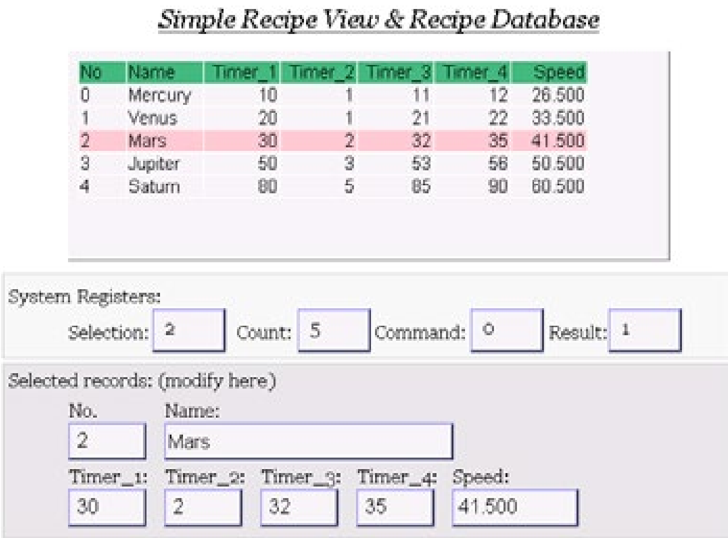
- Count
  Show the number of records in current recipe.
- Command
  Enter certain value will send command to the selected record.
  Enter "1": add a new recipe record to the last row.
  Enter "2": update the selected recipe record.
  Enter "3": delete the selected recipe record.
  Enter "4": delete all recipe records.
- Result
  View the result of executing commands.
  Displays "1": command successfully executed.
  Displays "2": the selected record does not exist.
  Displays "4": unknown command.
  Displays "8": records reach limit (10,000 records), no new records can be added.

Please go to (System Parameter Settings) » (Recipes) tab to create the recipe data before using Recipe View object. See "5 System Parameter Settings".

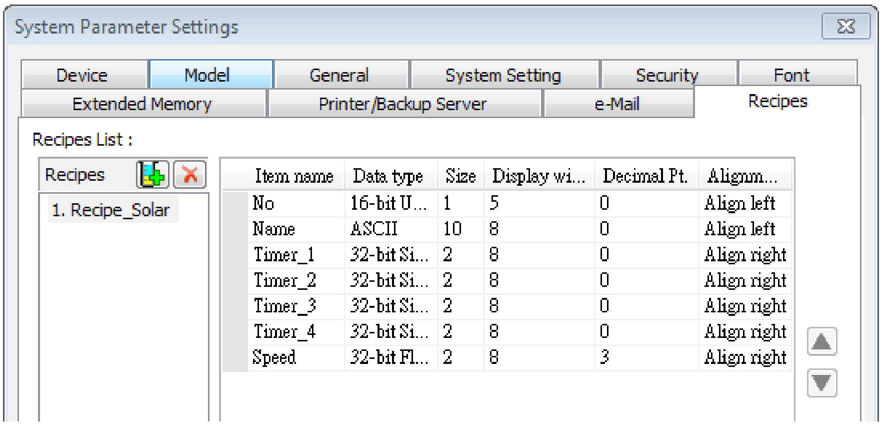
About creating recipes, see "24 Recipe Editor".

Example 1

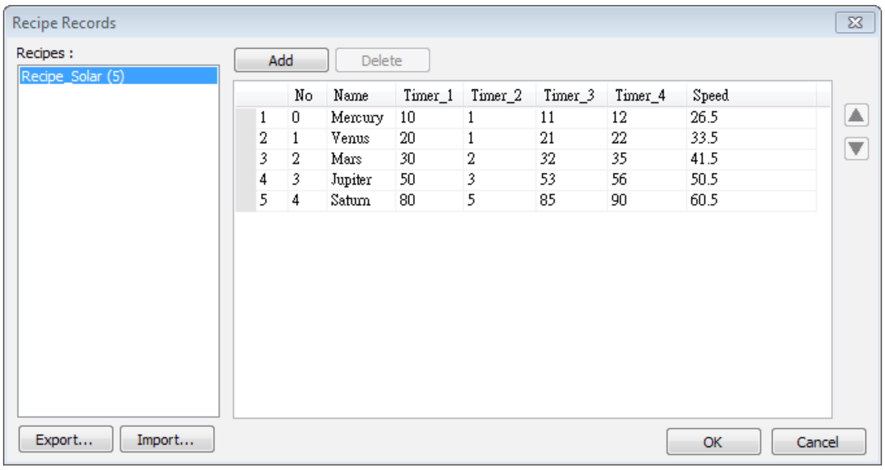
In this example, a recipe database is created to be displayed by Recipe View object. When you select a recipe record on Recipe View object, the value of (Selection) and the corresponding values will change accordingly. When finish designing, you can modify the recipe database by entering a value in (Command).



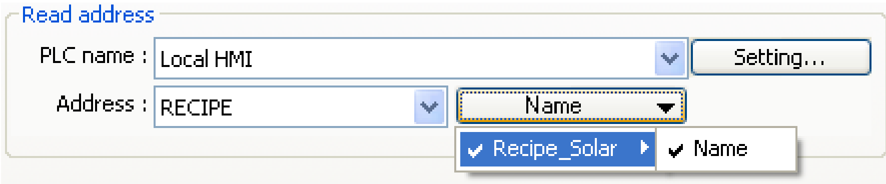
1. Create a recipe as shown in the following figure.



2. Use Recipe Records to create a number of records as shown in the following figure.



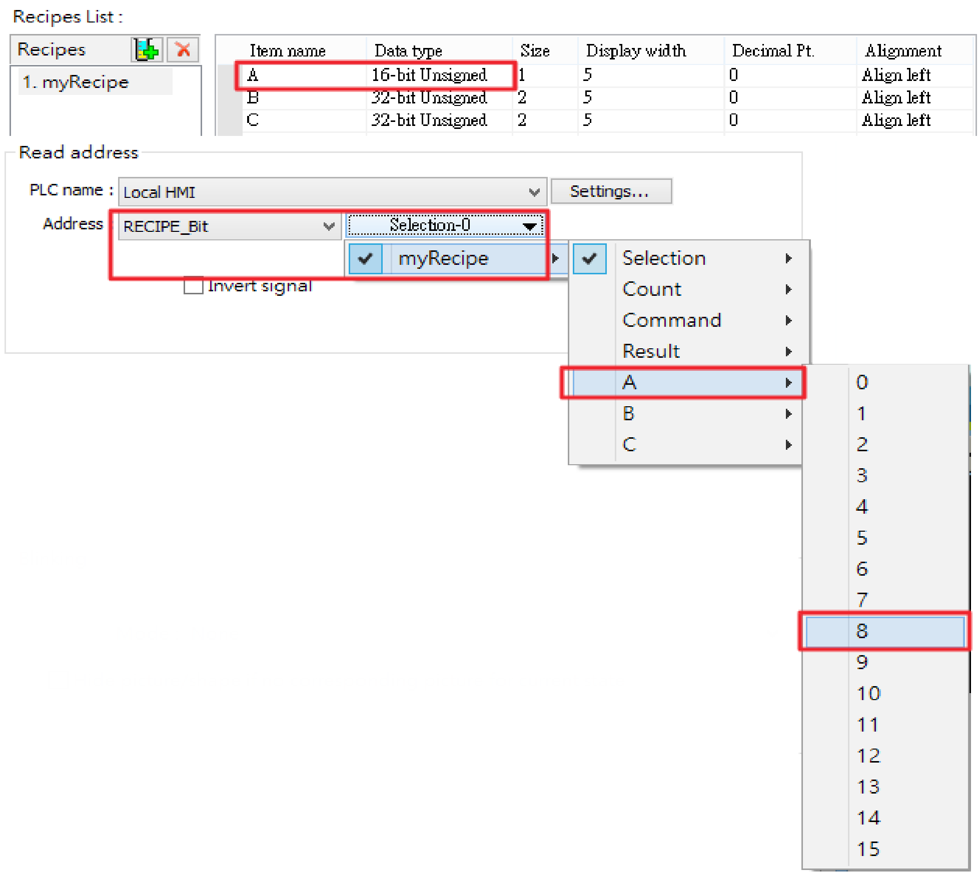
3. Create a Recipe View object and use the recipe database created in the preceding steps.
4. Create 4 Numeric objects using registers “Selection”, “Count”, “Command”, and “Result”.
5. Create corresponding input objects for “No”, “Name”, “Timer_1”, …, “Timer_4”, “Speed”.
   For example, “Name” is an ASCII item with size “10”. Create an ASCII object and set device type to “RECIPE” » “Name”.



6. The project is then completed.
7. As shown above, “Mars” is selected and the corresponding items are also updated. There are 5 records so the “Count” displays “5”. Try selecting different rows of the Recipe View object. Fields “Name”, “Timer_1”, … will change accordingly.
8. Try the following operations:
   - Add:
     - To add current data as a new record, enter “1” in “Command”
   - Update:
     - To update recipe database, enter “2” in “Command”
   - Delete:
     - To delete the selected record, enter “3” in “Command”
   - Sort the items
   - Click the title to change the order

Example 2

In this example, (RECIPE_Bit) can be used to read / write individual bits of Recipe data.
Although BOOL type items cannot be added to Recipe Database, individual bit access of 16bit / 32bit data is possible.
As shown in the following figure, select (RECIPE_Bit) for the read address of Bit object and point to the target item, and then the available Bit selections will be displayed. In this manner, Recipe Database can be used to record, read, and write bit data.



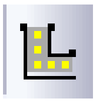
13.34. FLOW BLOCK

13.34.1. Overview

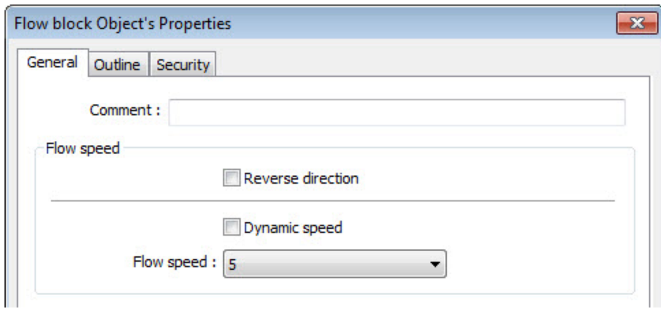
Flow Block object displays the flow status of the blocks in the pipe or the status of the transportation lines. Unlike Moving Shape object which requires a precise measurement between two points when drawing a straight line provided by users, the blocks flow at a fixed interval in a horizontal or vertical straight line.
The features of Flow Block:

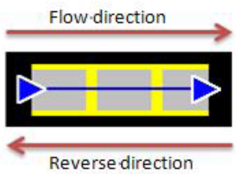
- Each section of the Flow Block must be a horizontal or vertical straight line and the blocks flow at a fixed interval within it
- Dynamic speed and direction adjustment (Speed and direction can be controlled by a designated register
- Security mechanism (Interlock), which hides Flow Block when the status of designated bit is invalid

13.34.2. Configuration

	Click on the Flow Block icon on the toolbar or select (Objects) » (Flow Block) to create object.

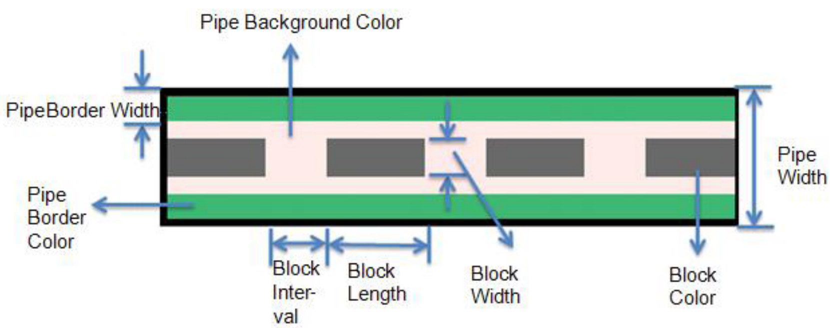
13.34.2.1. General Tab

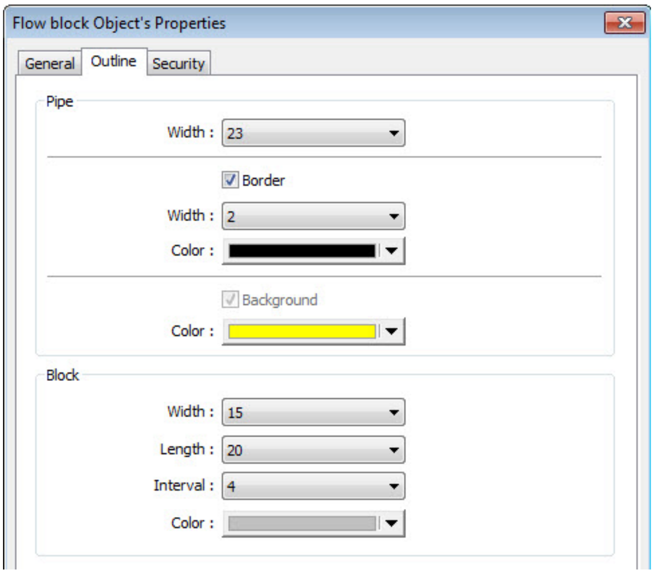


Setting	Description
Reverse direction	The blocks flow in the direction the object is drawn (the blue arrow). If select this check box, the blocks flow in the opposite direction. 
Dynamic speed	**Read address** The direction and speed at which the blocks flow can be controlled by a designated register. The valid rage is -25 to 25. When a negative value is entered, the blocks flow in a reversed direction. **Setting** Displays the address and format of the designated register. (System register), (Index register), and (Tag Library) can be set here.
Flow speed	25 flow speed levels, the valid range is 0 to 25 when (Dynamic speed) is not selected. A larger value indicates a faster speed.

13.34.2.2. Outline Tab

For setting the outline property of Flow Block. The following illustration shows each item.

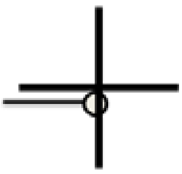




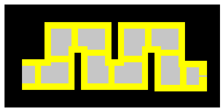
Setting	Description
Pipe	Sets the properties of the pipe within which the blocks flow. The background color, border width and color can be set. When the (Border) check box is selected, the background color must be set.
Block	Sets the properties of blocks. Width, length, interval and color can be set here.

Note

- If both (Reverse direction) and (Dynamic speed) check boxes are selected in (General) tab, when entering a negative value in the designated register of dynamic speed, the blocks flow in the direction the object is drawn.
- To avoid the pipe lines from overlapping when drawing a turn, there is a minimum width planned at each turn. As shown in Fig. 34.1, the sign on the cross cursor defines the minimum width. Fig. 34.2 demonstrates that each turn is drawn in the minimum width.

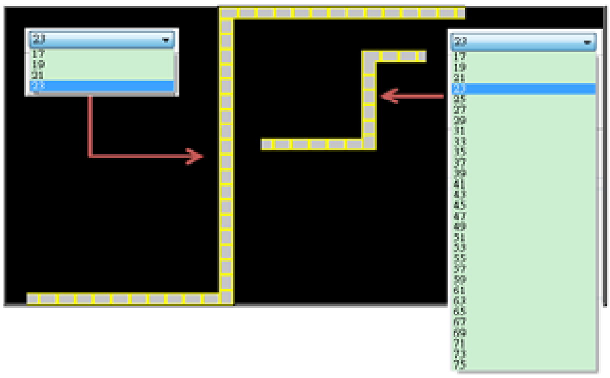


(Fig. 34.1)

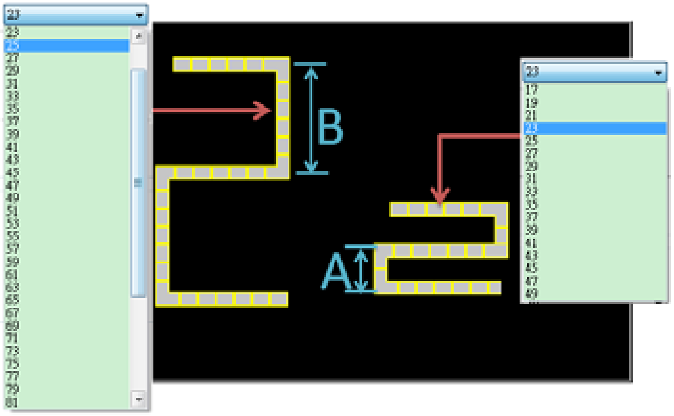


(Fig. 34.2)

The valid range of the length, width, and height of the Flow Block can be adjusted according to the size of the object drawn and the size of the window.
As shown in the following figure, when the size of the Flow Block is larger, the valid range is restricted to prevent the flow block from exceeding the window size. When the size of the object is smaller, the adjustment range will be larger.

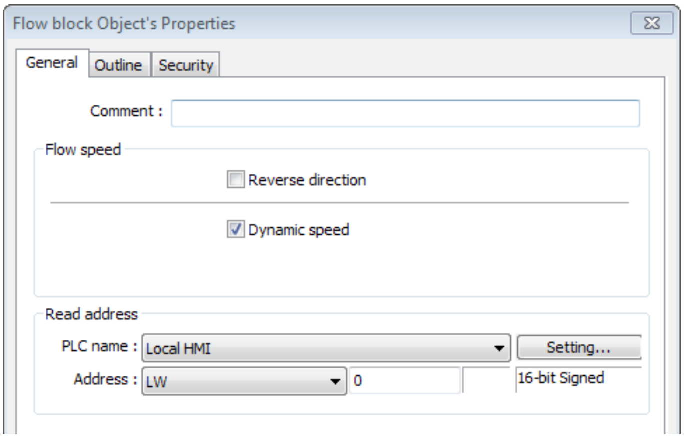


To prevent the flow block from overlapping itself, when the distance between two lines is shorter (Section A), the valid range is restricted. When the distance is longer (Section B), the adjustment range will be larger.

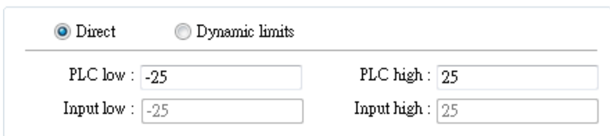


Example 1
The demonstration below shows how to use (Dynamic speed) to control the direction and speed of Flow Block by a designated word register.

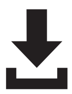
1. Create a Flow Block object and select (Dynamic speed) check box. Set (Address) to LW-0, and set the format to 16-bit Signed.



2. Create a Numeric object, set (Address) to LW-0. The high limit is 25, and the low limit is -25. The format is 16-bit Signed.



3. Execute simulation or download the project to HMI. When entering a positive value in LW-0, the blocks flow in the direction the section is drawn. A larger value indicates a faster speed. When a negative value is entered, the blocks flow in a reversed direction, and the smaller value indicates a faster speed. When 0 is entered, it stops flowing.

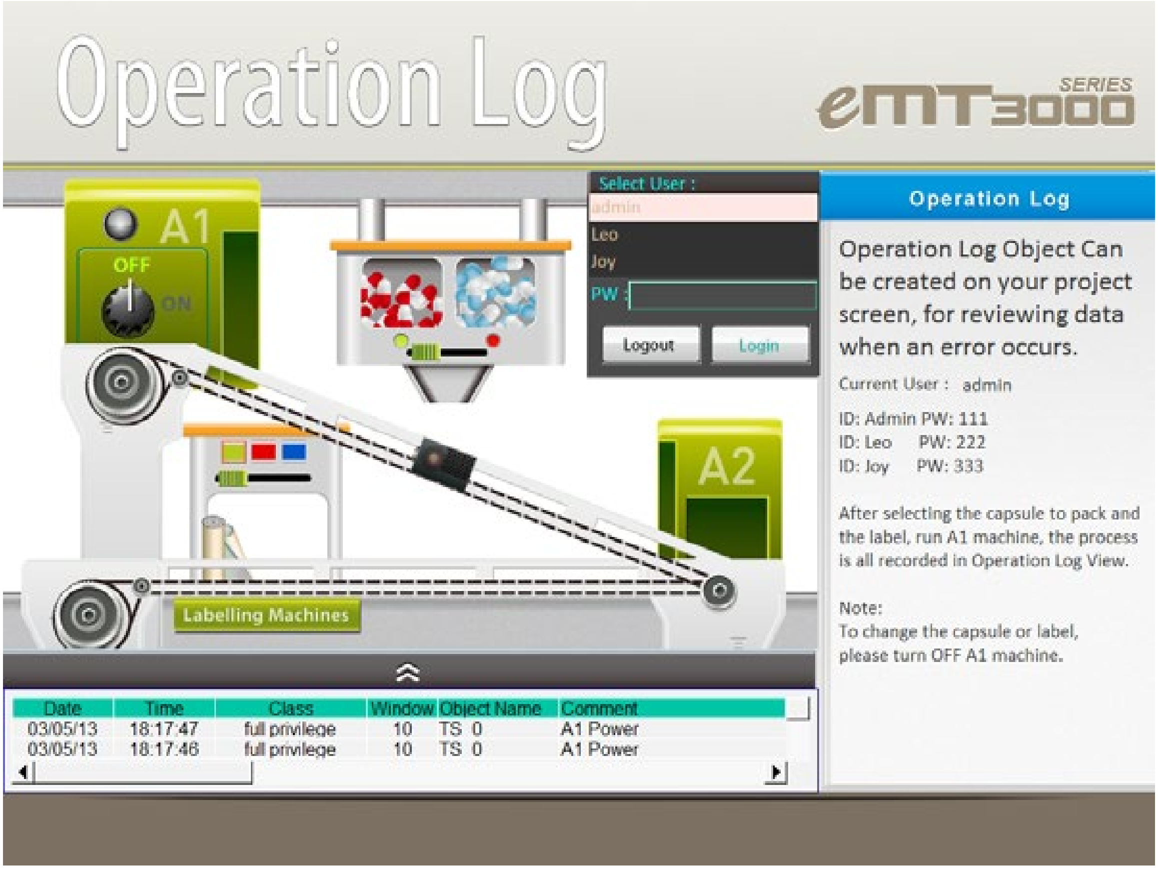
	**Download** Click the icon to download the demo project. Please confirm your internet connection.

13.35. OPERATION LOG

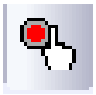
13.35.1. Operation Log Settings

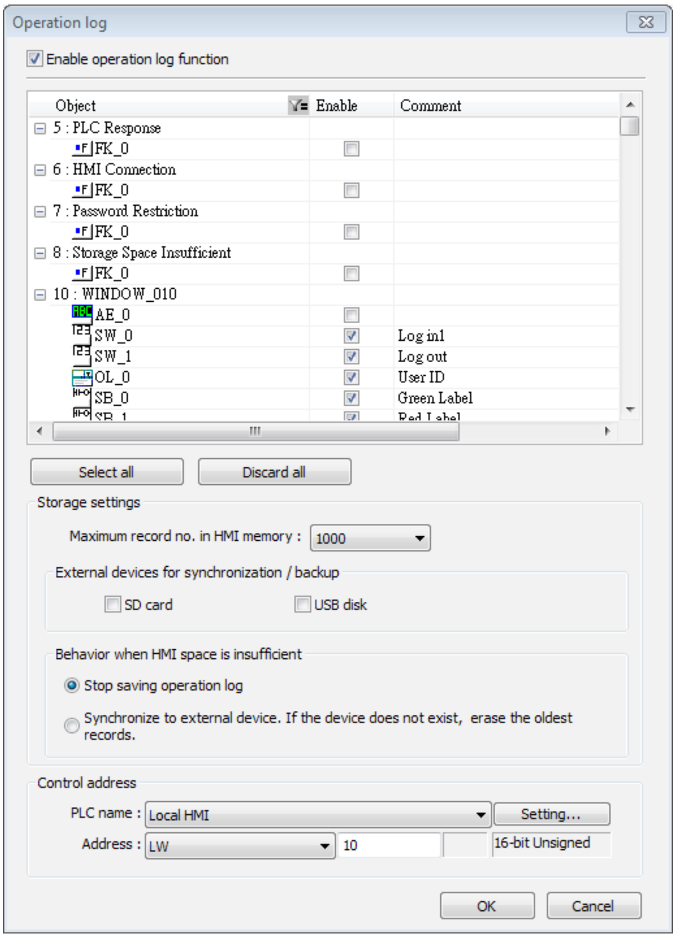
13.35.1.1. Overview

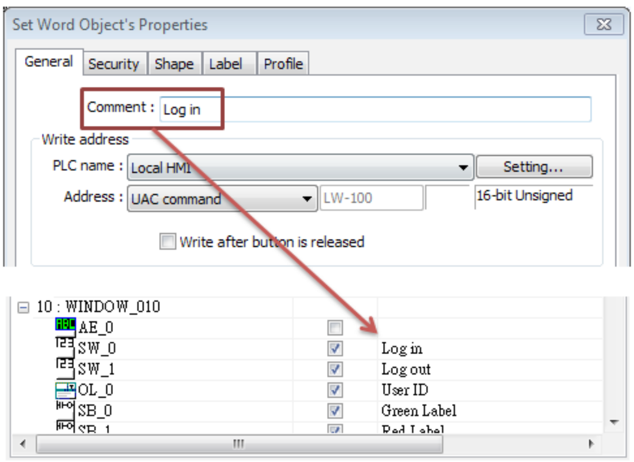
Operation Log records user's operation steps and displays the record in real-time. When an error occurs, use operation log to analyze the problem. The backup tables can be used to review the process in order to resolve the errors.



13.35.1.2. Configuration

	Select the objects to be recorded. Click (Objects) on the main menu, point to (Operation Log), click (Operation Log Settings), and then select (Enable operation log function) check box.



Setting	Description
Object	When Operation Log is enabled, the objects with write function are listed in the setting dialog box sorted by window numbers. (Filter): by clicking  icon, the objects with write function are listed. Users can filter out the objects that need not to be recorded, and the log displays only the selected objects.
Enable	The selected objects are recorded by Operation Log.
Comment	The description of the object as shown in the following figure. 

Select all	Selects all the listed objects. If (Filter) is used, clicking (Select all) only selects the objects in the list.
Discard all	Discards all the selected objects. If (Filter) is used, clicking (Discard all) only discards the objects in the list.
Storage settings	Sets the way the records are stored. **Maximum record no. in HMI memory** Sets the maximum number of records that can be stored in HMI memory. **External devices for synchronization / backup** Stores backup data to SD card or USB disk. **Behavior when HMI space is insufficient** When HMI memory space is insufficient, two options are provided: (Stop saving operation log): stops saving new records in order to keep the earlier records. (Synchronize to external device): stores the operation log to the external device. When the device does not exist, the HMI clears the oldest records in its memory.
Control address	Entering different values in the control address sends corresponding commands to Operation Log and returns the result of executing the command. If control address is LW-n (where n is an arbitrary number), the address that returns the result of executing the commend is LW-n+1. Control address (LW-n): (1): clear all records. (2): copy the records to the USB disk. (3): copy the records to the SD card. (4): copy the records to the USB disk and clear the records in HMI memory. (5): copy the records to the SD card and clear the records in HMI memory. Execution result (LW-n+1): (0): processing. (1): execution succeeded. (2): the device does not exist. (3): the record does not exist. (4): unknown error.

Note

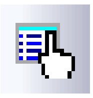
- Operation Log can only record the operation of the objects that are manually triggered. Objects that cannot be manually triggered are not recorded, such as Time Based Data Transfer object.
- When running off-line or on-line simulation, Operation Log is stored under EasyBuilder installation directory: HMI_memory\operationlog\operationlog.db
- Triggering Macro with a Set Bit object generates two records, the triggering of bit and the triggering of Macro.

13.35.2. Operation Log View

13.35.2.1. Overview

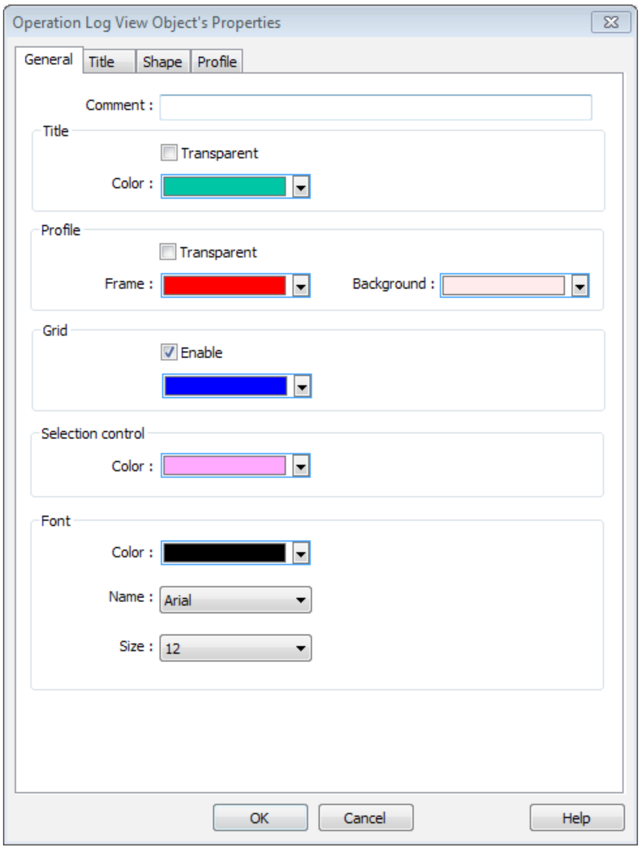
Operation Log View can be used to review the Operation Log.

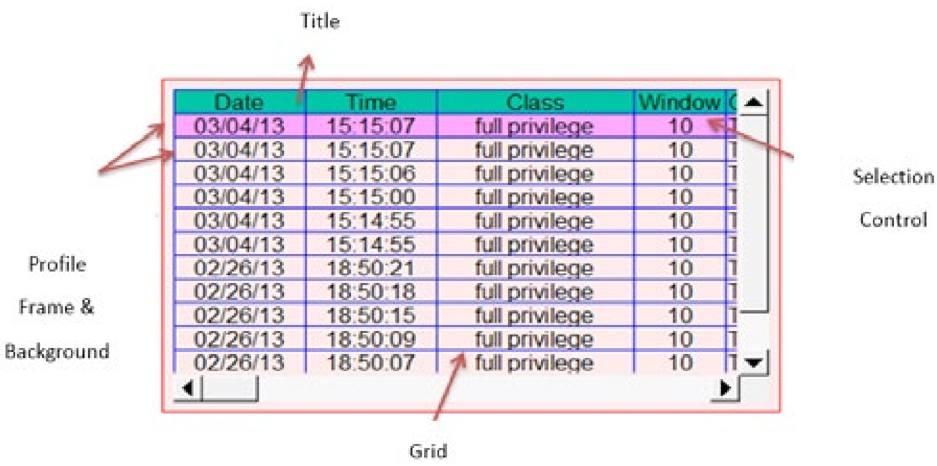
13.35.2.2. Configuration



Before using Operation Log View, please follow the steps described in the preceding part to finish Operation Log Settings. Click (Objects) on the main menu, point to (Operation Log), and then click (Operation Log View).

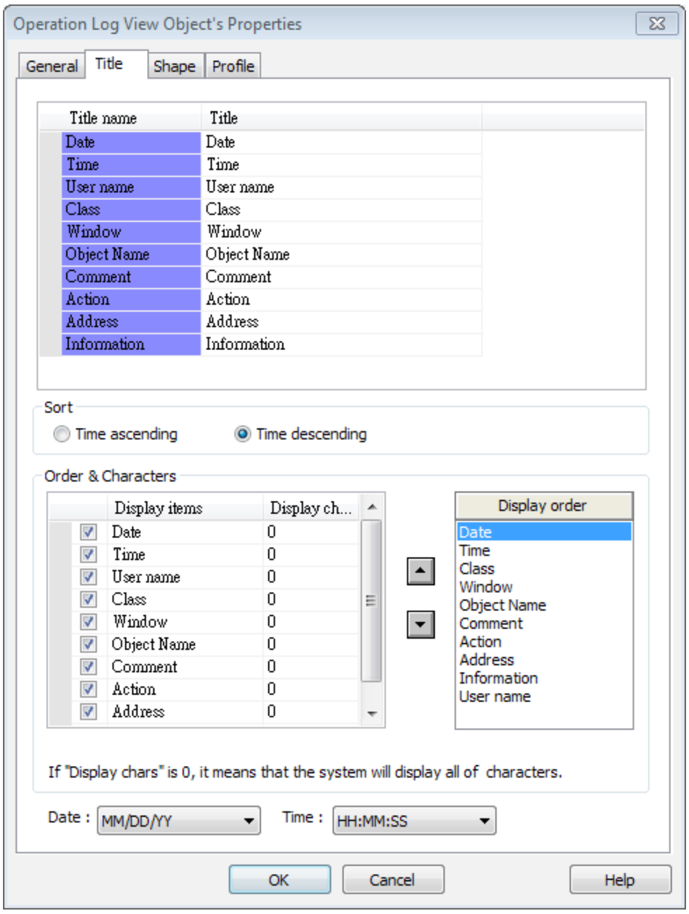
13.35.2.3. General Tab





Setting	Description
Title	Sets the color of the title row. **Transparent:** if selected, the title row will be transparent. The color selection is not available.
Profile	Sets the color of the frame and background of the object. **Transparent:** hides the frame and background. The color selection is not available.
Grid	Sets the color of the dividing lines between the columns and rows. **Enable:** if selected, displays the grid, otherwise, hides the grid.
Selection control	Sets the color of the selected row.
Font	Sets the color, font, and font size of the text displayed in Operation Log View object.

13.35.2.4. Title Tab



Setting	Description
Title	Sets the title displayed in Operation Log View object.
Sort	Sorts the records in time ascending or descending order.
Display order	Sets the order of the displayed item. If (Display chars.) is 0, all characters are displayed.
Date / Time	Sets the format of date and time displayed in Operation Log View object.

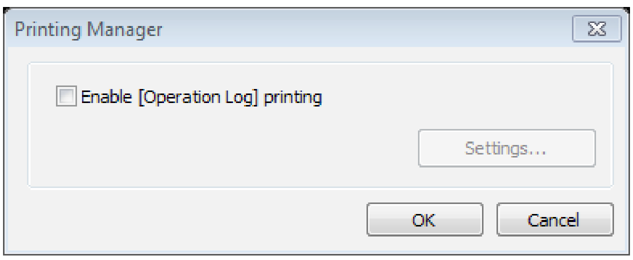
13.35.3. Operation Log Printing

13.35.3.1. Overview

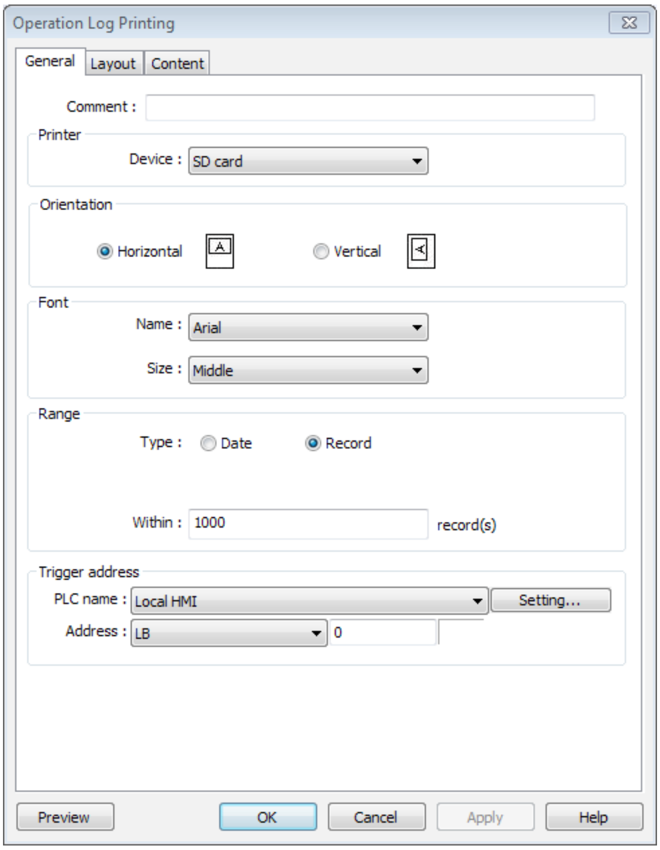
Operation Log Printing can generate an Operation Log sheet by printing out using a printer or by saving as JPEG file into an external device. Before using this function, please go to Operation Log Settings to finish the settings.

13.35.3.2. Configuration

	Select “Enable (Operation Log) printing” check box and click (Settings) button to open the Operation Log Printing dialog box.

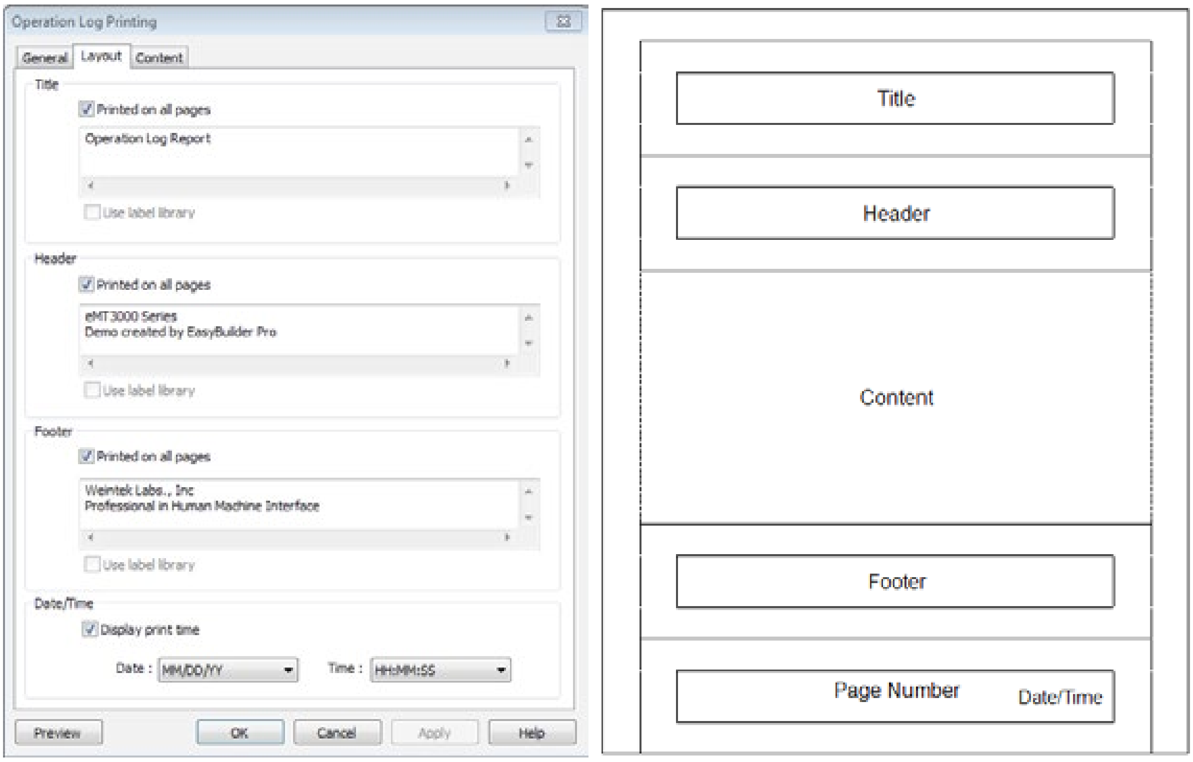


13.35.3.3. General Tab



<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Printer</td><td>Select the device to save the Operation Log sheet. If a printer is selected, the paper size should be A4. If an external device is selected, the Operation Log sheet is saved as a JPEG file. The system generates a folder named “operationlogsheet”, and the files saved in the folder are named “print date_sequence number”. For example, the first JPEG file saved on 2013/05/08 is named 130508_0000 and so on.</td></tr>
<tr><td>Orientation</td><td>Sets the layout of the Operation Log sheet to be horizontal or vertical.</td></tr>
<tr><td>Font</td><td>Sets the font and the font size to of the Operation Log sheet. The following table lists the corresponding size.
<table><tr><th>Size</th><th>Title</th><th>Content</th></tr><tr><td>Large</td><td>20 pt.</td><td>16 pt.</td></tr><tr><td>Middle</td><td>16 pt.</td><td>12 pt.</td></tr><tr><td>Small</td><td>12 pt.</td><td>8 pt.</td></tr></table></td></tr>
<tr><td>Range</td><td>Sets the range of the Operation Log to be included in the sheet.
Date
Sets the range by date, counted from the start day through the number of days entered. The maximum available range is 30 days.
Record
Sets the range by the number of records. The maximum available range is 10000 records.</td></tr>
<tr><td>Trigger address</td><td>Sets the register to control Operation Log Printing. When the register is set ON, it starts printing. When the printing is done, the register is set OFF automatically.</td></tr>
<tr><td>Preview</td><td>Preview the result before generating the Operation Log sheet.</td></tr>
</table>

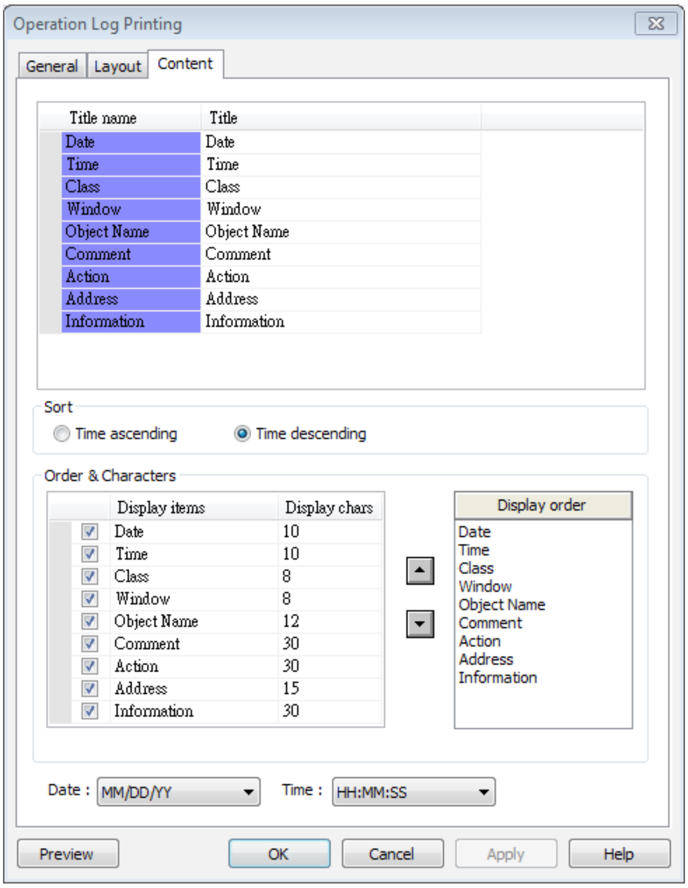
13.35.3.4. Layout Tab



The layout of each part is shown in the above figure.

Setting	Description
Title	Sets the content of the title. The title is limited to one line. **Printed on all pages** If selected, the title is shown on each page; otherwise, the title is shown on the first page.
Header	Sets the content of the header. The header can have 5 lines in maximum. **Printed on all pages** If selected, the header is shown on each page; otherwise, the header is shown on the first page.
Footer	Sets the content of the footer. The footer can have 5 lines in maximum. **Printed on all pages** If selected, the footer is shown on each page; otherwise, the footer is shown on the last page.
Date/Time	If selected, the date/time the in the sheet is shown on the lower-right corner of each page; otherwise, the date/time is not shown.
Page number	Shown on each page.

13.35.3.5. Content Tab



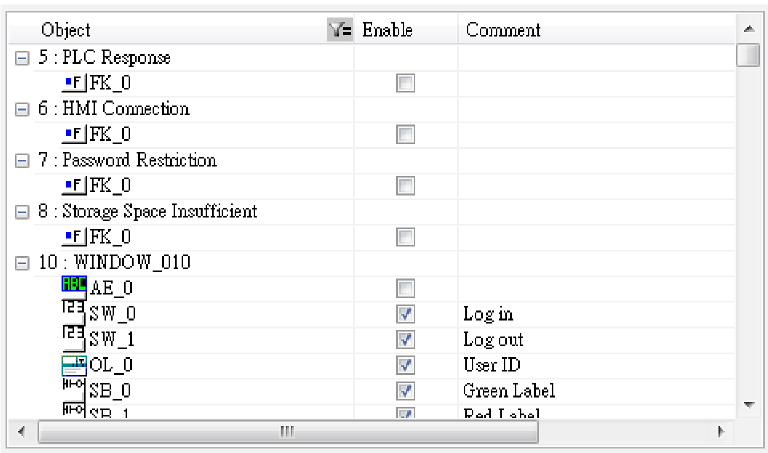
Setting	Description
Title	Sets the title displayed.
Sort	**Time ascending** The latest record is placed at the bottom. **Time descending** The latest record is placed at the top.
Date/Time	Sets the format of date and time displayed.

13.35.3.6. Demonstration

Example 1

The following demonstration explains how to create an Operation Log project.

1. Create a Toggle Switch object and a Numeric object on window number 10.
2. Go to Operation Log Settings; enable the Toggle Switch object and Numeric object on window number 10.



3. Create an Operation Log View object and finish relevant settings.
4. Run off-line simulation; trigger Toggle Switch and Numeric object. Operation Log is displayed by Operation Log View object.

Toggle 5 Numeric Input

Date	Time	Comment	Action	Address	Information
03/05/13	18:52:38	Numeric Input	Set word	Local HMI : LW-10	write 5
03/05/13	18:52:35	Toggle Switch	Toggle	Local HMI : LB-0	bit set OFF
03/05/13	18:52:35	Toggle Switch	Toggle	Local HMI : LB-0	bit set ON
03/05/13	18:52:34	Toggle Switch	Toggle	Local HMI : LB-0	bit set OFF
03/05/13	18:52:34	Toggle Switch	Toggle	Local HMI : LB-0	bit set ON

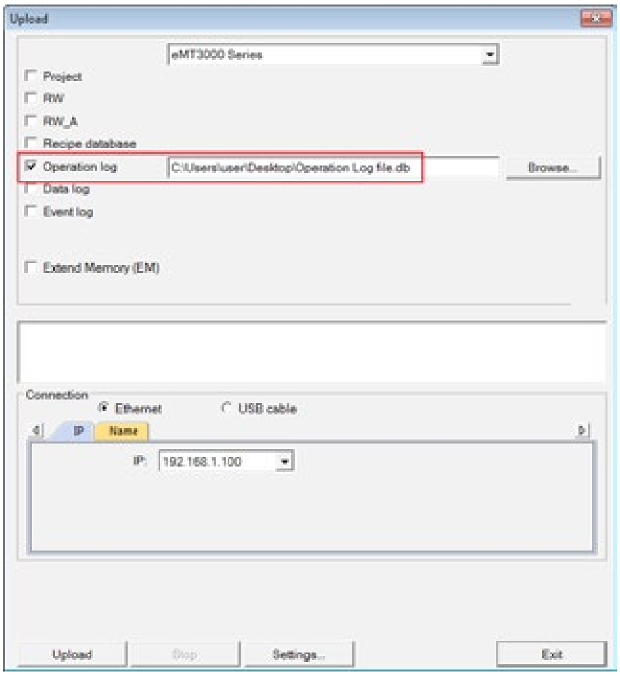
Download
Click the icon to download the demo project. Please confirm your internet connection.

Example 2

Upload Operation Log to PC by using Utility Manager or use Backup object to send the file by email.

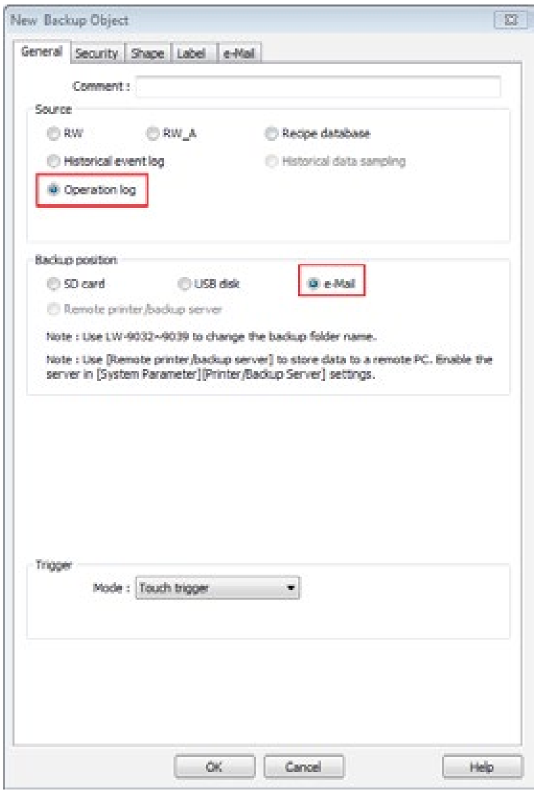
- Upload by Utility Manager

1. Open Utility Manager, click (Upload).
2. Select (Operation log), enter file name and HMI IP, and then click (Upload).



Send the sheet by e-mail

1. Open (System Parameter Settings) » (e-Mail) tab. Set e-mail server and the address of recipient and sender.
2. Create a Backup object, under (Source) select (Operation log), and under (Backup position) select (e-Mail).



For more information about e-Mail settings, see "5 System Parameter Settings".

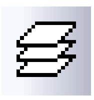
13.36. COMBO BUTTON

13.36.1. Overview

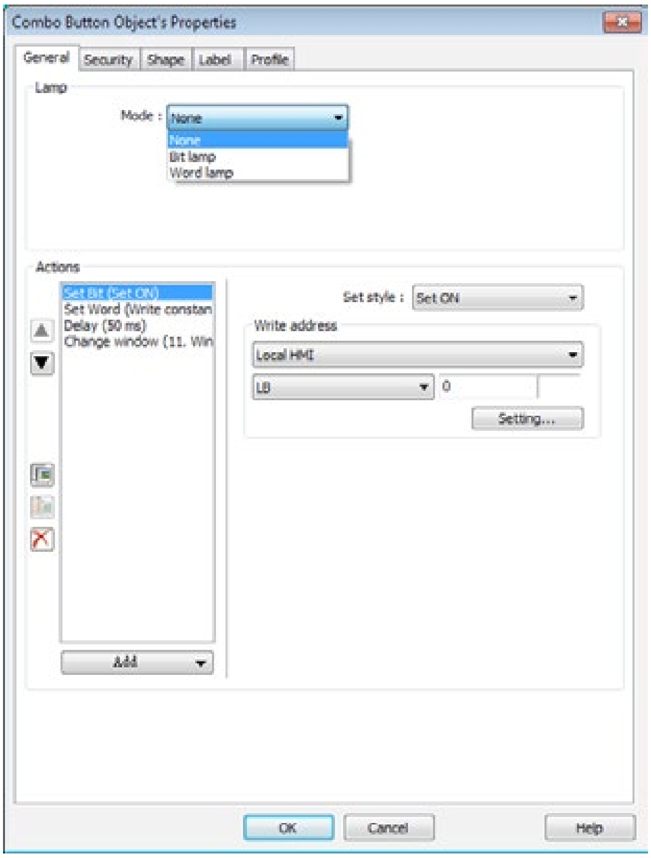
Combo Button can execute multiple commands. The former way was to overlay multiple objects in the same position, and the commands are executed in the order of the layer of the objects. This takes time to test the order when planning the project. Combo Button allows users to easily set multiple commands with one object, and freely adjust the order of executing commands.
The following are the features of Combo Button:

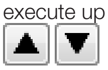
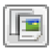
- Executes multiple commands
- Allows adjusting the order of executing multiple commands
- Displays the state in Bit or Word Lamp

13.36.2. Configuration

	Click the Combo Button icon on the toolbar to open a Combo Button object property dialog box. Set up the properties, press OK button, and a new Combo Button object will be created.

13.36.2.1. General Tab



Setting	Description
Lamp	The mode to display the state of a designated bit or word register. **None:** not using lamps to show states. **Bit Lamp** Displays the state of a designated bit address. (Invert Signal) Reverses the display of ON / OFF states. For example, if (Invert signal) check box is selected, when the designated bit is OFF, the object displays ON state. **Word Lamp** Displays the state according to the value of a designated word register. (No. of state): the number of states used by the object. The state is numbered from 0, so the number of states minus 1 will be the state number. If the value in the word register is ≥ (No. of states) defined in Attribute, the highest state will be displayed. If the number of states is set to 8, the valid states will be 0, 1, 2, …, 7. In this case if the word value is 8 or higher, the system will display the state 7 shape.
Actions	There are four types of actions: (Delay), (Set Bit), (Set Word), and (Change window). A combo button can execute up to 20 actions.  Change the order of the actions.  Copy  Paste  Delete Copy, paste, or delete the selected actions.

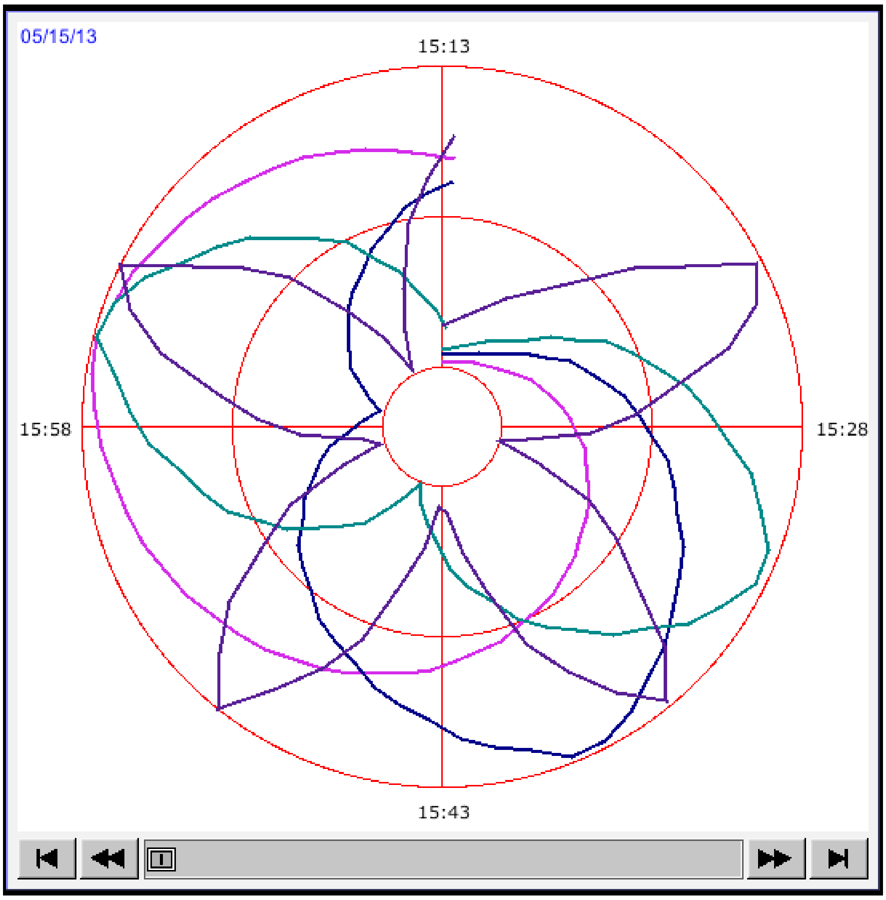
<table>
<tr>
<td>Add</td>
<td>
Delay
Delays the action for a few seconds. A combo button can set one (Delay) action only.

Set Bit
Sets the designated bit ON or OFF.
<table>
<tr><th>Set style</th><th>Description</th></tr>
<tr><td>Set ON</td><td>Set ON the designated bit of the device.</td></tr>
<tr><td>Set OFF</td><td>Set OFF the designated bit of the device.</td></tr>
<tr><td>Toggle</td><td>Alternates the bit state each time pressed.</td></tr>
<tr><td>Momentary</td><td>Holds the bit ON only while button is pressed.</td></tr>
</table>
Set Word
Sets the value in the designated register.
<table>
<tr><th>Set style</th><th>Description</th></tr>
<tr><td>Write Constant Value</td><td>Writes a constant value to the designated register.</td></tr>
<tr><td>JOG+</td><td>Increases value in register by a set amount in (Inc. value) each time when the button is pressed, to the (Upper limit).</td></tr>
<tr><td>JOG-</td><td>Decreases value in register by a set amount in (Dec. value) each time when the button is pressed, to the (Bottom limit).</td></tr>
<tr><td>Dynamic limits</td><td>Sets the Upper / Bottom limit by a designated register.
When Dynamic Address is LW-n, where n is an arbitrary number, set upper limit when using (JOG+), and bottom limit when using (JOG-).</td></tr>
</table>
Change Window
Switch to the designated window. A combo button can only set one (Change Window) action, and this action is always the last one executed.
</td>
</tr>
</table>

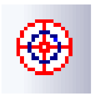
13.37. CIRCULAR TREND DISPLAY

13.37.1. Overview

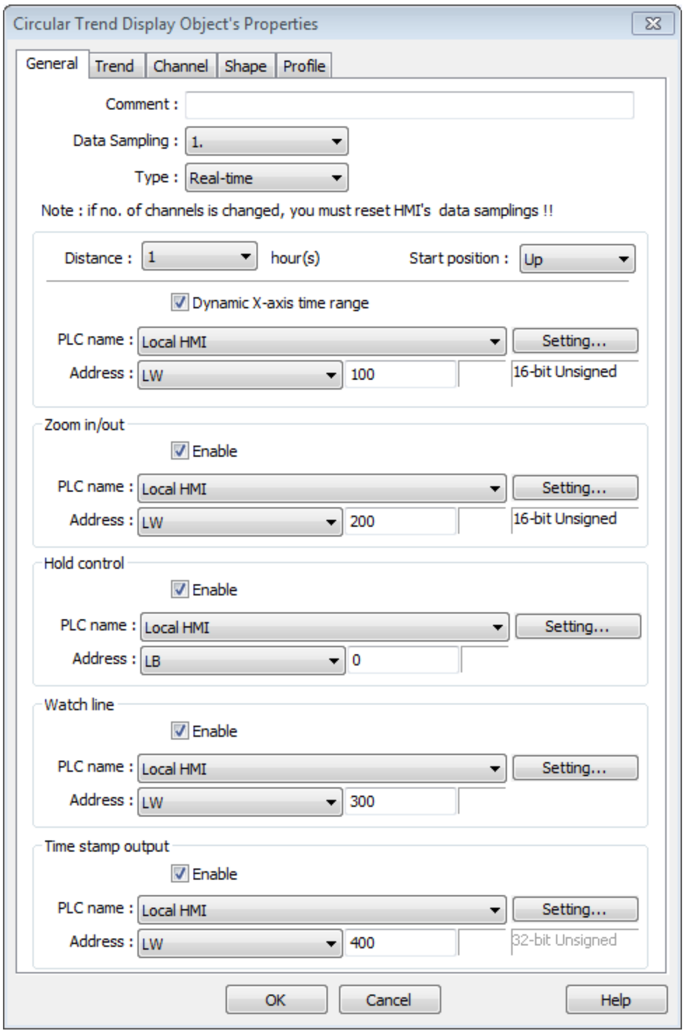
Circular Trend Display object draws the trend curve of Data Sampling in a polar coordinate system, where y-axis represents the radial coordinate and the x-axis represents the angular coordinate. The way to use this object is similar to using Trend Display object.



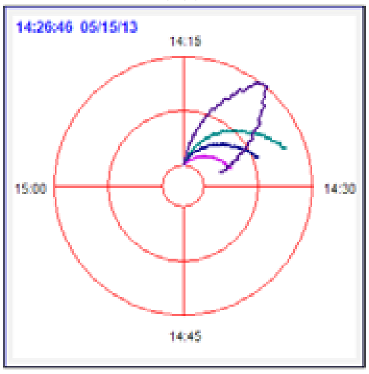
13.37.2. Configureation

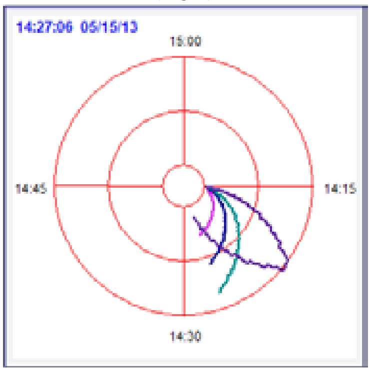
	Click the Circular Trend Display icon on the toolbar to open the property dialog box. Set up the properties, press OK button, and a new Circular Trend Display object will be created.

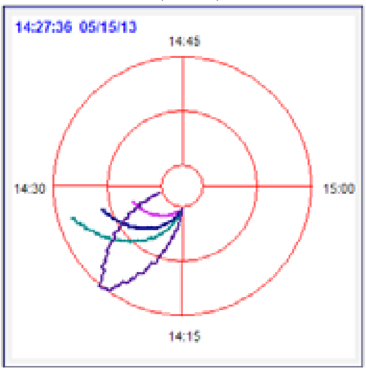
13.37.2.1. General Tab

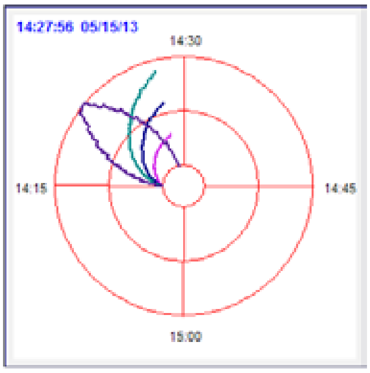


Setting	Description
Data sampling	Selects the data source for drawing the trend curve.
Type	Selects the type of the trend from (Real-time) or (History). **Real-time** In this mode, it displays a fixed number of sampling data from the moment HMI starts to present. The number of sampling data is determined by the (Max. data records (real-time mode)) setting of Data Sampling object. If the number of sampling data exceeds this number, the earlier data will not be displayed. To display earlier data or the data in other days, please select (History) mode. (Hold control) address can be used to pause refreshing the display. This only stops displaying new data in the Circular Trend Display object, and the data is still being sampled by Data Sampling object. **History** In this mode, it displays the sampled data sorted by date. Select the data source from (Data Sampling), and then use (History control) address to view the records of different dates. **Note** If (Show scroll control) check box in Trend Tab is not selected, the earlier data cannot be viewed when exceeding the specified (Distance). For example: set (Distance) to 1 (hour.), then sampling data earlier than one hour is not displayed.

<table>
<tr><td>Refresh data automatically</td><td>If enabled, the window in which the Circular Trend Display object (in history mode) is placed will be refreshed once per second.
▪ The scroll controls can be used to check the refresh status
If button is displayed, the Circular Trend Display will be automatically refreshed
If button is displayed, the Circular Trend Display will stop being refreshed
▪ Scrolling backward and viewing earlier data will disable (Refresh data automatically). The button displayed is at this moment
▪ If (Refresh data automatically) is selected, the display is refreshed when change back to this window, regardless of the use of scroll controls
Example: if (Refresh data automatically) is selected, scrolling to the earlier display stops auto-refresh. At this moment change to another window and then change back, the Circular Trend Display is still refreshed.
If (Refresh data automatically) is not enabled when building the project, to enable it directly on HMI, simply press . Please note that auto-refresh remains disabled after window change.</td></tr>
<tr><td>Distance</td><td>Defines the time length that the circle represents, the unit is hour, the range is 1 to 24 (hours).</td></tr>
<tr><td>Start position</td><td>The position from where to start drawing the trend curve.
(Up)


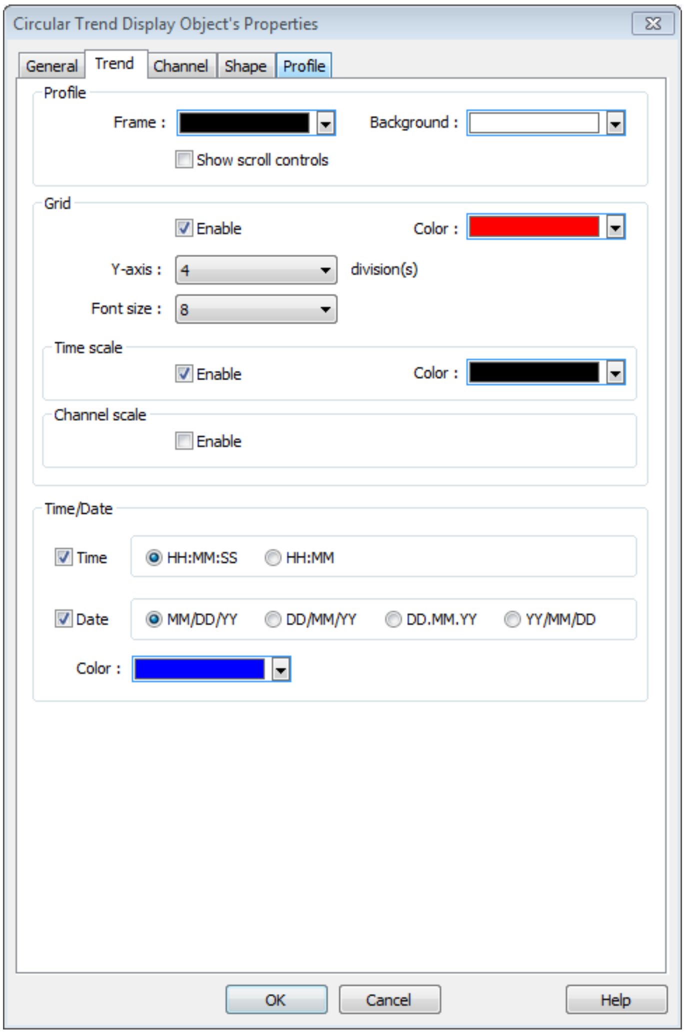
(Right)


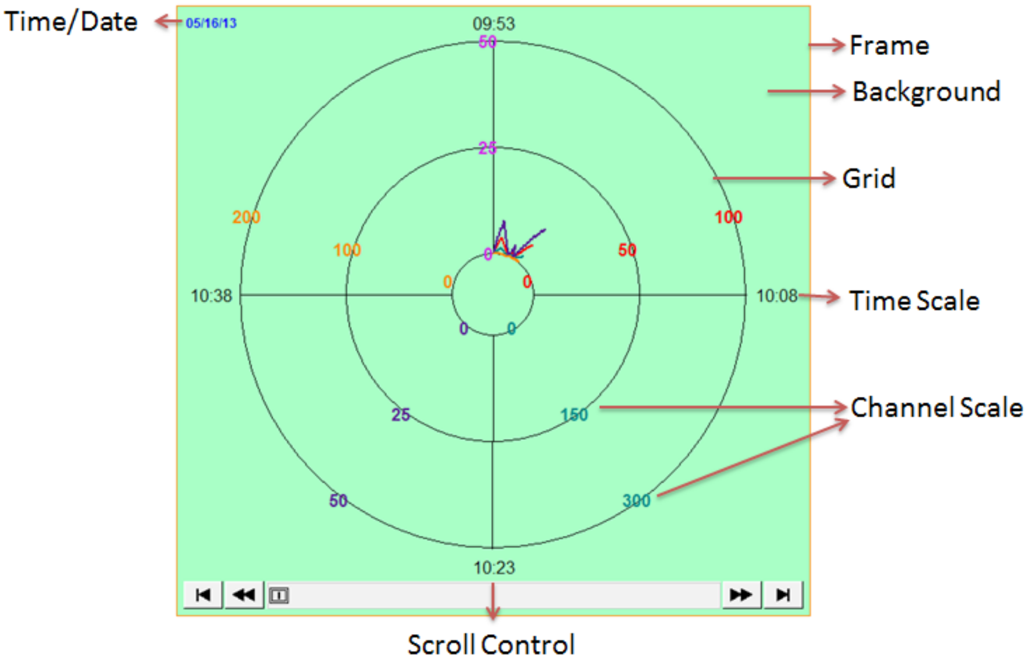
(Down)


(Left)

</td></tr>
<tr><td>Dynamic X-axis time range</td><td>If selected, a word register can be designated to control the time length of Circular Trend Display. The unit is hour. If no value is entered, the distance is set to default.</td></tr>
<tr><td>Zoom in / out</td><td>Enlarge or reduce the size of the object. The maximum size is 10 times larger. If 0 is entered in the designated register, the effect equals entering 1, the object remains the original size.</td></tr>
<tr><td>Hold control</td><td>When the register is set ON, suspend the update of Circular Trend Display. It does not stop the sampling process of Data Sampling object. This setting is available only in Real-time mode.</td></tr>
<tr><td>History control</td><td>History data is sorted by date. The system uses (History control) to select the history data that are created in different dates and then displays it by Circular Trend Display object.
If the value of the designated register in (History control) is 0, the Circular Trend Display object displays the latest record. If the value is 1, the second latest record is displayed and so on. This setting is available only in History mode.
If use with Option List object and select data source as (Dates of historical data), the history data will be sorted by date and displayed in Option List object, see “13.29 Option List”.
In the following example, when history control address is set to LW-0, and there are 4 sampling data: 20061120.dtl, 20061123.dtl, 0061127.dtl, 20061203.dtl. The corresponding data selected by the value in history control address is as the following list.
<table>
<tr><th>Value in LW-0</th><th>The sampling data displayed</th></tr>
<tr><td>0</td><td>20061203.dtl</td></tr>
<tr><td>1</td><td>20061127.dtl</td></tr>
<tr><td>2</td><td>20061123.dtl</td></tr>
<tr><td>3</td><td>20061120.dtl</td></tr>
</table></td></tr>
</table>

<table>
<tr>
<td>Watch line</td>
<td>Displays a watch line when user touches the Circular Trend Display object, and the sampling data at the position of the watch line is output to the designated register. To display sampling data with multiple channels, the system consecutively writes the data of each channel to the designated word register and the following registers. If the data format of each channel is different, the channels are sorted by the data format of its corresponding register.
In the following example, when watch address is set to LW-0, and there are 4 sampling data, the format of each data is: 16-bit Unsigned, 32-bit Unsigned, 32bit Signed, and 16-bit Signed. The corresponding watch address is as the following list.
<table>
<tr><th>Channel</th><th>Data Format</th><th>Data Length</th><th>Watch Address</th></tr>
<tr><td>0</td><td>16-bit Unsigned</td><td>1 Word</td><td>LW-0</td></tr>
<tr><td>1</td><td>32-bit Unsigned</td><td>2 Words</td><td>LW-1</td></tr>
<tr><td>2</td><td>32-bit Signed</td><td>2 Words</td><td>LW-3</td></tr>
<tr><td>3</td><td>16-bit Signed</td><td>1 Word</td><td>LW-5</td></tr>
</table>
</td>
</tr>
<tr>
<td>Time stamp output</td>
<td>If selected, the system will start counting time from the first data sampled, and output the elapsed time counted of the latest data sampled to the register designated in (Time stamp output + 2). When pressing a point on the trend curve, the relative time of the nearest data sample is then output to (Time stamp output address).
Note
The format of the register designated in (Time stamp output) and (Time stamp output + 2) must be 32-bit. (Time stamp output + 2) is only available for Real-time mode while (Time stamp output) is available for Real time mode and History mode.</td>
</tr>
</table>

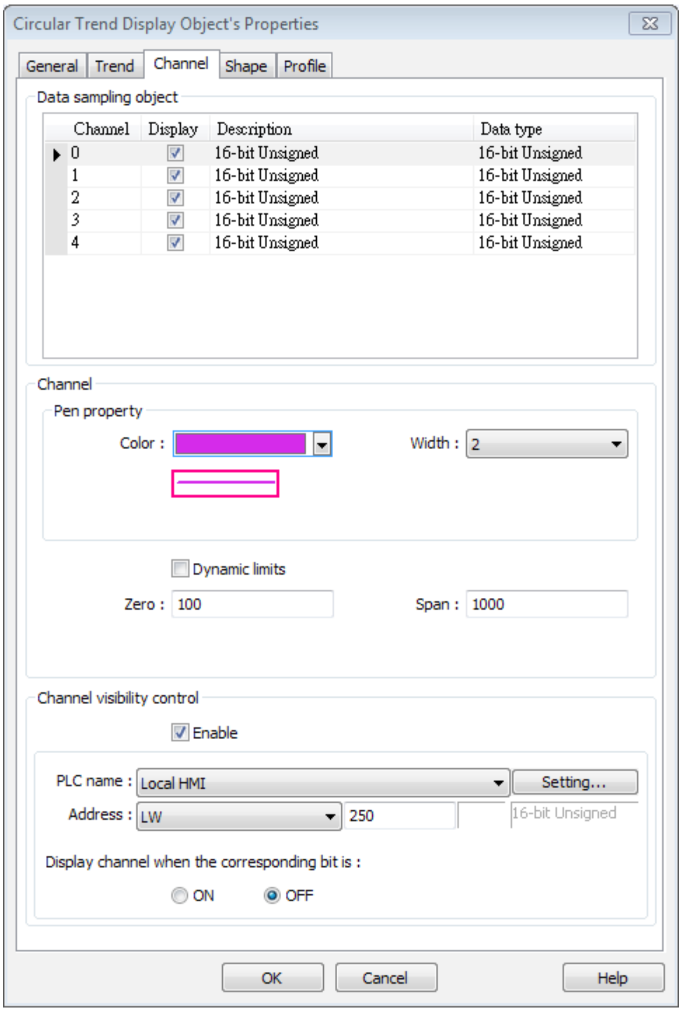
13.37.2.2. Trend Tab





Setting	Description
Profile	**Frame** Sets the color of the frame of the object. **Background** Sets the color of the background of the object. **Show scroll controls** The scroll controls are displayed for scrolling to view the sampling data of other time range. The minimum scrolling unit is determined by the setting of (Distance) in General tab. If (Show scroll controls) check box is not selected, the earlier data cannot be viewed when exceeding the specified (Distance). For example: set (Distance) to 1 (hour.), then sampling data earlier than one hour is not displayed.
Grid	Sets the number and the color of the dividing lines. **Y-axis** Sets the number of divisions on Y axis. **Font size** Sets the size of the font which labels the time scale or channel scale. **Time scale** If enabled, displays the time scale. When the time length is longer than 1 hour, the scale division is 1 hour. When the time length is set to 1 hour, the scale division is 15 minutes. **Channel scale** If enabled, displays the channel scale. The color of the font which labels the channel scale corresponds to the setting of the trend curve of each channel.
Time / Date	**Time** Sets the display format of time. **Date** Sets the display format of date.

13.37.2.3. Channel Tab



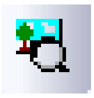
<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Channel</td><td>Sets the style and the color of the trend curve, and the upper and lower limit of data that can be drawn on the trend curve. Up to 8 channels are supported simultaneously.
Not selecting (Dynamic limits)
The upper and lower limits of the data are set by constants.
Selecting (Dynamic limits)
The upper and lower limits are set by the designated register. When the address is LW-n, the corresponding addresses are as the following list.
<table>
<tr><th>Data format</th><th>16-bit</th><th>32-bit</th></tr>
<tr><td>Lower limit</td><td>LW-n</td><td>LW-n</td></tr>
<tr><td>Upper limit</td><td>LW-n+1</td><td>LW-n+2</td></tr>
</table></td></tr>
<tr><td>Channel visibility control</td><td>If (Enable) is selected, the bits of the designated word register will be used to show/hide each channel. First bit (Bit-0) controls the first channel; second bit (Bit-1) controls the second channel, and so on.
Display channel when the corresponding bit is:
If (ON) is selected, when the corresponding bit is OFF, the channel is hidden. If (OFF) is selected, when the corresponding bit is ON, the channel is hidden.
In the following example, the control address of channel visibility is set to LW-0 and each channel shows when the corresponding bit is OFF. If there are 5 channels, the visibility of the channels is as the following list.
<table>
<tr><th>Channel</th><th>Control address</th><th>Bit state</th><th>Display</th></tr>
<tr><td>0</td><td>LW_bit-000</td><td>OFF</td><td>YES</td></tr>
<tr><td>1</td><td>LW_bit-001</td><td>ON</td><td>NO</td></tr>
<tr><td>2</td><td>LW_bit-002</td><td>ON</td><td>NO</td></tr>
<tr><td>3</td><td>LW_bit-003</td><td>OFF</td><td>YES</td></tr>
<tr><td>4</td><td>LW_bit-004</td><td>OFF</td><td>YES</td></tr>
</table></td></tr>
</table>

13.38. PICTURE VIEW

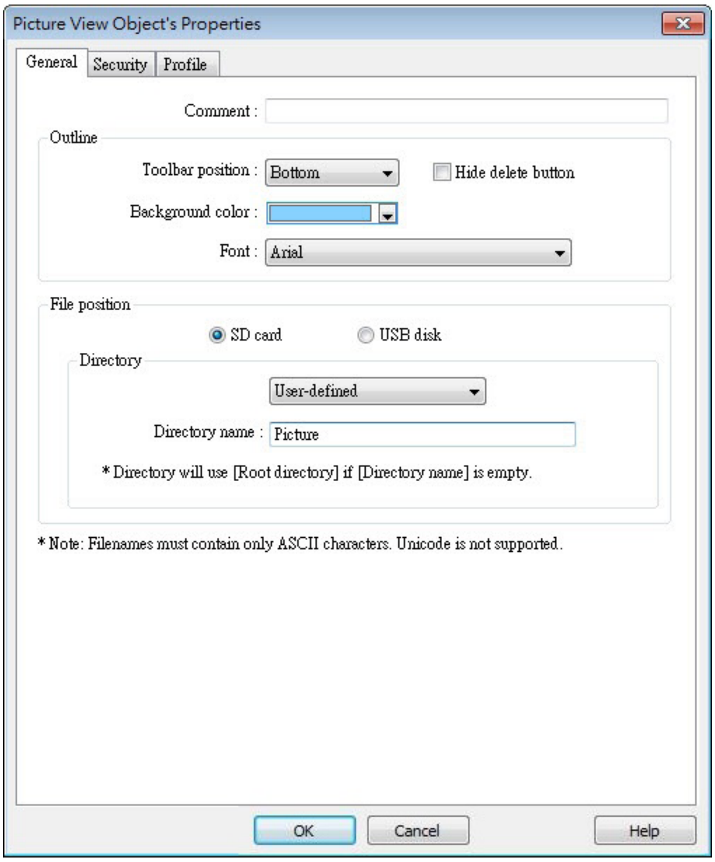
13.38.1. Overview

Picture View object plays slideshow of picture files saved in an external device such as a USB drive or SD card.

13.38.2. Configuration

	Click the Picture View icon on the toolbar to open the property dialog box. Set up the properties, press OK button, and a new Picture View object will be created.

13.38.2.1. General Tab



Setting	Description
Outline	Sets the toolbar position, background color, and text font of the Picture View object. **Hide delete button** If selected, the delete button will not be displayed on the Picture View object toolbar. The delete button is used to delete the picture currently viewed.
File position	Select the file source of the picture files from (SD card) or (USB disk). **Directory** Select the directory where the picture files are saved. If the selected directory is not the (Root directory), please enter the (Directory name). Please note that if the (Directory name) is left empty, the file source will be the root directory.

Note

- The file name must be all in ASCII characters, and the Unicode characters are not supported.
- The supported picture formats are: .jpg, .bmp, .gif, .png.

14. SHAPE LIBRARY AND PICTURE LIBRARY

This chapter explains how to build Shape Library and Picture Library.

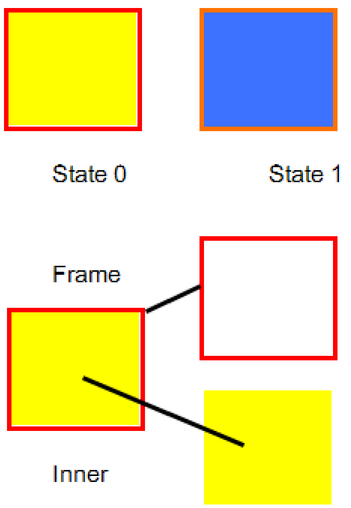
14.1. OVERVIEW

EasyBuilder Pro provides Shape Library and Picture Library for visual effects on objects.
Picture Manager provides two modes: (Project) and (Library). Pictures in (Project) mode will be stored in .emtp project file. Pictures in (Library) mode will be stored in EasyBuilder Pro libraries, or the user-defined directory.
Each Shape or Picture includes up to 256 states. This chapter explains how to build Shape Library and Picture Library.

☞ For more information about using libraries while creating an object, see “9 Object General Properties”.

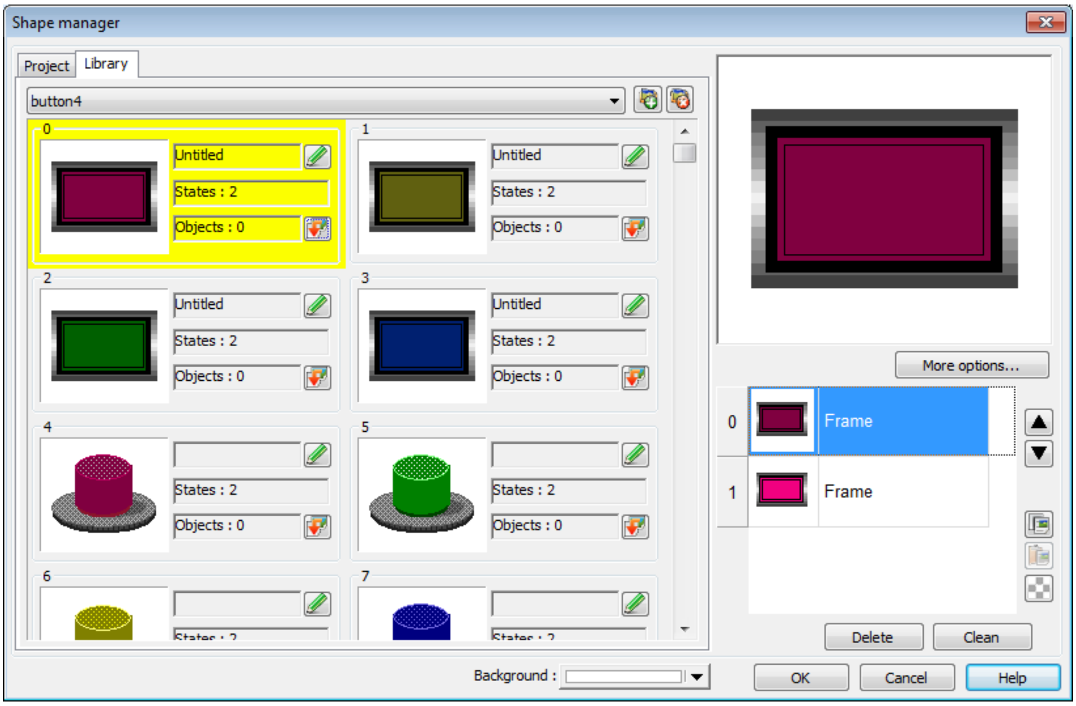
14.2. BUILDING SHAPE LIBRARY

Shapes are vector graphics constructed by lines, curves or polygons. A Shape can have more than one state, and each state includes two parts: frame and inner, as shown in the following figure.



14.2.1. Shape Manager

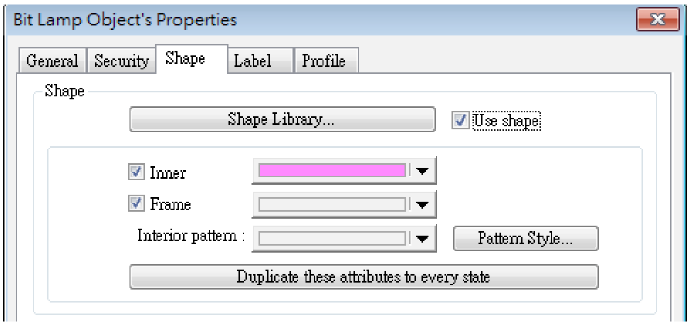
An object can use frame, inner or both. Click (Call up Shape Library), and the (Shape manager) dialog box appears.



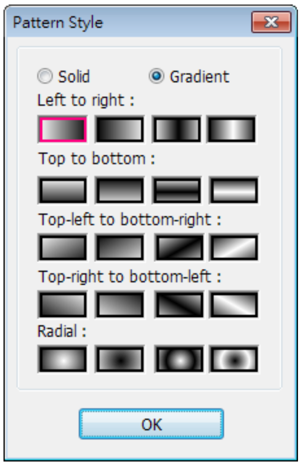
Setting	Description
Project	The Shape edited here will be saved in .emtp. Up to 1,000 Shapes can be added.
Library	The Shape edited here will be saved to the library directory on PC and will not be saved to .emtp project file.
New library	Include existing .plb shape library files or create a new one. To create an empty library, enter a new file name and click (Open). Up to 40 library files can be added.
Unattach library	Exclude current library.
Copy to project	Copy the selected Shape to (Project). Only the shapes that do not belong to the System Libraries can be copied. Shapes in System Frame/System Button/System Lamp/System Pipe cannot be copied.
Background	Select and preview the background color of the Shape. The color is only displayed in (Shape manager) dialog box, and is not displayed when placing the object in the screen.
Place	Add the selected Shape to window. Only available for libraries that are not in the system.
More options	Set the color and style of (Inner), (Frame), and (Pattern).
▲ ▼	Move the Shape to the previous / next state.
Copy	Copy the selected Shape.
Paste	Paste the copied Shape.
Insert transparent state	Insert a blank state after the selected state.
Delete	Delete the selected state of the shape.
Clean	Delete all the states of the selected shape.
OK	Confirm to save the edited Shape.
Cancel	Cancel the editing event.
Help	Open help files.

Note

- The color of (Inner) and (Frame) can be selected in Shape Library. The selection of (Pattern Style) is only available in System Frame / System Button Library.



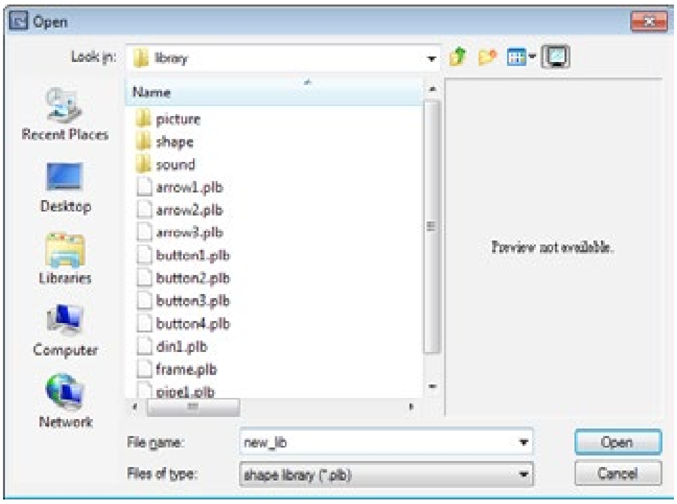
- cMT-SVR Series supports using gradient patterns in (Pattern Style), as follow:



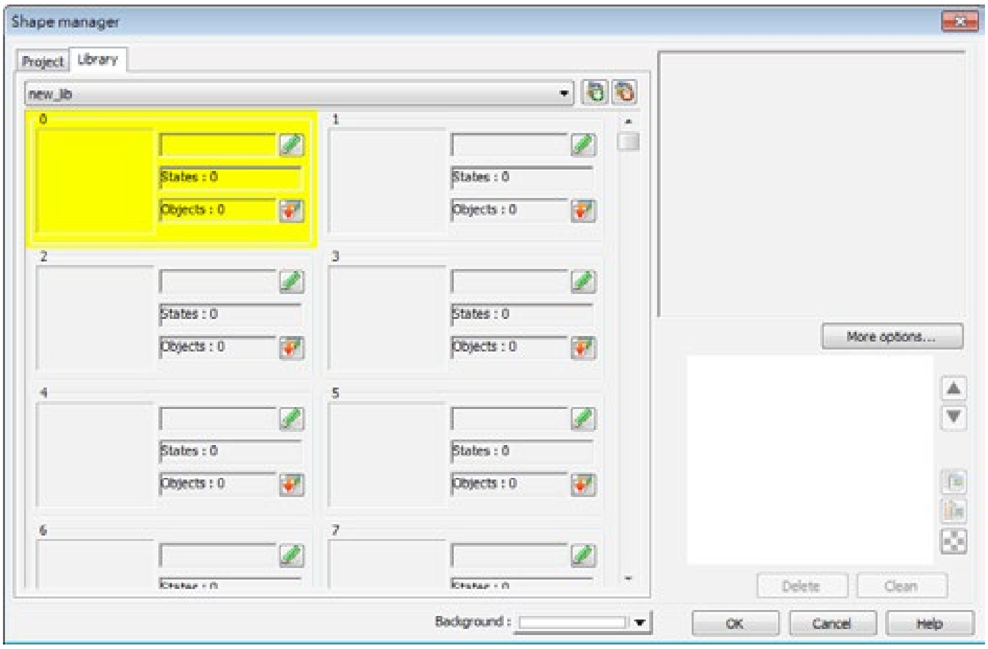
14.2.2. Steps to Build Shape Library

The following explains how to create a new Shape Library and add a Shape with two states into the library.

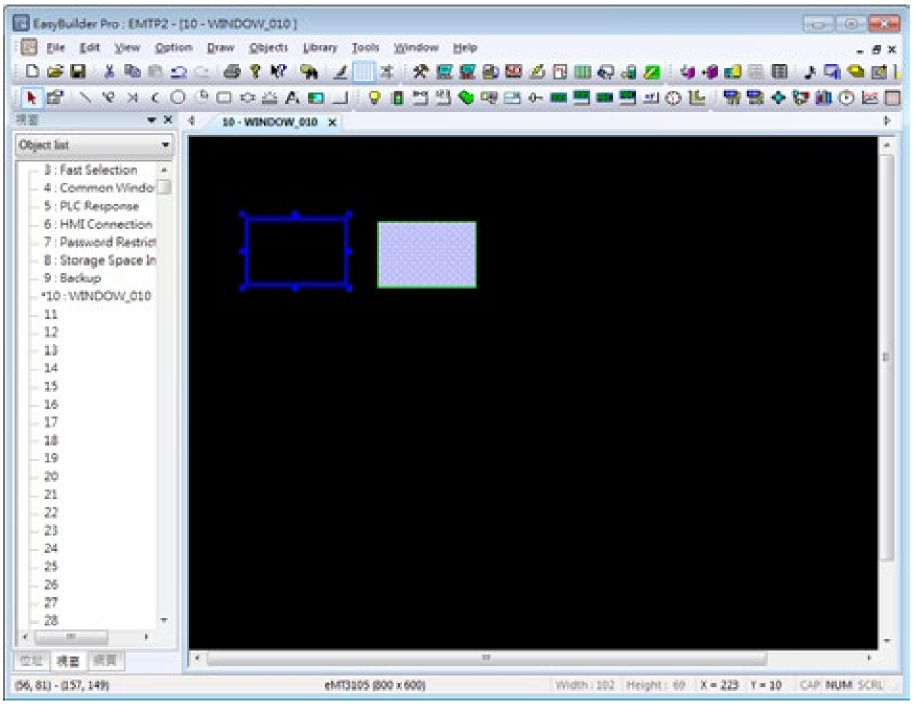
1. Click (New library) and enter the name of the new Shape Library.



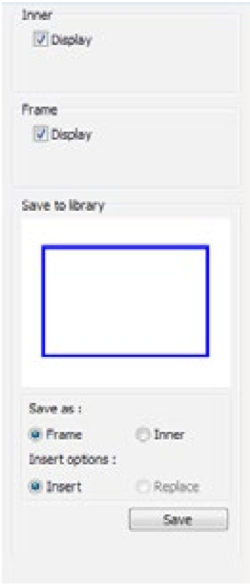
2. Click (Open), a popup dialog appears; click (Yes) to create the file.
3. A new Shape Library (new_lib) is added in (Shape manager). This library is empty as shown in the following figure.



4. Add a state to the selected Shape. First, use the drawing tools to draw a frame and inner in the window and select the frame to add to the Shape Library.

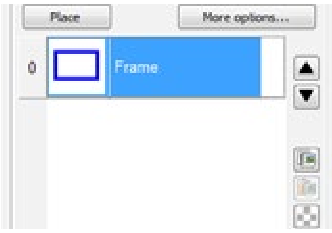


5. Click (Save to Shape Library) button in the toolbar, select (new_lib), and select a number in this library. The selected number is highlighted yellow.
6. Save the Shape as (Frame), select (Insert), and click (Save).

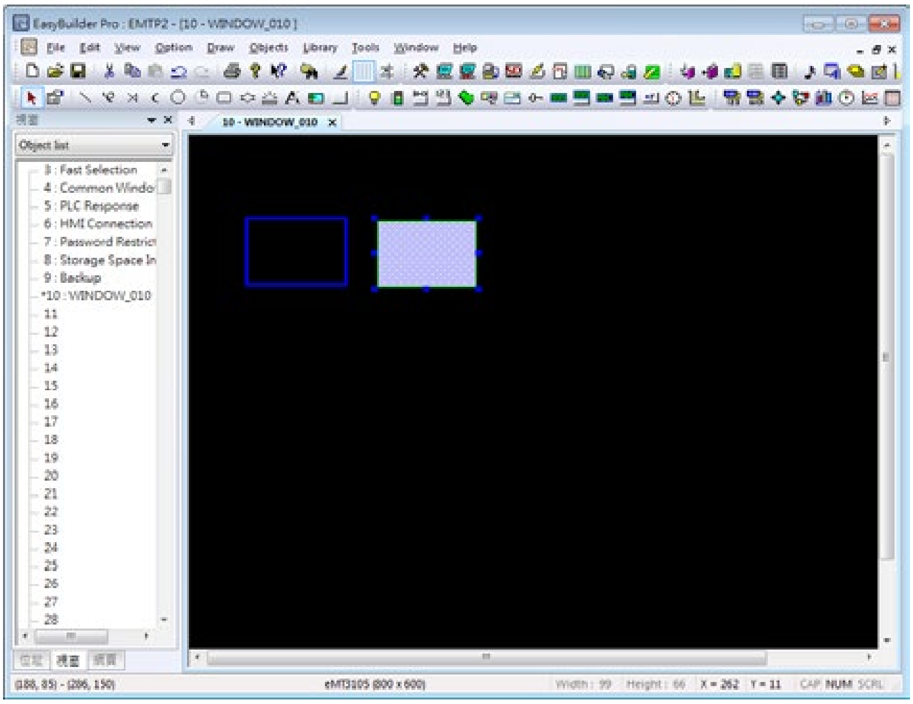


Setting	Description
Inner	Displays the inner of the Shape.
Frame	Displays the frame of the Shape.
Save to library	**Save as Frame** Saves the Shape as a frame. **Save as Inner** Saves the Shape as inner. **Insert** Inserts the Shape to be a new state. **Replace** Replaces a state with this Shape.
Save	Saves the settings above.

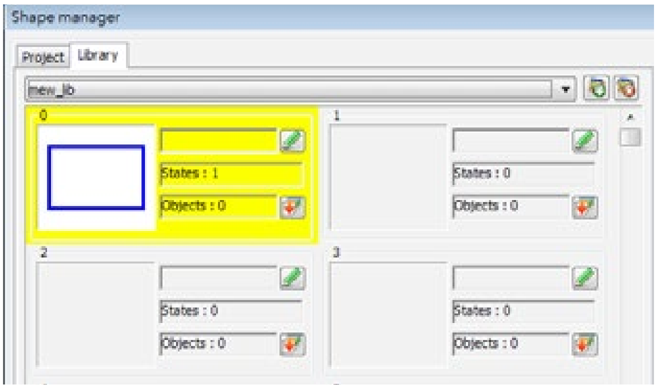
7. The following shows that a state of the Shape is added, and is defined as a frame.



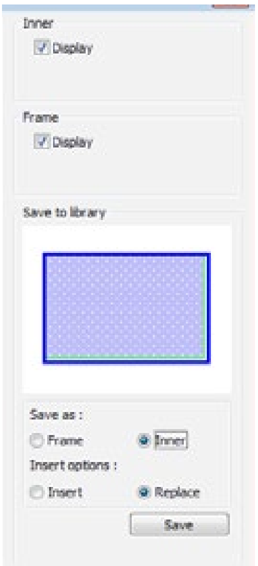
8. Create the shape to be saved as inner. Select the shape drawn in the window.



9. Click (Save to Shape Library) button in the toolbar, select (new_lib), and select the same number as in creating the frame in this library. The selected number is highlighted yellow.

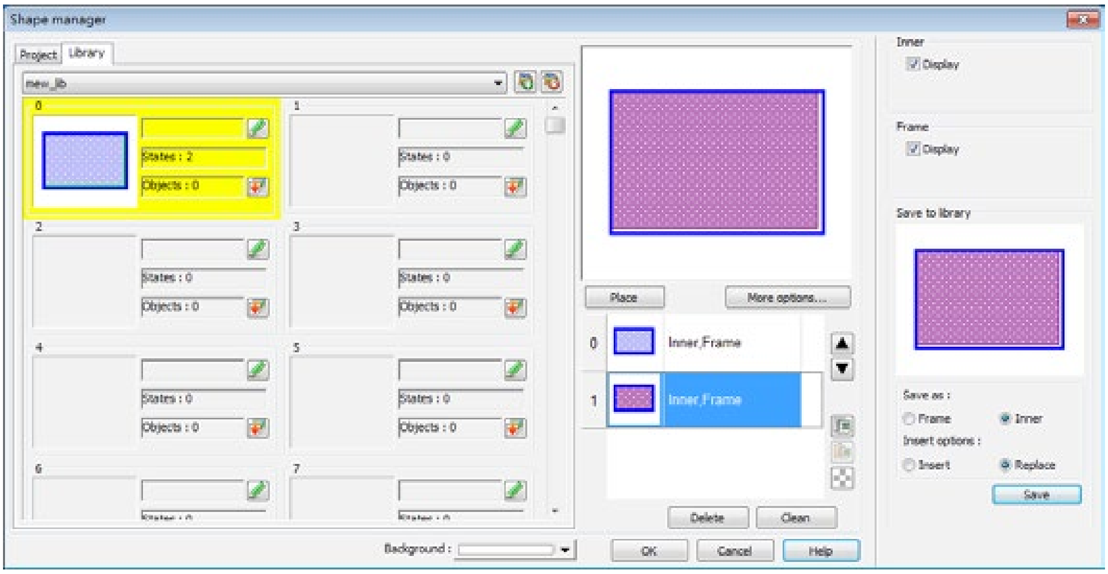


10. Save the Shape as (Inner), select (Replace), and click (Save).



11. A state of a Shape can include (Inner), (Frame), or both. The state 0 of the Shape shown in the following figure includes both frame and inner. Click (OK), the state 0 of the Shape is created.

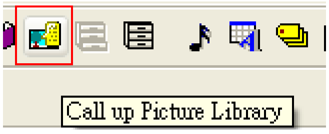
12. Follow the steps of creating state 0 and insert a new state set to state 1 as shown in the following figure. The Shape now has two states, click (OK) to finish setting.

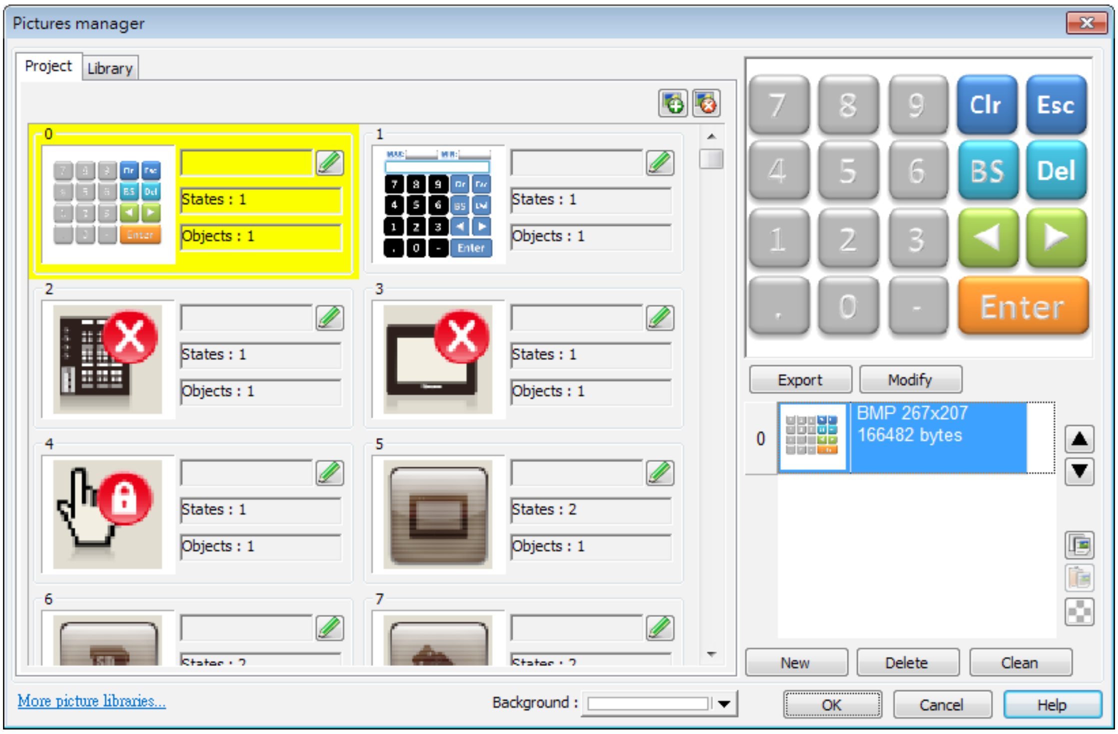


14.3. BUILDING PICTURE LIBRARY

14.3.1. Picture Manager

Click (Call up Picture Library) button in the toolbar and the (Picture manager) dialog box appears.





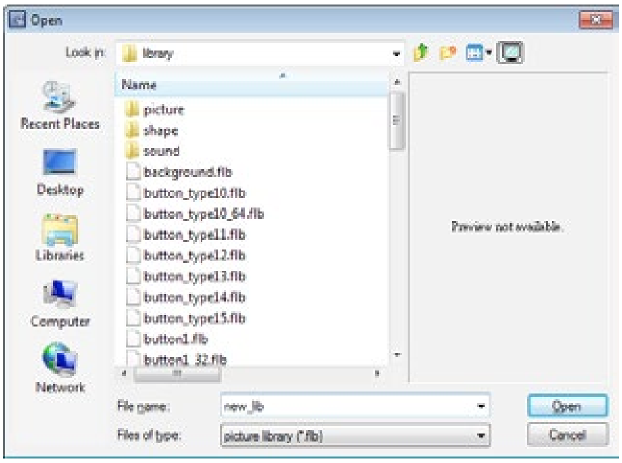
Setting	Description
Project	The Picture edited here will be saved in .emtp. Up to 1,000 Pictures can be added.
Library	The Picture edited here will be saved to the library directory on PC and will not be saved to .emtp project file.
New library	Add the existing .flb picture library files. To add a new library that does not exist, enter a new file name and click (Open), an empty library file is created. Up to 40 library files can be added.
Unattach library	Delete the current library.
Copy to project	Copy the Picture to (Project).
Background	Select the background color of the Picture. The color is only displayed in (Picture manager) dialog box, and is not displayed when placing the object in the screen.
Export	Export the selected Picture.
Modify	Modify the settings of the selected Picture.
	Move the Picture to the previous / next state.
Copy	Copy the selected Picture.
Paste	Paste the copied Picture. The Picture copied to the clipboard can be imported to the library by pasting.
Insert transparent state	Insert a blank state after the selected state.
New	Add a new Picture.
Delete	Delete the selected Picture.
Clean	Delete all the Pictures listed here.
OK	Confirm to save the edited Shape.
Cancel	Cancel the editing event.
Help	Open help files.

	Note The supported picture formats are .bmp, .jpg, .gif, .dpd, .svg and .png. When adding a gif animation file in Picture Library, the loop times of this animated Picture can be set. 

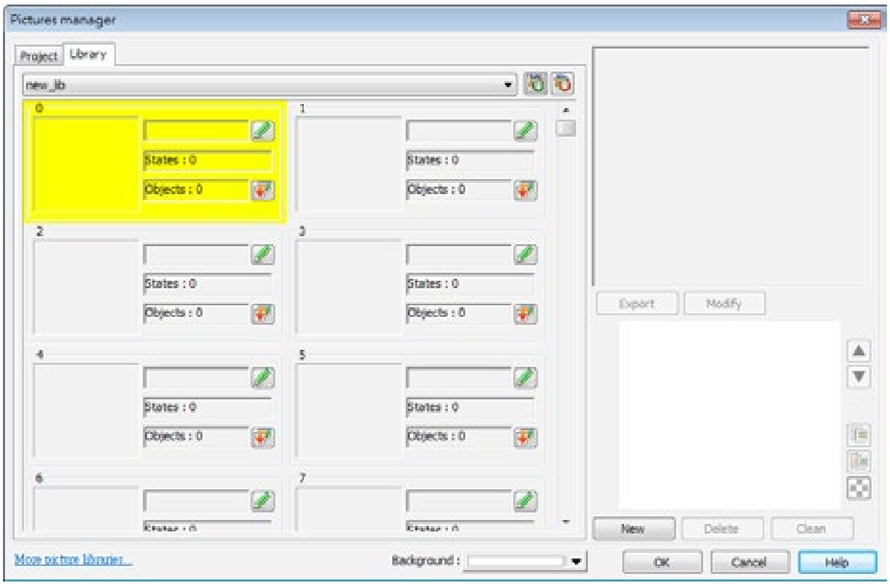
14.3.2. Steps to Build Picture Library

The following example explains how to create a new Picture Library and add a Picture with two states into the library.

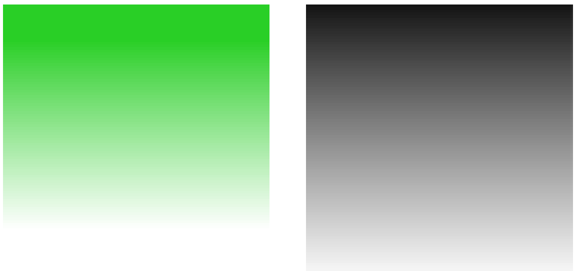
1. Click (New library) and enter the name of the new Picture Library.



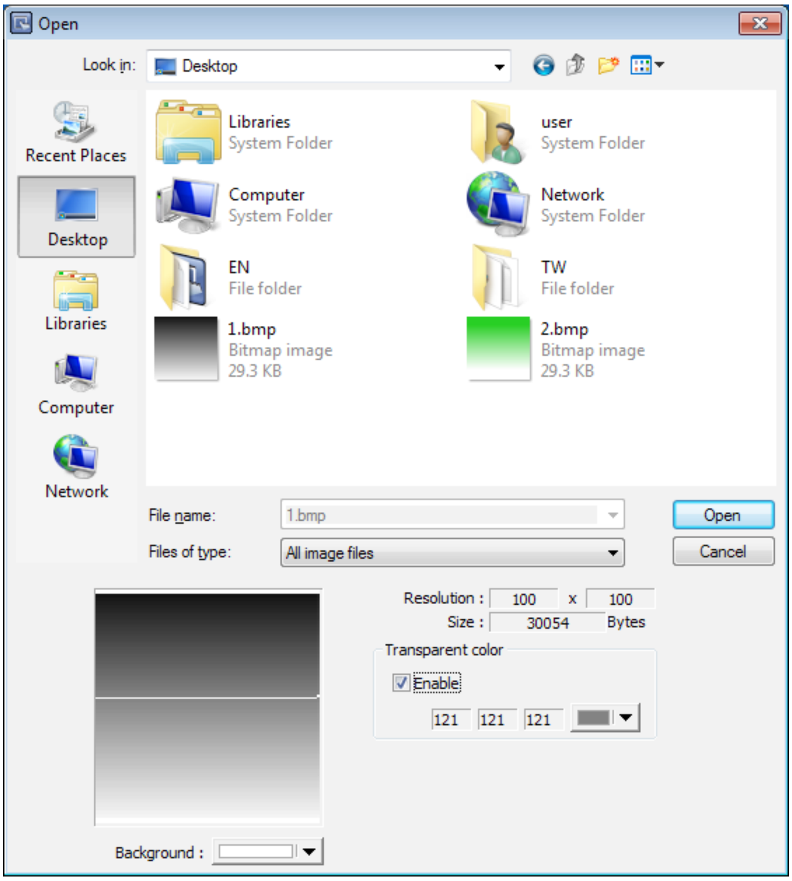
2. Click (Open), a popup dialog appears; click (Yes) to create the file.
3. A new Picture Library (new_lib) is added in (Picture manager). This library is empty as shown in the following figure.



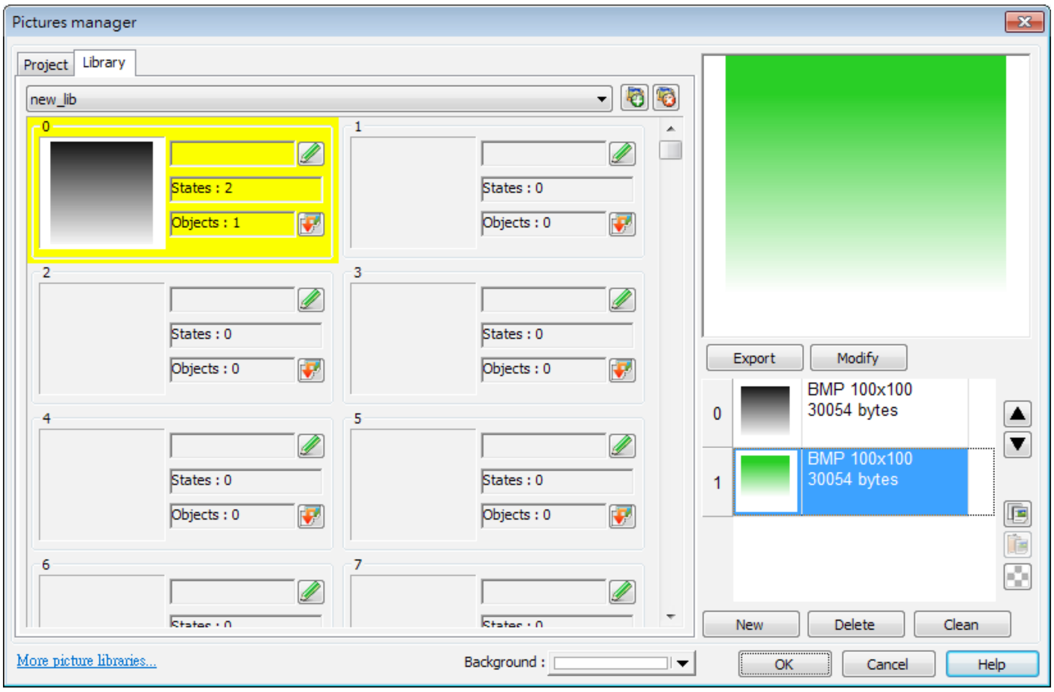
4. Draw the two pictures below to represent state 0 and state 1 respectively.



5. Select (new_lib), and select a number in this library. The selected number is highlighted yellow.
6. Click (New), and select the picture for state 0.
7. When the following dialog box is shown, select (Enable) check box to use transparent color. Set to RGB (121, 121, 121), the corresponding color in the picture below will be transparent. Or, click on a desired area with mouse to be the transparent area, the system will show the RGB of the clicked area automatically.



8. To set transparent color, select (Enable) check box first, and then click an area in the picture. The RGB value of that area is shown and it will be transparent. The displayed picture is shown as in the preceding figure.
9. The Picture of state 0 is created. Follow the steps of creating state 0 to create state 1 by clicking (New) as shown in the following figure.



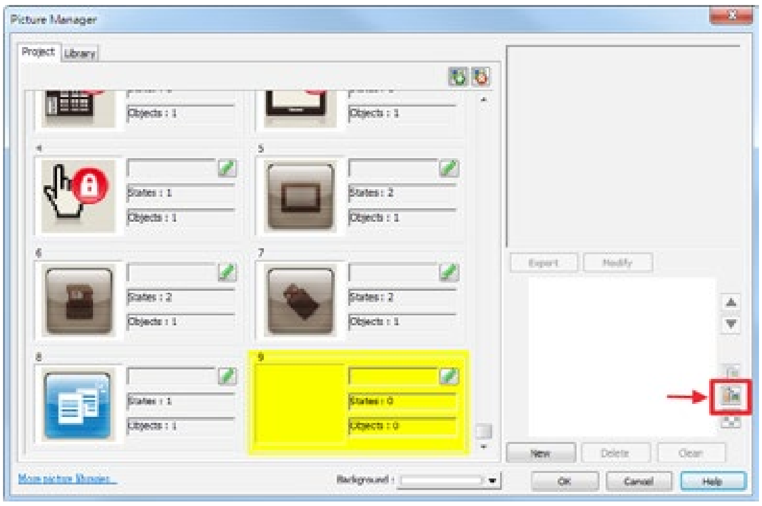
10. When finished, a complete Picture is created, click (OK). In (Picture manager) dialog box it shows that the newly added Picture Number 0 is a bitmap picture with two states.

14.3.3. Steps to Import Picture by Pasting

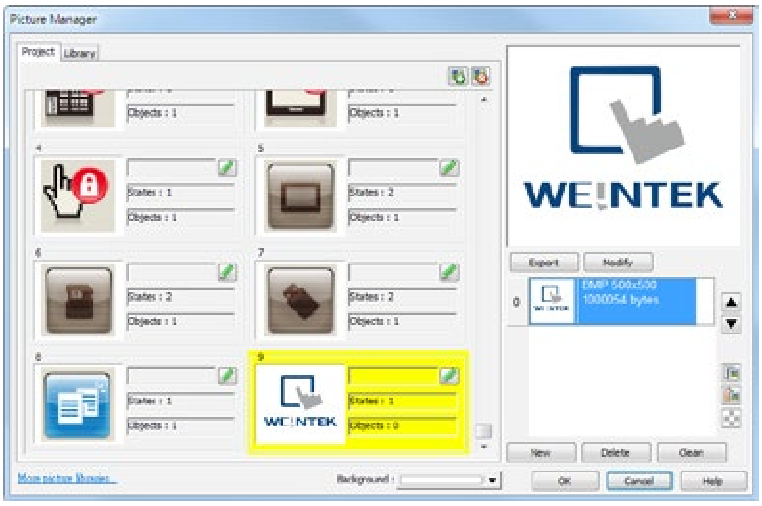
The following example explains how to import a Picture into the library by pasting the picture from the clipboard.

1. Copy the following picture to the clipboard.

2. Click the Paste icon on the right side.



3. The Pictur can be easily imported to the library.



	Note Transparent color can only be set for .bmp, .dpd, and .jpg picture files.

15. LABEL TAG LIBRARY AND MULTI-LANGUAGE

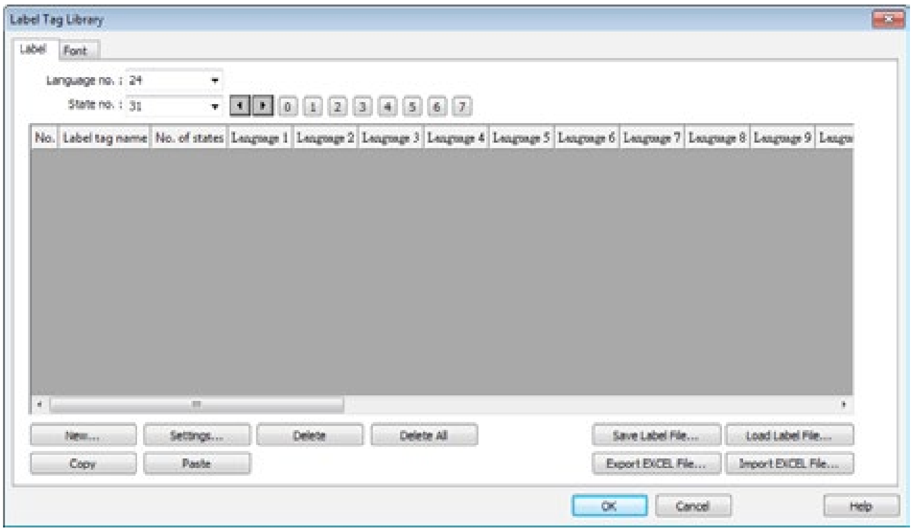
This chapter explains how to build and use Label Tag Library.

15.1. OVERVIEW

The Label Tag Library feature enables a multi-language environment. When multiple languages are required, users can create the Label Tag Library and then select a suitable label in the project. The project will display the corresponding language in runtime based on the settings. EasyBuilder Pro supports up to 8 different languages simultaneously. This chapter will explain how to create and use the Label Tag Library.

15.2. LABEL TAG LIBRARY MANAGER

Click (Library) » (Label) on the toolbar and the (Label Tag Library) dialog box appears.



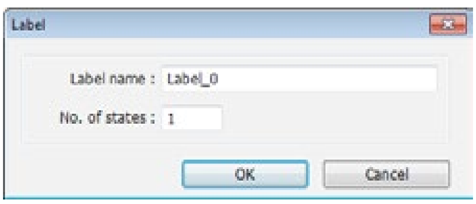
Setting	Description
Language no.	Specifies the number of languages used in a project.
State no.	Indicates the current state. Each Label has a maximum of 256 states (state no. 0 ~ 255). The state no. is determined by (Language no.). If less than 3 languages are used, the maximum state no. is 256. If more than 4 languages are used, divide 768 by the language number to get the maximum state no.. For example, the number of languages is 24, then there are only 768/24 = 32 states.
New	Adds a new Label.
Settings	Sets the selected Label.
Save label file	Saves all Labels in .lbl format.
Load label file	Loads the existing .lbl file to the Label Library.
Export Excel file	Saves all Labels in .csv, .xls, or .xlsx format.
Import Excel file	Loads the existing .csv, .xls, or .xlsx file to the Label Library.

Note
Unicode is not supported when importing and exporting an Excel file.

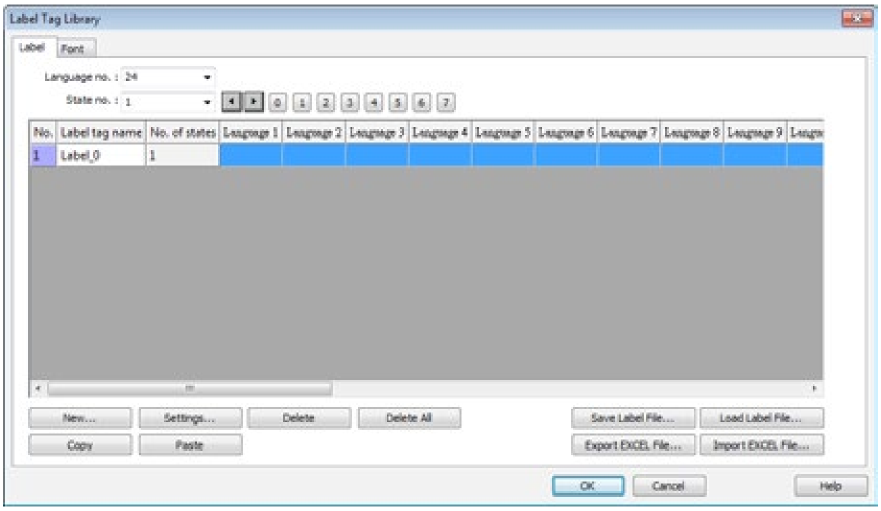
15.3. STEPS TO CREATE LABEL TAG LIBRARY

Please follow the steps to create a Label Tag Library.

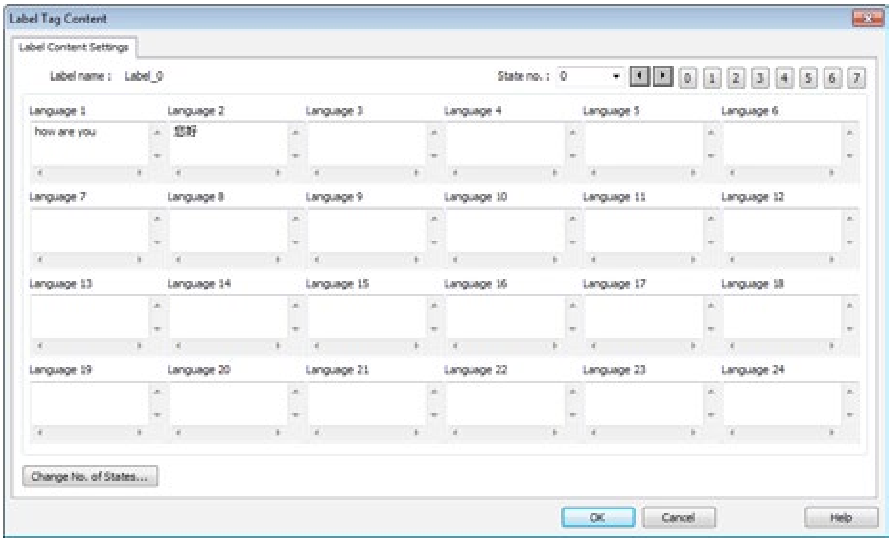
1. From the Library menu, click (Label). The Label Tag Library dialog box appears. Click (New) to specify the name of the Label and the number of states to be displayed by this Label.



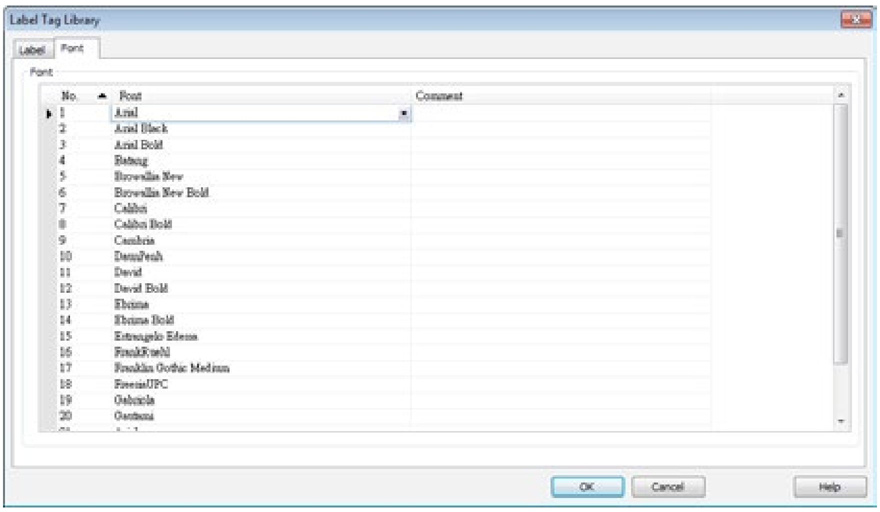
2. Click (OK) and a new label is added to the Label Tag Library. Select the label and click (Settings) to edit its content.



3. Edit the corresponding language content.

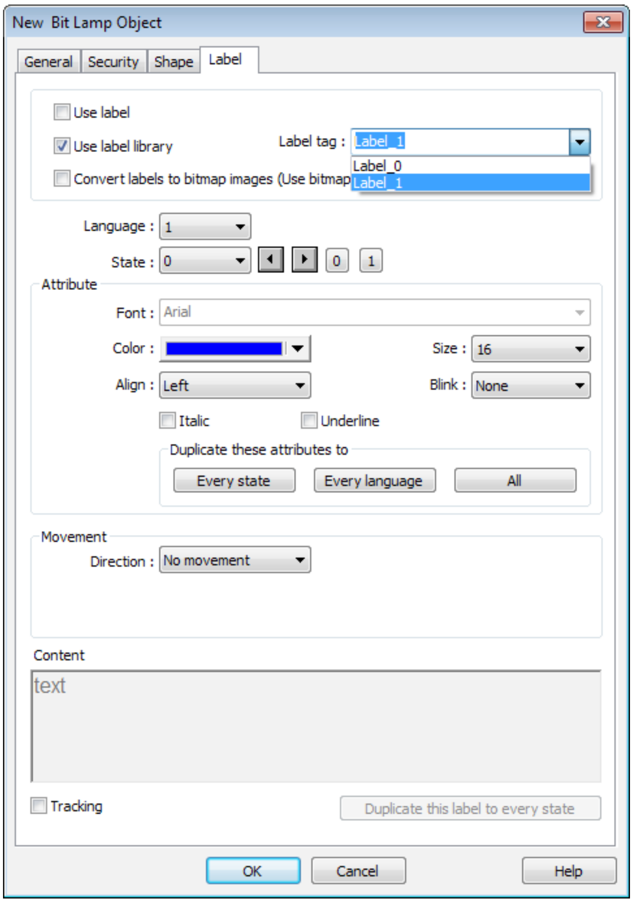


4. Select (Label Tag Library) » (Font) to view each label which contains different fonts for different languages. You can also enter the font description in the Comment field.



15.4. USING LABEL TAG LIBRARY

When there are defined labels in the Label Library, the labels can be found in the object's (Label) tab. Select (Use label library) check box, and select the label from the pull-down list (Label tag).



When a tag is selected, the content of the selected tag is shown in the (Content) field in its corresponding font style. Please note that from Language 2 to Language 24 can only be set the Font (Size), the others such as (Color), (Align), (Blink), etc. will follow the settings of Language 1.

15.5. SETTINGS OF MULTI-LANGUAGE

When displaying the texts in multiple languages, the system register "(LW-9134): language mode" should be used too.
The value of (LW-9134) is ranged from 0 to 23 (cMT Series is from 0 to 7). Different values correspond to different languages.
If not all languages are selected to compile and download, (LW-9134) will work differently.
For example, user defines 5 different languages in the Label Library:

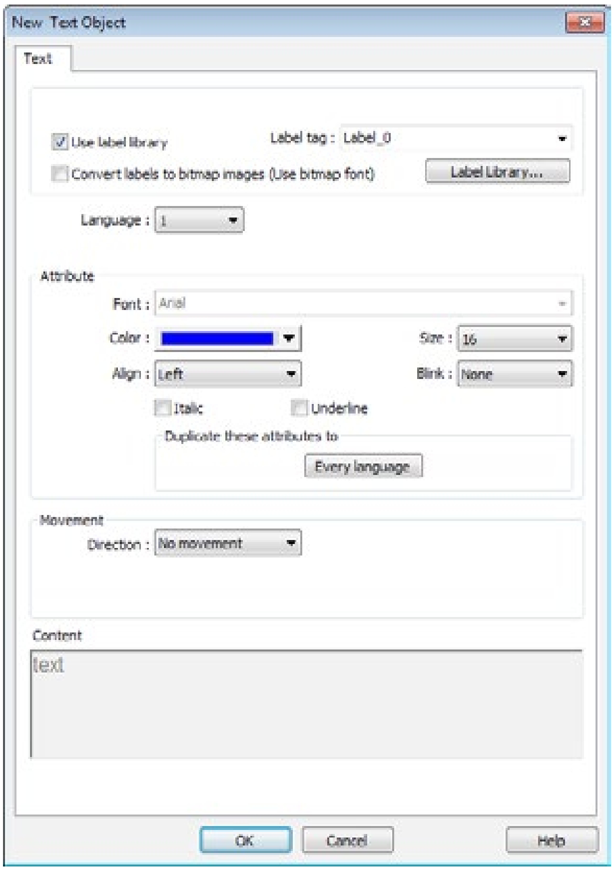
- 1: English, 2: Traditional Chinese, 3: Simplified Chinese, 4: French, 5: Korean

If only Language 1, Language 3, and Language 5 are selected to compile then the corresponding values of (LW-9134) are:

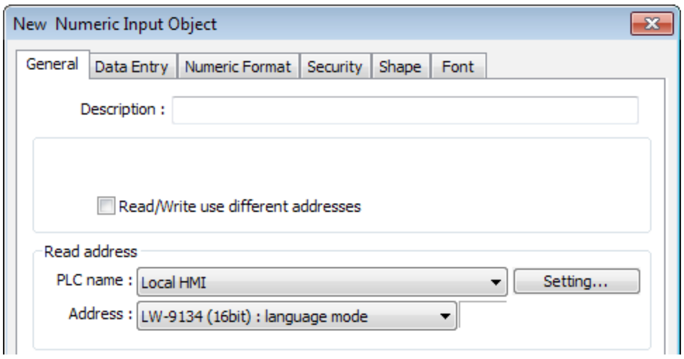
- 0: English, 1: Simplified Chinese, 2: Korean

Please follow the steps to use multiple languages.

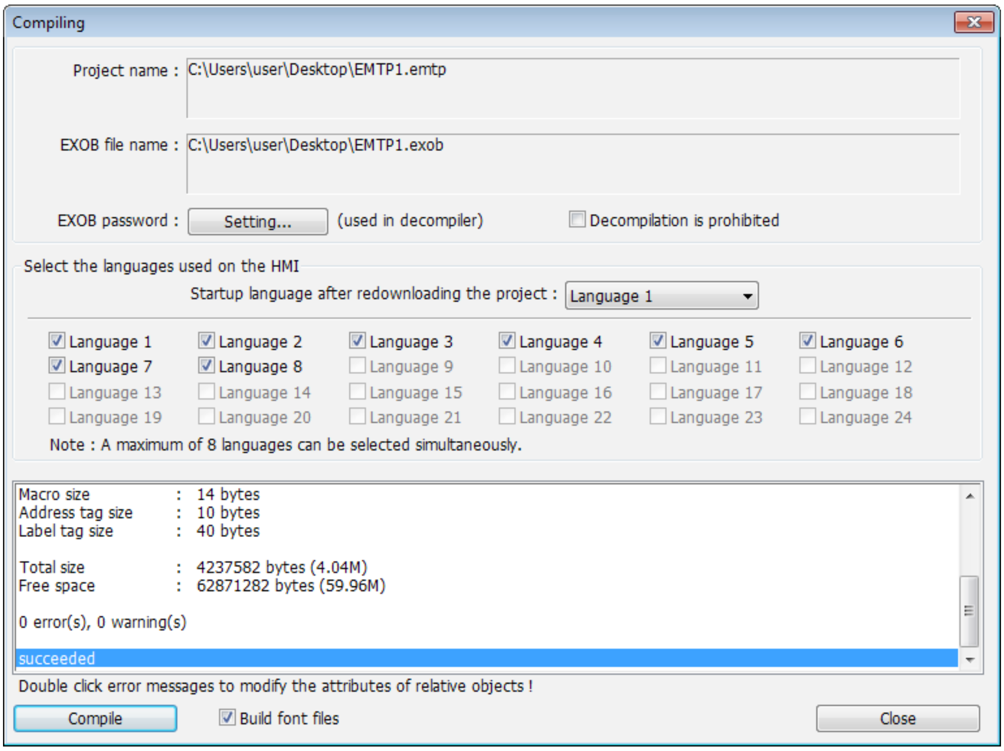
1. Create a Text object and select (Use label library) checkbox.



2. Create a Numeric Input Object and use the system register (LW-9134).



3. When compiling, select the defined languages.



4. The simulation is shown as followed: if the value of (LW-9134) is changed, the content of the Text object will be changed.

	Note ▪ For cMT Series, at most 8 different languages can be downloaded to the HMI. ▪ When selecting the HMI model: cMT, (LW-9134) is used to change the language mode on cMT model, while (PLW-9134) is to change the language mode on iPad.

	Download Click the icon to download the demo project that illustrates how to use the Option List object to switch between multiple languages. Please confirm your internet connection before downloading the demo project.

16. ADDRESS TAG LIBRARY

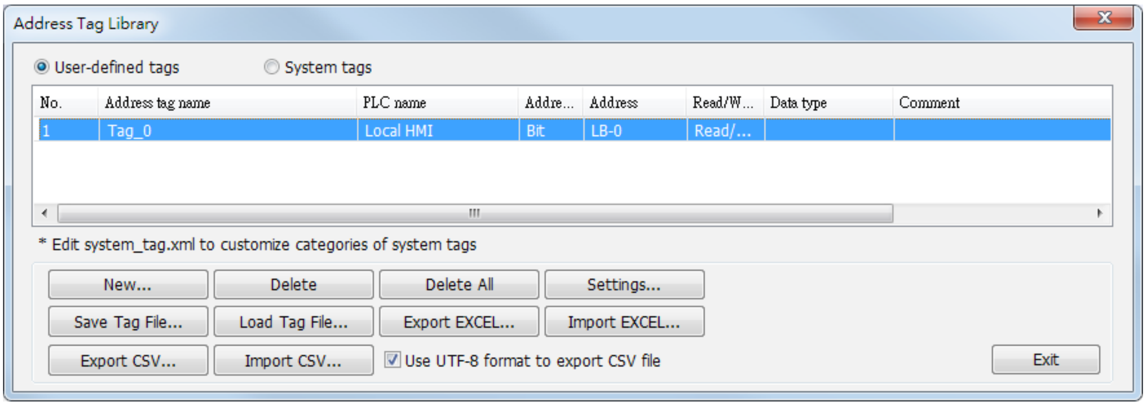
This chapter explains how to build and use Address Tag Library.

16.1. OVERVIEW

Generally it is recommended to define the commonly used addresses in Address Tag Library when starting to build a project. It not only avoids accidental reuse of addresses but also improves project readability.

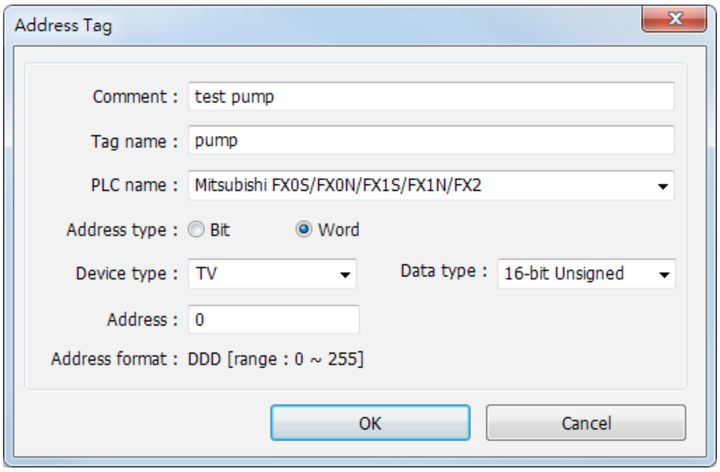
16.2. BUILDING ADDRESS TAG LIBRARY

Click (Library) » (Tag) on the toolbar and the (Address Tag Library) dialog box appears.



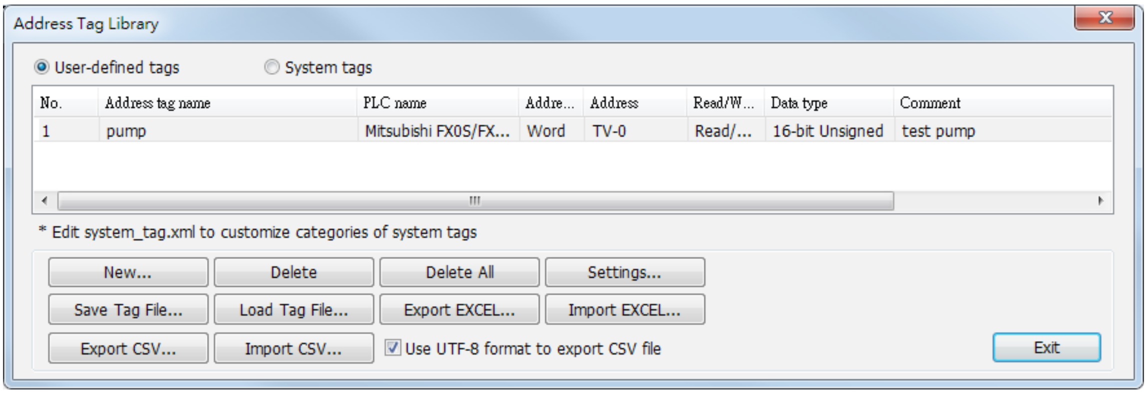
Setting	Description
Customized	Displays user-defined address tags.
System	Displays system registers. The registers listed cannot be deleted or changed.
New	Adds a new address tag. Please see the steps next page.
Settings	Sets the selected address tag.
Save tag file	Saves all current address tags as .tgl file.
Load tag file	Loads the existing .tgl file of address tag to the current project.
Export CSV	Saves all current address tags as .csv file.
Import CSV	Loads the existing .csv file of address tag to the current project.
Export Excel	Saves all current address tags as .xls file.
Import Excel	Loads the existing .xls file of address tag to the current project.
Use UTF-8 format to export CSV file	If selected, the .csv file will be exported in UTF-8 format. If not selected, in ANSI format.

1. Click (New) and set the relevant properties.



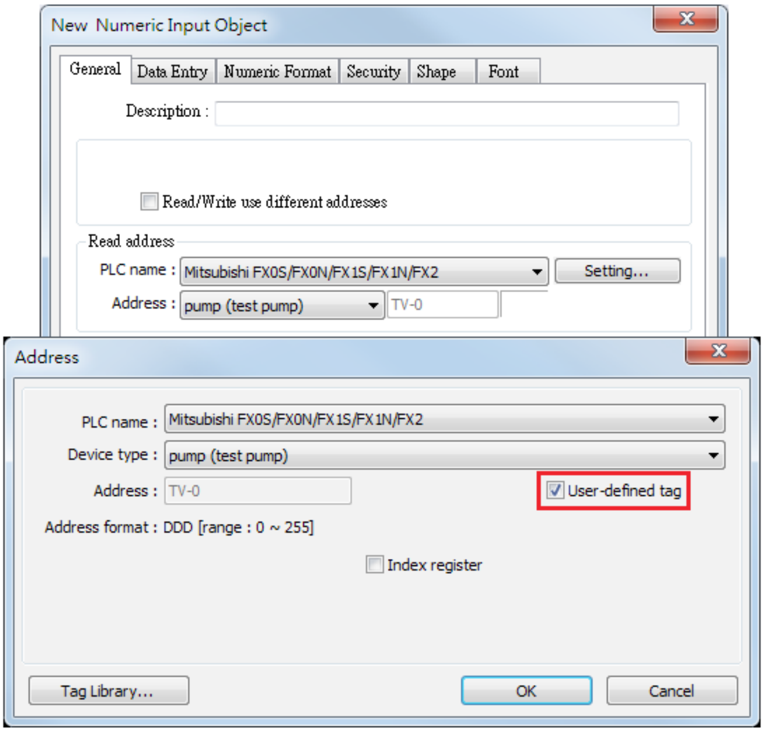
Setting	Description
Comment	The information of the address tag.
Tag name	The name of the address tag.
PLC name	As defined in (System Parameter Settings) » (Device list).
Address type	The tag address type; select (Bit) or (Word).
Device type	The available device types depend on (PLC name) and (Address type).
Address	Address of the tag.
Data type	If select (Word) in (Address type), the data type can be specified.

2. Click (OK), a newly added tag can be found in the (Customized) library.

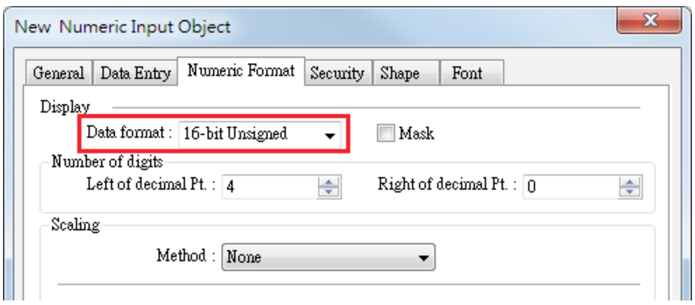


16.3. USING ADDRESS TAG LIBRARY

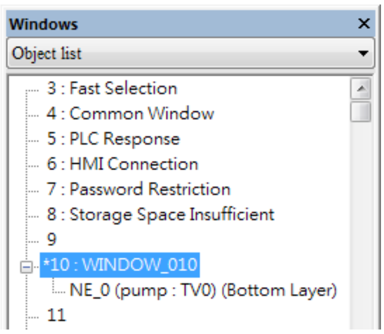
1. Create a tag in Address Tag Library.
2. Create an object, select (General) » (PLC name).
3. Click (Setting) to finish the settings.
4. Select (User-defined tag) check box.



5. In (Device type) select the defined tag.
6. If (Data type) is selected when creating the address tag, the system automatically restricts the data format to the one selected.



7. When finished, the window tree will show the address tag name used by the object.



17. TRANSFERRING RECIPE DATA

This chapter explains how to transfer recipe data.

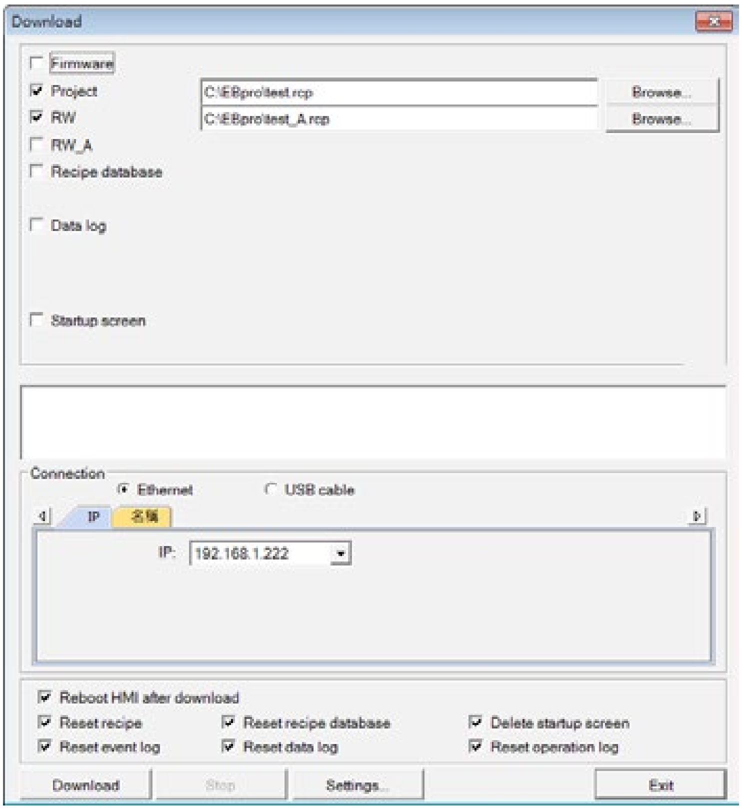
17.1. OVERVIEW

Recipe Data refers to the data stored in RW and RW_A addresses. The way of reading and writing these addresses is the same as operating a word register. The difference is that recipe data is stored in flash memory, when restarting HMI, the latest data records in RW and RW_A are kept.
The size of recipe data a RW address can store is 512K words, and RW_A is 64K words. Users can update recipe data with SD card, USB disk, USB cable or Ethernet and use the data to update PLC data. Recipe Data can also be uploaded to PC; furthermore, PLC data can be saved in recipe data. The following explains the ways of transferring recipe data.

17.2. STEPS TO UPDATE RECIPE DATA WITH ETHERNET OR USB CABLE

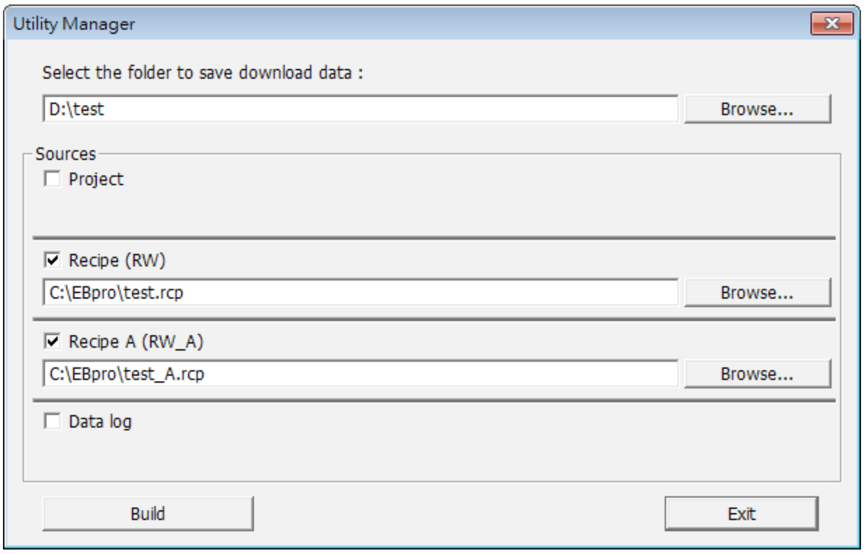
1. Open Utility Manager and click (Download).
2. Select (RW) and (RW_A) and (Browse) the source file.
3. After downloading, restart HMI, RW and RW_A will be updated.

When (Reboot HMI after download) is selected, users don't have to manually reboot HMI.
When (Reset recipe) check box is selected, the system will clear all the data in (RW) and (RW_A) before downloading.



17.3. STEPS TO UPDATE RECIPE DATA WITH SD CARD OR USB DISK

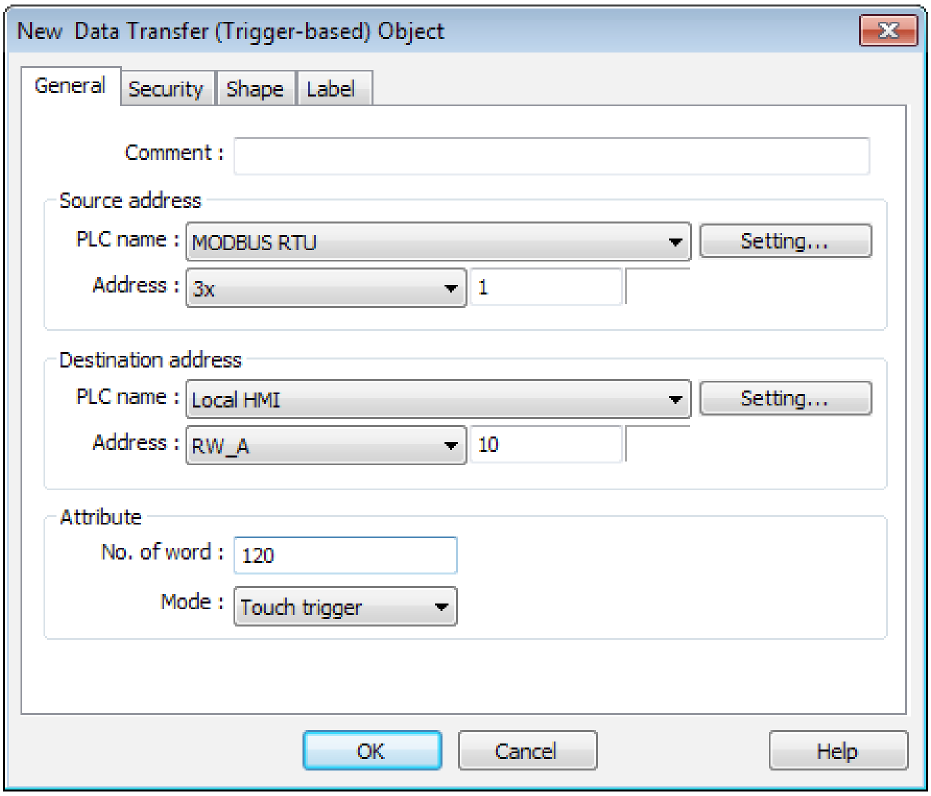
1. Open Utility Manager and click (Build Download Data for SD Card or USB Disk).
2. Insert a SD card or USB disk into PC.
3. Click (Browse) to designate the file path.
4. Click (Build), EasyBuilder Pro will save the data in SD card or USB disk.



	Note When download data is successfully built, two folders can be found: *history* and *emt3000. emt3000* is for storing project file; *history* is for storing recipe data and data sampling / event log records.

17.4. TRANSFERRING RECIPE DATA

Use (Data Transfer (Trigger-based) Object) to transfer recipe data to a specific address, or save the data of this address in (RW) and (RW_A).



Setting	Description
Source address	Sets the source of the data.
Destination address	Sets the destination of the data to transfer to.
Attribute	Sets the number of words to transfer from source to destination.

17.5. SAVING RECIPE DATA AUTOMATICALLY

In order to prolong the life span of HMI flash memory, the system will automatically save the recipe data to HMI every minute. To avoid losing data when turning HMI off during the interval of saving data, system register (LB-9029: save all recipe data to machine (set ON)) is provided. Set ON LB-9029 will make the system save recipe data once. Set ON (LB-9028: reset all recipe data (set ON)), the system will clear all recipe data.

18. MACRO REFERENCE

This chapter describes the syntax, programming methods and usage of macro commands.

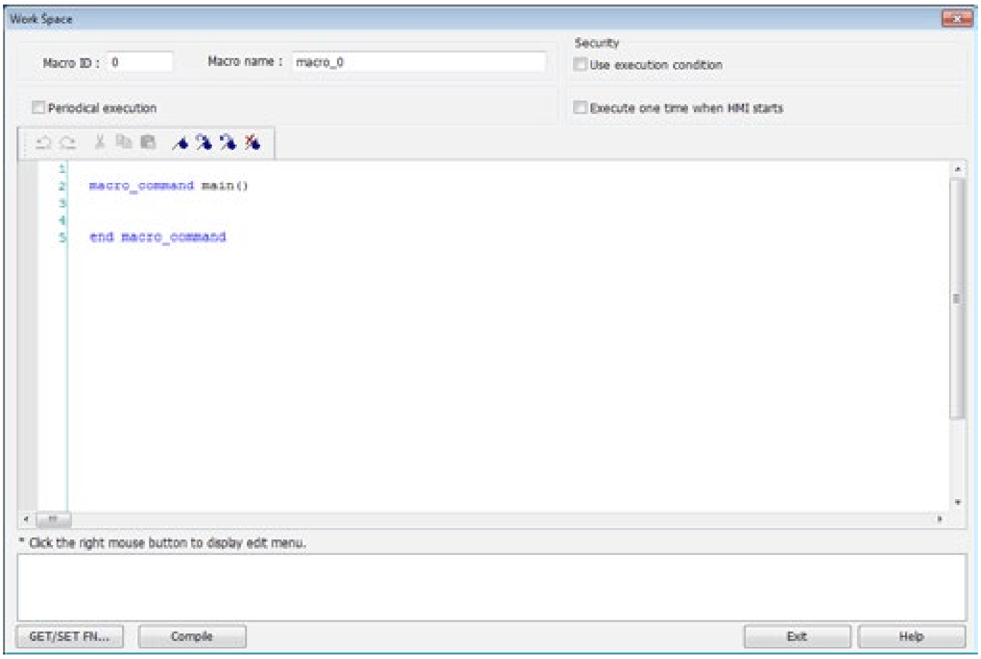
18.1. OVERVIEW

Macros provide the additional functionality your application may need. Macros are automated sequences of commands that are executed at run-time. Macros allow you to perform tasks such as complex scaling operations, string handling, and user interactions with your projects. This chapter describes syntax, usage, and programming methods of macro commands.

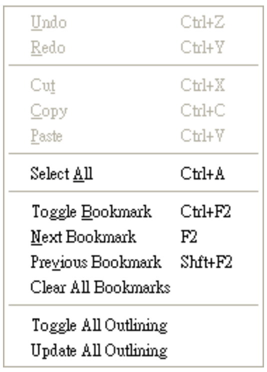
18.2. INSTRUCTIONS TO USE THE MACRO EDITOR

Macro editor provides the following functions:
- Display line number
- Undo / Redo
- Cut / Copy / Paste
- Select All
- Toggle Bookmark / Previous Bookmark / Next Bookmark / Clear All Bookmarks
- Toggle All Outlining
- Security -> Use execution condition
- Periodical execution
- Execute one time when HMI starts

The instructions in the following part show you how to use these functions.
1. Open the macro editor; you'll see the line numbers displayed on the left-hand side of the edit area.

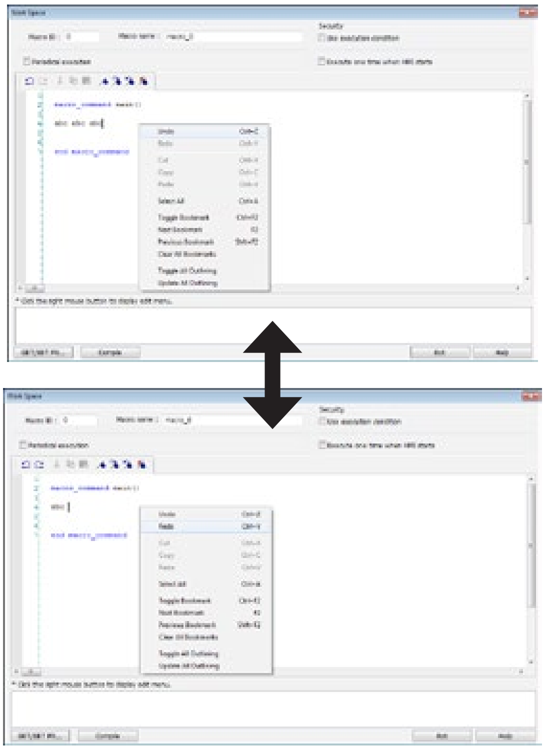


2. Right click on the edit area to open the pop-up menu as shown in the following figure. Disabled operations are colored grey, which indicates that it is not possible to use that function in the current status of the editor. For example, you should select some text to enable the copy function, otherwise it will be disabled. Keyboard shortcuts are also shown.

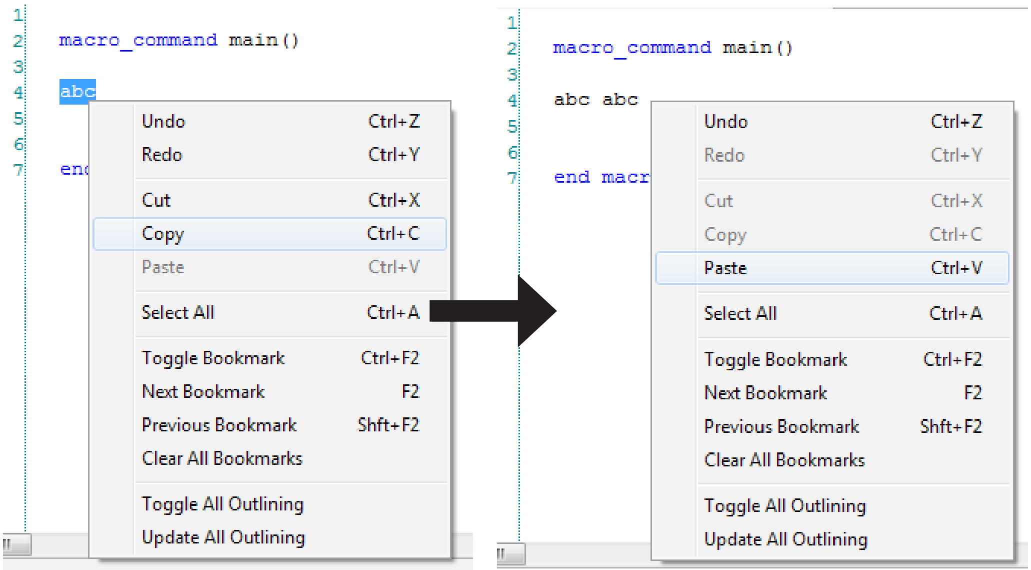


3. The toolbar provides (Undo), (Redo), (Cut), (Copy), (Paste), (Toggle Bookmark), (Next Bookmark), (Previous Bookmark) and (Clear All Bookmarks) buttons.

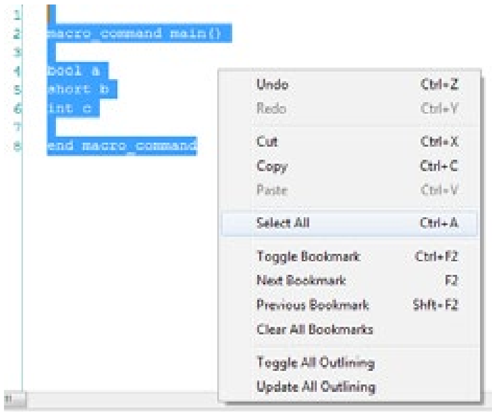
4. Any modification will enable the (Undo) function. (Redo) function will be enabled after the undo action is used. To perform the undo/redo, right click to select the item or use the keyboard shortcuts. (Undo: Ctrl+Z, Redo: Ctrl+Y).



5. Select a word in the editor to enable the (Cut) and (Copy) function. After (Cut) or (Copy) is performed, (Paste) function is enabled.

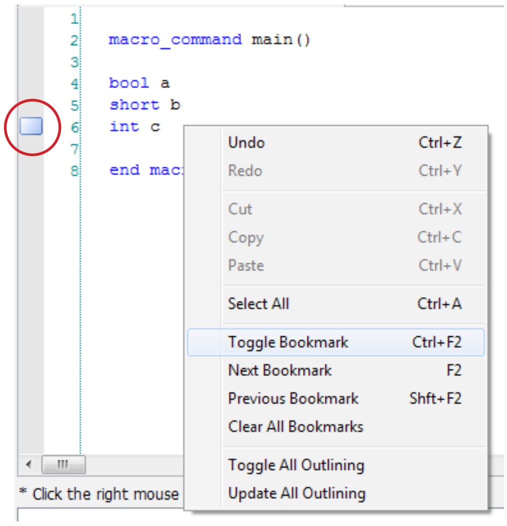


6. Use (Select All) to include all the content in the edit area.

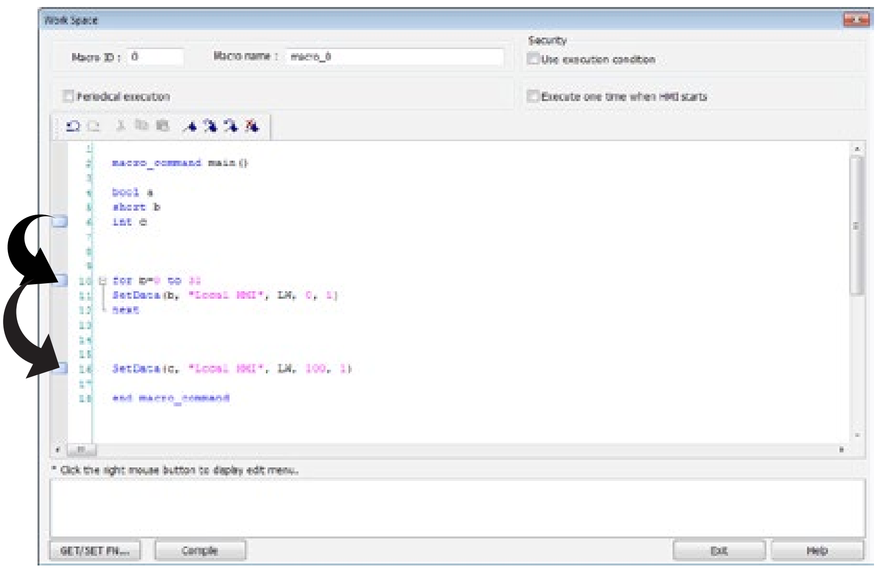


7. If the macro is too long, use bookmarks to manage and read the code with ease. The following illustration shows how it works.

- Move your cursor to the position in the edit area where to insert a bookmark. Right click, select (Toggle Bookmark). There will be a blue little square that represents a bookmark on the left hand side of edit area

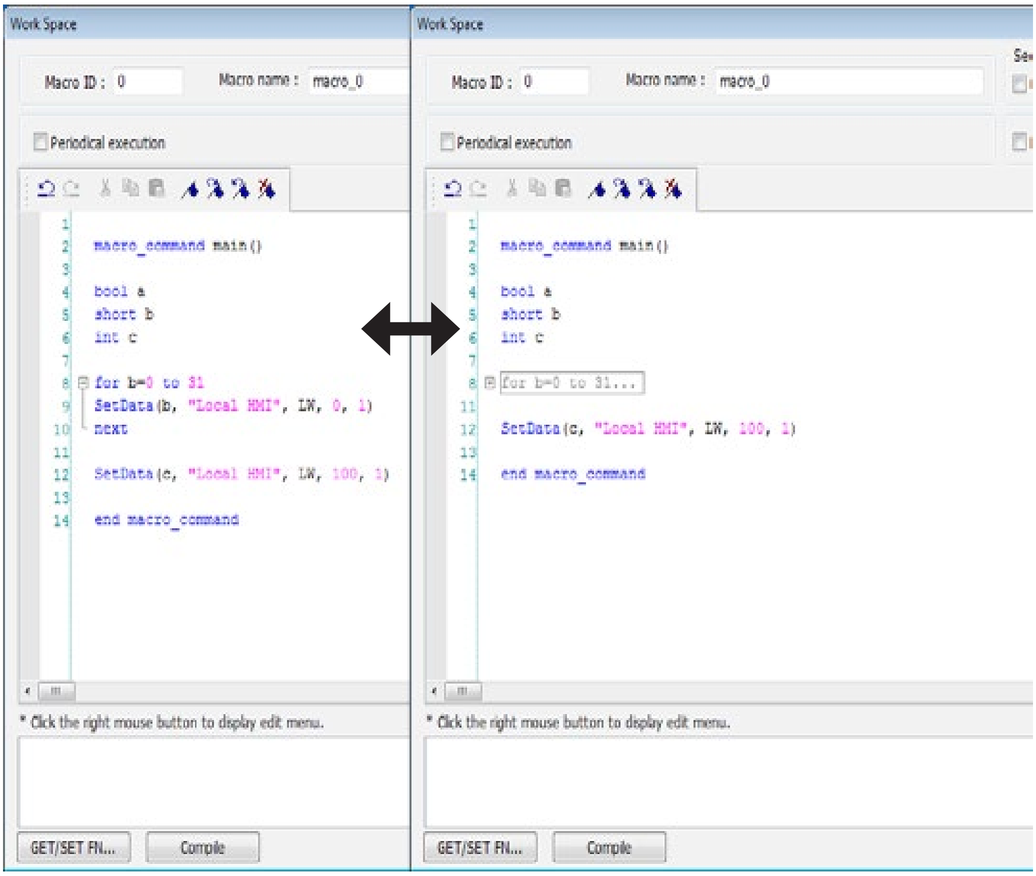


- If there is already a bookmark where the cursor is placed, select (Toggle Bookmark) to close it, otherwise to open it
- Right click and select (Next Bookmark), the cursor will move to where the next bookmark locates. Selecting (Previous Bookmark) will move the cursor to the previous bookmark

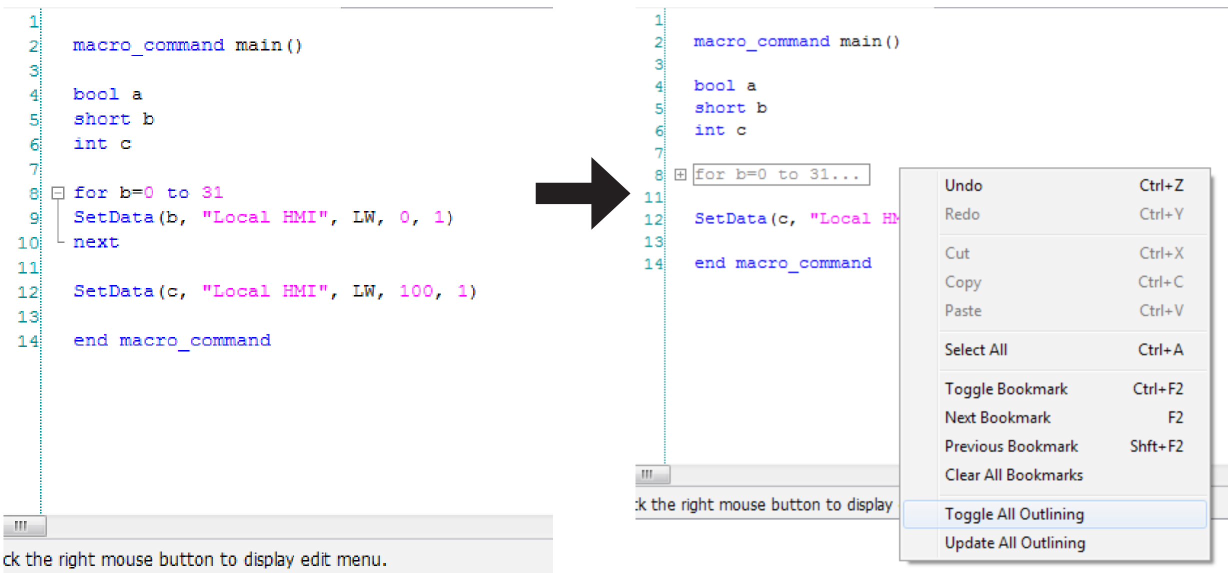


- Selecting (Clear All Bookmarks) will delete all bookmarks

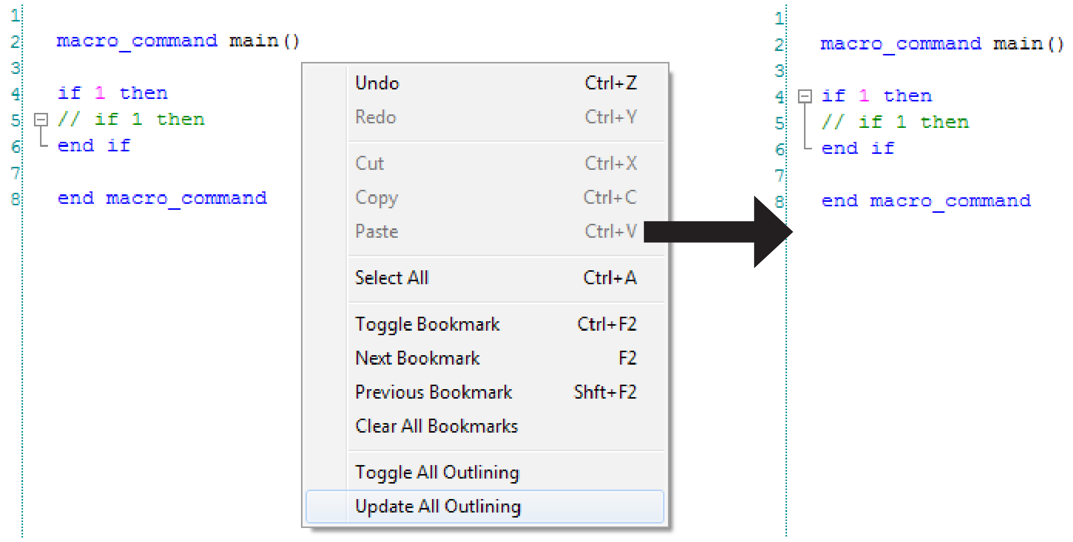
8. Macro editor provides outlining (or code-folding). Outlining will hide macro codes that belong to the same block, and display them as ⊞ [...]. There will be a tree diagram on the left hand side of edit area. Click ⊟ to hide the block or ⊞ to open, as shown in the following figure.



9. Right click to select (Toggle All Outlining) to open all folded macro code blocks.

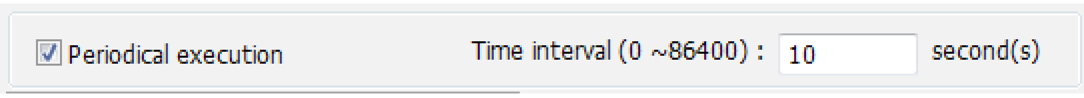


10. Sometimes the outlining might be incorrect since that the keywords are misjudged as shown in the following figure. To solve this problem, right click and select (Update All Outlining).

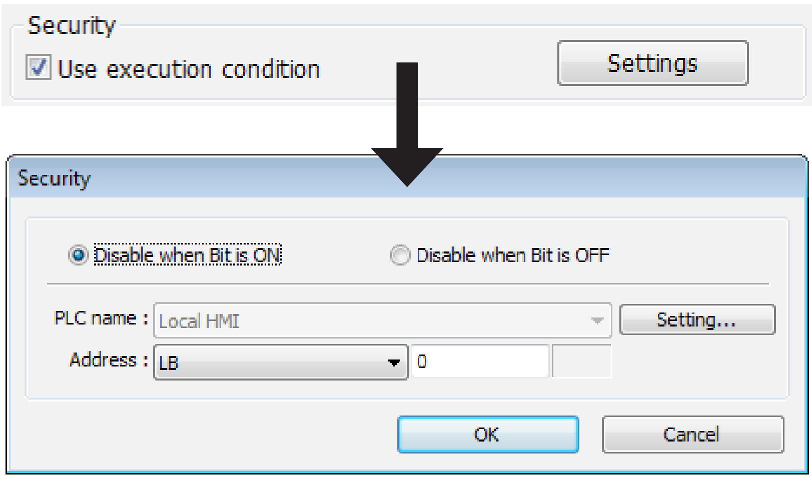


11. The statements enclosed in the following keywords are called a “block” of the macro code:
- Function block: sub – end sub
- Iterative statements:
  - for – next
  - while – wend
- Logical statements:
  - if – end if
- Selective statements: select case – end select

12. When (Periodical execution) is checked, this macro will be triggered periodically.



13. Select (Security) » (Use execution condition) » (Settings) to enable security settings:
- (Disable when Bit is ON): when Bit is ON, this macro is disabled
- (Disable when Bit is OFF): when Bit is OFF, this macro is disabled



14. Select (Execute one time when HMI starts), this macro will be executed once when HMI starts up.

18.3. CONFIGURATION

A macro contains statements. The statements contain constants, variables and operations. The statements are put in a specific order to create the desired output.
A macro has the following structure:

Global Variable Declaration	Optional
Sub Function Block Declarations Local Variable Declarations End Sub	Optional
macro_command main() Local Variable Declarations (Statements)	Required
end macro_command	Required

Macro must have one and only one main function which is the execution start point of macro. The format is:
- macro_command main()
- end macro_command

Local variables are used within the main macro function or in a defined function block. Its value remains valid only within the specific block.
Global variables are declared before any function blocks and are valid for all functions in the macro. When local variables and global variables have the same declaration of name, only the local variables are valid.

The following example shows a simple macro which includes a variable declaration and a function call.

```
macro_command main()
    short pressure = 10                              //    local variable declaration
    SetData(pressure, "Allen-Bradley DF1", N7, 0, 1) //    function calling
end macro_command
```

18.4. SYNTAX

18.4.1. Constants and Variables

18.4.1.1. Constants
Constants are fixed values and can be directly written into statements. The formats are:

Constant type	Note	Example
Decimal integer	-	345, -234, 0, 23456
Hexadecimal	Must begin with 0x	0x3b, 0xffff, 0x237
ASCII	Single character must be enclosed in single quotation marks and a string (group of characters) must be enclosed by double quotation marks.	'a', "data", "name"
Boolean	-	true, false

Here is an example using constants:

```
macro_command main()
short A, B          //   A and B are variables
A = 1234
B = 0x12            //   1234 and 0x12 are constants
end macro_command
```

18.4.1.2. Variables
Variables are names that represent information. The information can be changed as the variable is modified by statements.

Naming Rules for Variables
- A variable name must start with an alphabet
- Variable names longer than 32 characters are not allowed
- Reserved words cannot be used as variable names

There are 8 different Variable types, 5 for signed data types and 3 for unsigned data types:

Variable type	Description	Range
Bool (boolean)	1 bit (discrete)	0, 1
Char (character)	8 bits (byte)	+127 to -128
Short (short integer)	16 bits (word)	+32767 to -32768
Int (integer)	32 bits (double word)	+2147483647to -2147483648
Float (floating point)	32 bits (double word)	
Unsigned char	8 bits (byte)	0 to 255
Unsigned short	16 bits (word)	0 to 65535
Unsigned int	32 bits (double word)	0 to 4,294,967,295

Declaring Variables

Variables must be declared before being used. To declare a variable, specify the type before the variable name.
Example:

```
int             a
short           b, switch
float           pressure
unsigned short  c
```

Declaring Arrays

Macros support one-dimensional arrays (zero-based index). To declare an array of variables, specify the type and the variable name followed by the number of variables in the array enclosed in brackets "()". The length of an array could be 1 to 4,096. (Macros only support at most 4,096 variables per macro).
Example:

```
int             a[10]
short           b[20], switch[30]
float           pressure[15]
```

The minimum array index is 0 and the maximum is (array size – 1).
Example:

char data [100] // array size is 100

In this case, the minimum of array index is 0 and maximum of array index is 99 (=100-1).

Variable and Array Initialization

There are two ways variables can be initialized:

- By statement using the assignment operator (=)
  Example:
  int a
  float b[3]
  a = 10
  b[0] = 1

- During declaration
  - char a = '5', b = 9

The declaration of arrays is a special case. The entire array can be initialized during declaration by enclosing comma separated values inside curly brackets "{}".
Example:

- float data[4] = {11, 22, 33, 44} // now data[0] is 11, data[1] is 22….

18.4.2. Operators

Operators are used to designate how data is manipulated and calculated.

Operator	Description	Example
=	Assignment operator	pressure = 10

Arithmetic operators	Description	Example
+	Addition	A = B + C
-	Subtraction	A = B – C
*	Multiplication	A = B * C
/	Division	A = B / C
%	Modulo division (return remainder)	A = B % 5

Comparison operators	Description	Example
<	Less than	if A < 10 then B = 5
<=	Less than or equal to	if A <= 10 then B = 5
>	Greater than	if A > 10 then B = 5
>=	Greater than or equal to	if A >= 10 then B = 5
==	Equal to	if A == 10 then B = 5
<>	Not equal to	if A <> 10 then B = 5

Logic operators	Description	Example
and	Logical AND	if A < 10 and B > 5 then C = 10
or	Logical OR	if A >= 10 or B > 5 then C = 10
xor	Logical Exclusive OR	if A xor 256 then B = 5
not	Logical NOT	if not A then B = 5

Shift and bitwise operators are used to manipulate bits of signed/unsigned character and integer variables. The priority of these operators is from left to right within the statement.

Shift operators	Description	Example
<<	Shifts the bits in a bitset to the left a specified number of positions	A = B << 8
>>	Shifts the bits in a bitset to the right a specified number of positions	A = B >> 8

Bitwise operators	Description	Example
&	Bitwise AND	A = B & 0xf
|	Bitwise OR	A = B | C
^	Bitwise XOR	A = B ^ C
~	One's complement	A = ~B

Priority of All Operators

The overall priority of all operations from highest to lowest is as follows:

1. Operations within parenthesis are carried out first
2. Arithmetic operations
3. Shift and Bitwise operations
4. Comparison operations
5. Logic operations
6. Assignment

Reserved Keywords
The following keywords are reserved for system. These keywords cannot be used as variable, array, or function names.

+, -, *, /, %, >=, >, <=, <, <>, ==, and, or, xor, not, <<, >>,=, &, |, ^, ~
exit, macro_command, for, to, down, step, next, return, bool, short, int, char, float, void, if, then, else, break, continue, set, sub, end, while, wend, true, false
SQRT, CUBERT, LOG, LOG10, SIN, COS, TAN, COT, SEC, CSC, ASIN, ACOS, ATAN, BIN2BCD, BCD2BIN, DEC2ASCII, FLOAT2ASCII, HEX2ASCII, ASCII2DEC, ASCII2FLOAT, ASCII2HEX, FILL, RAND, DELAY, SWAPB, SWAPW, LOBYTE, HIBYTE, LOWORD, HIWORD, GETBIT, SETBITON, SETBITOFF, INVBIT, ADDSUM, XORSUM, CRC, INPORT, OUTPORT, POW, GetError, GetData, GetDataEx, SetData, SetDataEx, SetRTS, GetCTS, Beep, SYNC_TRIG_MACRO, ASYNC_TRIG_MACRO, TRACE, FindDataSamplingDate, FindDataSamplingIndex, FindEventLogDate, FindEventLogIndex
StringGet, StringGetEx, StringSet, StringSetEx, StringCopy, StringMid, StringDecAsc2Bin, StringBin2DecAsc, StringDecAsc2Float, StringFloat2DecAsc, StringHexAsc2Bin, StringBin2HexAsc, StringLength, StringCat, StringCompare, StringCompareNoCase, StringFind, StringReverseFind, StringFindOneOf, StringIncluding, StringExcluding, StringToUpper, StringToLower, StringToReverse, StringTrimLeft, StringTrimRight, StringInsert

18.5. STATEMENT

18.5.1. Definition Statement
This covers the declaration of variables and arrays. The formal construction is as follows:

```
type                name
```

This defines a variable with name as “name” and type as “type”.
Example:
int A   // define a variable A as an integer

```
type                name[constant]
```

This defines an array variable called “name” with size as “constant” and type as “type”.
Example:
int B[10]   // where define a variable B as a one-dimensional array of size 10

18.5.2. Assignment Statement
Assignment statements use the assignment operator to move data from the expression on the right side of the operator to the variable on the left side. An expression is the combination of variables, constants and operators to yield a value.

```
VariableName                Expression
```

Example:
A = 2   where a variable A is assigned to 2

18.5.3. Logical Statements
Logical statements perform actions depending on the condition of a boolean expression.
The syntax is as follows:

Single-Line Format

```
If <Condition> then
    [Statements]
else
    [Statements]
end if
```

Example:
```
if a == 2 then
            b = 1
else
            b = 2
end if
```

Block Format

```
If <Condition> then
    [Statements]
else if <Condition-n> then
[Statements]
else
    [Statements]
end if
```

Example:
```
if a == 2 then
            b = 1
else if a == 3 then
            b = 2
else
            b = 3
end if
```

Syntax description

if	Must be used to begin the statement.
<Condition>	Required. This is the controlling statement. It is FALSE when the <Condition> evaluates to 0 and TRUE when it evaluates to non- zero.
then	Must precede the statements to execute if the <Condition> evaluates to TRUE.
(Statements)	It is optional in block format but necessary in single-line format without else. The statement will be executed when the <Condition> is TRUE.
else if	Optional. The else if statement will be executed when the relative <Condition-n> is TRUE.
<Condition-n>	Optional. see <Condition>
else	Optional. The else statement will be executed when <Condition> and <Condition-n> are both FALSE.
end if	Must be used to end an if-then statement.

18.5.4. Selective Statements

The select-case construction can be used like multiple if-else statements and perform selected actions depending on the value of the given variable. When the matched value is found, all the actions below will be executed until a break statement is met. The syntax is as follows:

Format without a Default Case

```
Select Case [variable]
Case [value]
        [Statements]
        break
end Select
```

Example:

```
        Select Case A
            Case 1
                b=1
            break
        end Select
```

Format with a Default Case (Case else)

```
Select Case [variable]
Case [value]
        [Statements]
        break
        Case else
        [Statements]
        break

end Select
```

Example:

```
        Select Case A
            Case 1
                b=1
            break
            Case else
                b=0
            break
        end Select
```

Multiple cases in the same block

```
Select Case [variable]
Case [value1]
            [Statements]
Case [value2]
            [Statements]
            break

end Select
```

Example:

```
        Select Case A
             Case 1
             break
             Case 2
                 b=2
             break
             Case 3
                 b=3
             break
        end Select
```

Syntax description

Select Case	Must be used to begin the statement.
(variable)	Required. The value of this variable will be compared to the value of each case.
Case else	Optional. It represents the default case. If none of the cases above are matched, the statements under default case will be executed. When a default case is absent, it will skip directly to the end of the select-case statements if there is no matched case.
break	Optional. The statements under the matched case will be executed until the break command is reached. If a break command is absent, it simply keeps on executing next statement until the end command is reached.
end Select	Indicates the end of the select-case statements.

18.5.5. Iterative Statements

Iterative statements control loops and repetitive tasks depending on condition. There are two types of iterative statements.

18.5.5.1. For-Next Statements

The for-next statement runs for a fixed number of iterations. A variable is used as a counter to track the progress and test for ending conditions. Use this for fixed execution counts. The syntax is as follows:

```
for [Conunter] = <StartValue> to <EndValue> [step <StepValue>]
    [Statements]
next [Counter]
```

Or

```
for [Conunter] = <StartValue> to <EndValue> [step <StepValue>]
    [Statements]
next [Counter]
```

Example:

```
for a = 0 to 10 step 2
        b = a
next a
```

Syntax description

for	Must be used to begin the statement
(Counter)	Required. This is the controlling statement. The result of evaluating the variable is used as a test of comparison.
<StartValue>	Required. The initial value of (Counter)
to/down	Required. This determines if the <step> increments or decrements the <Counter>. “to” increments <Counter> by <StepValue>. “down” decrements <Counter> by <StepValue>.
<EndValue>	Required. The test point. If the <Counter> is greater than this value, the macro exits the loop.
step	Optional. Specifies that a <StepValue> other than one is to be used.
(StepValue)	Optional. The increment/decrement step of <Counter>. It can be omitted when the value is 1 If (step <StepValue>) are omitted the step value defaults to 1.
(Statements)	Optional. Statements to execute when the evaluation is TRUE. “for-next” loops may be nested.
next	Required.
(Counter)	Optional. This is used when nesting for-next loops.

18.5.5.2. While-Wend Statements

The while-wend statement runs for an unknown number of iterations. A variable is used to test for ending conditions. When the condition is TRUE, the statements inside are executed repetitively until the condition becomes FALSE. The syntax is as follows.

```
fwhile <Condition>
      [Statements]
wend
```

Example:

```
while a < 10
        a = a + 10
wend
```

Syntax description

while	Must be used to begin the statement.
continue	Required. This is the controlling statement. When it is TRUE, the loop begins execution. When it is FALSE, the loop terminates.
return (value)	Statements to execute when the evaluation is TRUE.
wend	Indicates the end of the while-end statements.

18.5.5.3. Other Control Commands

break	Used in for-next and while-wend. It skips immediately to the end of the iterative statement.
continue	Used in for-next and while-wend. It ends the current iteration of a loop and starts the next one.
return	The return command inside the main block can force the macro to stop anywhere. It skips immediately to the end of the main block.

18.6. FUNCTION BLOCKS

Function blocks are useful for reducing repetitive codes. It must be defined before use and supports any variable and statement type. A function block could be called by putting its name followed by parameters in parenthesis. After the function block is executed, it returns the value to the caller function where it is used as an assignment value or as a condition. A return type is not required in function definition, which means that a function block does not have to return a value. The parameters can also be ignored in function definition while the function has no need to take any parameters from the caller. The syntax is as follows:

Function definition with return type

```
sub type <name> [(parameters)]
    Local variable declarations
    [Statements]
    [return (value)]
end sub
```

Example:

```
        sub int Add(int x, int y)
                int result
                result = x +y
                return result
          end sub

          macro_command main()
                int a = 10, b = 20, sum
                sum = Add(a, b)
          end macro_command
```

or:

```
        sub int Add()
                int result, x=10, y=20
                result = x +y
                return result
          end sub
          macro_command main()
                int  sum
                sum = Add()
          end macro_command
```

Function definition without return type

```
sub <name> [(parameters)]
      Local variable declarations
      [Statements]
end sub
```

Example:

```
            sub Add(int x, int y)
                          int result
                          result = x +y
                          end sub

                        macro_command main()
                               int  a = 10, b = 20
                               Add(a, b)
                        end macro_command
```

or:

```
            sub Add()
                            int result, x=10, y=20
                            result = x +y
                      end sub

                        macro_command main()
                               Add()
                        end macro_command
```

Syntax description

sub	Must be used to begin the function block
type	Optional. This is the data type of value that the function returns. A function block is not always necessary to return a value.
(parameters)	Optional. The parameters hold values that are passed to the function. The passed parameters must have their type declared in the parameter field and assigned a variable name. For example: sub int MyFunction(int x, int y). x and y would be integers passed to the function. This function is called by a statement that looks similar to this: ret = MyFunction(456, pressure) where "pressure" must be integer according to the definition of function. Notice that the calling statement can pass hard coded values or variables to the function. After this function is executed, an integer values is return to 'ret'.
Local variable declaration	Variables that are used in the function block must be declared first. This is in addition to passed parameters. In the above example x and y are variables that the function can used. Global variables are also available for use in function block.
(Statements)	Statements to execute
(return (value))	Optional. Used to return a value to the calling statement. The value can be a constant or a variable. Return also ends function block execution. A function block is not always necessary to return a value, but, when the return type is defined in the beginning of the definition of function, the return command is needed.
end sub	Must be used to end a function block.

18.7. BUILT-IN FUNCTION BLOCK

EasyBuilder Pro has many built-in functions for retrieving and transferring data to the PLC, data management and mathematical functions.

18.7.1. Mathematical Functions

Name	SQRT
Syntax	SQRT(source, result)
Description	Calculate the square root of source and store the result into result. source can be a constant or a variable. result must be a variable. source must be a nonnegative value.
Example	macro_command main() float source, result SQRT(15, result) source = 9.0 SQRT(source, result)// result is 3.0 end macro_command

Name	CUBERT
Syntax	CUBERT(source, result)
Description	Calculate the cube root of source and store the result into result. source can be a constant or a variable. result must be a variable. source must be a nonnegative value.

Example	macro_command main()
	float source, result
	
	CUBERT (27, result) // result is 3.0
	
	source = 27.0
	CUBERT(source, result)// result is 3.0
	
	end macro_command

Name	POW
Syntax	POW(source1, source2, result)
Description	Calculate source1 to the power of source2. source1 and source2 can be a constant or a variable. result must be a variable. source1 and source2 must be a nonnegative value.
Example	macro_command main() float y, result y = 0.5 POW (25, y, result) // result = 5 end macro_command

Name	SIN
Syntax	SIN(source, result)
Description	Calculate the sine of source (degree) into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	float source, result
	
	SIN(90, result)// result is 1
	
	source = 30
	SIN(source, result)// result is 0.5
	
	end macro_command

Name	COS
Syntax	COS(source, result)
Description	Calculate the cosine of source (degree) into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	float source, result
	
	COS(90, result)// result is 0
	
	source = 60
	GetData(source, “Local HMI”, LW, 0, 1)
	COS(source, result)// result is 0.5
	
	end macro_command

Name	TAN
Syntax	TAN(source, result)
Description	Calculate the tangent of source (degree) into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	float source, result
	
	TAN(45, result)// result is 1
	
	source = 60
	TAN(source, result)// result is 1.732
	
	end macro_command

Name	COT
Syntax	COT(source, result)
Description	Calculate the cotangent of source (degree) into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	float source, result
	
	COT(45, result)// result is 1
	
	source = 60
	COT(source, result)// result is 0.5774
	
	end macro_command

Name	SEC
Syntax	SEC(source, result)
Description	Calculate the secant of source (degree) into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	float source, result
	
	SEC(45, result)// result is 1.414
	
	source = 60
	SEC(source, result)// if source is 60, result is 2
	
	end macro_command

Name	CSC
Syntax	CSC(source, result)
Description	Calculate the cosecant of source (degree) into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	float source, result
	
	CSC(45, result)// result is 1.414
	
	source = 30
	CSC(source, result)// result is 2
	
	end macro_command

Name	ASIN
Syntax	ASIN(source, result)
Description	Calculate the arc sine of source into result (degree). source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	float source, result
	
	ASIN(0.8660, result)// result is 60
	
	source = 0.5
	ASIN(source, result)// result is 30
	
	end macro_command

Name	ACOS
Syntax	ACOS(source, result)
Description	Calculate the arc cosine of source into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	float source, result
	
	ACOS(0.8660, result)// result is 30
	
	source = 0.5
	ACOS(source, result)// result is 60
	
	end macro_command

Name	ATAN
Syntax	ATAN(source, result)
Description	Calculate the arc tangent of source into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	float source, result
	
	ATAN(1, result)// result is 45
	
	source = 1.732
	ATAN(source, result)// result is 60
	
	end macro_command

Name	LOG
Syntax	LOG (source, result)
Description	Calculates the natural logarithm of a number. source can be either a variable or a constant. result must be a variable.
Example	macro_command main() float source = 100, result LOG (source, result)// result is approximately 4.6052 end macro_command

Name	LOG10
Syntax	LOG10(source, result)
Description	Calculates the base-10 logarithm of a number. source can be either a variable or a constant. result must be a variable.
Example	macro_command main() float source = 100, result LOG10 (source, result) // result is 2 end macro_command

Name	RAND
Syntax	RAND(result)
Description	Calculates a random integer and save into result. result must be a variable.
Example	macro_command main() short result RAND (result) //result is not a fixed value when executes macro every time end macro_command

18.7.2. Data Transformation

Name	BIN2BCD
Syntax	BIN2BCD(source, result)
Description	Transforms a binary-type value (source) into a BCD-type value (result). source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	
	short source, result
	
	BIN2BCD(1234, result)// result is 0x1234
	
	source = 5678
	BIN2BCD(source, result)// result is 0x5678
	
	end macro_command

Name	BCD2BIN
Syntax	BCD2BIN(source, result)
Description	Transforms a BCD-type value (source) into a binary-type value (result). source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	
	short source, result
	
	BCD2BIN(0x1234, result)// result is 1234
	
	source = 0x5678
	BCD2BIN(source, result)// result is 5678
	
	end macro_command

Name	DEC2ASCII
Syntax	DEC2ASCII(source, result[start], len)
Description	Transforms a decimal value (source) into an ASCII string and save it to an array (result). len represents the length of the string and the unit of length depends on result's type., i.e. if result's type is "char" (the size is byte), the length of the string is (byte * len). If result's type is "short" (the size is word), the length of the string is (word * len), and so on. The first character is put into result[start], the second character is put into result[start + 1], and the last character is put into result[start + (len -1)]. source and len can be a constant or a variable. result must be a variable. start must be a constant.

Example	macro_command main()
	short source
	char result1[4]
	short result2[4]
	char result3[6]
	source = 5678
	
	DEC2ASCII(source, result1[0], 4)
	// result1[0] is '5', result1[1] is '6', result1[2] is '7', result1[3] is '8'
	// the length of the string (result1) is 4 bytes(= 1 * 4)
	DEC2ASCII(source, result2[0], 4)
	// result2[0] is '5', result2[1] is '6', result2[2] is '7', result2[3] is '8'
	// the length of the string (result2) is 8 bytes(= 2 * 4)
	
	source=-123
	DEC2ASCII(source3, result3[0], 6)
	// result1[0] is '-', result1[1] is '0', result1[2] is '0', result1[3] is '1'
	// result1[4] is '2', result1[5] is '3'
	// the length of the string (result1) is 6 bytes(= 1 * 6)
	
	end macro_command

Name	HEX2ASCII
Syntax	HEX2ASCII(source, result[start], len)
Description	Transforms a hexadecimal value (source) into ASCII string saved to an array (result). len represents the length of the string and the unit of length depends on result's type., i.e. if result's type is "char" (the size is byte), the length of the string is (byte * len). If result's type is "short" (the size is word), the length of the string is (word * len), and so on. source and len can be a constant or a variable. result must be a variable. start must be a constant.
Example	macro_command main() short source char result[4] source = 0x5678 HEX2ASCII (source, result[0], 4) // result[0] is '5', result[1] is '6', result[2] is '7', result[3] is '8' end macro_command

Name	FLOAT2ASCII
Syntax	FLOAT2ASCII(source, result[start], len)
Description	Transforms a floating value (source) into ASCII string saved to an array (result). len represents the length of the string and the unit of length depends on result's type., i.e. if result's type is "char" (the size is byte), the length of the string is (byte * len). If result's type is "short" (the size is word), the length of the string is (word * len), and so on. source and len can be a constant or a variable. result must be a variable. start must be a constant.
Example	macro_command main() float source char result[4] source = 56.8 FLOAT2ASCII (source, result[0], 4) // result[0] is '5', result[1] is '6', result[2] is '.', result[3] is '8' end macro_command

Name	ASCII2DEC
Syntax	ASCII2DEC(source[start], result, len)
Description	Transforms a string (source) into a decimal value saved to a variable (result). The length of the string is len. The first character of the string is source[start]. source and len can be a constant or a variable. result must be a variable. start must be a constant.
Example	macro_command main() char source[4] short result source[0] = '5' source[1] = '6' source[2] = '7' source[3] = '8' ASCII2DEC(source[0], result, 4) // result is 5678 end macro_command

Name	ASCII2HEX
Syntax	ASCII2HEX (source[start], result, len)
Description	Transforms a string (source) into a hexadecimal value saved to a variable (result). The length of the string is len. The first character of the string is source[start]. source and len can be a constant or a variable. result must be a variable. start must be a constant.
Example	macro_command main() char source[4] short result source[0] = '5' source[1] = '6' source[2] = '7' source[3] = '8' ASCII2HEX (source[0], result, 4) // result is 0x5678 end macro_command

Name	ASCII2FLOAT
Syntax	ASCII2FLOAT(source[start], result, len)
Description	Transforms a string (source) into a float value saved to a variable (result). The length of the string is len. The first character of the string is source[start]. source and len can be a constant or a variable. result must be a variable. start must be a constant.
Example	macro_command main() char source[4] float result source[0] = '5' source[1] = '6' source[2] = '.' source[3] = '8' ASCII2FLOAT (source[0], result, 4) // result is 56.8 end macro_command

18.7.3. Data Manipulation

Name	FILL
Syntax	FILL(source[start], preset, count)
Description	Sets the first count elements of an array (source) to a specified value (preset). source and start must be a variable, and preset can be a constant or variable.
Example	macro_command main() char result[4] char preset FILL(result[0], 0x30, 4) // result[0] is 0x30, result[1] is 0x30, , result[2] is 0x30, , result[3] is 0x30 preset = 0x31 FILL(result[0], preset, 2) // result[0] is 0x31, result[1] is 0x31 end macro_command

Name	SWAPB
Syntax	SWAPB(source, result)
Description	Exchanges the high-byte and low-byte data of a 16-bit source into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	short source, result
	
	SWAPB(0x5678, result)// result is 0x7856
	
	source = 0x123
	SWAPB(source, result)// result is 0x2301
	
	end macro_command

Name	SWAPW
Syntax	SWAPW(source, result)
Description	Exchanges the high-word and low-word data of a 32-bit source into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	int source, result
	
	SWAPW (0x12345678, result)// result is 0x56781234
	
	source = 0x12345
	SWAPW (source, result)// result is 0x23450001
	
	end macro_command

Name	LOBYTE
Syntax	LOBYTE(source, result)
Description	Retrieves the low byte of a 16-bit source into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	short source, result
	
	LOBYTE(0x1234, result)// result is 0x34
	
	source = 0x123
	LOBYTE(source, result)// result is 0x23
	
	end macro_command

Name	HIBYTE
Syntax	HIBYTE(source, result)
Description	Retrieves the high byte of a 16-bit source into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	short source, result
	
	HIBYTE(0x1234, result)// result is 0x12
	
	source = 0x123
	HIBYTE(source, result)// result is 0x01
	
	end macro_command

Name	LOWORD
Syntax	LOWORD(source, result)
Description	Retrieves the low word of a 32-bit source into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	int source, result
	
	LOWORD(0x12345678, result)// result is 0x5678
	
	source = 0x12345
	LOWORD(source, result)// result is 0x2345
	
	end macro_command

Name	HIWORD
Syntax	HIWORD(source, result)
Description	Retrieves the high word of a 32-bit source into result. source can be a constant or a variable. result must be a variable.

Example	macro_command main()
	int source, result
	
	HIWORD(0x12345678, result)// result is 0x1234
	
	source = 0x12345
	HIWORD(source, result)// result is 0x0001
	
	end macro_command

18.7.4. Bit Transformation

Name	GETBIT
Syntax	GETBIT(source, result, bit_pos)
Description	Gets the state of designated bit position of a data (source) into result. result value will be 0 or 1. source and bit_pos can be a constant or a variable. result must be a variable.

Example	macro_command main()
	int source, result
	short bit_pos
	
	GETBIT(9, result, 3)// result is 1
	
	source = 4
	bit_pos = 2
	GETBIT(source, result, bit_pos)// result is 1
	
	end macro_command

Name	SETBITON
Syntax	SETBITON(source, result, bit_pos)
Description	Changes the state of designated bit position of a data (source) to 1, and put changed data into result. source and bit_pos can be a constant or a variable. result must be a variable.

Example	macro_command main()
	int source, result
	short bit_pos
	
	SETBITON(1, result, 3)// result is 9
	
	source = 0
	bit_pos = 2
	SETBITON (source, result, bit_pos)// result is 4
	
	end macro_command

Name	SETBITOFF
Syntax	SETBITOFF(source, result, bit_pos)
Description	Changes the state of designated bit position of a data (source) to 0, and put in changed data into result. source and bit_pos can be a constant or a variable. result must be a variable.

Example	macro_command main()
	int source, result
	short bit_pos
	
	SETBITOFF(9, result, 3)// result is 1
	
	source = 4
	bit_pos = 2
	SETBITOFF(source, result, bit_pos)// result is 0
	
	end macro_command

Name	INVBIT
Syntax	INVBIT(source, result, bit_pos)
Description	Inverts the state of designated bit position of a data (source), and put changed data into result. source and bit_pos can be a constant or a variable. result must be a variable.

Example	macro_command main()
	int source, result
	short bit_pos
	
	INVBIT(4, result, 1)// result = 6
	
	source = 6
	bit_pos = 1
	INVBIT(source, result, bit_pos)// result = 4
	
	end macro_command

18.7.5. Communication

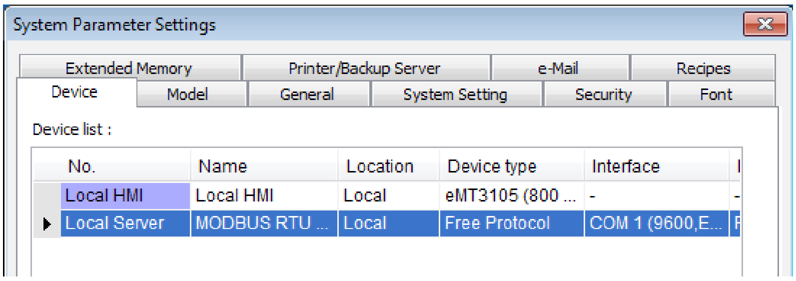
Name	DELAY
Syntax	DELAY(time)
Description	Suspends the execution of the current macro for at least the specified interval (time). The unit of time is millisecond. time can be a constant or a variable.

Example	macro_command main()
	int time == 500
	
	DELAY(100)// delay 100 ms
	DELAY(time)// delay 500 ms
	
	end macro_command

Name	ADDSUM
Syntax	ADDSUM(source[start], result, data_count)
Description	Adds up the elements of an array (source) from source[start] to source[start + data_count - 1] to generate a checksum. Puts in the checksum into result. result must be a variable. data_count is the amount of the accumulated elements and can be a constant or a variable.
Example	macro_command main() char data[5] short checksum data[0] = 0x1 data[1] = 0x2 data[2] = 0x3 data[3] = 0x4 data[4] = 0x5 ADDSUM(data[0], checksum, 5)// checksum is 0xf end macro_command

Name	XORSUM
Syntax	XORSUM(source[start], result, data_count)
Description	Uses an exclusion method to calculate the checksum from source[start] to source[start + data_count - 1]. Puts the checksum into result. result must be a variable. data_count is the amount of the calculated elements of the array and can be a constant or a variable.
Example	macro_command main() char data[5] = {0x1, 0x2, 0x3, 0x4, 0x5} short checksum XORSUM(data[0], checksum, 5)// checksum is 0x1 end macro_command

Name	CRC
Syntax	CRC(source[start], result, data_count)
Description	Calculates 16-bit CRC of the variables from source[start] to source[start + data_count - 1]. Puts in the 16-bit CRC into result. result must be a variable. data_count is the amount of the calculated elements of the array and can be a constant or a variable.
Example	macro_command main() char data[5] = {0x1, 0x2, 0x3, 0x4, 0x5} short 16bit_CRC CRC(data[0], 16bit_CRC, 5)// 16bit_CRC is 0xbb2a end macro_command

<table>
<tr><th>Name</th><th>OUTPORT</th></tr>
<tr><td>Syntax</td><td>OUTPORT(source[start], device_name, data_count)</td></tr>
<tr><td>Description</td><td>Sends out the specified data from source[start] to source[start + count -1] to PLC via a COM port or the ethernet.
device_name is the name of a device defined in the device table and the device must be a “Free Protocol”-type device.
data_count is the amount of sent data and can be a constant or a variable.</td></tr>
<tr><td>Example</td><td>To use an OUTPORT function, a “Free Protocol” device must be created first as follows:


The device is named ”MODBUS RTU Device”. The port attribute depends on the setting of this device. (the current setting is “19200,E, 8, 1”)

Below is an example of executing an action of writing single coil (SET ON) to a MODBUS device.

macro_command main()

char command[32]
short address, checksum

FILL(command[0], 0, 32)// command initialization

command[0] = 0x1// station no
command[1] = 0x5// function code : Write Single Coil

address = 0
HIBYTE(address, command[2])
LOBYTE(address, command[3])

command[4] = 0xff// force bit on
command[5] = 0

CRC(command[0], checksum, 6)

LOBYTE(checksum, command[6])
HIBYTE(checksum, command[7])

// send out a “Write Single Coil” command
OUTPORT(command[0], "MODBUS RTU Device", 8)

end macro_command</td></tr>
</table>

Name	INPORT
Syntax	INPORT(read_data[start], device_name, read_count, return_value)
Description	Reads data from a COM port or the ethernet. These data is stored to read_data[start]~ read_data[start + read_count - 1]. device_name is the name of a device defined in the device table and the device must be a "Free Protocol"-type device. read_count is the required amount of reading and can be a constant or a variable. If the function is used successfully to get sufficient data, return_value is 1, otherwise is 0.

Example	Below is an example of executing an action of reading holding registers of a MODBUS device.
	
	// Read Holding Registers
	macro_command main()
	char command[32], response[32]
	short address, checksum
	short read_no, return_value, read_data[2]
	
	FILL(command[0], 0, 32)// command initialization
	FILL(response[0], 0, 32)
	
	command[0] = 0x1// station no
	command[1] = 0x3// function code : Read Holding Registers
	
	address = 0
	HIBYTE(address, command[2])
	LOBYTE(address, command[3])
	
	read_no = 2// read 2 words (4x_1 and 4x_2)
	HIBYTE(read_no, command[4])
	LOBYTE(read_no, command[5])
	
	CRC(command[0], checksum, 6)
	
	LOBYTE(checksum, command[6])
	HIBYTE(checksum, command[7])
	
	// send out a 'Read Holding Registers" command
	OUTPORT(command[0], "MODBUS RTU Device", 8)
	
	// read responses for a 'Read Holding Registers" command
	INPORT(response[0], "MODBUS RTU Device", 9, return_value)
	
	if return_value > 0 then
	read_data[0] = response[4] + (response[3] << 8)// data in 4x_1
	read_data[1] = response[6] + (response[5] << 8)// data in 4x_2
	
	SetData(read_data[0], "Local HMI", LW, 100, 2)
	end if
	
	end macro_command

Name	INPORT2
Syntax	INPORT2(response[start], device_name, receive_len, wait_time)
Description	Read data from a communication port (COM Port or Ethernet Port). The data read will be saved in response. The description of device_name is the same as OUTPORT. receive_len stores the length of the data received, this must be a variable. receive_len total length can't exceed the size of response. wait_time (in millisecond) can be a constant or variable. After the data is read, if there's no upcoming data during the designated time interval, the function returns.

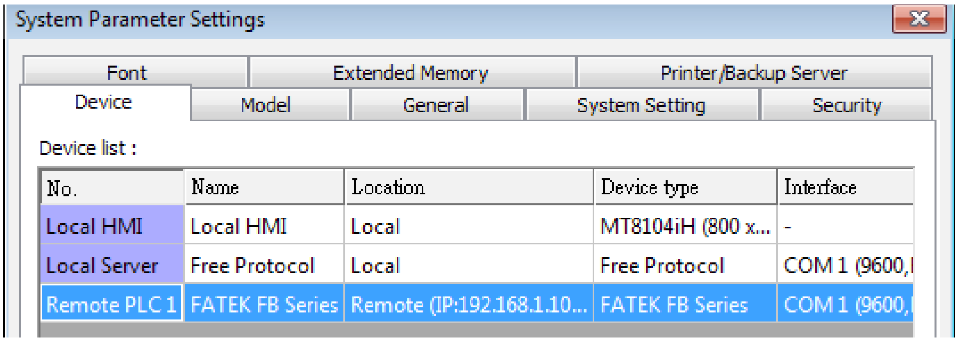
Example	macro_command main()
	
	short wResponse[6], receive_len, wait_time=20
	
	INPORT2(wResponse[0], "Free Protocol", receive_len, wait_time)
	// wait_time unit : millisecond
	
	if receive_len > 0 then
	SetData(wResponse[0], "Local HMI", LW, 0, 6)
	// set responses to LW0 end if
	
	end macro_command

Name	GetData
Syntax	GetData(read_data[start], device_name, device_type, address_offset, data_count) or GetData(read_data, device_name, device_type, address_offset, 1)

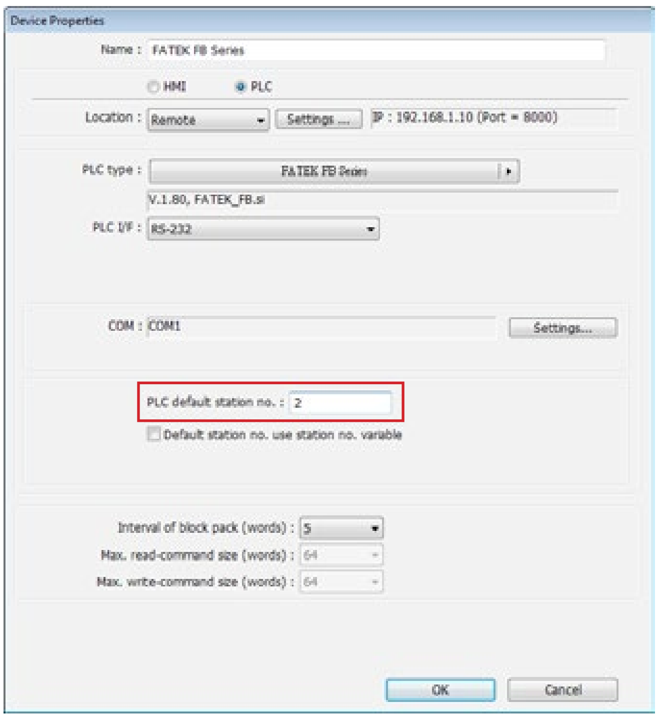
Receives data from the PLC. Data is stored into read_data[start]~ read_data[start + data_count - 1].
data_count is the amount of received data. In general, read_data is an array, but if data_count is 1, read_data can be an array or an ordinary variable. Below are two methods to read one word data from PLC.

```
macro_command main()
short read_data_1[2], read_data_2
GetData(read_data_1[0], "FATEK KB Series", RT, 5, 1)
GetData(read_data_2,   "FATEK KB Series", RT, 5, 1)
end macro_command
```

Device_name is the PLC name enclosed in the double quotation marks (") and this name has been defined in the device list of system parameters as follows (see FATEK KB Series):



Device_type is the device type and encoding method (binary or BCD) of the PLC data. For example, if device_type is LW_BIN, it means the register is LW and the encoding method is binary. If use BIN encoding method, "_BIN" can be ignored.
If device_type is LW_BCD, it means the register is LW and the encoding method is BCD.
Address_offset is the address offset in the PLC.
For example, GetData(read_data_1[0], "FATEK KB Series", RT, 5, 1) represents that the address offset is 5.
If address_offset uses the format – "N#AAAAA", N indicates that PLC's station number is N. AAAAA represents the address offset. This format is used while multiple PLCs or controllers are connected to a single serial port. For example, GetData(read_data_1[0], "FATEK KB Series", RT, 2#5, 1) represents that the PLC's station number is 2. If GetData() uses the default station number defined in the device list as follows, it is not necessary to define station number in address_offset.



	The number of registers actually read from depends on both the type of the read_data variable and the value of the number of data_count.

type of read_data	data_count	actual number of 16-bit register read
char (8-bit)	1	1
char (8-bit)	2	1
bool (8-bit)	1	1
bool (8-bit)	2	1
short (16-bit)	1	1
short (16-bit)	2	2
int (32-bit)	1	2
int (32-bit)	2	4
float (32-bit)	1	2
float (32-bit)	2	4

When a GetData() is executed using a 32-bit data type (int or float), the function will automatically convert the data. For example,

```
macro_command main()
float f
GetData(f, "MODBUS", 6x, 2, 1)   // f will contain a floating point value
end macro_command
```

Example

```
macro_command main()
bool a
bool b[30]
short c
short d[50]
int e
int f[10]
double g[10]

//  get the state of LB2 to the variable a
GetData(a, “Local HMI”, LB, 2, 1)

//  get 30 states of LB0 ~ LB29 to the variables b[0] ~ b[29]
GetData(b[0], “Local HMI”, LB, 0, 30)

//  get one word from LW-2 to the variable c
GetData(c, “Local HMI”, LW, 2, 1)

//  get 50 words from LW-0 ~ LW-49 to the variables d[0] ~ d[49]
GetData(d[0], “Local HMI”, LW, 0, 50)

//  get 2 words from LW-6 ~ LW-7 to the variable e
//  note that the type of e is int
GetData(e, “Local HMI”, LW, 6, 1)

//  get 20 words (10 integer values) from LW-0 ~ LW-19 to variables f[0] ~ f[9]
//  since each integer value occupies 2 words
GetData(f[0], “Local HMI”, LW, 0, 10)

//  get 2 words from LW-2 ~ LW-3 to the variable f
GetData(f, “Local HMI”, LW, 2, 1)

end macro_command
```

Name	GetDataEx
Syntax	GetDataEx(read_data[start], device_name, device_type, address_offset, data_count) or GetDataEx(read_data, device_name, device_type, address_offset, 1)
Description	Receives data from the PLC and continue executing next command even if no response from this device. Descriptions of read_data, device_name, device_type, address_offset and data_count are the same as GetData.

Example	macro_command main()
	bool a
	bool b[30]
	short c
	short d[50]
	int e
	int f[10]
	double g[10]
	
	// get the state of LB2 to the variable a
	GetDataEx (a, “Local HMI”, LB, 2, 1)
	
	// get 30 states of LB0 ~ LB29 to the variables b[0] ~ b[29]
	GetDataEx (b[0], “Local HMI”, LB, 0, 30)
	
	// get one word from LW-2 to the variable c
	GetDataEx (c, “Local HMI”, LW, 2, 1)
	
	// get 50 words from LW-0 ~ LW-49 to the variables d[0] ~ d[49]
	GetDataEx (d[0], “Local HMI”, LW, 0, 50)
	
	// get 2 words from LW-6 ~ LW-7 to the variable e
	// note that he type of e is int
	GetDataEx (e, “Local HMI”, LW, 6, 1)
	
	// get 20 words (10 integer values) from LW-0 ~ LW-19 to f[0] ~ f[9]
	// since each integer value occupies 2 words
	GetDataEx (f[0], “Local HMI”, LW, 0, 10)
	
	// get 2 words from LW-2 ~ LW-3 to the variable f
	GetDataEx (f, “Local HMI”, LW, 2, 1)
	
	end macro_command

<table>
<tr><th>Name</th><td>SetData</td></tr>
<tr><th>Syntax</th><td>SetData(send_data[start], device_name, device_type, address_offset, data_count)
or
SetData(send_data, device_name, device_type, address_offset, 1)</td></tr>
<tr><th>Description</th><td>Send data to the PLC. Data is defined in send_data[start]~ send_data[start + data_count - 1].
data_count is the amount of sent data. In general, send_data is an array, but if data_count is 1, send_data can be an array or an ordinary variable. Below are two methods to send one word data.

macro_command main()
short send_data_1[2] = { 5, 6}, send_data_2 = 5
SetData(send_data_1[0], “FATEK KB Series”, RT, 5, 1)
SetData(send_data_2, “FATEK KB Series”, RT, 5, 1)
end macro_command

device_name is the PLC name enclosed in the double quotation marks (“) and this name has been defined in the device list of system parameters.
device_type is the device type and encoding method (binary or BCD) of the PLC data. For example, if device_type is LW_BIN, it means the register is LW and the encoding method is binary. If use BIN encoding method, “_BIN” can be ignored.
If device_type is LW_BCD, it means the register is LW and the encoding method is BCD.
address_offset is the address offset in the PLC.
For example, SetData(read_data_1[0], “FATEK KB Series”, RT, 5, 1) represents that the address offset is 5.
If address_offset uses the format – “N#AAAAA”, N indicates that PLC’s station number is N. AAAAA represents the address offset. This format is used while multiple PLCs or controllers are connected to a single serial port. For example, SetData(read_data_1[0], “FATEK KB Series”, RT, 2#5, 1) represents that the PLC’s station number is 2. If SetData () uses the default station number defined in the device list, it is not necessary to define station number in address_offset.

The number of registers actually sends to depends on both the type of the send_data variable and the value of the number of data_count.

<table>
<tr><th>type of read_data</th><th>data_count</th><th>actual number of 16-bit register send</th></tr>
<tr><td>char (8-bit)</td><td>1</td><td>1</td></tr>
<tr><td>char (8-bit)</td><td>2</td><td>1</td></tr>
<tr><td>bool (8-bit)</td><td>1</td><td>1</td></tr>
<tr><td>bool (8-bit)</td><td>2</td><td>1</td></tr>
<tr><td>short (16-bit)</td><td>1</td><td>1</td></tr>
<tr><td>short (16-bit)</td><td>2</td><td>2</td></tr>
<tr><td>int (32-bit)</td><td>1</td><td>2</td></tr>
<tr><td>int (32-bit)</td><td>2</td><td>4</td></tr>
<tr><td>float (32-bit)</td><td>1</td><td>2</td></tr>
<tr><td>float (32-bit)</td><td>2</td><td>4</td></tr>
</table>

When a SetData() is executed using a 32-bit data type (int or float), the function will automatically send int-format or float-format data to the device. For example,

macro_command main()
float f = 2.6
SetData(f, "MODBUS", 6x, 2, 1)   // will send a floating point value to the device
end macro_command</td></tr>
</table>

Example	<pre>macro_command main()
	int i
	bool a = true
	bool b[30]
	short c = false
	short d[50]
	int e = 5
	int f[10]
	
	for i = 0 to 29
	b[i] = true
	next i
	
	for i = 0 to 49
	d[i] = i * 2
	next i
	
	for i = 0 to 9
	f [i] = i * 3
	next i
	
	// set the state of LB2
	SetData(a, "Local HMI", LB, 2, 1)
	
	// set the states of LB0 ~ LB29
	SetData(b[0], "Local HMI", LB, 0, 30)
	
	// set the value of LW-2
	SetData(c, "Local HMI", LW, 2, 1)
	
	// set the values of LW-0 ~ LW-49
	SetData(d[0], "Local HMI", LW, 0, 50)
	
	// set the values of LW-6 ~ LW-7, note that the type of e is int
	SetData(e, "Local HMI", LW, 6, 1)
	
	// set the values of LW-0 ~ LW-19
	// 10 integers equal to 20 words, since each integer value occupies 2 words.
	SetData(f[0], "Local HMI", LW, 0, 10)
	
	end macro_command</pre>

Name	SetDataEx
Syntax	SetDataEx (send_data[start], device_name, device_type, address_offset, data_count) or SetDataEx (send_data, device_name, device_type, address_offset, 1)
Description	Send data to the PLC and continue executing next command even if no response from this device. Descriptions of send_data, device_name, device_type, address_offset and data_count are the same as SetData.

Example	macro_command main()
	int i
	bool a = true
	bool b[30]
	short c = false
	short d[50]
	int e = 5
	int f[10]
	
	for i = 0 to 29
	b[i] = true
	next i
	
	for i = 0 to 49
	d[i] = i * 2
	next i
	
	for i = 0 to 9
	f [i] = i * 3
	next i
	
	// set the state of LB2
	SetDataEx (a, “Local HMI”, LB, 2, 1)
	
	// set the states of LB0 ~ LB29
	SetDataEx (b[0], “Local HMI”, LB, 0, 30)
	
	// set the value of LW-2
	SetDataEx (c, “Local HMI”, LW, 2, 1)
	
	// set the values of LW-0 ~ LW-49
	SetDataEx (d[0], “Local HMI”, LW, 0, 50)
	
	// set the values of LW-6 ~ LW-7, note that the type of e is int
	SetDataEx (e, “Local HMI”, LW, 6, 1)
	
	// set the values of LW-0 ~ LW-19
	// 10 integers equal to 20 words, since each integer value occupies 2 words.
	SetDataEx (f[0], “Local HMI”, LW, 0, 10)
	
	end macro_command

Name	GetError
Syntax	GetError (err)
Description	Get an error code.

Example	macro_command main()
	short err
	char byData[10]
	
	GetDataEx(byData[0], “MODBUS RTU”, 4x, 1, 10)// read 10 bytes
	
	// if err is equal to 0, it is successful to execute GetDataEx()
	GetErr(err)// save an error code to err
	
	end macro_command

Name	PURGE
Syntax	PURGE (com_port)
Description	com_port refers to the COM port number which ranges from 1 to 3. It can be either a variable or a constant. This function is used to clear the input and output buffers associated with the COM port.
Example	macro_command main() int com_port=3 PURGE (com_port) PURGE (1) end macro_command

Name	SetRTS
Syntax	SetRTS(com_port, source)
Description	Set RTS state for RS232. com_port refers to the COM port number. It can be either a variable or a constant. source can be either a variable or a constant. This command raise RTS signal while the value of source is greater than 0 and lower RTS signal while the value of source equals to 0.

Example	macro_command main()
	char com_port=1
	char value=1
	
	SetRTS(com_port, value) // raise RTS signal of COM1 while value>0
	
	SetRTS(1, 0) // lower RTS signal of COM1
	
	end macro_command

Name	GetCTS
Syntax	GetCTS(com_port, result)
Description	Get CTS state for RS232. com_port refers to the COM port number. It can be either a variable or a constant. result is used for receiving the CTS signal. It must be a variable. This command receives CTS signal and stores the received data in the result variable. When the CTS signal is pulled high, it writes 1 to result, otherwise, it writes 0.

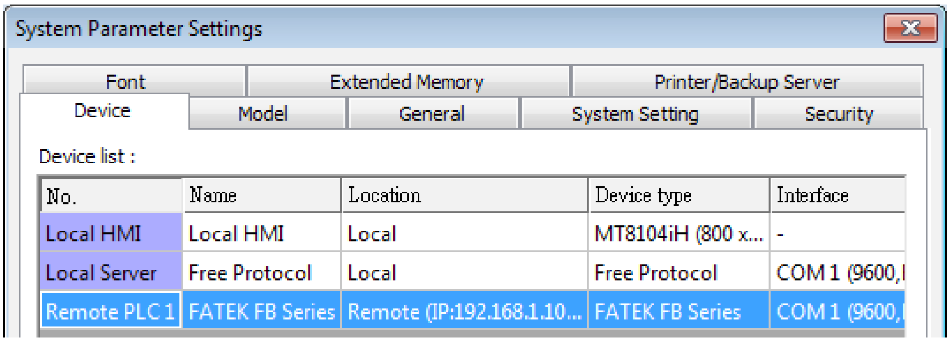
Example	macro_command main()
	char com_port=1
	char result
	
	GetCTS(com_port, result) // get CTS signal of COM1
	
	GetCTS (1, result) // get CTS signal of COM1
	
	end macro_command

18.7.6. String Operation Functions

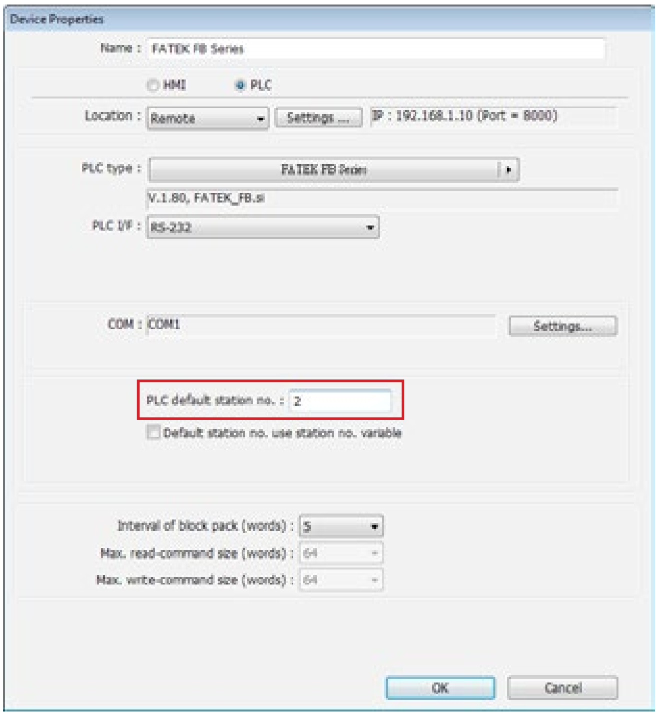
Name	StringGet
Syntax	StringGet(read_data[start], device_name, device_type, address_offset, data_count)

Description

Receives data from the PLC. The String data is stored into read_data[start]~ read_data[start + data_count - 1]. read_data must be a one-dimensional char array.
Data_count is the number of received characters, it can be either a constant or a variable.
Device_name is the PLC name enclosed in the double quotation marks (") and this name has been defined in the device list of system parameters as follows (see FATEK KB Series):



Device_type is the device type and encoding method (binary or BCD) of the PLC data. For example, if device_type is LW_BIN, it means the register is LW and the encoding method is binary. If use BIN encoding method, "_BIN" can be ignored.
If device_type is LW_BCD, it means the register is LW and the encoding method is BCD.
Address_offset is the address offset in the PLC.
For example, StringGet(read_data_1[0], "FATEK KB Series", RT, 5, 1) represents that the address offset is 5.
If address_offset uses the format – "N#AAAAA", N indicates that PLC's station number is N. AAAAA represents the address offset. This format is used while multiple PLCs or controllers are connected to a single serial port. For example, StringGet(read_data_1[0], "FATEK KB Series", RT, 2#5, 1) represents that the PLC's station number is 2. If StringGet() uses the default station number defined in the device list as follows, it is not necessary to define station number in address_offset.



The number of registers actually read from depends on the value of the number of data_count since that the read_data is restricted to char array.

Type of read_data	Data_count	Actual number of 16-bit register read
char (8-bit)	1	1
char (8-bit)	2	1

1 WORD register(16-bit) equals to the size of 2 ASCII characters. According to the above table, reading 2 ASCII characters is actually reading the content of one 16-bit register.

Example	macro_command main() char str1[20] // read 10 words from LW-0~LW-9 to the variables str1[0] to str1[19] // since that 1 word can store 2 ASCII characters, reading 20 ASCII // characters is actually reading 10 words of register StringGet(str1[0], “Local HMI”, LW, 0, 20) end macro_command

Name	StringGetEx
Syntax	StringGetEx (read_data[start], device_name, device_type, address_offset, data_count)
Description	Receives data from the PLC and continue executing next command even if no response from this device. Descriptions of read_data, device_name, device_type, address_offset and data_count are the same as GetData.

Example	macro_command main()
	char str1[20]
	short test=0
	
	// macro will continue executing test = 1 even if the MODBUS device is
	// not responding
	StringGetEx(str1[0], "MODBUS RTU", 4x, 0, 20)
	test = 1
	
	// macro will not continue executing test = 2 until MODBUS device responds
	StringGet(str1[0], "MODBUS RTU", 4x, 0, 20)
	test = 2
	
	end macro_command

Name	StringSet
Syntax	StringSet(send_data[start], device_name, device_type, address_offset, data_count)

Description

Send data to the PLC. Data is defined in send_data[start]~ send_data[start + data_count - 1]. send_data must be a one-dimensional char array.
data_count is the number of sent characters, it can be either a constant or a variable.
device_name is the PLC name enclosed in the double quotation marks (“) and this name has been defined in the device list of system parameters.
device_type is the device type and encoding method (binary or BCD) of the PLC data. For example, if device_type is LW_BIN, it means the register is LW and the encoding method is binary. If use BIN encoding method, “_BIN” can be ignored.
If device_type is LW_BCD, it means the register is LW and the encoding method is BCD.
address_offset is the address offset in the PLC.
For example, StringSet(read_data_1[0], “FATEK KB Series”, RT, 5, 1) represents that the address offset is 5.
If address_offset uses the format – “N#AAAAA”, N indicates that PLC’s station number is N. AAAAA represents the address offset. This format is used while multiple PLCs or controllers are connected to a single serial port. For example, StringSet(read_data_1[0], “FATEK KB Series”, RT, 2#5, 1) represents that the PLC’s station number is 2.
If SetData () uses the default station number defined in the device list, it is not necessary to define station number in address_offset.

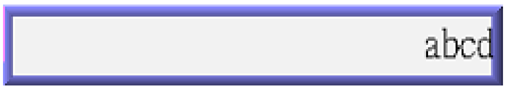
The number of registers actually sends to depends on the value of the number of data_count, since that send_data is restricted to char array.

Type of read_data	Data_count	Actual number of 16-bit register send
char (8-bit)	1	1
char (8-bit)	2	1

1 WORD register(16-bit) equals to the size of 2 ASCII characters. According to the above table, sending 2 ASCII characters is actually writing to one 16-bit register. The ASCII characters are stored into the WORD register from low byte to high byte. While using the ASCII Display object to display the string data stored in the registers, data_count must be a multiple of 2 in order to display full string content. For example:

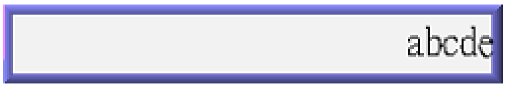
```
macro_command main()
char src1[10]="abcde"
StringSet(src1[0], "Local HMI", LW, 0, 5)
end macro_command
```

The ASCII Display object shows:



If data_count is an even number that is greater than or equal to the length of the string, the content of string can be completely shown:

```
macro_command main()
char src1[10]="abcde"
StringSet(src1[0], "Local HMI", LW, 0, 6)
end macro_command
```



Example

```
macro_command main()

char str1[10]=”abcde”

//  Send 3 words to LW-0~LW-2
//  Data are being sent until the end of string is reached.
//  Even though the value of data_count is larger than the length of string
//  , the function will automatically stop.
StringSet(str1[0], "Local HMI", LW, 0, 10)

end macro_command
```

Name	StringSetEx
Syntax	StringSetEx (send_data[start], device_name, device_type, address_offset, data_count)
Description	Send data to the PLC and continue executing next command even if no response from this device. Descriptions of send_data, device_name, device_type, address_offset and data_count are the same as StringSet.

Example	macro_command main()
	char str1[20]="abcde"
	short test=0
	
	// macro will continue executing test = 1 even if the MODBUS device is
	// not responding
	StringSetEx(str1[0], "MODBUS RTU", 4x, 0, 20)
	test = 1
	
	// macro will not continue executing test = 2 until MODBUS device responds
	StringSet(str1[0], "MODBUS RTU", 4x, 0, 20)
	test = 2
	
	end macro_command

Name	StringCopy
Syntax	success = StringCopy ("source", destination[start]) or success = StringCopy (source(start), destination[start])
Description	Copy one string to another. This function copies a static string (which is enclosed in quotes) or a string that is stored in an array to the destination buffer. The source string parameter accepts both static string (in the form: "source") and char array (in the form: source[start]). destination[start] must be an one-dimensional char array. This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of source string exceeds the max. size of destination buffer, it returns false and the content of destination remains the same. The success field is optional.

Example	macro_command main()
	char src1[5[="abcde"
	char dest1]5]
	bool success1
	success1 = StringCopy(src1[0], dest1[0])
	// success1=true, dest1="abcde"
	
	char dest2[5]
	bool success2
	success2 = StringCopy("12345", dest2[0])
	// success2=true, dest2="12345"
	
	char src3[10]="abcdefghij"
	char dest3[5]
	bool success3
	success3 = StringCopy(src3[0], dest3[0])
	// success3=false, dest3 remains the same.
	
	char src4[10]="abcdefghij"
	char dest4[5]
	bool success4
	success4 = StringCopy(src4[5], dest4[0])
	// success4=true, dest4="fghij"
	
	end macro_command

Name	StringDecAsc2Bin
Syntax	success = StringDecAsc2Bin(source[start], destination) or success = StringDecAsc2Bin(“source”, destination)
Description	This function converts a decimal string to an integer. It converts the decimal string in source parameter into an integer, and stores it in the destination variable. The source string parameter accepts both static string (in the form: “source”) and char array (in the form: source[start]). Destination must be a variable, to store the result of conversion. This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the source string contains characters other than ‘0’ to ‘9’, it returns false. The success field is optional.

Example	macro_command main()
	char src1[5]="12345"
	int result1
	bool success1
	success1 = StringDecAsc2Bin(src1[0], result1)
	// success1=true, result1 is 12345
	
	char result2
	bool success2
	success2 = StringDecAsc2Bin("32768", result2)
	// success2=true, but the result exceeds the data range of result2
	
	char src3[2]="4b"
	char result3
	bool success3
	success3 = StringDecAsc2Bin (src3[0], result3)
	// success3=false, because src3 contains characters other than ‘0’ to ‘9’
	
	end macro_command

Name	StringBin2DecAsc
Syntax	success = StringBin2DecAsc (source, destination[start])
Description	This function converts an integer to a decimal string. It converts the integer in source parameter into a decimal string, and stores it in the destination buffer. Source can be either a constant or a variable. Destination must be an one-dimensional char array, to store the result of conversion. This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of decimal string after conversion exceeds the size of destination buffer, it returns false. The success field is optional.

Example	macro_command main()
	int src1 = 2147483647
	char dest1[20]
	bool success1
	success1 = StringBin2DecAsc(src1, dest1[0])
	// success1=true, dest1=”2147483647”
	
	short src2 = 0x3c
	char dest2[20]
	bool success2
	success2 = StringBin2DecAsc(src2, dest2[0])
	// success2=true, dest2="60"
	
	int src3 = 2147483647
	char dest3[5]
	bool success3
	success3 = StringBin2DecAsc(src3, dest3[0])
	// success3=false, dest3 remains the same.
	
	end macro_command

Name	StringDecAsc2Float
Syntax	success = StringDecAsc2Float (source[start], destination) or success = StringDecAsc2Float (“source”, destination)
Description	This function converts a decimal string to floats. It converts the decimal string in source parameter into float, and stores it in the destination variable. The source string parameter accepts both static string (in the form: “source”) and char array (in the form: source[start]). Destination must be a variable, to store the result of conversion. This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the source string contains characters other than ‘0’ to ‘9’ or ‘.’, it returns false. The success field is optional.

Example	macro_command main()
	char src1[10]="12.345"
	float result1
	bool success1
	success1 = StringDecAsc2Float(src1[0], result1)
	// success1=true, result1 is 12.345
	
	float result2
	bool success2
	success2 = StringDecAsc2Float("1.234567890", result2)
	// success2=true, but the result exceeds the data range of result2, which
	// might result in loss of precision
	
	char src3[2]="4b"
	float result3
	bool success3
	success3 = StringDecAsc2Float(src3[0], result3)
	// success3=false, because src3 contains characters other than ‘0’ to ‘9’ or
	// ‘.’
	end macro_command

Name	StringFloat2DecAsc
Syntax	success = StringFloat2DecAsc(source, destination[start])
Description	This function converts a float to a decimal string. It converts the float in source parameter into a decimal string, and stores it in the destination buffer. Source can be either a constant or a variable. Destination must be an one-dimensional char array, to store the result of conversion. This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of decimal string after conversion exceeds the size of destination buffer, it returns false. The success field is optional.

Example	macro_command main()
	float src1 = 1.2345
	char dest1[20]
	bool success1
	success1 = StringFloat2DecAsc(src1, dest1[0])
	// success1=true, dest1="1.2345"
	
	float src2 = 1.23456789
	char dest2 [20]
	bool success2
	success2 = StringFloat2DecAsc(src2, dest2 [0])
	// success2=true, but it might lose precision
	
	float src3 = 1.2345
	char dest3[5]
	bool success3
	success3 = StringFloat2DecAsc(src3, dest3 [0])
	// success3=false, dest3 remains the same.
	
	end macro_command

Name	StringHexAsc2Bin
Syntax	success = StringHexAsc2Bin (source[start], destination) or success = StringHexAsc2Bin (“source”, destination)
Description	This function converts a hexadecimal string to binary data. It converts the hexadecimal string in source parameter into binary data, and stores it in the destination variable. The source string parameter accepts both static string (in the form: “source”) and char array (in the form: source[start]). Destination must be a variable, to store the result of conversion. This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the source string contains characters other than ‘0’ to ‘9’, ‘a’ to ‘f’ or ‘A’ to ‘F’, it returns false. The success field is optional.

Example	macro_command main()
	char src1[5]="0x3c"
	int result1
	bool success1
	success1 = StringHexAsc2Bin(src1[0], result1)
	// success1=true, result1 is 3c
	
	short result2
	bool success2
	success2 = StringDecAsc2Bin("1a2b3c4d", result2)
	// success2=true, result2=3c4d.The result exceeds the data range of
	// result2
	
	char src3[2]="4g"
	char result3
	bool success3
	success3 = StringDecAsc2Bin (src3[0], result3)
	// success3=false, because src3 contains characters other than ‘0’ to ‘9’
	// , ‘a’ to ‘f’ or ‘A’ to ‘F’
	
	end macro_command

Name	StringBin2HexAsc
Syntax	success = StringBin2HexAsc (source, destination[start])
Description	This function converts binary data to a hexadecimal string. It converts the binary data in source parameter into a hexadecimal string, and stores it in the destination buffer. Source can be either a constant or a variable. Destination must be an one-dimensional char array, to store the result of conversion. This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of hexadecimal string after conversion exceeds the size of destination buffer, it returns false. The success field is optional.

Example	macro_command main()
	int src1 = 20
	char dest1[20]
	bool success1
	success1 = StringBin2HexAsc(src1, dest1[0])
	// success1=true, dest1="14"
	
	short src2 = 0x3c
	char dest2[20]
	bool success2
	success2 = StringBin2HexAsc(src2, dest2[0])
	// success2=true, dest2="3c"
	
	int src3 = 0x1a2b3c4d
	char dest3[6]
	bool success3
	success3 = StringBin2HexAsc(src3, dest3[0])
	// success3=false, dest3 remains the same.
	
	end macro_command

Name	StringMid
Syntax	success = StringMid (source[start], count, destination[start]) or success = StringMid (“string”, start, count, destination[start])
Description	Retrieve a character sequence from the specified offset of the source string and store it in the destination buffer. The source string parameter accepts both static string (in the form: “source”) and char array (in the form: source[start]). For source[start], the start offset of the substring is specified by the index value. For static source string(“source”), the second parameter(start) specifies the start offset of the substring. The count parameter specifies the length of substring being retrieved. Destination must be an one-dimensional char array, to store the retrieved substring. This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of retrieved substring exceeds the size of destination buffer, it returns false. The success field is optional.

Example	macro_command main()
	char src1[20]="abcdefghijklmnopqrst"
	char dest1[20]
	bool success1
	success1 = StringMid(src1[5], 6, dest1[0])
	// success1=true, dest1="fghijk"
	
	char src2[20]="abcdefghijklmnopqrst"
	char dest2[5]
	bool success2
	success2 = StringMid(src2[5], 6, dest2[0])
	// success2=false, dest2 remains the same.
	
	char dest3[20]="12345678901234567890"
	bool success3
	success3 = StringMid("abcdefghijklmnopqrst", 5, 5, dest3[15])
	// success3= true, dest3="123456789012345fghij"
	
	end macro_command

Name	StringLength
Syntax	length = StringLength (source[start]) or length = StringLength (“source”)
Description	Obtain the length of a string. It returns the length of source string and stores it in the length field on the left-hand side of ‘=’ operator. The source string parameter accepts both static string (in the form: “source”) and char array (in the form: source[start]). The return value of this function indicates the length of the source string.

Example	macro_command main()
	char src1[20]="abcde"
	int length1
	length1= StringLength(src1[0])
	// length1=5
	
	char src2[20]={'a', 'b', 'c', 'd', 'e'}
	int length2
	length2= StringLength(src2[0])
	// length2=20
	
	char src3[20]="abcdefghij"
	int length3
	length3= StringLength(src3 [2])
	// length3=8
	
	end macro_command

Name	StringCat
Syntax	success = StringCat (source(start), destination[start]) or success = StringCat (“source”, destination[start])
Description	This function appends source string to destination string. It adds the contents of source string to the last of the contents of destination string. The source string parameter accepts both static string (in the form: “source”) and char array (in the form: source[start]). Destination must be an one-dimensional char array. This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of result string after concatenation exceeds the max. size of destination buffer, it returns false. The success field is optional.

Example	macro_command main()
	char src1[20]="abcdefghij"
	char dest1[20]="1234567890"
	bool success1
	success1= StringCat(src1[0], dest1[0])
	// success1=true, dest1="123456790abcdefghij"
	
	char dest2 [10]="1234567890"
	bool success2
	success2= StringCat("abcde", dest2 [0])
	// success2=false, dest2 remains the same.
	
	char src3[20]="abcdefghij"
	char dest3[20]
	bool success3
	success3= StringCat(src3[0], dest3[15])
	// success3=false, dest3 remains the same.
	
	end macro_command

Name	StringCompare
Syntax	ret = StringCompare (str1[start], str2[start]) ret = StringCompare (“string1”, str2[start]) ret = StringCompare (str1[start], “string2”) ret = StringCompare (“string1”, “string2”)
Description	Do a case-sensitive comparison of two strings. The two string parameters accept both static string (in the form: “string1”) and char array (in the form: str1[start]). This function returns a Boolean indicating the result of comparison. If two strings are identical, it returns true. Otherwise it returns false. The ret field is optional.

Example	macro_command main()
	char a1[20]="abcde"
	char b1[20]="ABCDE"
	bool ret1
	ret1= StringCompare(a1[0], b1[0])
	// ret1=false
	
	char a2[20]="abcde"
	char b2[20]="abcde"
	bool ret2
	ret2= StringCompare(a2[0], b2[0])
	// ret2=true
	
	char a3 [20]="abcde"
	char b3[20]="abcdefg"
	bool ret3
	ret3= StringCompare(a3[0], b3[0])
	// ret3=false
	
	end macro_command

Name	StringCompareNoCase
Syntax	ret = StringCompareNoCase(str1[start], str2[start]) ret = StringCompareNoCase("string1", str2[start]) ret = StringCompareNoCase(str1[start], "string2") ret = StringCompareNoCase("string1", "string2")
Description	Do a case-insensitive comparison of two strings. The two string parameters accept both static string (in the form: "string1") and char array (in the form: str1[start]). This function returns a Boolean indicating the result of comparison. If two strings are identical, it returns true. Otherwise it returns false. The ret field is optional.

Example	macro_command main()
	char a1[20]="abcde"
	char b1[20]="ABCDE"
	bool ret1
	ret1= StringCompareNoCase(a1[0], b1[0])
	// ret1=true
	
	char a2[20]="abcde"
	char b2[20]="abcde"
	bool ret2
	ret2= StringCompareNoCase(a2[0], b2[0])
	// ret2=true
	
	char a3 [20]="abcde"
	char b3[20]="abcdefg"
	bool ret3
	ret3= StringCompareNoCase(a3[0], b3[0])
	// ret3=false
	
	end macro_command

Name	StringFind
Syntax	position = StringFind (source[start], target[start]) position = StringFind ("source", target[start]) position = StringFind (source[start], "target") position = StringFind ("source", "target")
Description	Return the position of the first occurrence of target string in the source string. The two string parameters accept both static string (in the form: "source") and char array (in the form: source[start]). This function returns the zero-based index of the first character of substring in the source string that matches the target string. Notice that the entire sequence of characters to find must be matched. If there is no matched substring, it returns -1.

Example	macro_command main()
	char src1[20]="abcde"
	char target1[20]="cd"
	bool pos1
	pos1= StringFind(src1[0], target1[0])
	// pos1=2
	
	char target2[20]="ce"
	bool pos2
	pos2= StringFind("abcde", target2[0])
	// pos2=-1
	
	char src3[20]="abcde"
	bool pos3
	pos3= StringFind(src3[3], "cd")
	// pos3=-1
	
	end macro_command

Name	StringReverseFind
Syntax	position = StringReverseFind (source[start], target[start]) position = StringReverseFind (“source”, target[start]) position = StringReverseFind (source[start], “target”) position = StringReverseFind (“source”, “target”)
Description	Return the position of the last occurrence of target string in the source string. The two string parameters accept both static string (in the form: “source”) and char array (in the form: source[start]). This function returns the zero-based index of the first character of substring in the source string that matches the target string. Notice that the entire sequence of characters to find must be matched. If there exists multiple substrings that matches the target string, function will return the position of the last matched substring. If there is no matched substring, it returns -1.

Example	macro_command main()
	char src1[20]="abcdeabcde"
	char target1[20]="cd"
	bool pos1
	pos1= StringReverseFind(src1[0], target1[0])
	// pos1=7
	
	char target2[20]="ce"
	bool pos2
	pos2= StringReverseFind("abcdeabcde", target2[0])
	// pos2=-1
	
	char src3[20]="abcdeabcde"
	bool pos3
	pos3= StringReverseFind(src3[6], "ab")
	// pos3=-1
	
	end macro_command

Name	StringFindOneOf
Syntax	position = StringFindOneOf (source[start], target[start]) position = StringFindOneOf (“source”, target[start]) position = StringFindOneOf (source[start], “target”) position = StringFindOneOf (“source”, “target”)
Description	Return the position of the first character in the source string that matches any character contained in the target string. The two string parameters accept both static string (in the form: “source”) and char array (in the form: source[start]). This function returns the zero-based index of the first character in the source string that is also in the target string. If there is no match, it returns -1.

Example	macro_command main()
	char src1[20]="abcdeabcde"
	char target1[20]="sdf"
	bool pos1
	pos1= StringFindOneOf(src1[0], target1[0])
	// pos1=3
	
	char src2[20]="abcdeabcde"
	bool pos2
	pos2= StringFindOneOf(src2[1], "agi")
	// pos2=4
	
	char target3 [20]="bus"
	bool pos3
	pos3= StringFindOneOf("abcdeabcde", target3[1])
	// pos3=-1
	
	end macro_command

Name	StringIncluding
Syntax	success = StringIncluding (source[start], set[start], destination[start]) success = StringIncluding (“source”, set[start], destination[start]) success = StringIncluding (source[start], “set”, destination[start]) success = StringIncluding (“source”, “set”, destination[start])
Description	Retrieve a substring of the source string that contains characters in the set string, beginning with the first character in the source string and ending when a character is found in the source string that is not in the target string. The source string and set string parameters accept both static string (in the form: “source”) and char array (in the form: source[start]). This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of retrieved substring exceeds the size of destination buffer, it returns false.

Example	macro_command main()
	char src1[20]="cabbageabc"
	char set1[20]="abc"
	char dest1[20]
	bool success1
	success1 = StringIncluding(src1[0], set1[0], dest1[0])
	// success1=true, dest1="cabba"
	
	char src2[20]="gecabba"
	char dest2[20]
	bool success2
	success2 = StringIncluding(src2[0], "abc", dest2[0])
	// success2=true, dest2=""
	
	char set3[20]="abc"
	char dest3[4]
	bool success3
	success3 = StringIncluding("cabbage", set3[0], dest3[0])
	// success3=false, dest3 remains the same.
	
	end macro_command

Name	StringExcluding
Syntax	success = StringExcluding (source[start], set[start], destination[start]) success = StringExcluding (“source”, set[start], destination[start]) success = StringExcluding (source[start], “set”, destination[start]) success = StringExcluding (“source”, “set”, destination[start])
Description	Retrieve a substring of the source string that contains characters that are not in the set string, beginning with the first character in the source string and ending when a character is found in the source string that is also in the target string. The source string and set string parameters accept both static string (in the form: “source”) and char array (in the form: source[start]). This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of retrieved substring exceeds the size of destination buffer, it returns false.

Example	macro_command main()
	char src1[20]="cabbageabc"
	char set1[20]="ge"
	char dest1[20]
	bool success1
	success1 = StringExcluding(src1[0], set1[0], dest1[0])
	// success1=true, dest1="cabba"
	
	char src2[20]="cabbage"
	char dest2[20]
	bool success2
	success2 = StringExcluding(src2[0], "abc", dest2[0])
	// success2=true, dest2=""
	
	char set3[20]="ge"
	char dest3[4]
	bool success3
	success3 = StringExcluding("cabbage", set3[0], dest3[0])
	// success3=false, dest3 remains the same.
	
	end macro_command

Name	StringToUpper
Syntax	success = StringToUpper (source[start], destination[start]) success = StringToUpper (“source”, destination[start])
Description	Convert all the characters in the source string to uppercase characters and store the result in the destination buffer. The source string parameter accepts both static string (in the form: “source”) and char array (in the form: source[start]). This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of result string after conversion exceeds the size of destination buffer, it returns false.

Example	macro_command main()
	char src1[20]="aBcDe"
	char dest1[20]
	bool success1
	success1 = StringToUpper(src1[0], dest1[0])
	// success1=true, dest1="ABCDE"
	
	char dest2[4]
	bool success2
	success2 = StringToUpper("aBcDe", dest2[0])
	// success2=false, dest2 remains the same.
	
	end macro_command

Name	StringToLower
Syntax	success = StringToLower (source[start], destination[start]) success = StringToLower (“source”, destination[start])
Description	Convert all the characters in the source string to lowercase characters and store the result in the destination buffer. The source string parameter accepts both static string (in the form: "source") and char array (in the form: source[start]). This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of result string after conversion exceeds the size of destination buffer, it returns false.

Example	macro_command main()
	char src1[20]="aBcDe"
	char dest1[20]
	bool success1
	success1 = StringToUpper(src1[0], dest1[0])
	// success1=true, dest1="abcde"
	
	char dest2[4]
	bool success2
	success2 = StringToUpper("aBcDe", dest2[0])
	// success2=false, dest2 remains the same.
	
	end macro_command

Name	StringToReverse
Syntax	success = StringToReverse (source[start], destination[start]) success = StringToReverse (“source”, destination[start])
Description	Reverse the characters in the source string and store it in the destination buffer. The source string parameter accepts both static string (in the form: "source") and char array (in the form: source[start]). This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of reversed string exceeds the size of destination buffer, it returns false.

Example	macro_command main()
	char src1[20]="abcde"
	char dest1[20]
	bool success1
	success1 = StringToUpper(src1[0], dest1[0])
	// success1=true, dest1="edcba"
	
	char dest2[4]
	bool success2
	success2 = StringToUpper("abcde", dest2[0])
	// success2=false, dest2 remains the same.
	
	end macro_command

Name	StringTrimLeft
Syntax	success = StringTrimLeft (source[start], set[start], destination[start]) success = StringTrimLeft (“source”, set[start], destination[start]) success = StringTrimLeft (source[start], “set”, destination[start]) success = StringTrimLeft (“source”, “set”, destination[start])
Description	Trim the leading specified characters in the set buffer from the source string. The source string and set string parameters accept both static string (in the form: “source”) and char array (in the form: source[start]). This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of trimmed string exceeds the size of destination buffer, it returns false.

Example	macro_command main()
	char src1[20]= "# *a*#bc"
	char set1[20]="# *"
	char dest1[20]
	bool success1
	success1 = StringTrimLeft (src1[0], set1[0], dest1[0])
	// success1=true, dest1="a*#bc"
	
	char set2[20]={'#', ' ', '*'}
	char dest2[4]
	success2 = StringTrimLeft ("# *a*#bc", set2[0], dest2[0])
	// success2=false, dest2 remains the same.
	
	char src3[20]="abc *#"
	char dest3[20]
	bool success3
	success3 = StringTrimLeft (src3[0], "# *", dest3[0])
	// success3=true, dest3="abc *#"
	
	end macro_command

Name	StringTrimRight
Syntax	success = StringTrimRight (source[start], set[start], destination[start]) success = StringTrimRight (“source”, set[start], destination[start]) success = StringTrimRight (source[start], “set”, destination[start]) success = StringTrimRight (“source”, “set”, destination[start])
Description	Trim the trailing specified characters in the set buffer from the source string. The source string and set string parameters accept both static string (in the form: “source”) and char array (in the form: source[start]). This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of trimmed string exceeds the size of destination buffer, it returns false.

Example	macro_command main()
	char src1[20]= "# *a*#bc# * "
	char set1[20]="# *"
	char dest1[20]
	bool success1
	success1 = StringTrimRight(src1[0], set1[0], dest1[0])
	// success1=true, dest1="# *a*#bc"
	
	char set2[20]={'#', ' ', '*'}
	char dest2[20]
	success2 = StringTrimRight("# *a*#bc", set2[0], dest2[0])
	// success2=true, dest2="# *a*#bc"
	
	char src3[20]="ab**c *#"
	char dest3[4]
	bool success3
	success3 = StringTrimRight(src3[0], "# *", dest3[0])
	// success3=false, dest3 remains the same.
	
	end macro_command

Name	StringInsert
Syntax	success = StringInsert (pos, insert[start], destination[start]) success = StringInsert (pos, “insert”, destination[start]) success = StringInsert (pos, insert[start], length, destination[start]) success = StringInsert (pos, “insert”, length, destination[start])
Description	Insert a string in a specific location within the destination string content. The insert location is specified by the pos parameter. The insert string parameter accepts both static string (in the form: “source”) and char array (in the form: source[start]). The number of characters to insert can be specified by the length parameter. This function returns a Boolean indicating whether the process is successfully done or not. If successful, it returns true, otherwise it returns false. If the length of string after insertion exceeds the size of destination buffer, it returns false.

Example	macro_command main()
	
	char str1[20]="but the question is"
	char str2[10]=", that is"
	char dest[40]="to be or not to be"
	bool success
	
	success = StringInsert(18, str1[3], 13, dest[0])
	// success=true, dest="to be or not to be the question"
	
	success = StringInsert(18, str2[0], dest[0])
	// success=true, dest="to be or not to be, that is the question"
	
	success = StringInsert(0, "Hamlet:", dest[0])
	// success=false, dest remains the same.
	
	end macro_command

18.7.7. Recipe Query Function

Name	RecipeGetData
Syntax	RecipeGetData(destination, recipe_address, record_ID)
Description	Get Recipe Data. The gained data will be stored in destination, and must be a variable. recipe_address consists of recipe name and item name: “recipe_name.item_name”. record_ID specifies the ID number of the record in recipe being gained.

Example	macro_command main()
	int data=0
	char str[20]
	int recordID
	bool result
	
	recordID = 0
	result = RecipeGetData(data, "TypeA.item_weight", recordID)
	// From recipe "TypeA" get the data of the item “item_weight” in record 0.
	
	recordID = 1
	result = RecipeGetData(str[0], "TypeB.item_name", recordID)
	// From recipe "TypeB" get the data of the item “item_name” in record 1.
	
	end macro_command

Name	RecipeQuery
Syntax	RecipeQuery (SQL_command, destination)
Description	Use SQL statement to query recipe data. The number of records of query result will be stored in the destination. This must be a variable. SQL command can be static string or char array. Example: RecipeQuery(“SELECT * FROM TypeA”, destination) or RecipeQuery(sql[0], destination) SQL statement must start with ”SELECT * FROM” followed by recipe name and query condition.

Example	macro_command main()
	
	int total_row=0
	char sql[100]="SELECT * FROM TypeB"
	bool result
	
	result = RecipeQuery("SELECT * FROM TypeA", total_row)
	// Query Recipe "TypeA". Store the number of records of query result in total_row.
	
	result = RecipeQuery(sql[0], total_row)
	// Query Recipe "TypeB". Store the number of records of query result in total_row.
	
	end macro_command

Name	RecipeQueryGetData
Syntax	RecipeQueryGetData (destination, recipe_address, result_row_no)
Description	Get the data in the query result obtained by RecipeQuery. This function must be called after calling RecipeQuery, and specify the same recipe name in recipe_address as RecipeQuery. result_row_no specifies the sequence row number in query result

Example	macro_command main()
	
	int data=0
	int total_row=0
	int row_number=0
	bool result_query
	bool result_data
	
	result_query = RecipeQuery("SELECT * FROM TypeA", total_row)
	// Query Recipe "TypeA". Store the number of records of query result in total_row.
	if (result_query) then
	for row_number=0 to total_row-1
	result_data = RecipeQueryGetData(data, "TypeA.item_weight", row_number)
	next row_number
	end if
	
	end macro_command

Name	RecipeQueryGetRecordID
Syntax	RecipeQueryGetRecordID (destination, result_row_no)
Description	Get the record ID numbers of those records gained by RecipeQuery. This function must be called after calling RecipeQuery. result_row_no specifies the sequence row number in query result, and write the obtained record ID to destination.

Example	macro_command main()
	
	int recordID=0
	int total_row=0
	int row_number=0
	bool result_query
	bool result_id
	
	result_query = RecipeQuery("SELECT * FROM TypeA", total_row)
	// Query Recipe "TypeA". Store the number of records of query result in total_row.
	if (result_query) then
	for row_number=0 to total_row-1
	result_id = RecipeQueryGetRecordID(recordID, row_number)
	next row_number
	end if
	
	end macro_command

Name	RecipeSetData
Syntax	RecipeSetData(source, recipe address, record_ID)
Description	Write data to recipe. If success, returns true, else, returns false. recipe_address consists of recipe name and item name: "recipe_name.item_name". record_ID specifies the ID number of the record in recipe being modified.

Example	macro_command main()
	
	int data=99
	char str[20]="abc"
	int recordID
	bool result
	
	recordID = 0
	result = RecipeSetData(data, "TypeA.item_weight", recordID)
	// set data to recipe "TypeA", where item name is "item_weight" and the record ID is 0.
	
	recordID = 1
	result = RecipeSetData(str[0], "TypeB.item_name", recordID)
	// set data to recipe "TypeB", where item name is "item_name" and the record ID is 1.
	
	end macro_command

18.7.8. Miscellaneous

Name	Beep
Syntax	Beep ()
Description	Plays beep sound. This command plays a beep sound with frequency of 800 hertz and duration of 30 milliseconds.
Example	macro_command main() Beep() end macro_command

Name	Buzzer
Syntax	Buzzer ()
Description	Turn ON / OFF the buzzer.

Example	char on = 1, off = 0
	
	Buzzer(on) // turn on the buzzer
	
	DELAY1,000) // delay 1 second
	
	Buzzer(off) // turn off the buzzer
	
	DELAY(500) // delay 500ms
	
	Buzzer(1) // turn on the buzzer
	
	DELAY1,000) // delay 1 second
	
	Buzzer(0) // turn off the buzzer

Name	SYNC_TRIG_MACRO
Syntax	SYNC_TRIG_MACRO(macro_id or name)
Description	Trigger the execution of a macro synchronously (use macro_id or macro name to designate this macro) in a running macro. The current macro will pause until the end of execution of this called macro. macro_id can be a constant or a variable.

Example	macro_command main()
	char ON = 1, OFF = 0
	
	SetData(ON, “Local HMI”, LB, 0, 1)
	
	SYNC_TRIG_MACRO(5)// call a macro (its ID is 5)
	
	SYNC_TRIG_MACRO(“macro_1”) // call a macro (its name is macro_1)
	
	SetData(OFF, “Local HMI”, LB, 0, 1)
	
	end macro_command

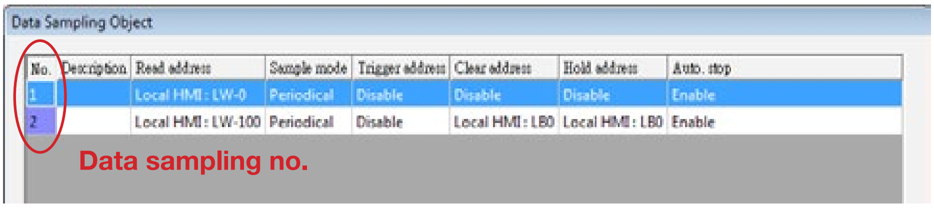
Name	ASYNC_TRIG_MACRO
Syntax	ASYNC_TRIG_MACRO (macro_id or name)
Description	Trigger the execution of a macro asynchronously (use macro_id or macro name to designate this macro) in a running macro. The current macro will continue executing the following instructions after triggering the designated macro; in other words, the two macros will be active simultaneously. macro_id can be a constant or a variable.

Example	macro_command main()
	char ON = 1, OFF = 0
	
	SetData(ON, “Local HMI”, LB, 0, 1)
	
	ASYNC_TRIG_MACRO(5)// call a macro (its ID is 5)
	
	ASYNC_TRIG_MACRO(“macro_1”) // call a macro (its name is macro_1)
	
	SetData(OFF, “Local HMI”, LB, 0, 1)
	
	end macro_command

Name	TRACE
Syntax	TRACE(format, argument)
Description	Use this function to send specified string to the EasyDiagnoser. Users can print out the current value of variables during run-time of macro for debugging. When TRACE encounters the first format specification (if any), it converts the value of the first argument after format and outputs it accordingly. format refers to the format control of output string. A format specification, which consists of optional (in [ ]) and required fields (in bold), has the following form: %[flags] [width] [.precision] type Each field of the format specification is described as below: flags (optional): - + width (optional): A nonnegative decimal integer controlling the minimum number of characters printed. precision (optional): A nonnegative decimal integer which specifies the precision and the number of characters to be printed. type: C or c : specifies a single-byte character. d : signed decimal integer. i : signed decimal integer. o : unsigned octal integer. u : unsigned decimal integer. X or x : unsigned hexadecimal integer. E or e : Signed value having the form. [–]d.dddd e [sign]ddd where d is a single decimal digit, dddd is one or more decimal digits, ddd is exactly three decimal digits, and sign is + or –. f : Signed value having the form [–]dddd.dddd, where dddd is one or more decimal digits. The length of output string is limited to 256 characters. The extra characters will be ignored. The argument part is optional. One format specification converts exactly one argument.

Example	macro_command main()
	char c1 = ’a’
	short s1 = 32767
	float f1 = 1.234567
	
	TRACE(“The results are”) // output: The results are
	TRACE(“c1 = %c, s1 = %d, f1 = %f”, c1, s1, f1)
	// output: c1 = a, s1 = 32767, f1 = 1.234567
	
	end macro_command

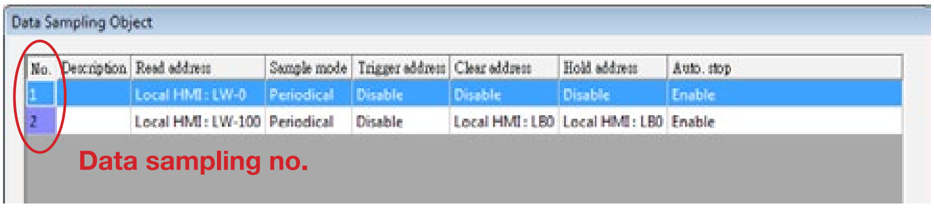
<table>
<tr><th>Name</th><td>FindDataSamplingDate</td></tr>
<tr><th>Syntax</th><td>return_value = FindDataSamplingDate (data_log_number, index, year, month, day)
or
FindDataSamplingDate (data_log_number, index, year, month, day)</td></tr>
<tr><th>Description</th><td>A query function for finding the date of specified data sampling file according to the data sampling no. and the file index. The date is stored into year, month and day respectively in the format of YYYY, MM and DD.



The directory of saved data: [Storage location]\[filename]\yyyymmdd.dtl. The data sampling files under the same directory are sorted according to the file name and are indexed starting from 0. The most recently saved file has the smallest file index number. For example, if there are four data sampling files as follows:
20101210.dtl
20101230.dtl
20110110.dtl
20110111.dtl
The file index are:
20101210.dtl -> index is 3
20101230.dtl -> index is 2
20110110.dtl -> index is 1
20110111.dtl -> index is 0
return_value equals to 1 if referred data sampling file is successfully found, otherwise it equals to 0.
data_log_number and index can be constant or variable. year, month, day and return_value must be variable. return_value is optional.</td></tr>
<tr><th>Example</th><td>macro_command main()
short data_log_number = 1, index = 2, year, month, day
short success

// if there exists a data sampling file named 20101230.dtl, with data sampling // number 1 and file index 2.
// the result after execution: success == 1, year == 2010, month == 12 and //day == 30
success = FindDataSamplingDate(data_log_number, index, year, month, day)

end macro_command</td></tr>
</table>

Name	FindDataSamplingIndex
Syntax	return_value = FindDataSamplingIndex (data_log_number, year, month, day, index) or FindDataSamplingIndex (data_log_number, year, month, day, index)
Description	A query function for finding the file index of specified data sampling file according to the data sampling no. and the date. The file index is stored into index. year, month and day are in the format of YYYY, MM and DD respectively.  The directory of saved data: [Storage location]\[filename]\yyyymmdd.dtl. The data sampling files under the same directory are sorted according to the file name and are indexed starting from 0. The most recently saved file has the smallest file index number. For example, if there are four data sampling files as follows: 20101210.dtl 20101230.dtl 20110110.dtl 20110111.dtl The file index are: 20101210.dtl -> index is 3 20101230.dtl -> index is 2 20110110.dtl -> index is 1 20110111.dtl -> index is 0 return_value equals to 1 if referred data sampling file is successfully found, otherwise it equals to 0. data_log_number, year, month and day can be constant or variable. index and return_value must be variable. return_value is optional.

Example	macro_command main()
	short data_log_number = 1, year = 2010, month = 12, day = 10, index
	short success
	
	// if there exists a data sampling file named 20101210.dtl, with data sampling // number 1 and file index 2.
	// the result after execution: success == 1 and index == 2
	success = FindDataSamplingIndex (data_log_number, year, month, day, index)
	
	end macro_command

Name	FindEventLogDate
Syntax	return_value = FindEventLogDate (index, year, month, day) or FindEventLogDate (index, year, month, day)
Description	A query function for finding the date of specified event log file according to file index. The date is stored into year, month and day respectively in the format of YYYY, MM and DD. The event log files stored in the designated position (such as HMI memory storage or external memory device) are sorted according to the file name and are indexed starting from 0. The most recently saved file has the smallest file index number. For example, if there are four event log files as follows: EL_20101210.evt EL_20101230.evt EL_20110110.evt EL_20110111.evt The file index are: EL_20101210.evt -> index is 3 EL_20101230.evt -> index is 2 EL_20110110.evt -> index is 1 EL_20110111.evt -> index is 0 return_value equals to 1 if referred data sampling file is successfully found, otherwise it equals to 0. index can be constant or variable. year, month, day and return_value must be variable. return_value is optional.

Example	macro_command main()
	short index = 1, year, month, day
	short success
	
	// if there exists an event log file named EL_20101230.evt with index 1
	// the result after execution: success == 1, year == 2010, month == 12, day //== 30
	success = FindEventLogDate (index, year, month, day)
	
	end macro_command

Name	FindEventLogIndex
Syntax	return_value = FindEventLogIndex (year, month, day, index) or FindEventLogIndex (year, month, day, index)
Description	A query function for finding the file index of specified event log file according to date. The file index is stored into index. year, month and day are in the format of YYYY, MM and DD respectively. The event log files stored in the designated position (such as HMI memory storage or external memory device) are sorted according to the file name and are indexed starting from 0. The most recently saved file has the smallest file index number. For example, if there are four event log files as follows: EL_20101210.evt EL_20101230.evt EL_20110110.evt EL_20110111.evt The file index are: EL_20101210.evt -> index is 3 EL_20101230.evt -> index is 2 EL_20110110.evt -> index is 1 EL_20110111.evt -> index is 0 return_value equals to 1 if referred data sampling file is successfully found, otherwise it equals to 0. index can be constant or variable. year, month, day and return_value must be variable. return_value is optional.

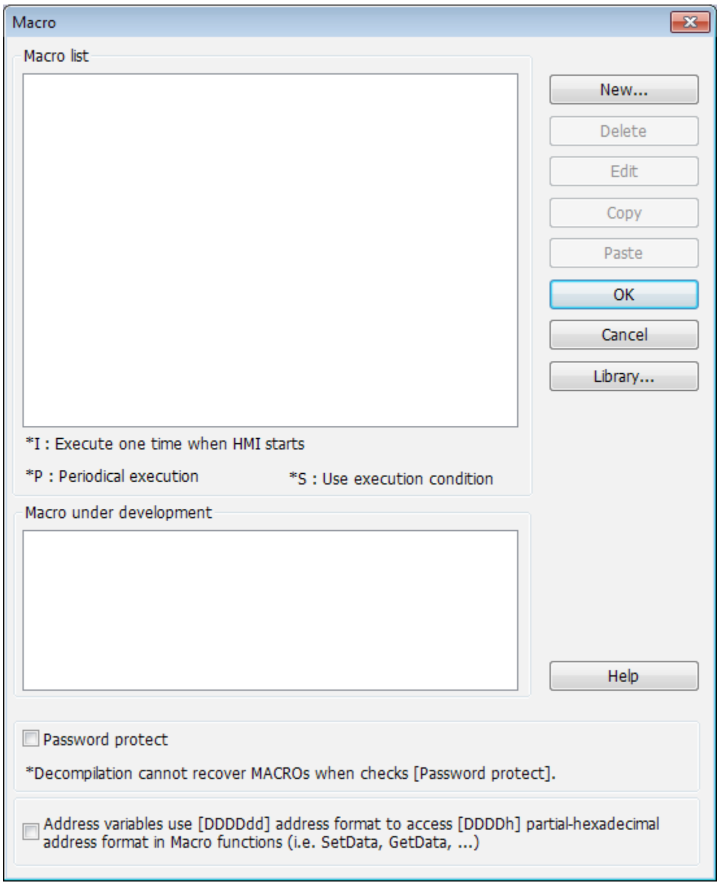
Example	macro_command main()
	short year = 2010, month = 12, day = 10, index
	short success
	
	// if there exists an event log file named EL_20101210.evt, with index 2
	// the result after execution: success == 1, index == 2
	success = FindEventLogIndex (year, month, day, index)
	
	end macro_command

18.8. HOW TO CREATE AND EXECUTE A MACRO

18.8.1. How to Create a Macro

Please follow the steps below to create a macro.

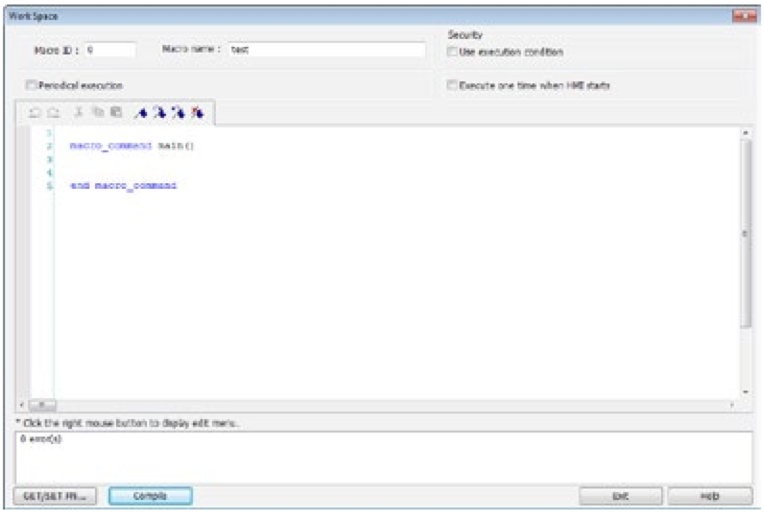
1. Click on (Macro Manager) icon on the tool bar in EasyBuilder Pro to open Macro Manager dialog box as follows.



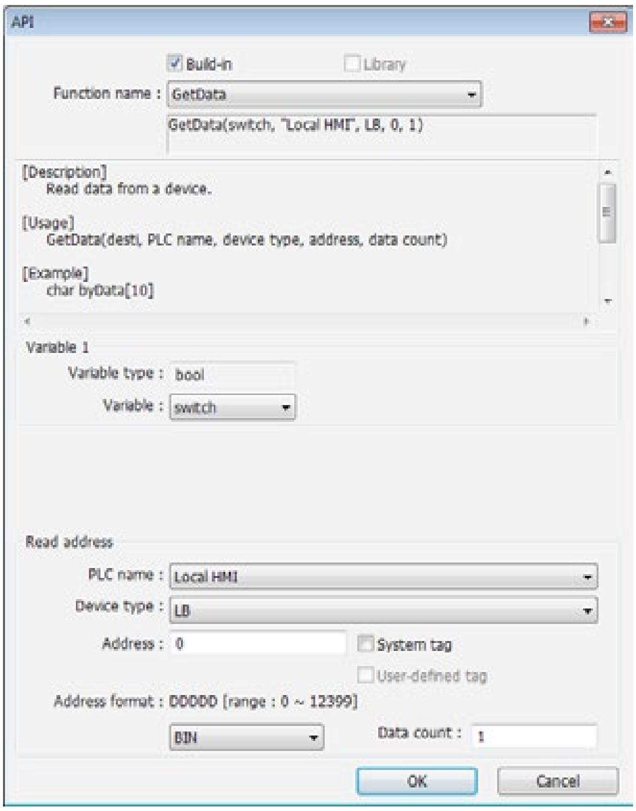
In Macro Manager, all macros compiled successfully are displayed in “Macro list”, and all macros under development or cannot be compiled are displayed in “Macro under development”. The following is a description of the various buttons.

Setting	Description
New	Opens a blank “WorkSpace” editor for creating a new macro.
Delete	Deletes the selected macro.
Edit	Opens the “WorkSpace” editor, and loads the selected macro.
Copy	Copies the selected macro into the clipboard.
Paste	Pastes the macro in the clipboard into the list, and creates a new name for the macro.
OK	Confirm all the edited Macros and click this button to save the new contents before leaving this dialog .
Cancel	Cancel the editing and leave Macro editing dialog.
Library	Open Macro Function Library managing dialog.

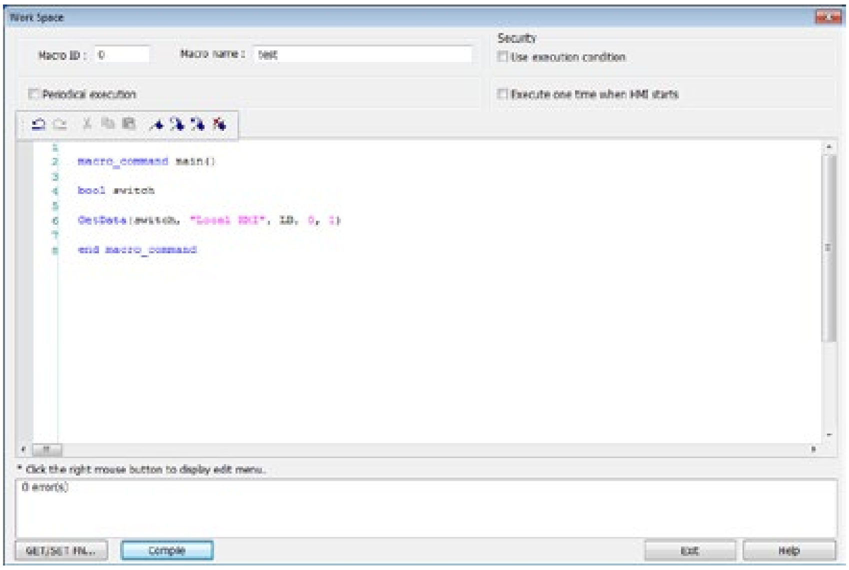
2. Press the (New) button to create an empty macro and open the macro editor. Every macro has a unique number defined at (Macro ID), and must have a macro name, otherwise an error will appear while compiling.



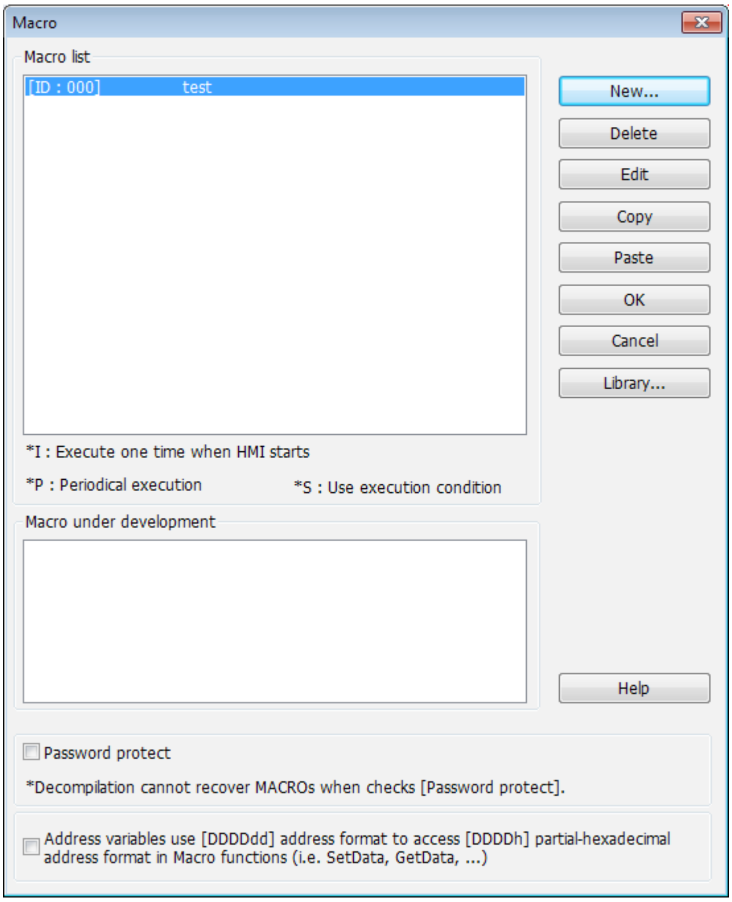
3. Design your macro. To use built-in functions (like SetData() or Getdata()), press (Get/Set FN…) button to open API dialog box and select the function and set essential parameters.



4. After the completion of a new macro, press (Compile) button to compile the macro.



5. If there is no error, press (Exit) button and a new macro “macro_test” will be in “Macro list”.



18.8.2. Execute a Macro
There are several ways to execute a macro.

Use a PLC Control object

1. Open (PLC Control) and add one PLC Control object with the (Type of control) as (Execute macro program).
2. Select the macro in (Macro name). Choose a bit and select a trigger condition to trigger the macro. In order to guarantee that the macro will run only once, consider latching the trigger bit, and then resetting the trigger condition within the macro.
3. Use a (Set Bit) or Toggle Switch object to change the bit to activate the macro.

Use a (Set Bit) or Toggle Switch object

1. On the (General) tab of the (Set Bit) or (Toggle Switch) dialog box, select the (Execute Macro) option.
2. Select the macro to execute. The macro will be executed one time when the button is activated.

	Note If (Set Bit) uses (Periodic Toggle), the macro will be executed every time (Set Bit) toggles.

Use a Function Key object

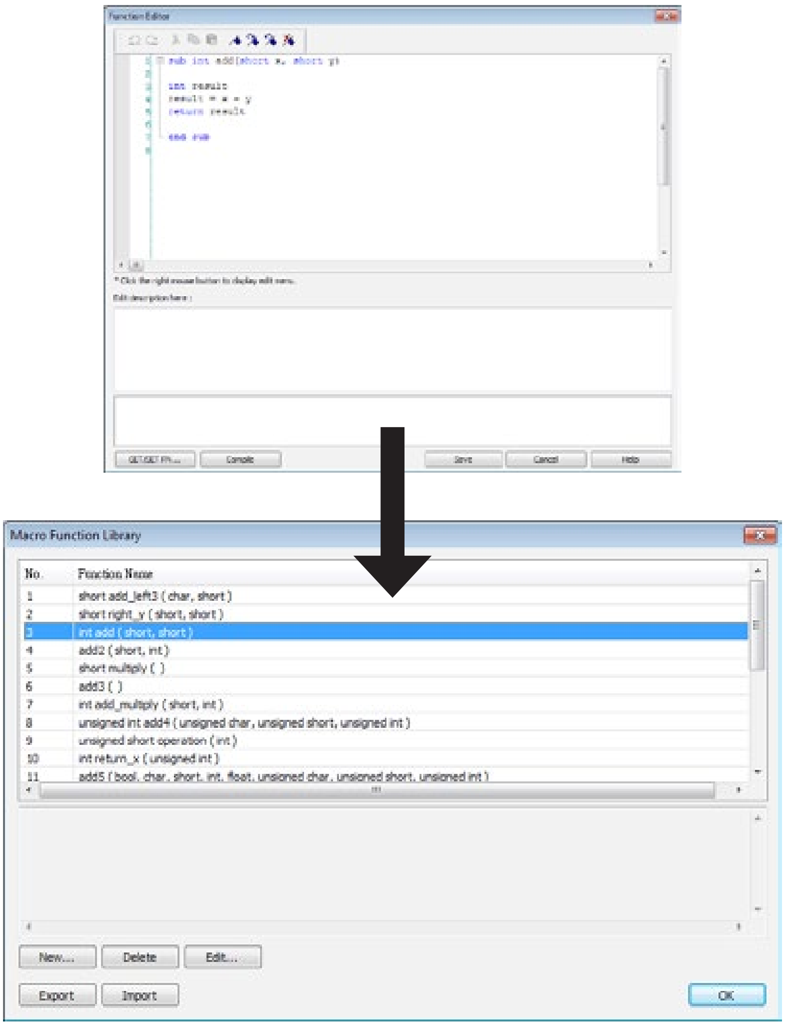
1. On the (General) tab of the (Function Key) dialog, select the (Execute Macro) option.
2. Select the macro to execute. The macro will execute one time when the button is activated.

In macro editor, use

1. (Periodical Execution): macro will be triggered periodically.
2. (Execute one time when HMI starts): macro will be executed once HMI starts.

18.9. USER DEFINED MACRO FUNCTION

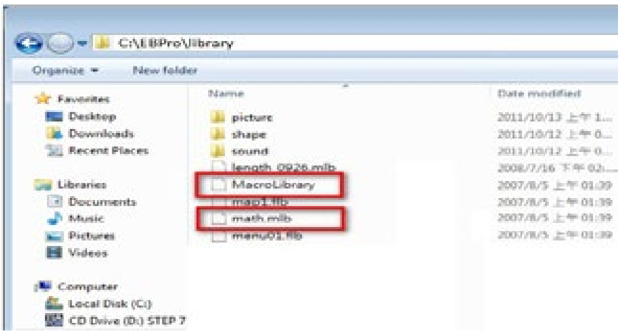
When editing Macro, to save time of defining functions, user may search for the needed from built-in Macro Function Library. However, certain functions, though frequently used, may not be found there. In this case, user may define the needed function and save it for future use. Next time when the same function is required, the saved functions can be called from (Macro Function Library) for easier editing. Additionally, (Macro Function Library) greatly enhances the portability of user-defined functions. Before building a function please check the built-in functions or online function library to see if it exists.



18.9.1. Import Function Library File

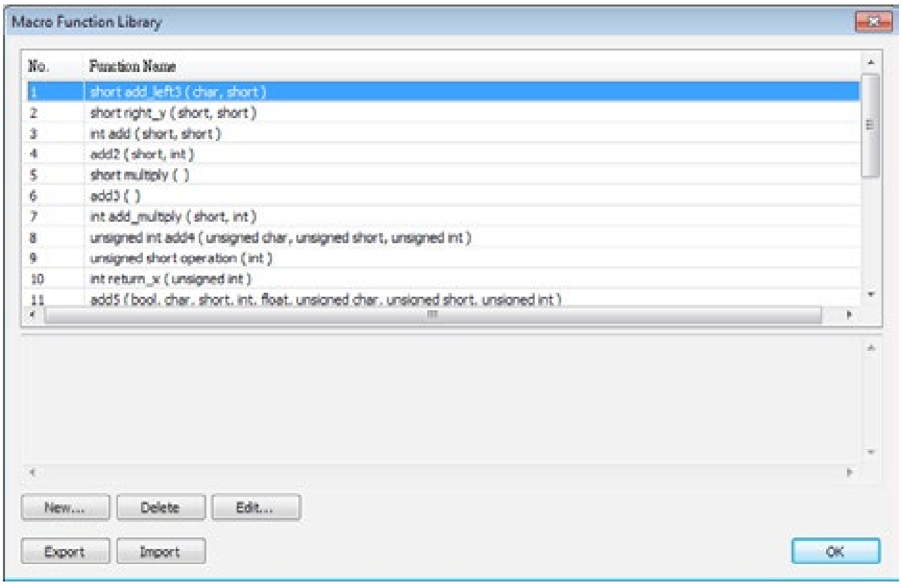
Open a project in HMI programming software, the default Function Library File will be read automatically and the function information will be loaded in. At this moment if a user-defined function is called, the relevant .mlb file must be imported first.

1. Default Function Library File Name: MacroLibrary (without filename extension)
2. Function Library Directory: HMI programming software installation directory\library (folder)
   - \library (folder) contains two types of function library files:
     - Without filename extension: MacroLibrary, the Default Function Library for HMI programming software to read at the beginning.
     - With filename extension (.mlb): such as "math.mlb". The files to be read / written when users import / export. These files are portable and can be called from the folder when needed.
3. When opening HMI programming software, only the functions in Default Function Library will be loaded in, to use functions in .mlb files, please import them first.

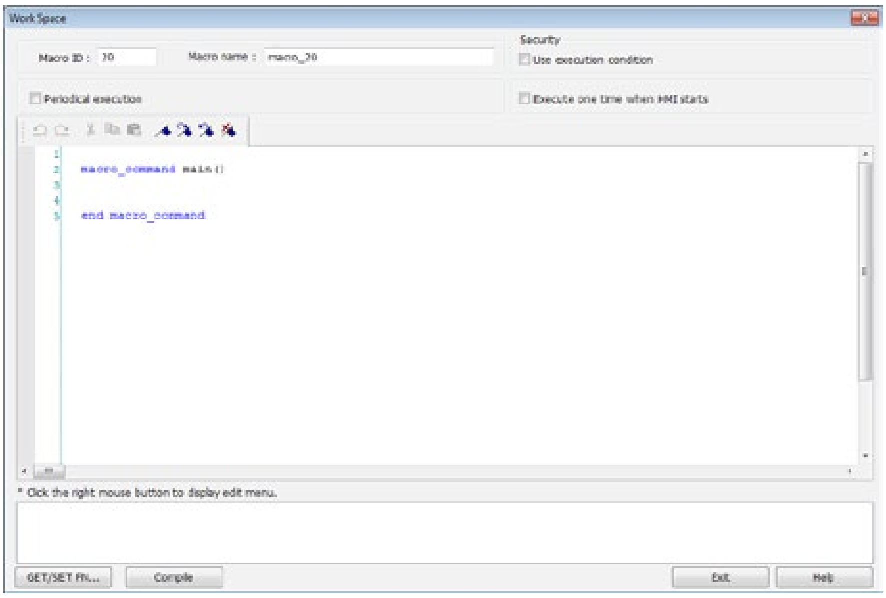


18.9.2. How to Use Macro Function Library

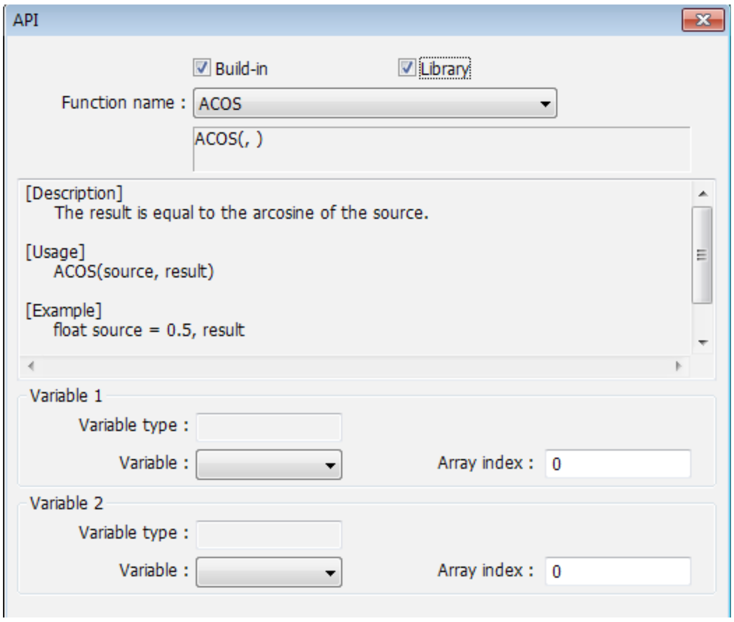
1. Select the function directly from Macro Function Library.



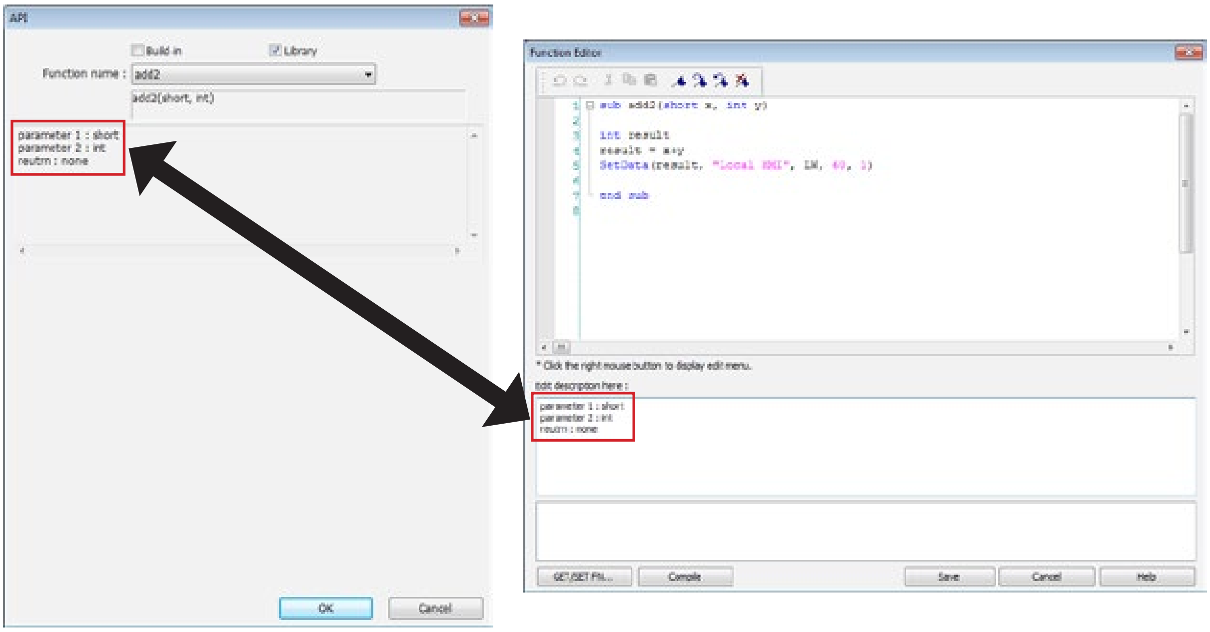
2. In WorkSpace click (GET/SET FN…) to open API dialog box.



3. At least check one from (Library) or (Build-in) and select the function to be used.



4. The description displayed in API dialog box is the same as written in Function Editor.



5. Select the function to be used, fill in the corresponding variables according to the data type.

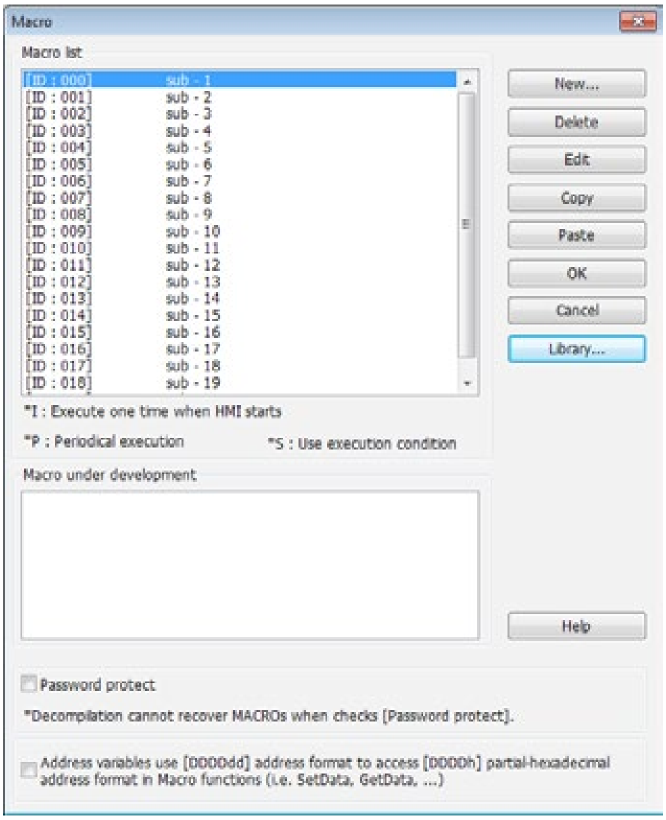
```
macro_command main()

short a
int b,result

add2(short, int)

end macro_command
```

```
macro_command main()

short a
int b,result

result = add2(a, b)

end macro_command
```

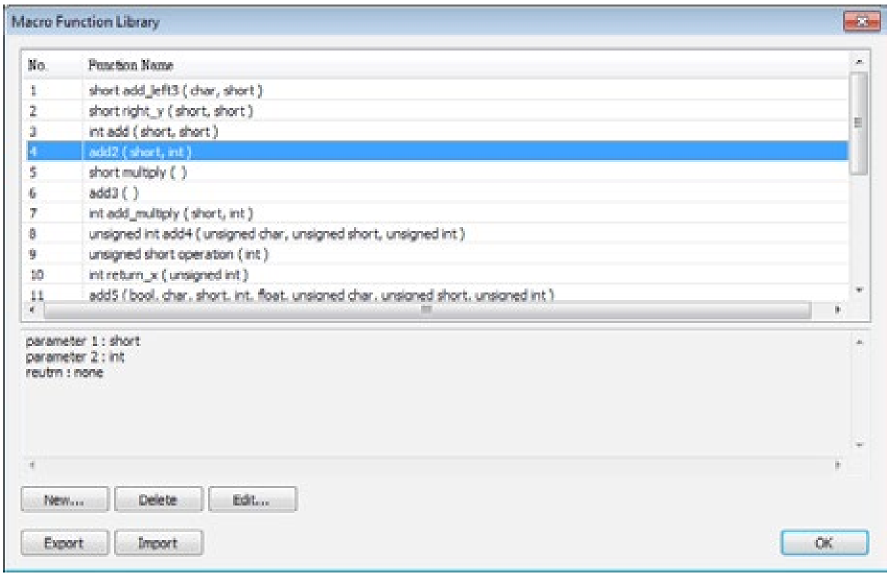
6. Upon completion of the steps above, user-defined functions can be used freely without defining the same functions repeatedly.

18.9.3. Function Library Management Interface

1. Open macro management dialog, click (Library) to open (Macro Function Library) dialog box.



2. A list of functions is shown. When the project is opened, the software will load all the functions in the Default Function Library.



3. Each listed function has the following format:

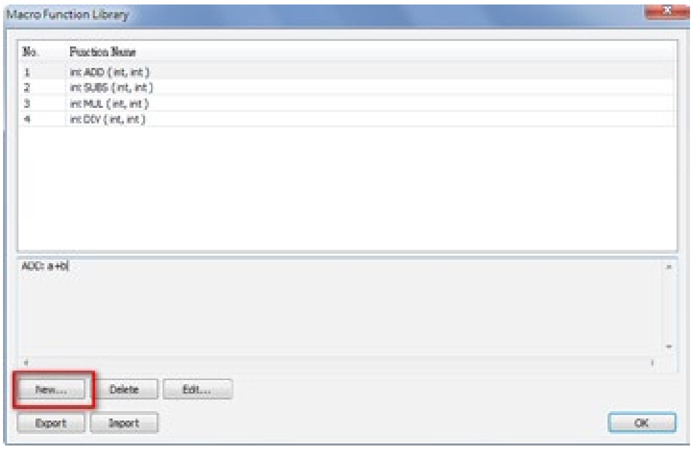
return_type function_name (parameter_type1, …, parameter_typeN)

return_type indicates the type of the return value. If this value does not exist, this column will be omitted. function_name indicates the name of the function. “N” in parameter_typeN stands for the number of parameter types. If this function does not need any parameter, this column will be omitted.

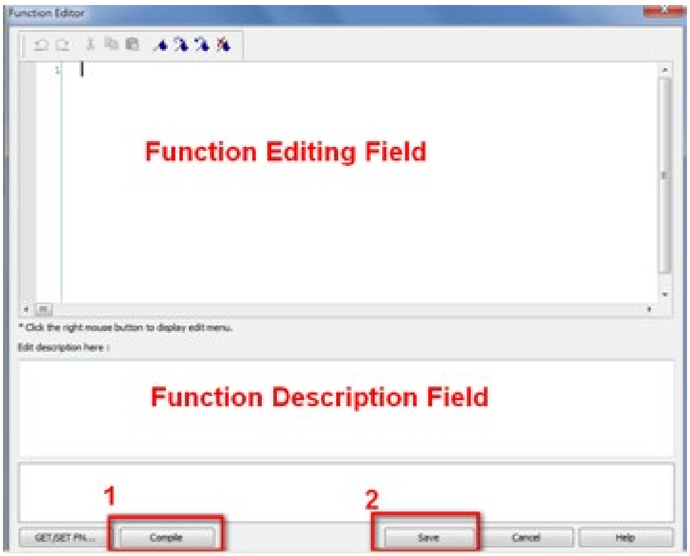
```
sub int ADD(int a, int b)
  int ret
  ret = a+b
  return ret
end sub
```

18.9.3.1. Create a Function

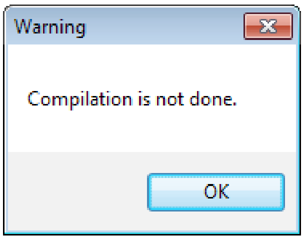
1. Click (New) to enter Function Editor.



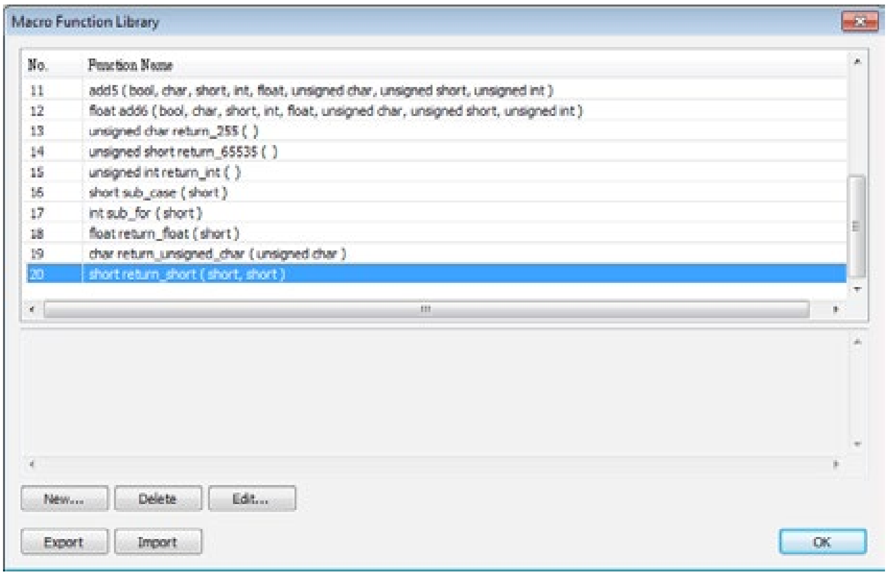
2. Edit function in Function Editor.



3. Edit the function description to describe what the specification is, how to use … etc.
4. After editing, click (Compile) and (Save) to save this function to the Library. Otherwise, a warning is shown.



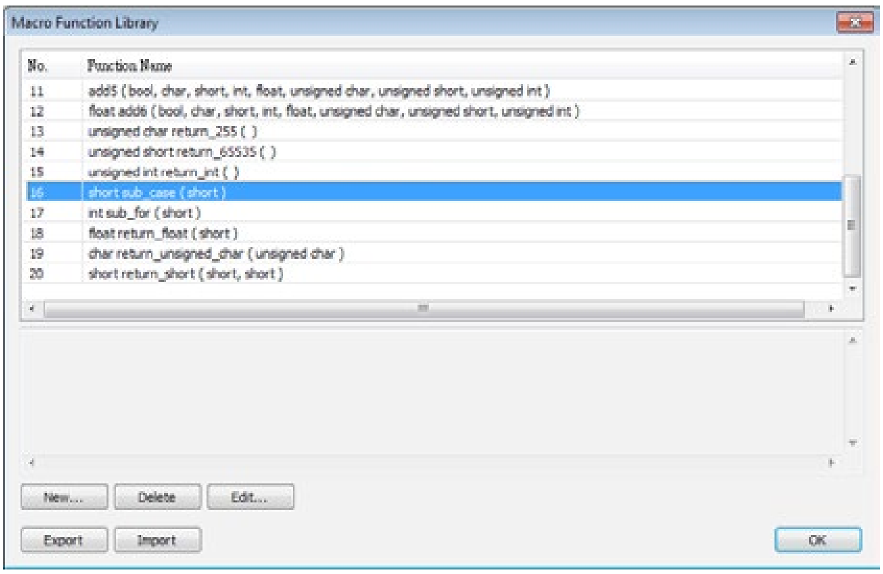
5. Successfully add a function into Macro Function Library.



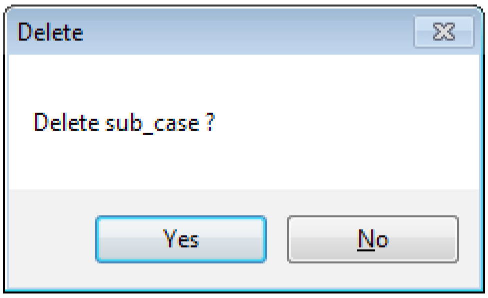
	Note The total size of data type can be declared in a function is 4096 bytes. Function name must only contain alphanumeric characters, and cannot start with a number.

18.9.3.2. Delete a Function

1. In function list select the function to be deleted and click (Delete).

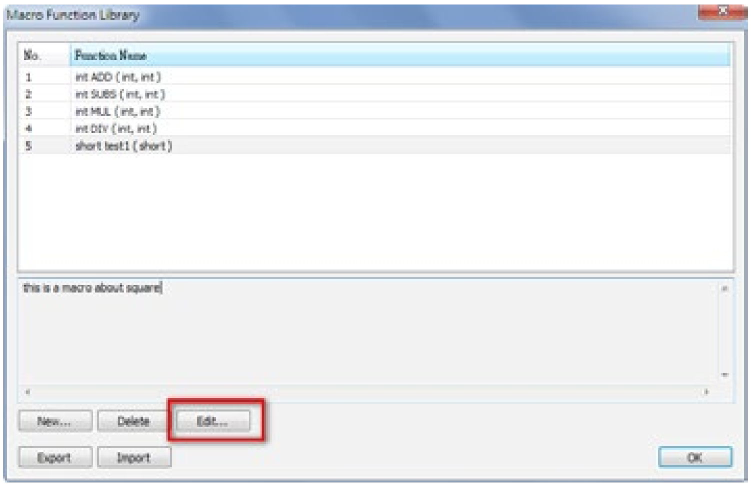


2. Click (Yes) to confirm, (No) to cancel the deletion. Click (Yes) to delete MAX_SHORT function.

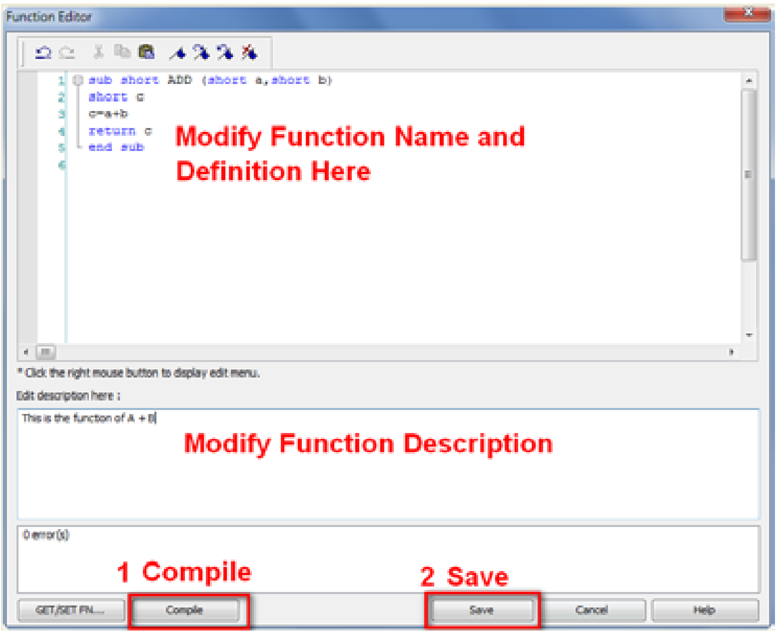


18.9.3.3. Modify a Function

1. Users can modify the functions exist in the Library.
2. Select a function to modify by clicking (Edit) to enter Function Editor.



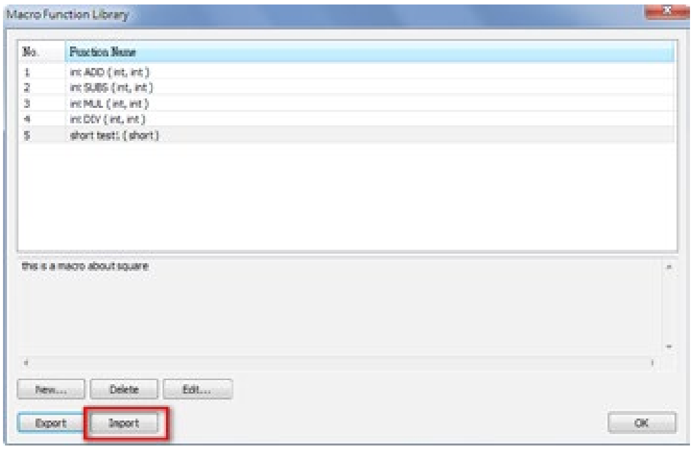
3. Double click the function to be modified can also enter Function Editor.



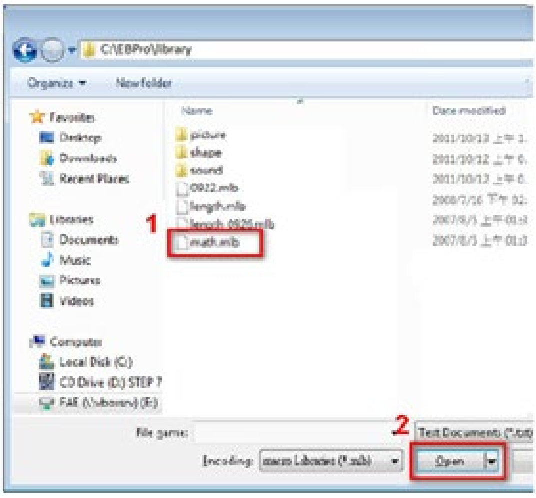
4. After modifying, (Compile) then (Save) before leaving.

18.9.3.4. Import a Function

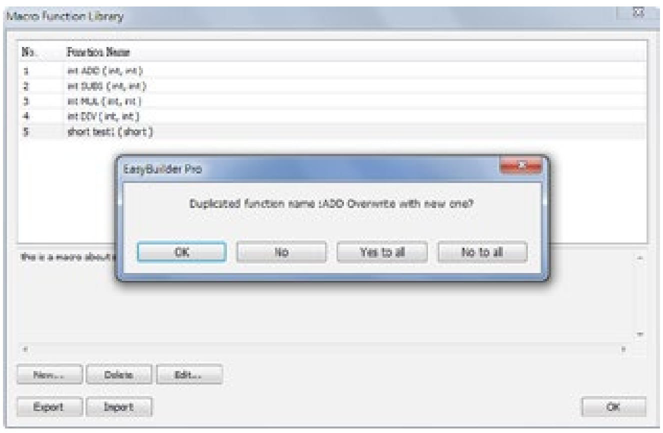
1. Functions can be imported using an external .mlb file.



2. For example, import a function library “math.mlb” which contains a function “test1”. Click (Open).

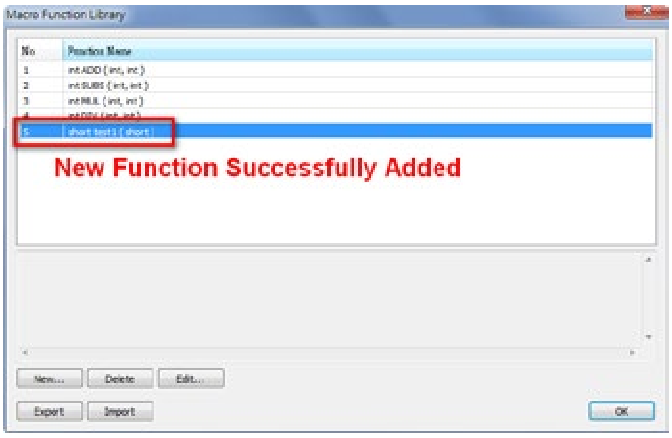


3. When importing a function which already exists in the Library, a confirmation pop-up will be shown. The buttons are:



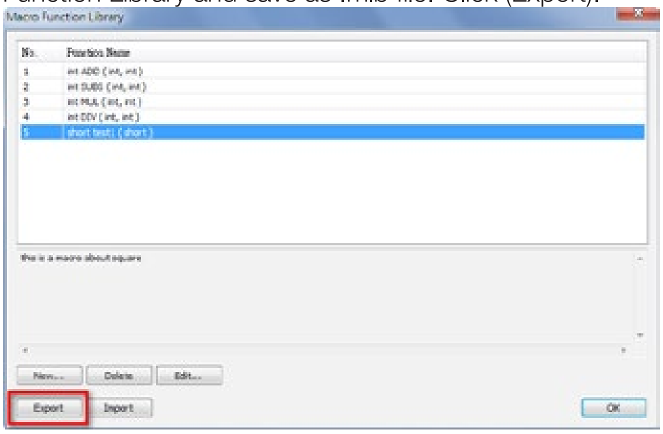
- (OK): overwrite the existing function with the imported one
- (NO): cancel the importing of the function with the same name
- (Yes to all): overwrite using all the imported functions with the same name
- (No to all): cancel the importing of all the functions with the same name

4. The imported functions will be saved in Default Function Library, so if “math.mlb” file is deleted, “test1” will still exist in the Library, even restarting EasyBuilder Pro.

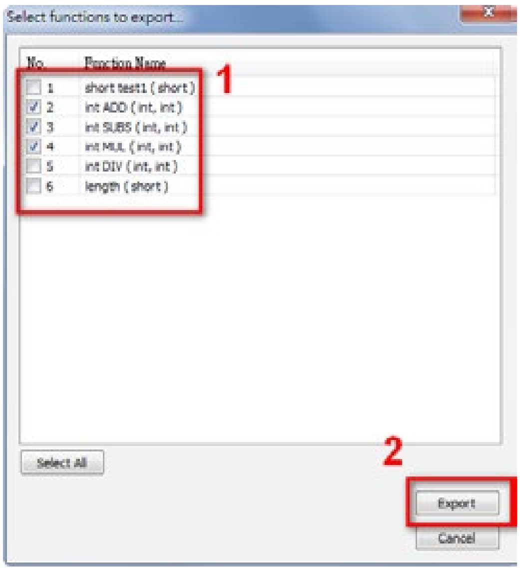


18.9.3.5. Export a Function

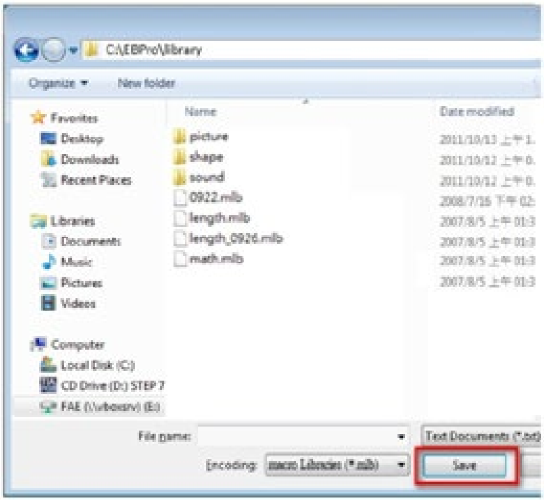
1. Export the function from Function Library and save as .mlb file. Click (Export).



2. Select the function to be exported, and click (Export).



3. A “math.mlb” file can be found under export directory. This file contains 4 functions: ADD, SUBS, MUL, and DIV.
4. The exported .mlb file can be imported on another PC. Open HMI programming software, import, then the functions in this file can be used.



18.10. SOME NOTES ABOUT USING THE MACRO

1. The maximum storage space of local variables in a macro is 4K bytes. So the maximum array size of different variable types are as follows:
   - char a(4096)
   - bool b(4096)
   - short c(2048)
   - int d(1024)
   - float e(1024)

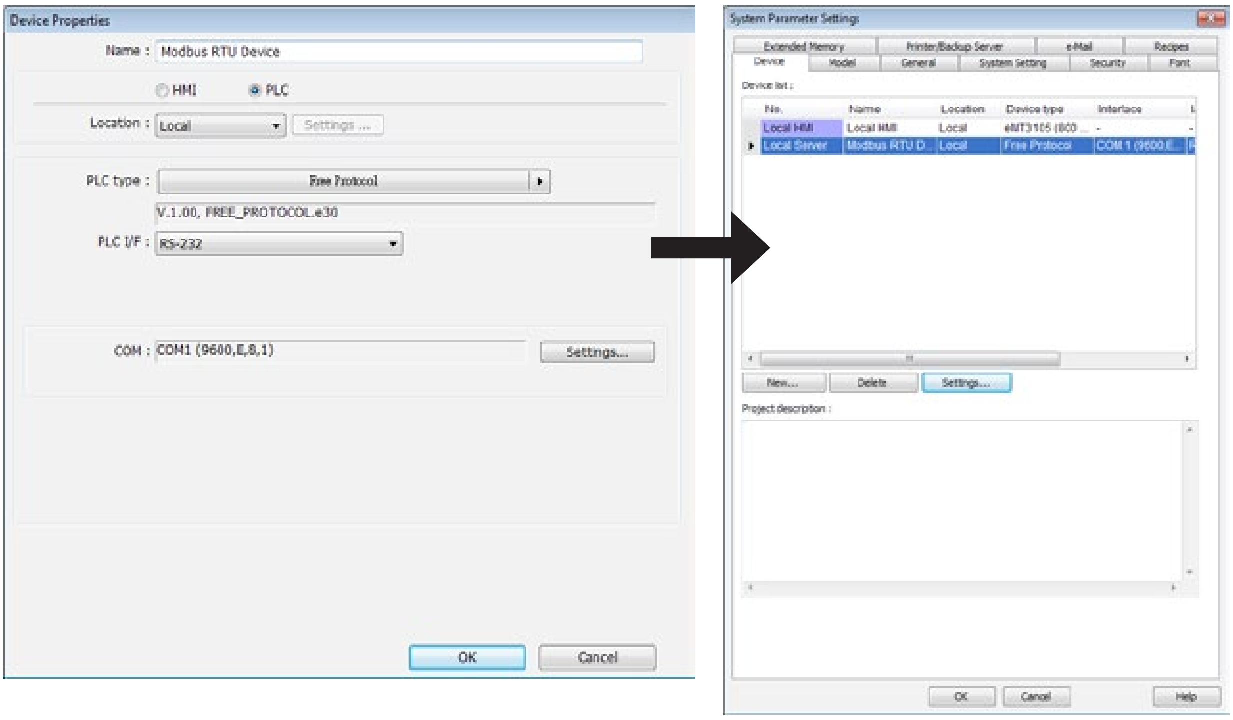
2. A maximum of 255 macros are allowed in an EasyBuilder Pro project.
3. A macro may cause the HMI unresponsive. Possible reasons are:
   - A macro contains an infinite loop with no PLC communication
   - The size of an array exceeds the storage space in a macro

4. The PLC communication speed affects the running time for the macro to execute. Also, too many macros may slow down the communication between HMI and PLC.

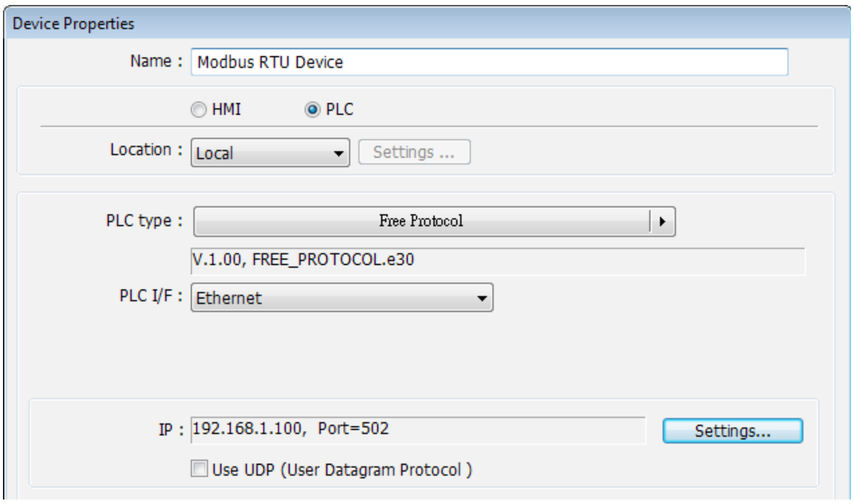
18.11. USE THE FREE PROTOCOL TO CONTROL A DEVICE

If EasyBuilder Pro does not provide a driver for a specific device, users can use OUTPORT and INPORT built-in functions to control the device. The data sent by OUTPORT and INPORT must follow the communication protocol of the device. The following example explains how to use these two functions to control a Modbus RTU device.

1. First, create a new device in the device table. The device type of the new device is set to "Free Protocol" and named with "Modbus RTU device" as follows:



2. The interface of the device (PLC I/F) uses (RS232). If a Modbus TCP/IP device is connected, the interface should be (Ethernet) with correct IP and port number as follows:



Suppose that the HMI will read the data of 4x_1 and 4x_2 on the device. First, utilize OUTPORT to send out a read request to the device. The format of OUTPORT is:
OUTPORT(command(start), device_name, cmd_count)
Since "Modbus RTU device" is a Modbus RTU device, the read request must follow Modbus RTU protocol. The request uses"Reading Holding Registers (0x03)" command to read data. The following picture displays the content of the command. (The items of the station number (byte 0) and the last two bytes (CRC) are ignored).

Request

Function code	1 Byte	0x03
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	1 to 125 (0x7D)

Response

Function code	1 Byte	0x03
Byte count	1 Byte	2 x N*
Register value	N* x 2 Bytes	

*N = Quantity of Registers

Error

Error code	1 Byte	0x83
Exception code	1 Byte	01 or 02 or 03 or 04

Depending on the protocol, the content of a read command as follows (The total is 8 bytes):

command(0): station number	(BYTE 0)
command(1): function code	(BYTE 1)
command(2): high byte of starting address	(BYTE 2)
command(3): low byte of starting address	(BYTE 3)
command(4): high byte of quantity of registers	(BYTE 4)
command(5): low byte of quantity of registers	(BYTE 5)
command(6): low byte of 16-bit CRC	(BYTE 6)
command(7): high byte of 16-bit CRC	(BYTE 7)

So a read request is designed as follows:

```
char command[32]
short address, checksum

FILL(command[0], 0, 32)  //  initialize command[0]~command[31] to 0

command[0] = 0x1  //  station number
command[1] = 0x3  //  read holding registers (function code  is 0x3)

address =  //  starting address (4x_1) is 0
HIBYTE(address, command[2])
LOBYTE(address, command[3])

read_no = 2  //  the total words of rading is 2 words
HIBYTE(read_no, command[4])
LOBYTE(read_no, command[5])

CRC(command[0], checksum, 6)  //  calculate 16-bit CRC

LOBYTE(checksum, command[6])
HIBYTE(checksum, command[7])
```

Lastly, use OUPORT to send out this read request to PLC.

```
OUTPORT(command[0], "MODBUS RTU Device", 8)  //  send read request
```

After sending out the request, use INPORT to get the response from PLC. Depending on the protocol, the content of the response is as follows (the total byte is 9):

command(0): station number	(BYTE 0)
command(1): function code	(BYTE 1)
command(2): byte count	(BYTE 2)
command(3): high byte of 4x_1	(BYTE 3)
command(4): low byte of 4x_1	(BYTE 4)
command(5): high byte of 4x_2	(BYTE 5)
command(6): high byte of 4x_2	(BYTE 6)
command(7): low byte of 16-bit CRC	(BYTE 7)
command(8): high byte of 16-bit CRC	(BYTE 8)

The format of INPORT is:

```
INPORT(response[0], “MODBUS RTU Device”, 9, return_value)  //  read reponse
```

Where the real read count is restored to the variable return_value (unit is byte). If return_value is 0, it means reading fails in executing INPORT.
According to the MODBUS RTU protocol specification, the correct response[1] must be equal to 0x03. After getting correct response, calculate the data of 4x_1 and 4x_2 and put in the data into LW-100 and LW-101 of HMI.

```
If (return_value) >0 and response[1] == 0x3) then
   read_data[0] = response[4] + (response[3] << 8) // 4x_1
   read_data[1] = response[6] + (response[5] << 8) // 4x_2

   SetData(read_data[0], “Local HMI”, LW, 100, 2)
endif
```

The complete macro is as follows:

```
// Read Holding Registers
macro_command main()

    char command[32], response[32]
    short address, checksum
    short read_no, return_value, read_data[2], i

    FILL(command[0], 0, 32)//  initialize command[0]~command[31] to 0
    FILL(response[0], 0, 32)

    command[0] = 0x1//  station number
    command[1] = 0x3//  read holding registers (function code is 0x3)

    address = 0
    address =  0//  starting address (4x_1) is 0
    HIBYTE(address, command[2])
    LOBYTE(address, command[3])

    read_no = 2/  the total words of reading is 2 words
    HIBYTE(read_no, command[4])
    LOBYTE(read_no, command[5])

    CRC(command[0], checksum, 6)//  calculate 16-bit CRC

    LOBYTE(checksum, command[6])
    HIBYTE(checksum, command[7])

    OUTPORT(command[0], "MODBUS RTU Device", 8 )//  send request
    INPORT(response[0], "MODBUS RTU Device", 9, return_value)//  read response

    if (return_value > 0 and response[1] == 0x3) then
       read_data[0] = response[4] + (response[3] << 8)//  4x_1
       read_data[1] = response[6] + (response[5] << 8)//  4x_2

       SetData(read_data[0], "Local HMI", LW, 100, 2)
    end if

    end macro_command
```

The following example explains how to design a request to set the status of 0x_1. The request uses ”Write Single Coil(0x5)” command.

Request

Function code	1 Byte	0x05
Output Address	2 Bytes	0x0000 to 0xFFFF
Output Value	2 Bytes	0x0000 or 0xFF00

Response

Function code	1 Byte	0x05
Output Address	2 Bytes	0x0000 to 0xFFFF
Output Value	2 Bytes	0x0000 or 0xFF00

Error

Error code	1 Byte	0x85
Exception code	1 Byte	01 or 02 or 03 or 04

The complete macro is as follows:

```
//  Write Single Coil (ON)
macro_command main()

char command[32], response[32]
short address, checksum
short i, return_value

FILL(command[0], 0, 32)//  initialize command[0]~ command[31] to 0
FILL(response[0], 0, 32)

command[0] = 0x1//  station number
command[1] = 0x5//  function code : write single coil

address = 0
HIBYTE(address, command[2])
LOBYTE(address, command[3])

command[4] = 0xff//  force 0x_1 on
command[5] = 0

CRC(command[0], checksum, 6)

LOBYTE(checksum, command[6])
HIBYTE(checksum, command[7])

OUTPORT(command[0], "MODBUS RTU Device", 8)//  send request
INPORT(response[0], "MODBUS RTU Device", 8, return_value)//  read response

end macro_command
```

18.12. COMPILER ERROR MESSAGE

Error Message Format

error C#: error description
(# is the error message number)
Example: error C37: undeclared identifier: i
When there are compile errors, the description of the error can be found by the compiler error message number.

Error Description

(C1) syntax error:’identifier’
There are many possibilities to cause compiler error.

For example:

```
macro_command main()
char i, 123xyz  //  this is an unsupported variable name
end macro_command
```

(C2) ‘identifier’ used without having been initialized
Macro must define the size of an array during declaration.

For example:

```
macro_command main()
char i
int g[i]  //  i must be a numeric constant
end macro_command
```

(C3) redefinition error: ‘identifier’
The name of variable and function within its scope must be unique.

For example:
macro_command main()
int g[10]/g // error
end macro_command

(C4) function name error: ‘identifier’
Reserved keywords and constant cannot be the name of a function

For example:
sub int if() // error

(C5) parentheses have not come in pairs
Statement missing “(“ or “)”
For example:
macro_command main) // missing “(“

(C6) illegal expression without matching ‘if’
Missing expression in “if” statement

(C7) illegal expression (no ‘then’) without matching ‘if’
Missing “then” in “if” statement

(C8) illegal expression (no ‘end if’)
Missing “end if”

(C9) illegal ‘end if’ without matching ‘if’
Unfinished “If’ statement before “End If”

(C10) illegal ‘else’
The format of “if” statement is:
if [logic expression] then
[else [if [logic expression] then]]

end if

Any format other than this format will cause a compile error.

(C17) illegal expression (no ‘for’) without matching ‘next’
“for” statement error: missing “for” before “next”

(C18) illegal variable type (not integer or char)
Should be integer or char variable

(C19) variable type error
Missing assign statement

(C20) must be keyword ‘to’ or ‘down’
Missing keyword “to” or “down”

(C21) illegal expression (no ‘next’)
The format of “for” statement is:
for [variable] = [initial value] to [end value] [step]

next [variable]

Any format other than this format will cause a compile error.

(C22) ‘wend’ statement contains no ‘while’
“While” statement error: missing “while” before “Wend”

(C23) illegal expression without matching ‘wend’
The format of “While” statement is:

while [logic expression]

wend

Any format other than this format will cause a compile error.

(C24) syntax error: ‘break’
“break” statement can only be used in “for”, “while” statement.

(C25) syntax error: ‘continue’
“continue” statement can only be used in “for” statement, or “while” statement.

(C26) syntax error
Error in expression.

(C27) syntax error
The mismatch of an operation object in expression can cause a compile error.

For example:

```
macro_command main( )
int a, b
for a = 0 to 2
b = 4 + xyz //  illegal: xyz is undefined
next a
end macro_command
```

(C28) must be ‘macro_command’
There must be ’macro_command’

(C29) must be key word ‘sub’
The format of function declaration is:

```
sub [data type] function_name(…)
            ………..
end sub
```

For example::

```
sub int pow(int exp)
…….
end sub
```

format other than this format will cause a compile error.

(C30) number of parameters is incorrect
Mismatch of the number of parameters

(C31) parameter type is incorrect
Mismatch of data type of parameter. When a function is called, the data type and the number of parameters should match the declaration of function, otherwise it will cause a compile error.

(C32) variable is incorrect
The parameters of a function must be equivalent to the arguments passing to a function to avoid compile error.

(C33) function name: undeclared function

(C34) expected constant expression
Illegal array index format.

(C35) invalid array declaration

(C36) array index error

(C37) undeclared identifier: i ‘identifier’
Any variable or function should be declared before use.

(C38) un-supported PLC data address
The parameter of GetData(…) , SetData(…) should be legal PLC address. If the address is illegal, this error message will be shown.

(C39) ‘idenifier’ must be integer, char or constant
The format of array is:
Declaration: array_name[constant] (constant is the size of the array)
Usage: array_name[integer, character or constant]
Any format other than this format will cause a compile error.

(C40) execution syntax should not exist before variable declaration or constant definition

For example:

```
macro_command main( )
int a, b
for a = 0 To 2
  b = 4 + a
int h , k  //  illegal – definitions must occur before any statements or expressions
                    //  for example, b = 4 + a
next a
end macro_command
```

(C41) float variables cannot be contained in shift calculation

(C42) function must return a value

(C43) function should not return a value

(C44) float variables cannot be contained in calculation

(C45) PLC address error

(C46) array size overflow (max. 4k)

(C47) macro command entry function is not only one

(C48) macro command entry function must be only one
The only one main entrance of macro is:

```
macro_command function_name( )
end macro_command
```

(C49) an extended addressee’s station number must be between 0 and 255

For example:

```
SetData(bits[0] , “PLC 1”, LB , 300#123, 100)
//  illegal : 300#123 means the station number is 300, but the maximum is 255
```

(C50) an invalid PLC name
PLC name is not defined in the device list of system parameters.

(C51) macro command do not control a remote device
A macro can only control a local machine.

For example:
SetData(bits[0] , “PLC 1”, LB , 300#123, 100)
“PLC 1“ is connected with the remote HMI ,so it cannot work.

18.13. SAMPLE MACRO CODE

“for” statement and other expressions (arithmetic, bitwise shift, logic and comparison)

```
macro_command main()
int a[10], b[10], i

b[0] = (400 + 400 << 2) / 401
b[1] = 22 *2 - 30 % 7
b[2] = 111 >> 2
b[3] = 403 > 9 + 3 >= 9 + 3 < 4 + 3 <= 8 + 8 == 8
b[4] = not 8 + 1 and 2 + 1 or 0 + 1 xor 2
b[5] = 405 and 3 and not 0
b[6] = 8 & 4 + 4 & 4 + 8 | 4 + 8 ^ 4
b[7] = 6 – (~4)
b[8] = 0x11
b[9] = 409

for i = 0 to 4 step 1
    if (a[0] == 400) then
         GetData(a[0],”Device 1”, 4x, 0,9)
         GetData(b[0],”Device 1”, 4x, 11,10)
    end If
    next i
    end macro_command
```

“while”, “if” and “break” statements

```
macro_command main()
int b[10], i
i = 5
while i == 5 - 20 % 3
  GetData(b[1], ”Device 1”, 4x, 11, 1)

  if b[1] == 100 then
    break
    end if
wend
end macro_command
```

Global variables and function call

```
char g
sub int fun(int j, int k)
  int y

  SetData(j, “Local HMI”, LB, 14, 1)
  GetData(y, “Local HMI”, LB, 15, 1)
  g = y

  return y
end Sub

macro_command main()
  int a, b, i

  a = 2
  b = 3
```

```
    i = fun(a, b)
    SetData(i, "Local HMI", LB, 16, 1)
end macro_command
```

"if" statement

```
macro_command main()
  int k[10], j

  for j = 0 to 10
                  k[j] = j
  next j

  if k[0] == 0 then
  SetData(k[1], "Device 1", 4x, 0, 1)
  end if

if k[0] == 0 then
    SetData(k[1], "Device 1", 4x, 0, 1)
  else
  SetData(k[2], "Device 1", 4x, 0, 1)
end if

    if k[0] == 0 then
       SetData(k[1], "Device 1", 4x, 1, 1)
    else if k[2] == 1 then
    SetData(k[3], "Device 1", 4x, 2, 1)
end If

     if k([0] == 0 then
    SetData(k[1], "Device 1", 4x, 3, 1)
else if k[2] == 2 then
    SetData(k[3], "Device 1", 4x, 4, 1)
else
    SetData(k[4], "Device 1", 4x, 5, 1)
end If
end macro_command
```

"while" and "wend" statements

```
macro_command main()
char i = 0
int a[13], b[14], c = 4848

b[0] = 13

while b[0]
     a[i] = 20 + i * 10

    if a[i] == 120 then
      c =200
       break
    end if

    i = i + 1
wend

SetData(c, "Device 1", 4x, 2, 1)
end macro_command
```

"break" and "continue" statements

```
macro_command main()
char i = 0
int a[13], b[14], c = 4848
```

```
b[0] = 13

while b[0]
    a[i] = 20 + i * 10

    if a[i] == 120 then
    c =200
    i = i + 1
      continue
    end if

i = i + 1

if c == 200 then
    SetData(c, "Device 1", 4x, 2, 1)
break
    end if
wend
end macro_command
```

Array

```
macro_command main()
int a[25], b[25], i

b[0] = 13

for i = 0 to b[0] step 1
a[i] = 20 + i * 10
next i

SetData(a[0], "Device 1", 4x, 0, 13)
end macro_command
```

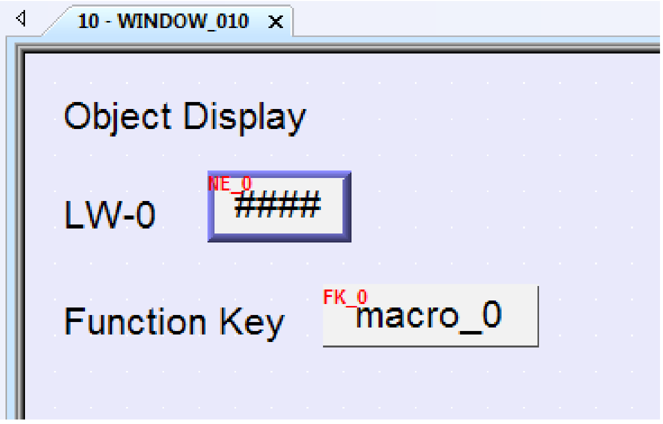
18.14. MACRO TRACE FUNCTION

TRACE function can be used with EasyDiagnoser to show the current content of the variables. The following example illustrates how TRACE function could be used in macro.

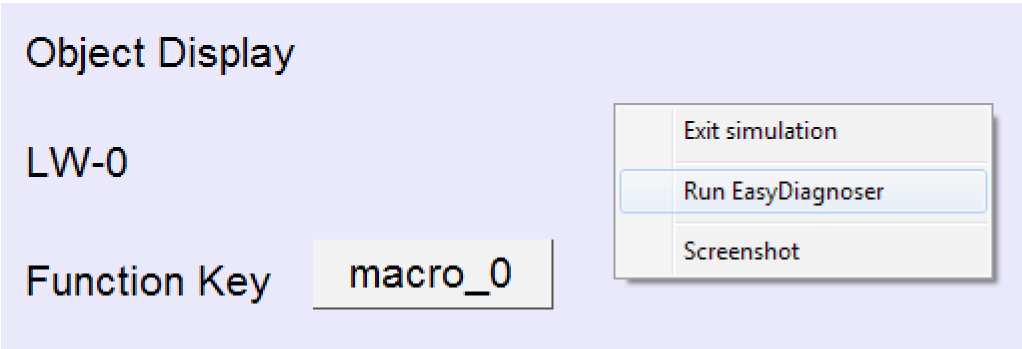
1. First of all, add a new macro "macro_0" in the project, and in "macro_0" add TRACE ("LW = %d", a). "%d" indicates display current value of LW in decimal format. The content of "macro_0" is as follows:

```

macro_command main()

short a
GetData(a, "Local HMI", LW, 0, 1)
a=a+1
SetData(a, "Local HMI", LW, 0, 1)
TRACE ("LW0 = %d" , a)

end macro_command
```

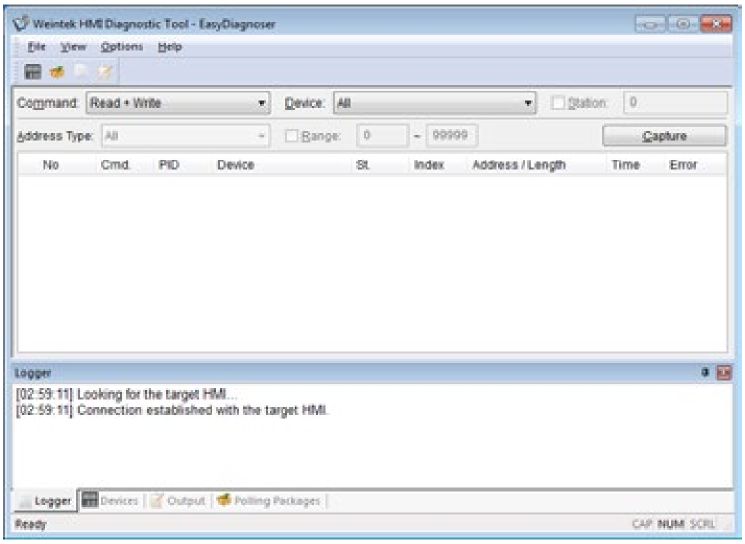
2. Secondly, add a Numeric Display object and a Function Key object in window no. 10 of the project. The Function Key object is used to execute macro_0.



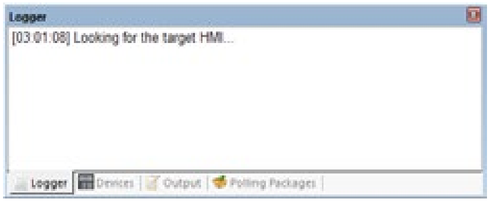
3. Lastly, compile the project and execute (Off-line simulation) or (On-line simulation).
4. When processing simulation on PC, right click and select “Run EasyDiagnoser” in the pop-up menu.



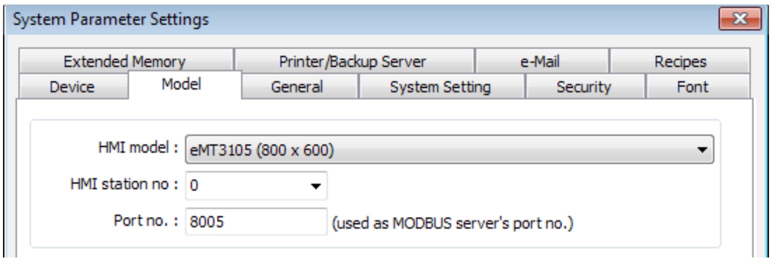
5. Afterwards, EasyDiagnoser will be started. (Logger) window displays whether EasyDiagnoser is able to connect with the HMI to be watched or not. (Output) window displays the output of the TRACE function. The illustration below shows that EasyDiagnoser succeeds in connecting with HMI.



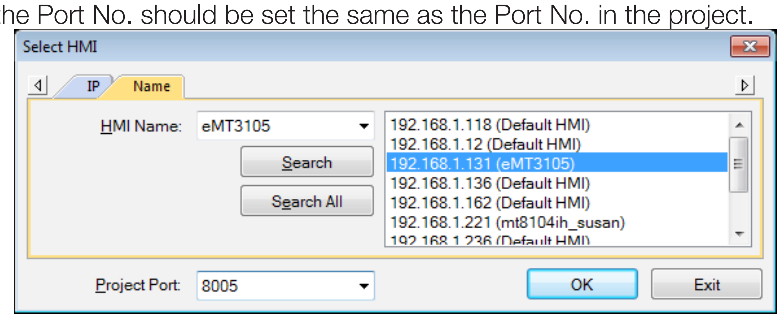
- When EasyDiagnoser is not able to connect with HMI, (Logger) window displays content as shown in the following figure:



6. The possible reason of not being able to get connection with HMI can be failure in executing simulation on PC. Another reason is that the Port No. used in project for simulation on PC is incorrect (or occupied by system). Please change Port No. as shown, compile project then do simulation again.



7. In EasyDiagnoser, the Port No. should be set the same as the Port No. in the project.



- The three consecutive ports of the project port no. are preserved for HMI communication. In the setting above as an example, Port No. is set as 8,005. Port 8,005, 8,006 and 8,007 should be reserved. In this case when executing simulation on PC, please make sure that these ports are not occupied by other programs.

- TRACE Syntax List

<table>
<tr><th>Name</th><td>TRACE</td></tr>
<tr><th>Syntax</th><td>TRACE(format, argument)</td></tr>
<tr><th>Description</th><td>Use this function to send specified string to the EasyDiagnoser. Users can print out the current value of variables during run-time of macro for debugging.
When TRACE encounters the first format specification (if any), it converts the value of the first argument after format and outputs it accordingly.
format refers to the format control of output string. A format specification, which consists of optional (in [ ]) and required fields (in bold), has the following form:
%[flags] [width] [.precision] type
Each field of the format specification is described as below:
flags (optional):
▪ -
▪ +
width (optional):
▪ A nonnegative decimal integer controlling the minimum number of characters printed

precision (optional):
▪ A nonnegative decimal integer which specifies the precision and the number of characters to be printed

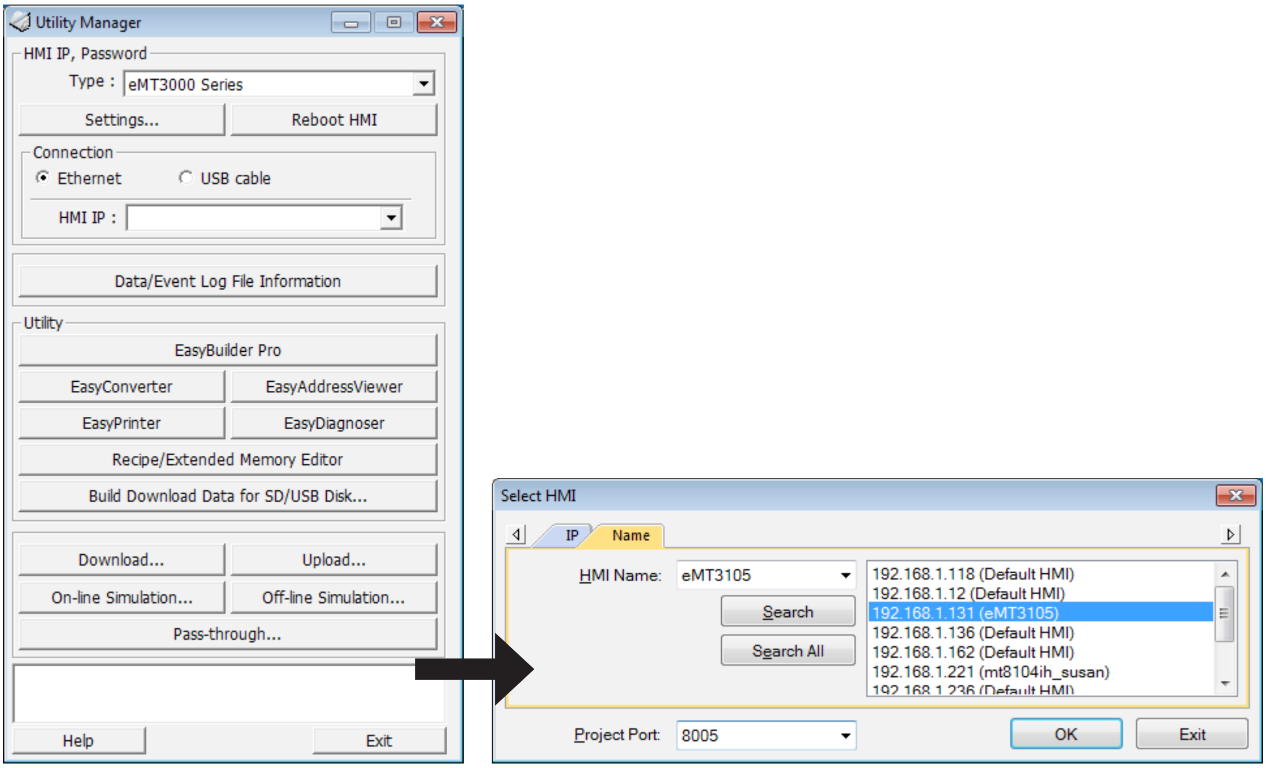
Type:
<table>
<tr><td>C or c</td><td>specifies a single-byte character</td></tr>
<tr><td>d</td><td>signed decimal integer</td></tr>
<tr><td>i</td><td>signed decimal integer</td></tr>
<tr><td>o</td><td>unsigned octal integer</td></tr>
<tr><td>u</td><td>unsigned decimal integer</td></tr>
<tr><td>X or x</td><td>unsigned hexadecimal integer</td></tr>
<tr><td>E or e</td><td>Signed value having the form. [–]d.dddd e [sign]ddd where d is a single decimal digit, dddd is one or more decimal digits, ddd i exactly three decimal digits, and sign is + or –</td></tr>
<tr><td>f</td><td>Signed value having the form [–]dddd.dddd, where dddd is one or more decimal digits</td></tr>
</table>

The length of output string is limited to 256 characters.
The argument part is optional.</td></tr>
<tr><th>Example</th><td>macro_command main()
char c1 = ’a’
short s1 = 32767
float f1 = 1.234567

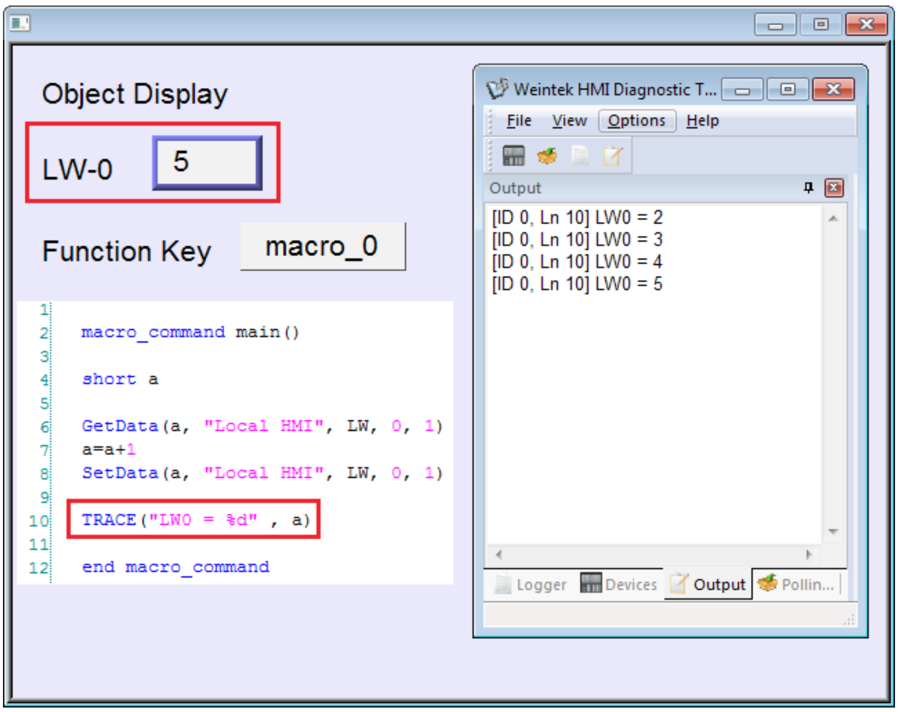
TRACE(“The results are”) // output: The results are
TRACE(“c1 = %c, s1 = %d, f1 = %f”, c1, s1, f1)
// output: c1 = a, s1 = 32767, f1 = 1.234567

end macro_command</td></tr>
</table>

8. Use LB-9059 to disable MACRO TRACE function (when ON). When set ON, the output message of TRACE won't be sent to EasyDiagnoser.
9. Users can directly execute EasyDiagnoser.exe from Utility Manager. In Utility Manager, current HMI on line will be listed; users can simply select the HMI to be watched. Please note that Project Port should be the same as Port No. used in project file.

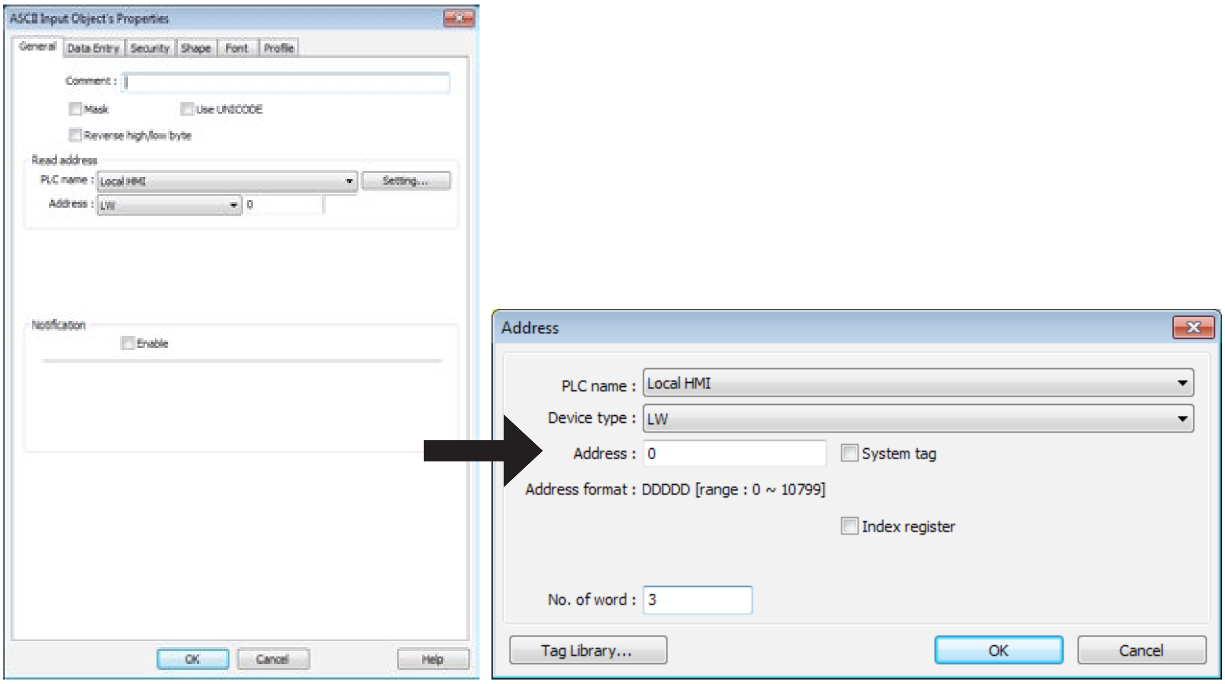


10. Download the project to HMI and start the project. If EasyDiagnoser is unable to get connection with the HMI to be watched, it is possible that HMI power is not ON, or Port No. is incorrect. This may cause EasyDiagnoser to connect then disconnect with HMI continuously. Please check the Port No. in EasyDiagnoser settings.
11. When EasyDiagnoser succeeds in connecting with HMI, simply execute macro_0, (Output) window will then display the output of the TRACE function.



18.15. EXAMPLE OF STRING OPERATION FUNCTIONS

String operation functions are added to macro to provide a convenient way to operate strings. The term “string” means a sequence of ASCII characters, and each of them occupies 1 byte. The sequence of characters can be stored into 16-bit registers with least significant byte first. For example, create an ASCII Input object and setup as follows:



Run simulation and input “abcdef”:

The string “abcdef” is stored in LW-0~LW-2 as follows (LB represents low byte and HB represents high byte):

	HB	LB
LW0	'B'	'A'
LW1	'D'	'C'
LW2	'F'	'E'
LW3		
LW4		
LW5		

The ASCII Input object reads 1 word (2 bytes) at a time as described in the previous chapter. Suppose an ASCII Input object is set to read 3 words as shown in the above example, it can actually read at most 6 ASCII characters since that one ASCII character occupies 1 byte.
The functionality of each string operation function is described in the following table:

Function name	Description
StringGet	Read string data from a device.
StringGetEx	Read string data from a device and continue executing next command even if no response from that device.
StringSet	Write string data to a device.
StringSetEx	Write string data to a device and continue executing next command even if no response from that device.
StringCopy	Copy one string to another.
StringMid	Retrieve a substring.
StringDecAsc2Bin	Convert a decimal string to an integer.
StringBin2DecAsc	Convert an integer to a decimal string.
StringDecAsc2Float	Convert a decimal string to floats.
StringFloat2DecAsc	Convert a float to a decimal string.
StringHexAsc2Bin	Convert a hexadecimal string to binary data.
StringBin2HexAsc	Convert binary data into a hexadecimal string.
StringLength	Obtain the length of a string.
StringCat	Append source string to destination string.
StringCompare	Do a case-sensitive comparison of two strings.
StringCompareNoCase	Do a case-insensitive comparison of two strings.
StringFind	Find a substring inside a larger string.
StringReverseFind	Find a substring inside a larger string; starts from the end.
StringFindOneOf	Find the first matching character from a set.
StringIncluding	Extracts a substring that contains only the characters in a set.
StringExcluding	Extracts a substring that contains only the characters not in a set.
StringToUpper	Convert the characters of a string to uppercase.
StringToLower	Convert the characters of a string to lowercase.
StringToReverse	Reverse the characters of a string.
StringTrimLeft	Trim the leading specified characters in a set from the source string.
StringTrimRight	Trim the trailing specified characters in a set from the source string.
StringInsert	Insert a string in a specific location within another string.

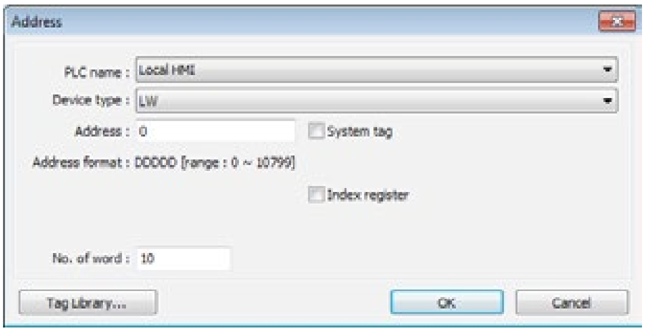
For more detailed information of the above string operation functions, please check out the "Built-In Function Block" section. In order to demonstrate the powerful usage of string operation functions, the following examples will show you step by step how to create executable project files using the new functions; starts from creating a macro, ends in executing simulation.

1. To read (or write) a string from a device:
   - Create a new macro:

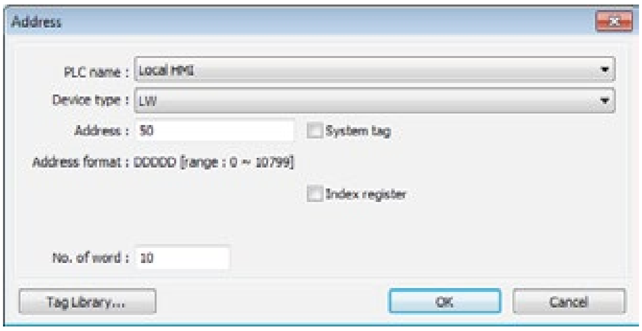
   - Edit the content:

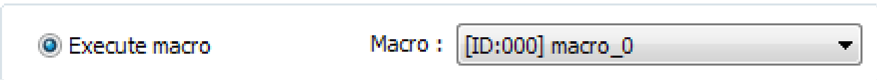
```

macro_command main()

char str[20]

StringGet(str[0], "Local HMI", LW, 0, 20)
StringSet(str[0], "Local HMI", LW, 50, 20)

end macro_command
```

The first function “StringGet” is used to read a string from LW-0~LW-19, and store it into the str array. The second function “StringSet” is used to output the content of str array.
Add one ASCII Input object and one Function Key object in window 10 of the project. The settings of these objects are shown as below. Function Key object is used to execute macro_0.
ASCII Input object:



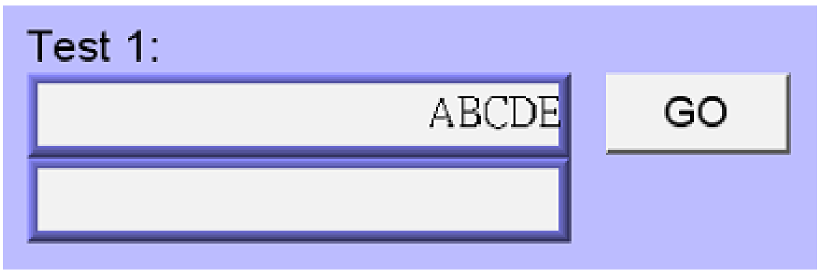
Function Key object:



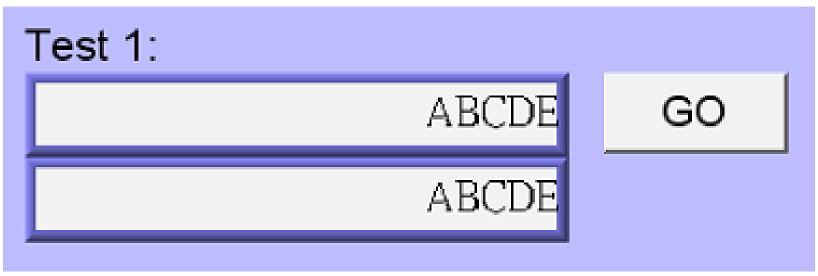


Lastly, use (Compile) to compile the project and execute (Off-line simulation) or (On-line simulation).
Follow the steps below to operate the executing project:

- Step 1. Input string
- Step 2. Press “GO” button



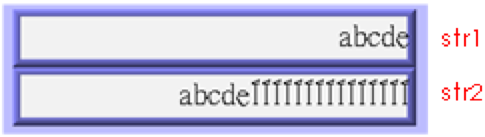
- Step 3. Output string



2. Initialization of a string.
   Create a new macro and edit the content:

```
macro_command main()

char str1[20]="abcde"
char str2[20]={'a','b','c','d','e'}

StringSet(str1[0], "Local HMI", LW, 0, 20)
StringSet(str2[0], "Local HMI", LW, 50, 20)

end macro_command
```

The data enclosed in double quotation mark (“”) is viewed as a string. str1 is initialized as a string while str2 is initialized as a char array. The following snapshot of simulation shows the difference between str1 and str2 using two ASCII Input objects.

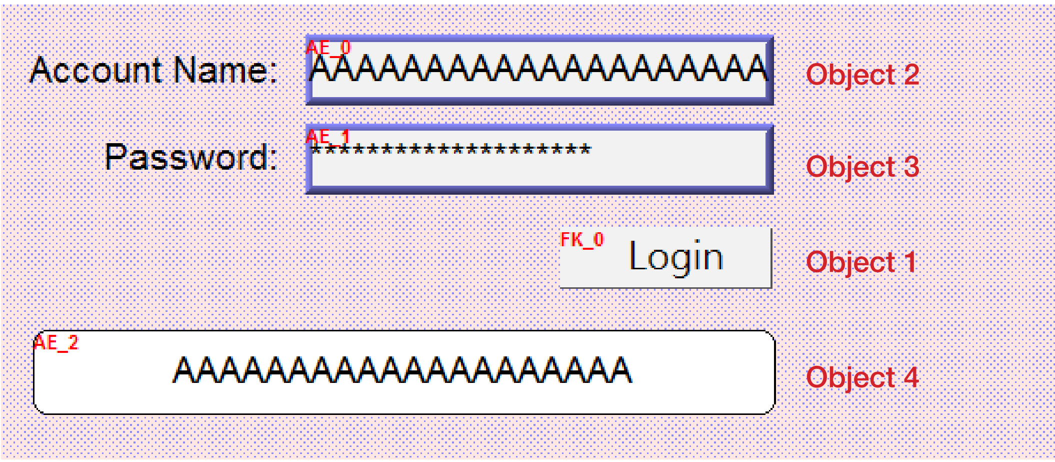


Macro compiler will add a terminating null character (‘\0’) at the end of a string. The function “StringSet” will send each character of str1 to registers until a null character is reached. The extra characters following the null character will be ignored even if the data count is set to a larger value than the length of string.
On the contrary, macro compiler will not add a terminating null character (‘\0’) at the end of a char array. The actual number of characters of str2 being sent to registers depends on the value of data count that is passed to the “StringSet” function.

3. A simple login page.
   Create a new macro and edit the content, for example, Macro (ID:001) macro_1.

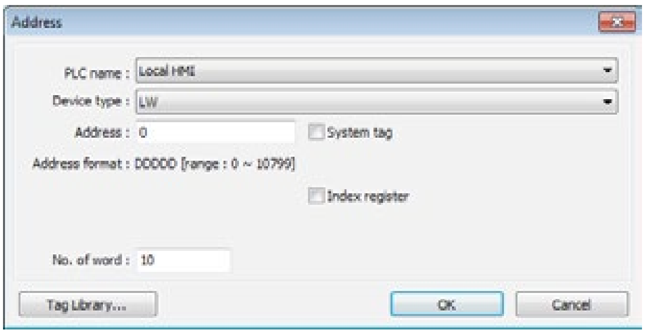
```
macro_command main()

char name[20]="admin"
char password[20]="123456"
char name_input[20], password_input[20]
char message_success[40]="Success! Access Accepted."
char message_fail[40]="Fail! Access Denied."
char message_clear[40]
bool name_match=false, password_match=false

StringGet(name_input[0], "Local HMI", LW, 0, 20)
StringGet(password_input[0], "Local HMI", LW, 50, 20)

name_match = StringCompare(name_input[0], name[0])
password_match = StringCompare(password_input[0], password[0])

FILL(message_clear[0], 0x20, 40) //FILL with white space
StringSet(message_clear[0], "Local HMI", LW, 100, 40)

if (name_match==true and password_match==true) then
    StringSet(message_success[0], "Local HMI", LW, 100, 40)
else
    StringSet(message_fail[0], "Local HMI", LW, 100, 40)
end if

end macro_command
```

The first two “StringGet” functions will read the strings input by users and store them into arrays named name_ input and password_input separately. Use the function “StringCompare” to check if the input account name and password are matched. If the account name is matched, name_match is set true; if the password is matched, password_match is set true. If both name_match and password_match are true, output the string “Success! Access Accepted.”. Otherwise, output the string “Fail! Access Denied.”.
Add ASCII Input and Function Key objects in window 10 of the project. The settings of these objects are shown as below. Function Key object is used to execute macro_1.

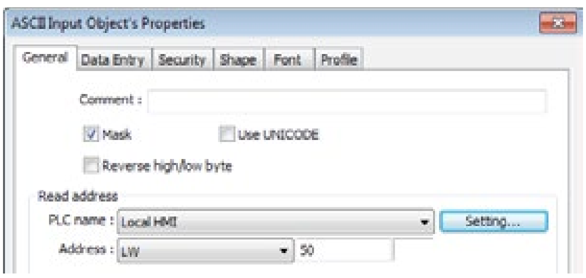


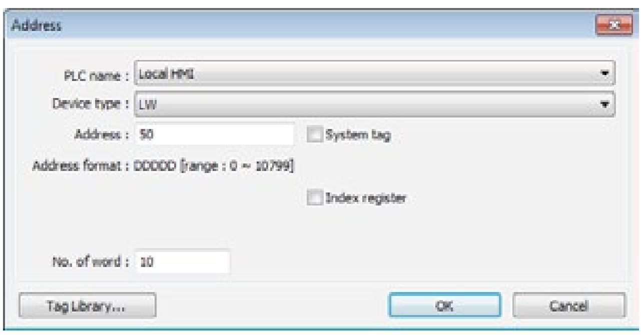
- Object 1: function Key
  Select (Execute macro) and Macro: (ID:000) macro_1

- Object 2: ASCII Input

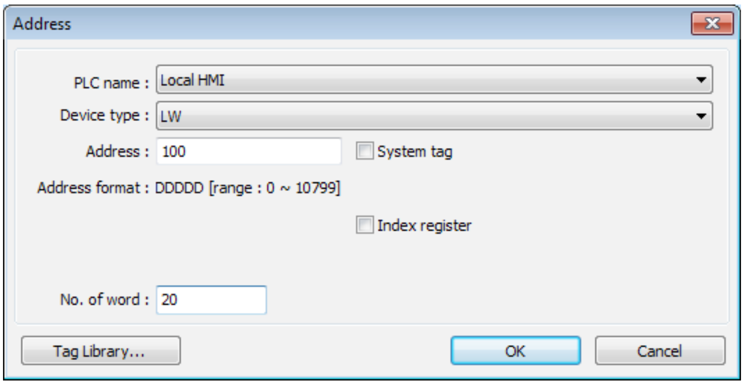


- Object 3: ASCII Input



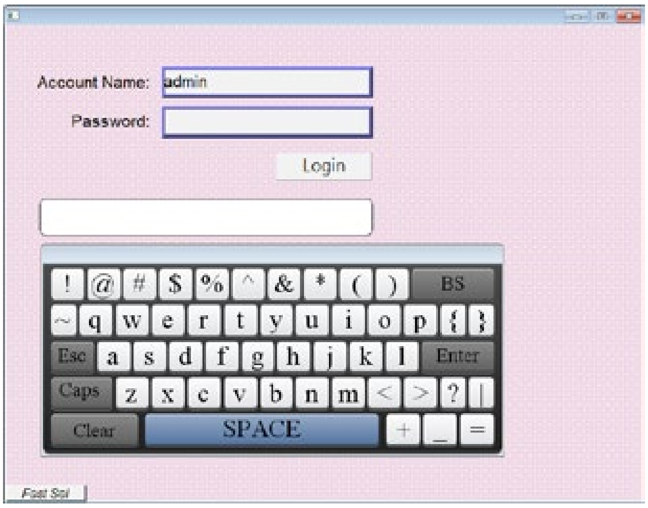


- Object 4: ASCII Display



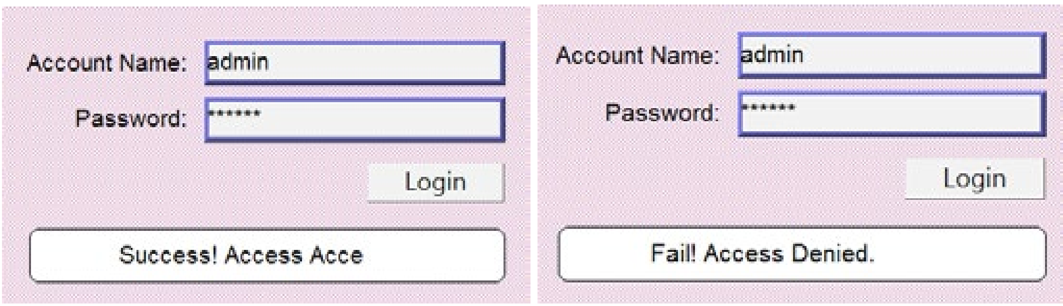
Lastly, use (Compile) to compile the project and execute (Off-line simulation) or (On-line simulation). Follow the steps below to operate the executing project:

- Step 1. Enter account name

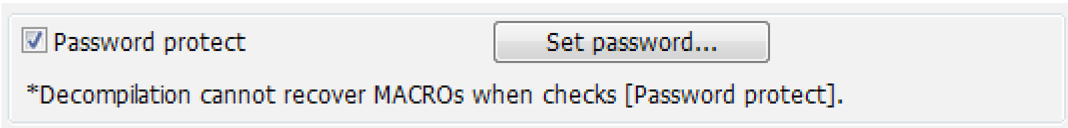


- Step 2. Enter password and press (Login) button

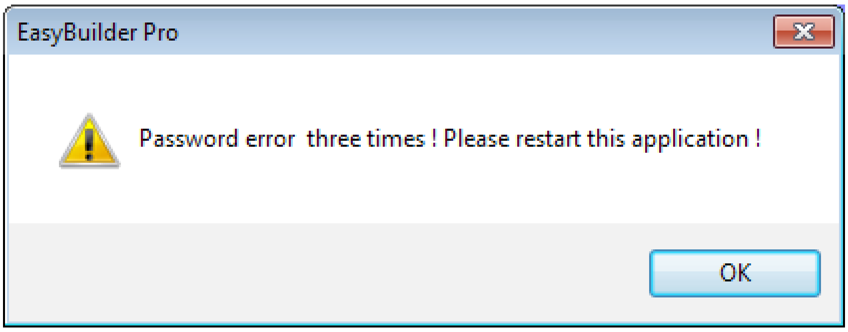
- Step 3. Login succeeded or failed



18.16. MACRO PASSWORD PROTECTION



On MACRO editing window there's the (Password protect) selection, tick it and click (Set password…) to set a password less than or equals to 10 characters (support ASCII character only, ex. "a$#*hFds").
After setting MACRO password, users will have to input correct password when opening MACRO editing window. EasyBuilder Pro should be rebooted for typing the password again after 3 incorrect attempts.



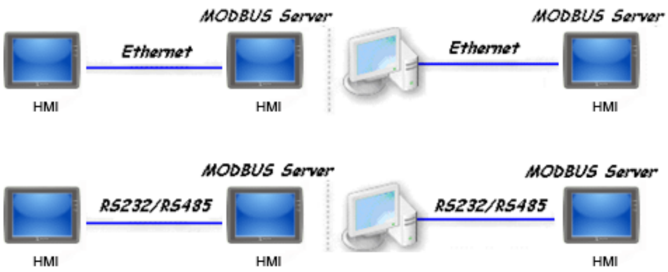
	Note When MACRO is password protected, de-compilation of EXOB file will not be able to restore MACRO contents.

19. CONFIGURE HMI AS A MODBUS SERVER

This chapter explains how to configure HMI as a Modbus Server.

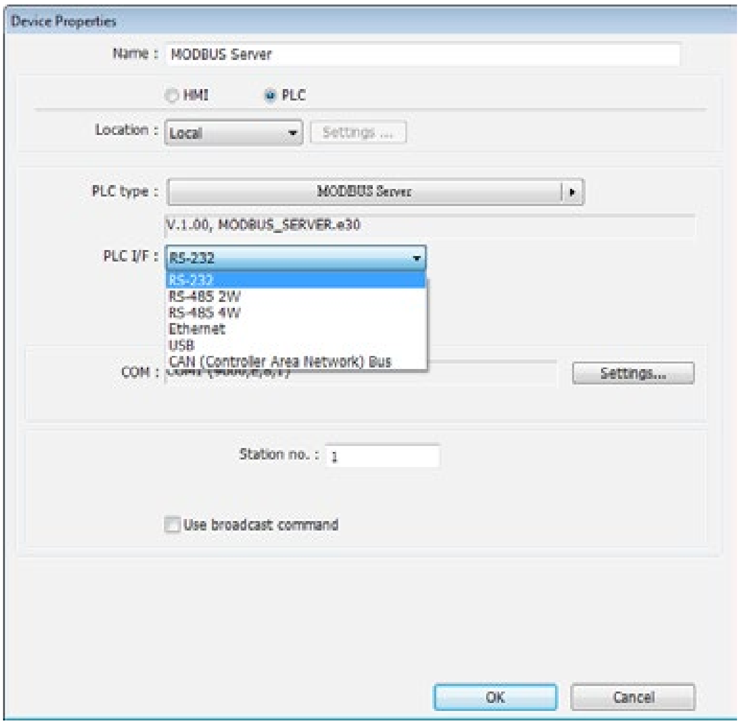
19.1. OVERVIEW

Once the HMI is configured as a Modbus device, the data of HMI can be read or written via Modbus protocol. As shown in the following figure, the HMI is configured as a Modbus device (also called Modbus Server). The HMI, PC or other devices can use Modbus protocol to read or write HMI data via Ethernet or RS232 / RS485 interface.

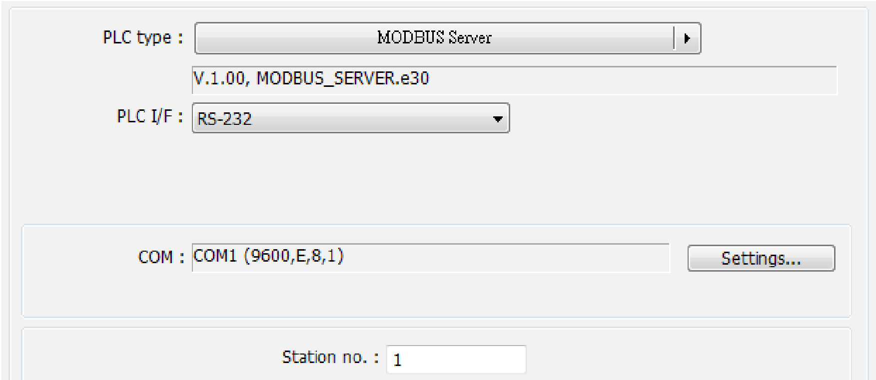


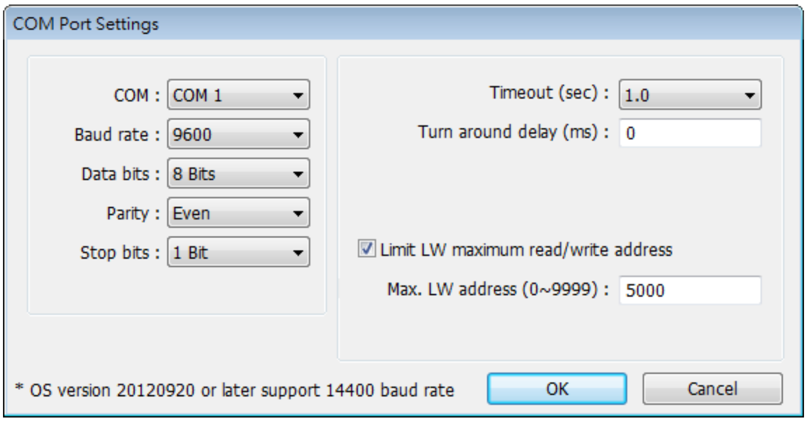
19.2. STEPS TO CREATE A MODBUS SERVER

1. To configure the HMI as a Modbus device, add a new device in the device list first. Click (PLC type) drop-down box and select “Modbus Server” driver. There are six selections for (PLC I/F): RS232 / RS485 2W / RS485 4W / Ethernet / USB / CAN. Choose the PLC interface appropriate for your PLC model.

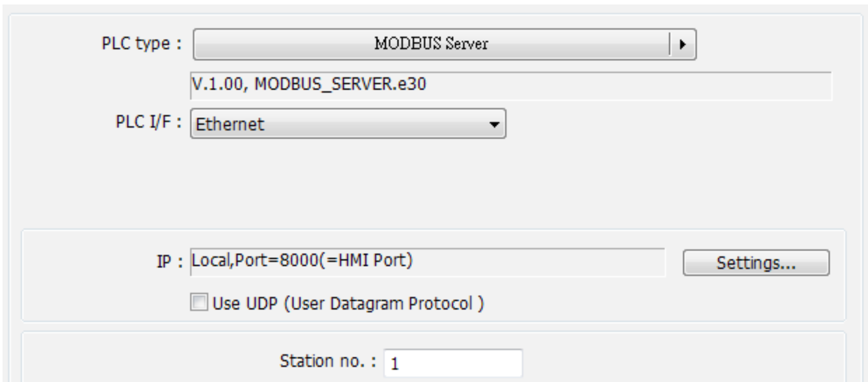


2. If (PLC I/F) is set to (RS232) or (RS485), please select (COM) (COM 1 ~ COM 3) and set correct communication parameters as shown in the following figure. Modbus Server (Station no.) is set to 1. Click (Settings), the maximum LW address range read / written by Modbus Client can be set. When the object in the project uses a LW register, the Modbus Client will not be able to read or write an address that is not within the specified range.

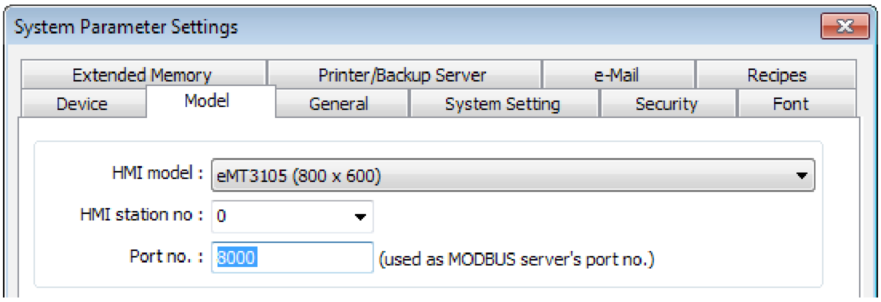




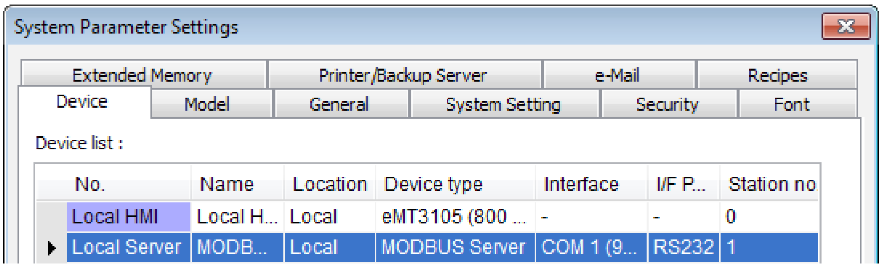
▪ If (PLC I/F) is set to (Ethernet), please set (Port no.)



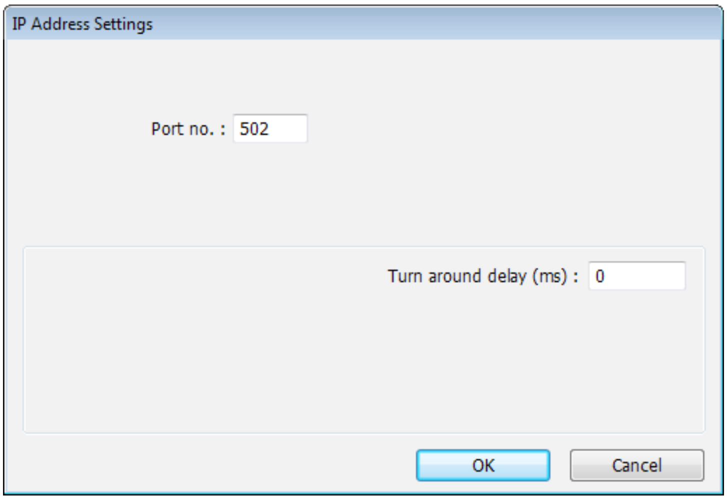
▪ The (Port no.) of Modbus Server and HMI must be the same. To change the port number, please set in the (Model) tab



3. When finished, Modbus Server is listed in (Device) tab. The configuration of Modbus device is completed. Compile the .emtp file and download the compiled .exob file to the HMI, then HMI data can be read or written by using Modbus protocol.



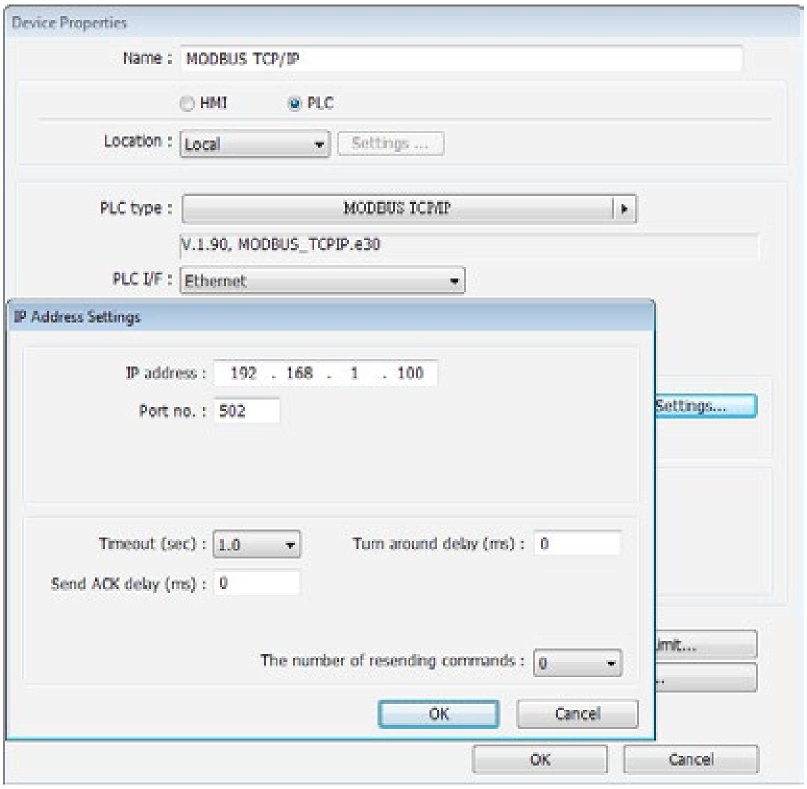
Note
For cMT-SVR, if (Ethernet) PLC interface is chosen, then enter the port number.



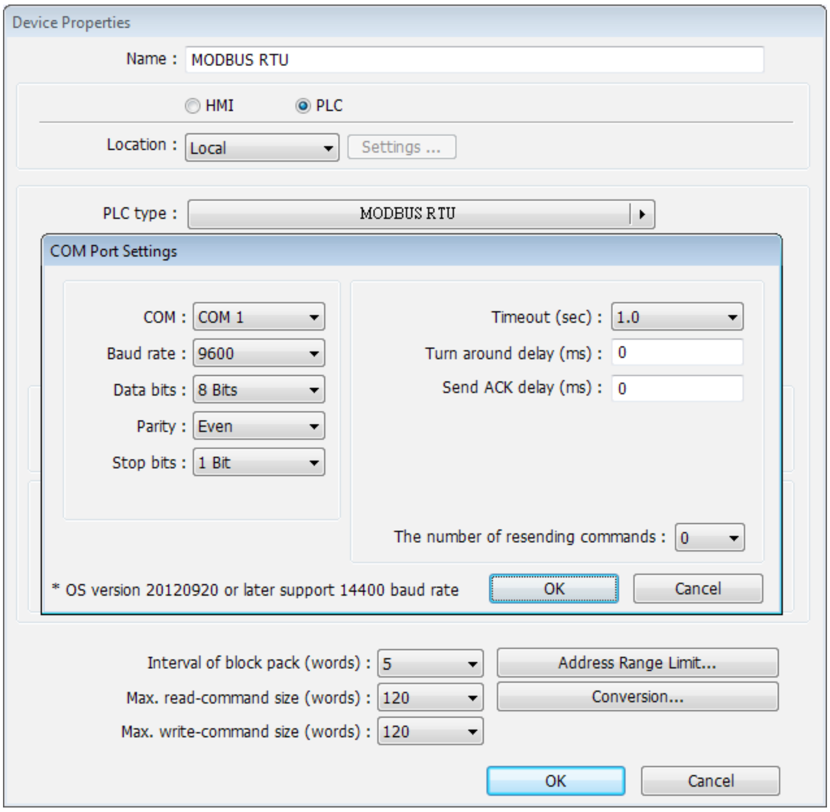
19.3. STEPS TO ACCESS A MODBUS SERVER

Two HMIs can be configured as one Modbus client and one Modbus server to communicate and exchange data.

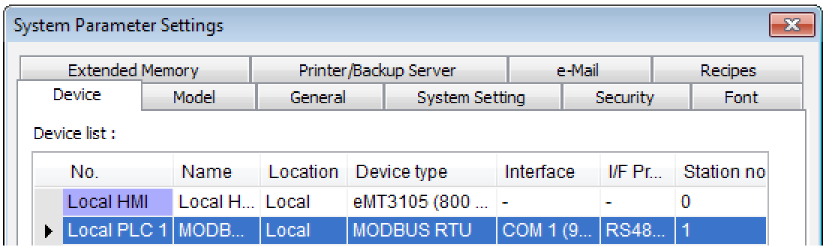
1. Add a new device in client's device list. If the client chooses (Ethernet) PLC interface, set (PLC type) to "Modbus TCP/IP" and fill in the correct (IP address) (the IP of Modbus Server), (Port no.), and (Station no.).



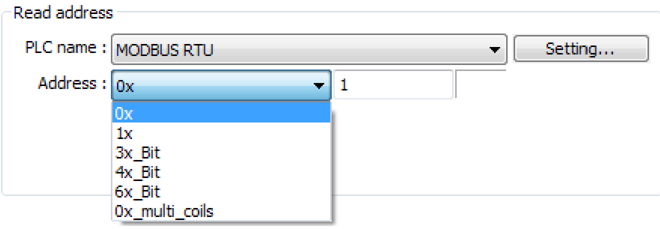
If the client chooses (RS232) or (RS485) PLC interface, the (PLC type) must be set to "Modbus RTU", and its communication parameters also must be configured correctly.



2. When finished, click (OK), then a new device “Modbus RTU” is listed in the (Device) tab.



3. In the setting page of each object, select “Modbus RTU” in (PLC name), and set the address of Modbus RTU.



- Since the server is an HMI, the corresponding read and write addresses are listed below:

0x/1x (1 ~ 12096)	LB (0 ~ 12095)
3x/4x/5x (1 ~ 9999)	LW (0 ~ 9998)
3x/4x/5x (10000 ~ 65535)	RW (0 ~ 55535)

19.4. CHANGING MODBUS SERVER STATION NUMBER ONLINE

EasyBuilder Pro provides the following system registers to change Modbus Server station number online.

LW-9541	Modbus/ASCII server station no. (COM 1)
LW-9542	Modbus/ASCII server station no. (COM 2)
LW-9543	Modbus/ASCII server station no. (COM 3)
LW-9544	Modbus/ASCII server station no. (Ethernet)

19.5. MODBUS ADDRESS TYPE

In the EasyBuilder Pro, the address types of Modbus protocol are 0x, 1x, 3x, 4x, 5x, 6x, 3x_bit and 4x_bit. Modbus RTU function codes are listed below:

0x: Coils	A read and write device type. When reading a bit with this device type, the function code is 01H. When writing a bit, the function code is 05H. When writing multiple bits, the function code is 0fH.
1x: Discrete Inputs	A read only device type. When reading a bit the function code is 02H.
3x: Input Registers	A read only device type. When reading data, the function code is 04H.
4x: Holding Register	A read and write device type. When reading data, the function code is 03H. When writing data, the function code is 10H.
5x	The function code is the same as 4x. The difference is that 5x makes double word swap when the format is 32-bit unsigned. If the data read by 4x is 0x1234, the data read by 5x is 0x3412.
6x	A read and write device type. When reading data, the function code is 03H. The difference from 4x is that when writing data, the function code is 06H, meaning to write a single register.
3x_bit	The function code is the same as 3x. The difference is that 3x_bit reads a single bit in the data.
4x_bit	The function code is the same as 4x. The difference is that 4x_bit reads a single bit in the data.

☞ For more information, see “37 Modbus TCP/IP Gateway”.

20. HOW TO CONNECT A BARCODE READER

This chapter explains how to connect a Barcode reader and the relevant settings.

20.1. OVERVIEW

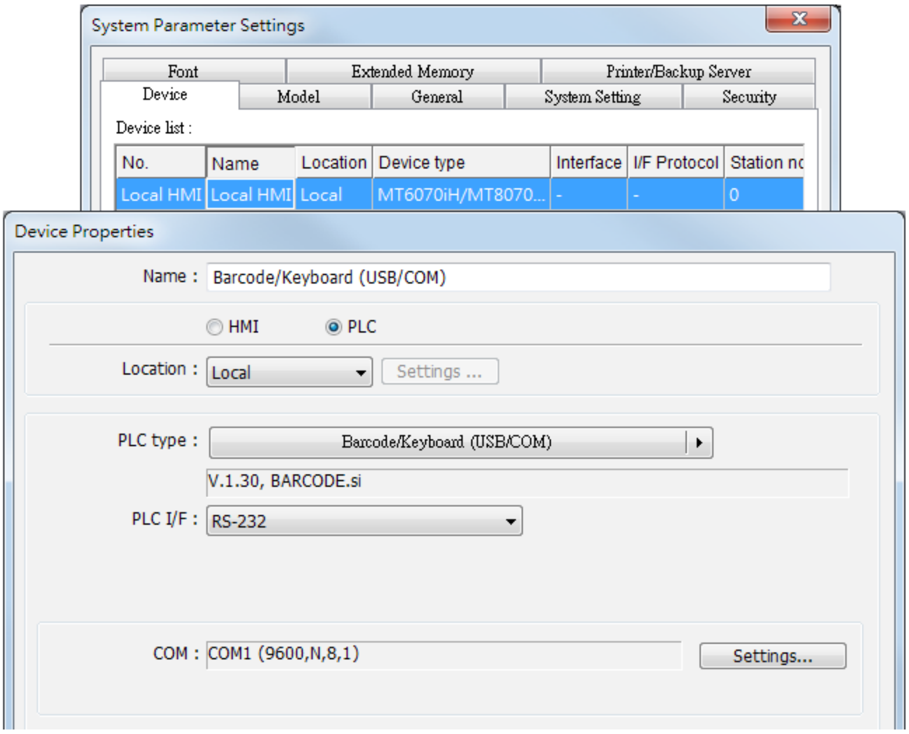
HMI can connect with barcode reader via the following interfaces:

- USB
- COM port

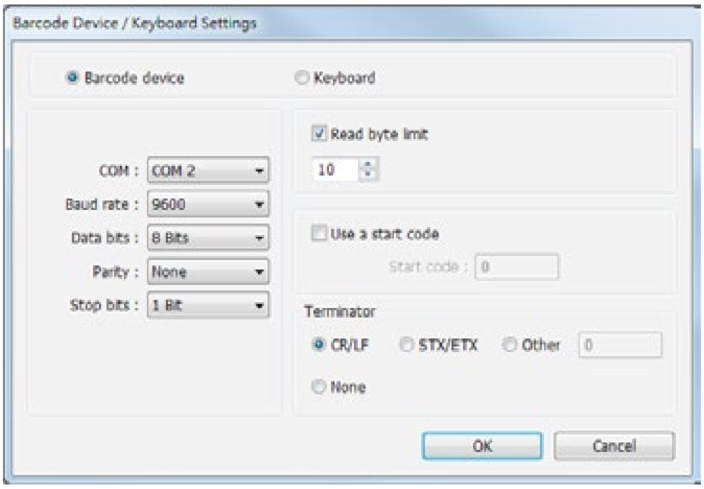
To connect a barcode reader, please add a new device by the following steps.

20.2. STEPS TO CONNECT A BARCODE READER

1. In EasyBuilder Pro » (Edit) » (System Parameter Settings) » (Device list), add a new device.



2. Click (Settings) and finish (Barcode Device / Keyboard Settings).



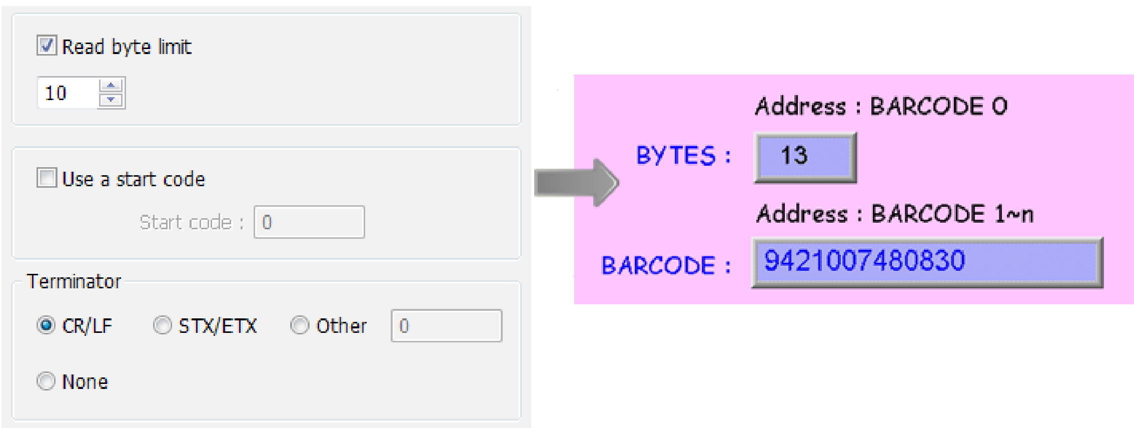
Setting	Description
Timeout	If select (Keyboard), set a time range for keyboard entries. The system starts counting time from the first entry.
COM **Baud rate** **Data bits** **Parity** **Stop bits**	When using COM port, please set the communication parameters correctly. When using USB, there is no need to set the parameters.
Read byte limit	If this check box is selected, the number of bytes a barcode reader reads is restricted in order to prevent overloading. The range is 10 to 512. Please note that the data cannot be read if it exceeds the limit.
Use a start code	If this check box is selected, the data is only valid when the first data is identical to the start code, otherwise the data will be ignored. The start code will not be stored in the address of barcode reader. For example: if the start code is 255 (0xff), and the data read is: 0xff 0x34 0x39 0x31 0x32 0x30 0x30 0x34 0x37 The data saved in the designated barcode reader address will be: 0x34 0x39 0x31 0x32 0x30 0x30 0x34 0x37
Terminator	Terminator means the end of data. When a terminator is detected, it stands for the end of data stream.
CR/LF	0x0a or 0x0d stands for the end of data stream.
STX/ETX	0x02 or 0x03 stands for the end of data stream.
Other	Users can set the terminator.
None	If this check box is selected, HMI will save all the data to the designated address of barcode reader.

When finish setting, a new device is added to the (Device list).
Now the barcode reader can be selected in (PLC type) when creating an object. The address types are listed in the following table.

Address Type	Address Name	Description
Bit	FLAG	**FLAG 0** indicates the status of data reading. When reading data, the status of FLAG 0 is set OFF and will return ON after reading successfully.
	RESET	**RESET 0** clears the data of BARCODE and RESULT when set ON.
	CONNECT_STATUS	CONNECT_STATUS 0 indicates whether the barcode reader (USB interface) is connected. When the status is ON, the barcode reader is connected.
Word	BARCODE	**BARCODE 0:** number of bytes currently read. **BARCODE 1 ~ n:** stores the data read.
	RESULT	**RESULT 0** indicates the result of data reading. The following codes indicate: **0x00** Waiting to read BARCODE. **0x01** BARCODE successfully read. **0x02** Invalid BARCODE format. **0x03** The number of bytes specified in (Read byte limit) exceeded. **0x04** The Start Code of the data read does not match the setting. **0x05** The Terminator of the data read does not match the setting.

Example 1

The following is a setting example, the barcode is 9421007480830. BARCODE 0 is the address of Numeric Display Object (BYTES) and BARCODE 1 ~ n is the address of ASCII Display object (BARCODE).



In the example the data stored in the barcode reader address is listed in the following table:

Barcode Reader Address	Data
BARCODE 0	13 bytes (decimal) However, the data saved is 14 bytes = 7 words. It is because when the number of bytes is an odd number, the system adds a byte (0x00) to make it an even number.
BARCODE 1	3439 (HEX)
BARCODE 2	3132 (HEX)
BARCODE 3	3030 (HEX)
BARCODE 4	3437 (HEX)
BARCODE 5	3038 (HEX)
BARCODE 6	3338 (HEX)
BARCODE 7	0030 (HEX)

Note
HMI can only connect with one USB barcode reader. When the device list in the project includes this kind of device, the system register LB-9064: (enable USB barcode device (disable keyboard) (when ON)) is set ON. To enable USB keyboard again and stop using USB barcode reader, please set LB-9064 OFF.

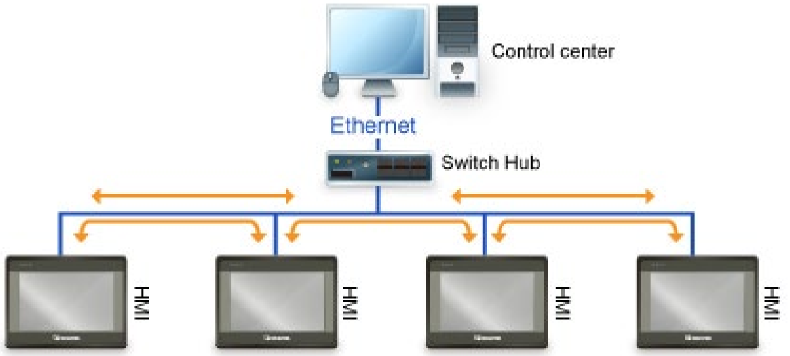
Download
Click the icon to download the demo project. Please confirm your internet connection.

21. ETHERNET COMMUNICATION AND MULTI-HMI CONNECTION

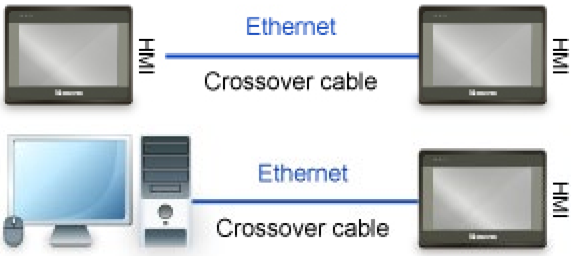
This chapter explains how to connect multiple devices via Ethernet.

21.1. OVERVIEW

There are two ways of Ethernet communication:
- Use RJ45 straight through cable and hub



- Use RJ45 crossover cable and without hub, but this is limited to point-to-point connection (HMI to HMI or PC to HMI)

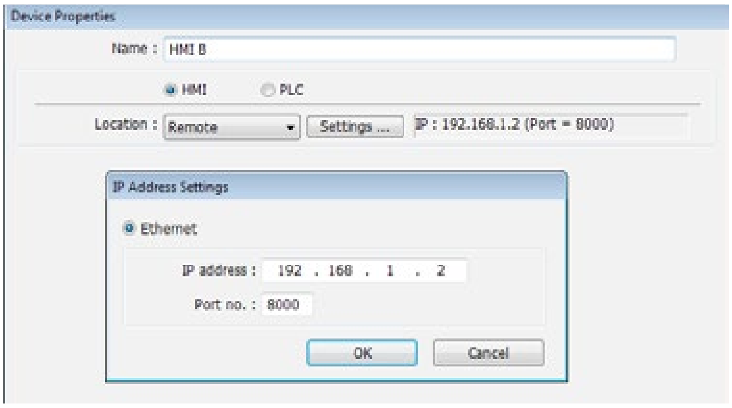


Through Ethernet network, the system provides the following methods for data transmission:
- HMI to HMI communication
- PC to HMI communication
- Operating the PLC connected to another HMI

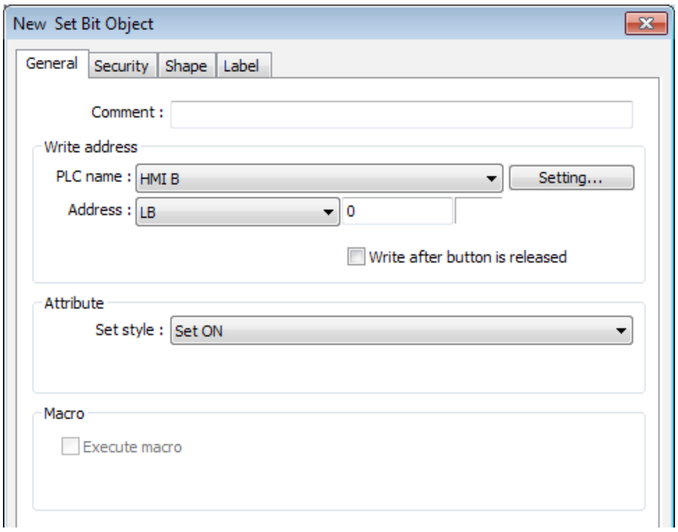
21.2. HMI TO HMI COMMUNICATION

To exchange data between one HMI and another HMI, add a new remote HMI device in (System Parameter Settings). If there are 2 HMIs (HMI A and HMI B), in order to use a Set Bit object on HMI A to control (LB-0) on HMI B, the setting of the project of HMI A is explained in the following part.

1. Set the IP address of the two HMIs, for example, HMI A: 192.168.1.1, HMI B: 192.168.1.2.
2. In (System Parameter Settings) » (Device list), add a remote HMI B (IP: 192.168.1.2).



3. Create a Set Bit Object, select “HMI B” in (PLC name) to control the address of the remote HMI.

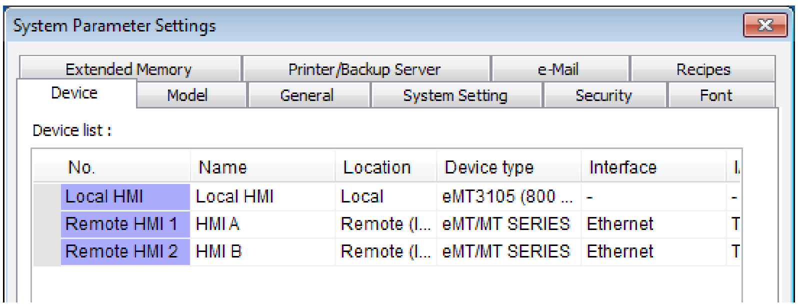


Note
- One HMI can handle requests from a maximum of 64 HMIs simultaneously.
- One cMT-SVR can handle requests from a maximum of 32 HMIs simultaneously.

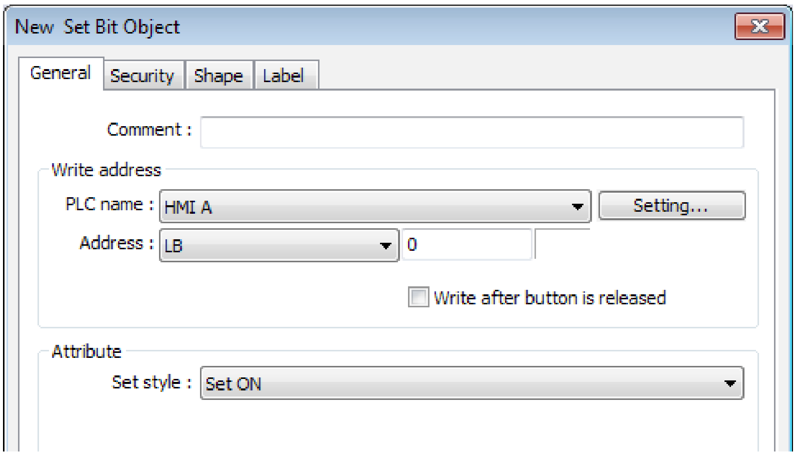
21.3. PC TO HMI COMMUNICATION

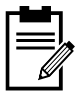
With On-line Simulation, PC can collect data from HMI through Ethernet network and save the data files to PC. To connect PC with two HMIs (HMI A and HMI B), the setting of the project on PC is explained in the following part.

1. Set the IP address of the two HMIs, for example, HMI A: 192.168.1.1, HMI B: 192.168.1.2.
2. In (System Parameter Settings) » (Device list), add a remote HMI A (IP: 192.168.1.1) & HMI B (IP: 192.168.1.2).



3. Create a Set Bit Object, select “HMI A” in (PLC name) to control the address of the remote HMI A. Same for the HMI B.

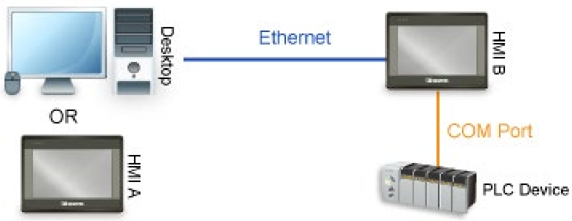




Note
- A PC can control at most 64 HMIs simultaneously.
- As shown above, HMI can also control PC. PC can be seen as another HMI, that is, adding a remote HMI in the project of HMI A / HMI B, and the IP of the remote HMI is set to the IP of PC.

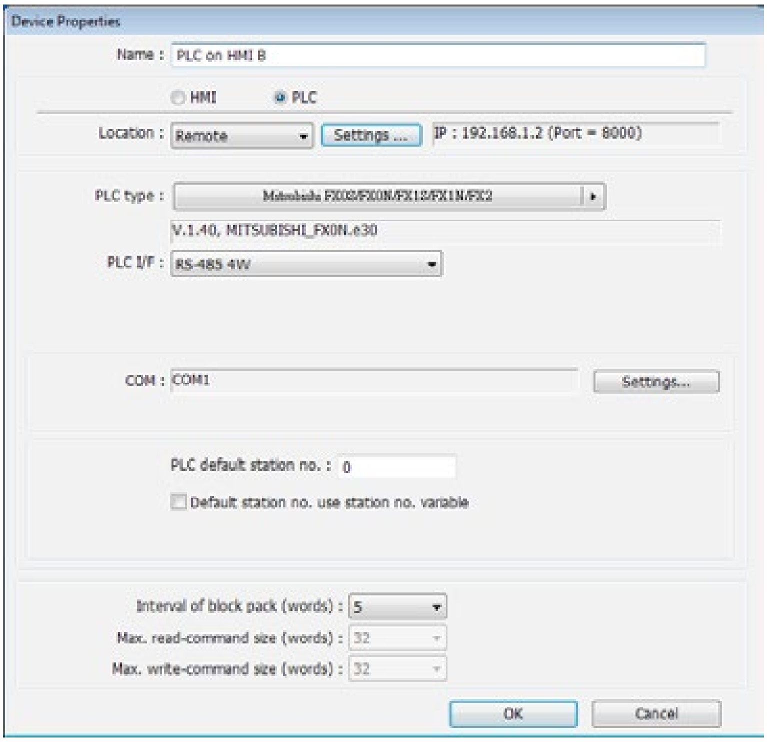
21.4. OPERATING THE PLC CONNECTED WITH OTHER HMI

Through Ethernet network, PC or HMI can operate the PLC that is connected to another HMI. If PLC is connected to COM 1of HMI B, when using PC or HMI A to read PLC data, the setting of the project of PC or HMI A is explained in the following part.

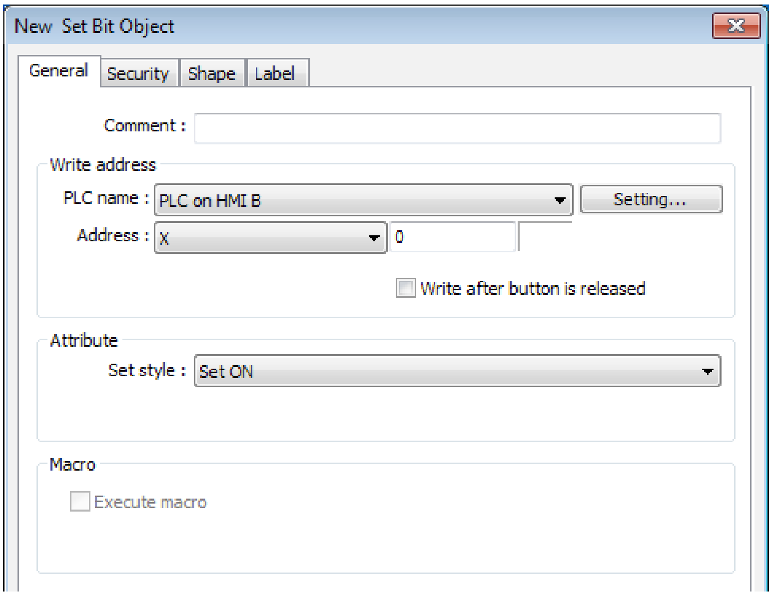


21.4.1. Settings of eMT / mTV Series

1. Set the IP address of HMI B, for example, 192.168.1.2.
2. In (System Parameter Settings) » (Device list), add a remote PLC, and set (Name) to “PLC on HMI B”. Set correct parameters. Since this PLC is connected to remote HMI B, set the IP address to HMI B (IP: 192.168.1.2).

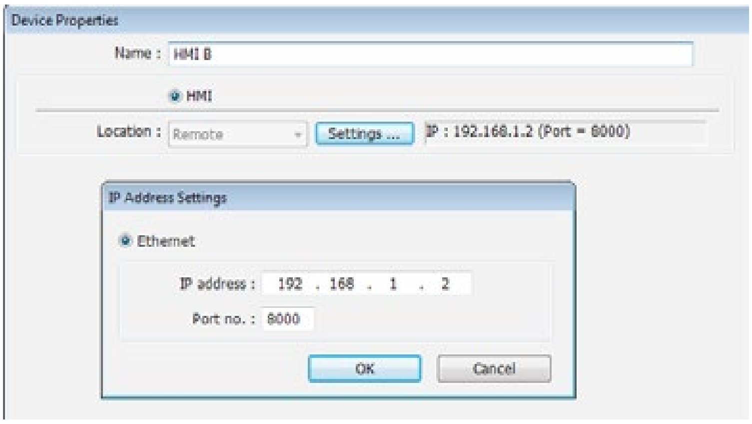


3. Create a Set Bit Object, select “PLC on HMI B” in (PLC name) to control the PLC connected with the remote HMI B.

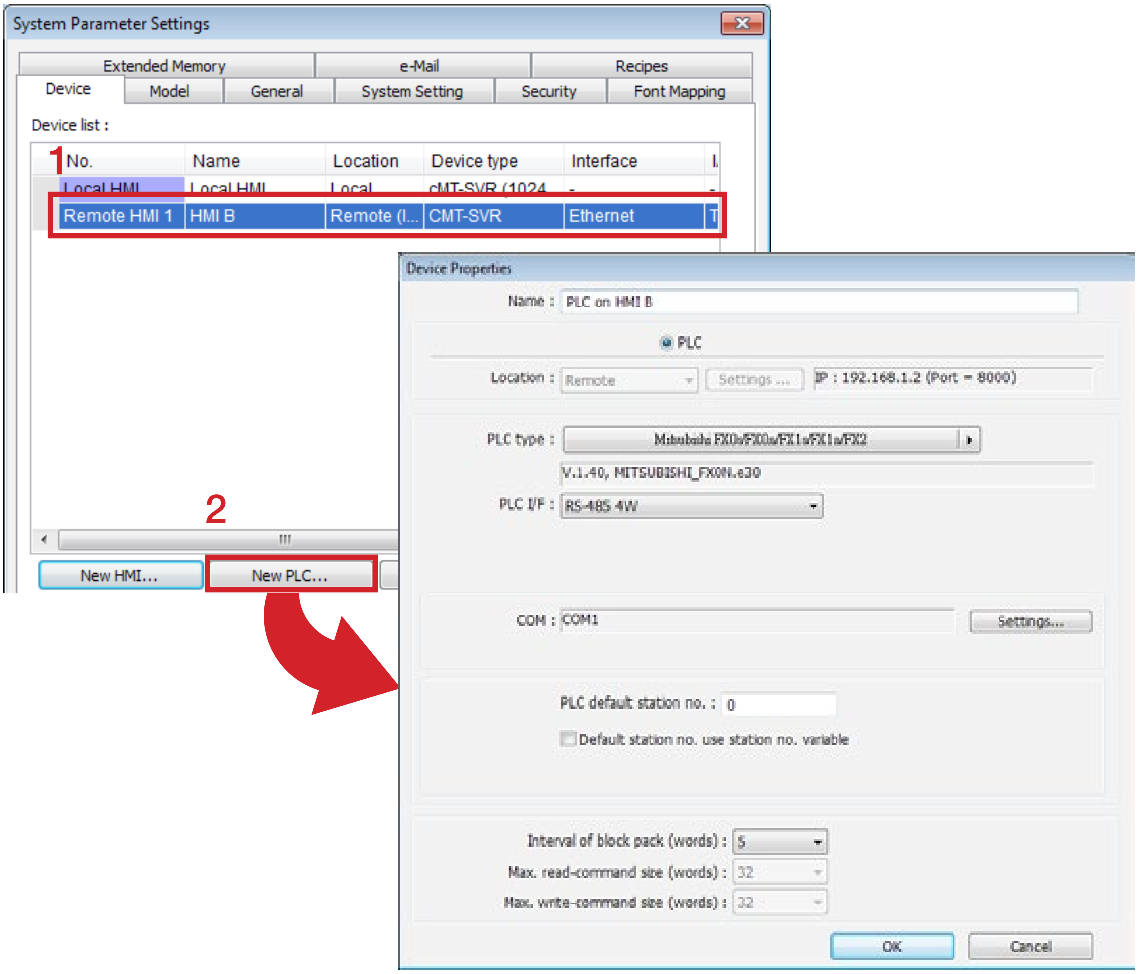


21.4.2. Settings of cMT-SVR Series

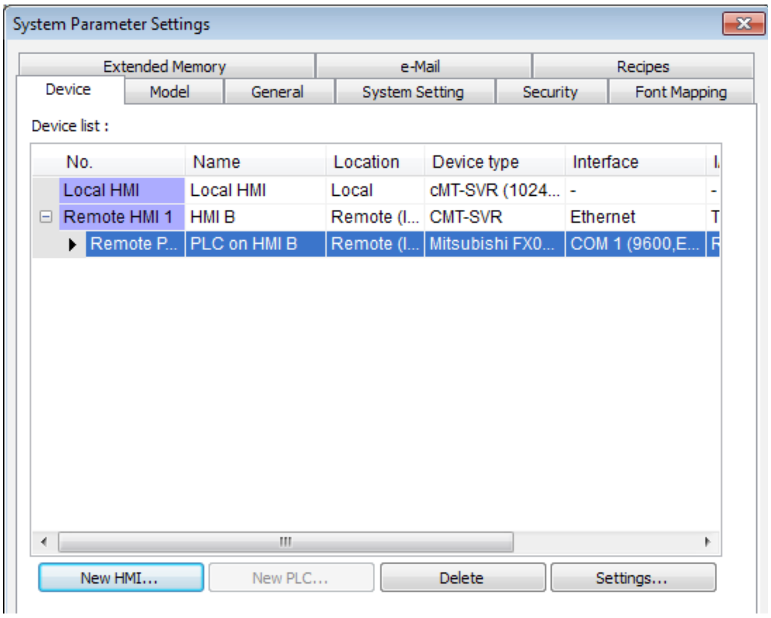
1. Set the IP address of HMI B, for example, 192.168.1.2.
2. In (System Parameter Settings) » (Device list), click (New HMI). Set the IP address to HMI B (IP: 192.168.1.2).



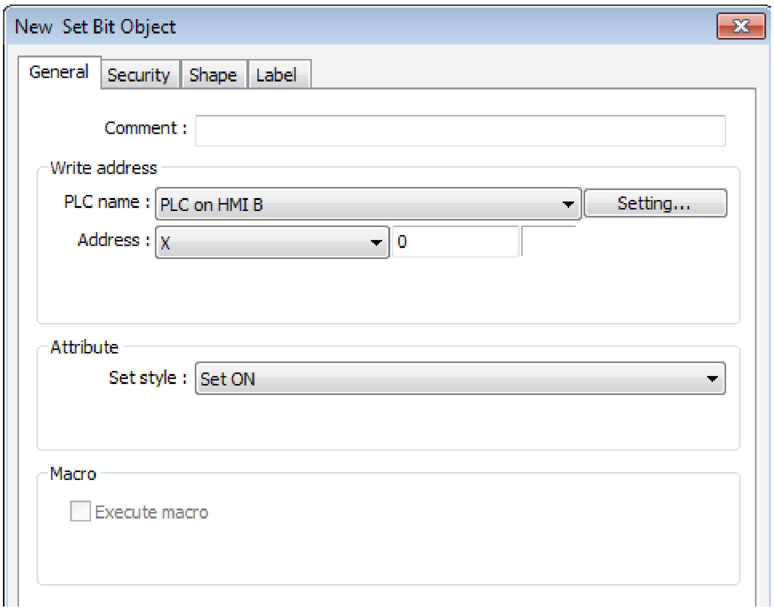
3. In the project of HMI B, go to (System Parameter Settings) » (Device list), click (New PLC), set (Name) to “PLC on HMI B”. Set correct parameters.



4. When finished, a remote PLC can be found under Remote HMI 1. Local HMI 1 stands for HMI A, Remote HMI 1 stands for HMI B, and Remote PLC 1 is connected with HMI B.



5. Create a Set Bit Object, select “PLC on HMI B” in (PLC name) to control the PLC connected with the remote HMI B.



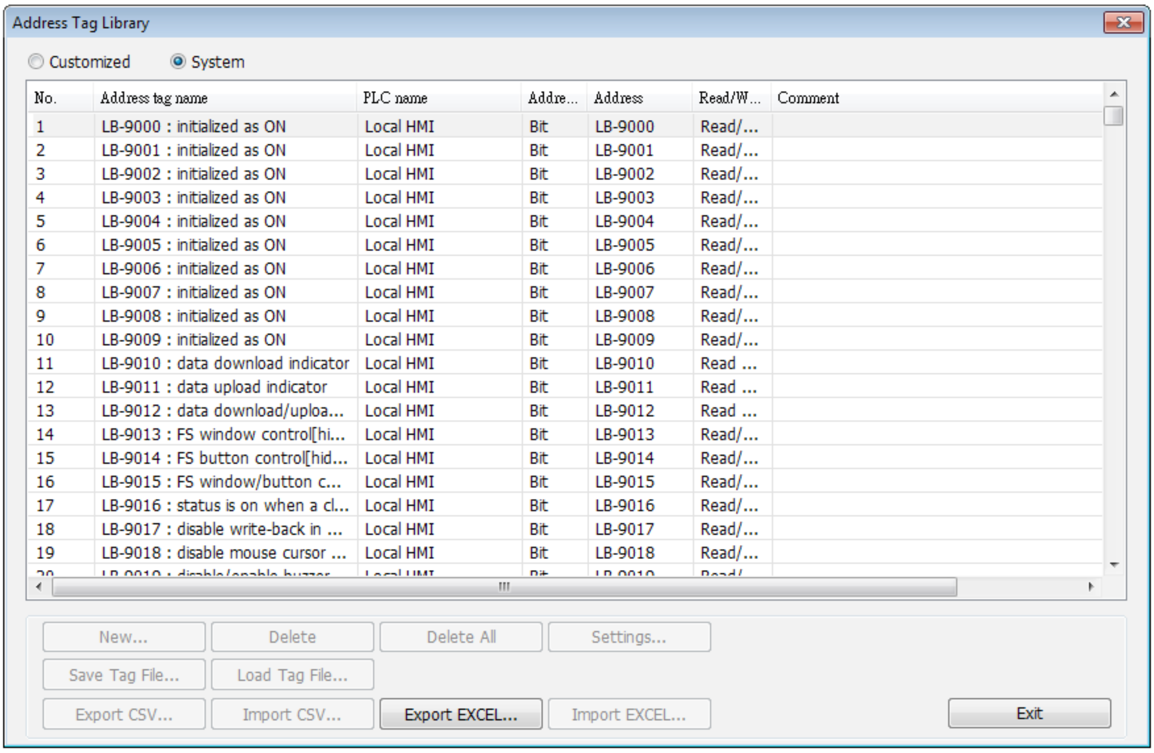
Note
Remote HMI in a cMT-SVR project must be a cMT-SVR machine. Thus, a cMT-SVR cannot communicate with PLCs connected with other series, such as eMT, mTV-series.

22. SYSTEM REGISTERS

This chapter introduces different types of registers.

22.1. OVERVIEW

Some Word and Bit addresses are reserved in EasyBuilder Pro. These registers are reserved for different functions. This chapter introduces different types of registers.
The “C” letter in the register tables stands for “Control”, which means that this register not only allows write operation, but also can be controlled by Macro or a remote HMI.



22.2. THE ADDRESS RANGES OF LOCAL HMI

22.2.1. Bits

Register	Device type	Range	Format
Local Bits	LB	0 ~ 12095	DDDDD
Local Word Bits	LW_BIT	0 ~ 1079915	DDDDDdd DDDDD: address dd: bit no. (00 ~ 15)
Retentive Bit Index	RBI	0 ~ 65535f	DDDDDh DDDDD: address h: bit no. (0 ~ f) Use LW-9000 as Index Register, and correspond to RW_Bit
Retentive Word Bits	RW_Bit	0 ~ 524287f	DDDDDh DDDDD: address h: bit no. (0 ~ f)
Retentive A Word Bits	RW_A_Bit	0 ~ 65535f	DDDDDh DDDDD: address h: bit no. (0 ~ f)

22.2.2. Words

Register	Device type	Range	Format
Local Words	LW	0 ~ 10799	DDDDD
Retentive Words	RW	0 ~ 524287	DDDDDD
Retentive Word Index	RWI	0 ~ 65535	DDDDD Use LW-9000 as Index Register, and correspond to RW
Retentive A Words	RW_A	0 ~ 65535	DDDDD
Extended Memory Words	EM0 ~ EM9	0 ~ 1073741823	DDDDDDDDDD

22.3. SYSTEM REGISTERS

22.3.1. HMI Time

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-11958	time setting error (when ON) *Note 3	R	R	R
LW-9010	(16bit-BCD): local second	R/W	R/C	R/C
LW-9011	(16bit-BCD): local minute	R/W	R/C	R/C
LW-9012	(16bit-BCD): local hour	R/W	R/C	R/C
LW-9013	(16bit-BCD): local day	R/W	R/C	R/C
LW-9014	(16bit-BCD): local month	R/W	R/C	R/C
LW-9015	(16bit-BCD): local year	R/W	R/C	R/C
LW-9016	(16bit-BCD): local week	R	R	R
LW-9017	(16bit): local second	R/W	R/C	R/C
LW-9018	(16bit): local minute	R/W	R/C	R/C
LW-9019	(16bit): local hour	R/W	R/C	R/C
LW-9020	(16bit): local day	R/W	R/C	R/C
LW-9021	(16bit): local month	R/W	R/C	R/C
LW-9022	(16bit): local year *Note 1	R/W	R/C	R/C
LW-9023	(16bit): local week *Note 2	R	R	R
LW-9030	(32bit): system time (unit: 0.1 second)	R	R	R
LW-9048	(16bit): time (0: AM, 1: PM)	R/W	R/C	R/C
LW-9049	(16bit): local hour (12-hour format)	R/W	R/C	R/C

Notes

1. Value range: 2000 ~ 2037.
2. Value range: 0 ~ 6, stand for Sunday ~ Saturday.
3. When use LW-9010 to LW-9023 to update RTC time, the system will check if RTC time is successfully updated. If the system still fails to update RTC time, the system register (LB-11958: time setting error) will be set ON, and restore to the time before update. Updating time on PC during simulation by using LW-9010 to LW-9023 is ineffective.

22.3.2. HMI Operation

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9018	disable mouse cursor (set ON)	R/W	R/C	R/C
LB-9019	disable/enable buzzer	R/W	R/C	R/C
LB-9020	show (set ON)/ hide (set OFF) system setting bar	R/W	R/C	R/C
LB-9033	disable(when on)/enable (when off) HMI upload function *Note 1	R/W	R/C	R
LB-9040	backlight up (set ON) *Note 2	W	C	C
LB-9041	backlight down (set ON) *Note 2	W	C	C
LB-9047	reboot HMI (set ON when LB-9048 is on)	W	C	C
LB-9048	reboot-HMI protection	R/W	R/C	R/C
LB-9062	open hardware setting dialog (set ON)	W	C	C
LB-9063	disable(set ON)/enable(set OFF) popuping information dialog while finding an USB disk	R/W	R/C	R/C
LB-9064	enable USB barcode device (disable keyboard) (when ON) *Note 5	R/W	R/C	R
LB-11959	LED indicator control *Note 4	R/W	R/C	R/C
LB-12051	buzzer status (active when ON)	R/W	R/C	R/C
LW-9007	(16bit): hardware index	R	R	R
LW-9008	(32bit-float): battery voltage *Note 3	R	R	R
LW-9025	(16bit): CPU loading (x 100%)	R	R	R
LW-9026	(16bit): OS version (year)	R	R	R
LW-9027	(16bit): OS version (month)	R	R	R
LW-9028	(16bit): OS version (day)	R	R	R
LW-9040	(16bit): backlight index *Note 2	R	R	R
LW-9051	(16bit): audio volume (0 ~100)	R/W	R/C	R/C
LW-9080	(16bit): backlight saver time (unit: minute)	R/W	R/C	R/C
LW-9081	(16bit): screen saver time (unit: minute)	R/W	R/C	R/C
LW-9199	(16bit): external keyboard layout: 0 (QWERTY), 1 (AZERTY)	R/W	R/C	R/C
LW-9350	(16bit): pending command no. in local HMI	R	R	R
LW-10884	(16 words): HMI name	R/W	R/C	R/C
LW-11155	(32bit): the total size of HMI memory (unit: KB)	R	R	R

LW-11157	(32bit): the free size of HMI memory (unit: KB)	R	R	R
LW-11159	(16bit): memory loading (x 100%)	R	R	R

Notes
1. After changing the settings, please reboot HMI for the updates to take effect.
2. Use LW-9040 together with LB-9040 ~ LB-9041 to adjust the backlight brightness, range: 0 ~ 31.
3. Only supported by eMT Series. When the battery voltage level, indicated by LW-9008, drops below 2.8V, battery replacement is recommended.
4. When multiple mTV or cMT-SVR devices are used, this register can be triggered to make the LED indicator blink for identifying the device.
5. LB-9064: Enable USB barcode device (disable keyboard).

Download
Click the icon to download the demo project. Please confirm your internet connection.

22.3.3. Touch Position

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-9041	(16bit): touch status word (bit 0 on = user is touching the screen)	R	R	R
LW-9042	(16bit): touch x position	R	R	R
LW-9043	(16bit): touch y position	R	R	R
LW-9044	(16bit): leave x position	R	R	R
LW-9045	(16bit): leave y position	R	R	R

Download
Click the icon to download the demo project that explains how to how to trigger relevant registers to change page with finger slide. Please confirm your internet connection.

22.3.4. Local HMI Network Information

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-12041	refresh HMI IP (LW-9129~9132) (set ON)	R/W	R/C	R/C
LB-12094	update ethernet 1 setting (IP, netmask, gateway) (set ON)	R/W	R/C	R/C
LB-12095	update ethernet 2 setting (IP, netmask, gateway) (set ON)	R/W	R/C	R/C
LW-9125	(16bit): HMI ethernet 1 gateway 0 (machine used only)	R/W	R/C	R/C
LW-9126	(16bit): HMI ethernet 1 gateway 1 (machine used only)	R/W	R/C	R/C
LW-9127	(16bit): HMI ethernet 1 gateway 2 (machine used only)	R/W	R/C	R/C
LW-9128	(16bit): HMI ethernet 1 gateway 3 (machine used only)	R/W	R/C	R/C
LW-9129	(16bit): HMI ethernet 1 IP 0 (machine used only)	R/W	R/C	R/C
LW-9130	(16bit): HMI ethernet 1 IP 1 (machine used only)	R/W	R/C	R/C
LW-9131	(16bit): HMI ethernet 1 IP 2 (machine used only)	R/W	R/C	R/C
LW-9132	(16bit): HMI ethernet 1 IP 3 (machine used only)	R/W	R/C	R/C
LW-9133	(16bit): ethernet port no.	R	R	R
LW-9135	(16bit): ethernet 1 media access control (MAC) address 0	R	R	R
LW-9136	(16bit): ethernet 1 media access control (MAC) address 1	R	R	R
LW-9137	(16bit): ethernet 1 media access control (MAC) address 2	R	R	R
LW-9138	(16bit): ethernet 1 media access control (MAC) address 3	R	R	R
LW-9139	(16bit): ethernet 1 media access control (MAC) address 4	R	R	R
LW-9140	(16bit): ethernet 1 media access control (MAC) address 5	R	R	R
LW-9141	(16bit): HMI station no.	R/W	R/C	R/C
LW-10750	(16bit): HMI ethernet 1 Mask 0 (machine used only)	R/W	R/C	R/C
LW-10751	(16bit): HMI ethernet 1 Mask 1 (machine used only)	R/W	R/C	R/C
LW-10752	(16bit): HMI ethernet 1 Mask 2 (machine used only)	R/W	R/C	R/C
LW-10753	(16bit): HMI ethernet 1 Mask 3 (machine used only)	R/W	R/C	R/C
LW-10786	(16bit): HMI ethernet 2 IP 0 (machine used only)	R/W	R/C	R/C
LW-10787	(16bit): HMI ethernet 2 IP 1 (machine used only)	R/W	R/C	R/C
LW-10788	(16bit): HMI ethernet 2 IP 2 (machine used only)	R/W	R/C	R/C
LW-10789	(16bit): HMI ethernet 2 IP 3 (machine used only)	R/W	R/C	R/C
LW-10790	(16bit): HMI ethernet 2 netmask 0 (machine used only)	R/W	R/C	R/C

LW-10791	(16bit): HMI ethernet 2 netmask 1 (machine used only)	R/W	R/C	R/C
LW-10792	(16bit): HMI ethernet 2 netmask 2 (machine used only)	R/W	R/C	R/C
LW-10793	(16bit): HMI ethernet 2 netmask 3 (machine used only)	R/W	R/C	R/C
LW-10794	(16bit): HMI ethernet 2 gateway 0 (machine used only)	R/W	R/C	R/C
LW-10795	(16bit): HMI ethernet 2 gateway 1 (machine used only)	R/W	R/C	R/C
LW-10796	(16bit): HMI ethernet 2 gateway 2 (machine used only)	R/W	R/C	R/C
LW-10797	(16bit): HMI ethernet 2 gateway 3 (machine used only)	R/W	R/C	R/C
LW-10798	(16bit): ethernet 2 media access control (MAC) address 0	R	R	R
LW-10799	(16bit): ethernet 2 media access control (MAC) address 1	R	R	R
LW-10800	(16bit): ethernet 2 media access control (MAC) address 2	R	R	R
LW-10801	(16bit): ethernet 2 media access control (MAC) address 3	R	R	R
LW-10802	(16bit): ethernet 2 media access control (MAC) address 4	R	R	R
LW-10803	(16bit): ethernet 2 media access control (MAC) address 5	R	R	R
LW-10804	(16bit): HMI ethernet 1 domain name system (DNS) server IP0	R	R	R
LW-10805	(16bit): HMI Ethernet 1 domain name system (DNS) server IP1	R	R	R
LW-10806	(16bit): HMI Ethernet 1 domain name system (DNS) server IP2	R	R	R
LW-10807	(16bit): HMI Ethernet 1 domain name system (DNS) server IP3	R	R	R
LW-10808	(16bit): HMI ethernet 2 domain name system (DNS) server IP0	R	R	R
LW-10809	(16bit): HMI ethernet 2 domain name system (DNS) server IP1	R	R	R
LW-10810	(16bit): HMI ethernet 2 domain name system (DNS) server IP2	R	R	R
LW-10811	(16bit): HMI ethernet 2 domain name system (DNS) server IP3	R	R	R
LW-10812	(16bit): obtain an IP address automatically (DHCP => 0: off, 1: on)	R/W	R/C	R/C

Note
Registers relevant to Ethernet 2 are only available for cMT-SVR model.

22.3.5. Project File Information

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-9100	(16bit): project name (16 words)	R	R	R
LW-9116	(32bit): project size in bytes	R	R	R
LW-9118	(32bit): project size in K bytes	R	R	R
LW-9120	(32bit): compiler version	R	R	R
LW-9122	(16bit): project compiled date (year)	R	R	R
LW-9123	(16bit): project compiled date (month)	R	R	R
LW-9124	(16bit): project compiled date (day)	R	R	R

22.3.6. Storage Space Management

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9035	HMI free space insufficiency alarm (when ON)	R	R	R
LB-9036	SD card free space insufficiency alarm (when ON)	R	R	R
LB-9037	USB disk free space insufficiency alarm (when ON)	R	R	R
LB-12048	USB disk status (exists when ON)	R	R	R
LB-12050	SD card status (exists when ON)	R	R	R
LW-9070	(16bit): free space insufficiency warning (Mega bytes)	R	R	R
LW-9071	(16bit): reserved free space size (Mega bytes)	R	R	R
LW-9072	(32bit): HMI current free space (K bytes)	R	R	R
LW-9074	(32bit): SD current free space (K bytes)	R	R	R
LW-9076	(32bit): USB disk current free space (K bytes)	R	R	R

Download
Click the icon to download the demo project that explains how to use LW-9072 ~ LW-9076 with Backup Object. Please confirm your internet connection.

22.3.7. Recipe and Extended Memory

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9028	reset all recipe data (set ON)	W	C	C
LB-9029	save all recipe data to machine (set ON)	W	C	C
LB-9460	EM0's storage device (SD card) does not exist (when ON)	R	R	R
LB-9461	EM1's storage device (SD card) does not exist (when ON)	R	R	R
LB-9462	EM2's storage device (SD card) does not exist (when ON)	R	R	R
LB-9463	EM3's storage device (SD card) does not exist (when ON)	R	R	R
LB-9464	EM4's storage device (SD card) does not exist (when ON)	R	R	R
LB-9465	EM5's storage device (SD card) does not exist (when ON)	R	R	R
LB-9466	EM6's storage device (SD card) does not exist (when ON)	R	R	R
LB-9467	EM7's storage device (SD card) does not exist (when ON)	R	R	R
LB-9468	EM8's storage device (SD card) does not exist (when ON)	R	R	R
LB-9469	EM9's storage device (SD card) does not exist (when ON)	R	R	R
LB-9470	EM0's storage device (USB disk) does not exist (when ON)	R	R	R
LB-9471	EM1's storage device (USB disk) does not exist (when ON)	R	R	R
LB-9472	EM2's storage device (USB disk) does not exist (when ON)	R	R	R
LB-9473	EM3's storage device (USB disk) does not exist (when ON)	R	R	R
LB-9474	EM4's storage device (USB disk) does not exist (when ON)	R	R	R
LB-9475	EM5's storage device (USB disk) does not exist (when ON)	R	R	R
LB-9476	EM6's storage device (USB disk) does not exist (when ON)	R	R	R
LB-9477	EM7's storage device (USB disk) does not exist (when ON)	R	R	R
LB-9478	EM8's storage device (USB disk) does not exist (when ON)	R	R	R
LB-9479	EM9's storage device (USB disk) does not exist (when ON)	R	R	R

22.3.8. Data Sampling

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9025	delete the earliest data sampling file on HMI memory (set ON)	W	C	C
LB-9026	delete all data sampling files on HMI memory (set ON)	W	C	C
LB-9027	refresh data sampling information on HMI memory (set ON)	W	C	C
LB-9034	save event/data sampling to HMI, USB disk, SD card (set ON)	W	C	C
LB-11949	delete the earliest data sampling file on SD card (set ON)	W	C	C
LB-11950	delete all data sampling files on SD card (set ON)	W	C	C
LB-11951	refresh data sampling information on SD card (set ON)	W	C	C
LB-11952	delete the earliest data sampling file on USB disk (set ON)	W	C	C
LB-11953	delete all data sampling files on USB disk (set ON)	W	C	C
LB-11954	refresh data sampling information on USB disk (set ON)	W	C	C
LW-9063	(16bit): no. of data sampling files on HMI memory	R	R	R
LW-9064	(32bit): size of data sampling files on HMI memory	R	R	R
LW-10489	(16bit): no. of data sampling files on SD card	R	R	R
LW-10490	(32bit): size of data sampling files on SD card	R	R	R
LW-10492	(16bit): no. of data sampling files on USB disk	R	R	R
LW-10493	(32bit) : size of data sampling files on USB disk	R	R	R

Note
1. The registers for deleting or updating data samplings do not work during simulation on PC.

22.3.9. Event Log

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9021	reset current event log (set ON)	W	C	C
LB-9022	delete the earliest event log file on HMI memory (set ON)	W	C	C
LB-9023	delete all event log files on HMI memory (set ON)	W	C	C
LB-9024	refresh event log information on HMI memory (set ON)	W	C	C
LB-9034	save event/data sampling to HMI, USB disk, SD card (set ON)	W	C	C
LB-9042	acknowledge all alarm events (set ON)	W	C	C
LB-9043	unacknowledged events exist (when ON)	R	R	R
LB-11940	delete the earliest event log file on SD card (set ON)	W	C	C
LB-11941	delete all event log files on SD card (set ON)	W	C	C
LB-11942	refresh event log information on SD card (set ON)	W	C	C
LB-11943	delete the earliest event log file on USB disk (set ON)	W	C	C
LB-11944	delete all event log files on USB disk (set ON)	W	C	C
LB-11945	refresh event log information on USB disk (set ON)	W	C	C
LW-9060	(16bit): no. of event log files on HMI memory	R	R	R
LW-9061	(32bit): size of event log files on HMI memory	R	R	R
LW-9450	(16bit): time tag of event log – second *Note 1	R/W	R/C	R/C
LW-9451	(16bit): time tag of event log – minute *Note 1	R/W	R/C	R/C
LW-9452	(16bit): time tag of event log – hour *Note 1	R/W	R/C	R/C
LW-9453	(16bit): time tag of event log – day *Note 1	R/W	R/C	R/C
LW-9454	(16bit): time tag of event log – month *Note 1	R/W	R/C	R/C
LW-9455	(16bit): time tag of event log – year *Note 1	R/W	R/C	R/C
LW-10480	(16bit): no. of event log files on SD card	R	R	R
LW-10481	(32bit): size of event log files on SD card	R	R	R
LW-10483	(16bit): no. of event log files on USB disk	R	R	R
LW-10484	(32bit): size of event log files on USB disk	R	R	R

Note
1. If LW-9450 ~ LW-9455 are used to get Event Log time, please enable in (system parameters) » (General).
2. The registers for deleting or updating event logs do not work during simulation on PC.

Download
Click the icon to download the demo project that explains how to use the system registers LW-9450 to LW-9455 to be the time tag of event log. Please confirm your internet connection.

22.3.10. Station Number Variables

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-10000	(16bit): var0 - station no variable (usage: var0#address)	R/W	R/C	R/C
LW-10001	(16bit): var1 - station no variable (usage: var1#address)	R/W	R/C	R/C
LW-10002	(16bit): var2 - station no variable (usage: var2#address)	R/W	R/C	R/C
LW-10003	(16bit): var3 - station no variable (usage: var3#address)	R/W	R/C	R/C
LW-10004	(16bit): var4 - station no variable (usage: var4#address)	R/W	R/C	R/C
LW-10005	(16bit): var5 - station no variable (usage: var5#address)	R/W	R/C	R/C
LW-10006	(16bit): var6 - station no variable (usage: var6#address)	R/W	R/C	R/C
LW-10007	(16bit): var7 - station no variable (usage: var7#address)	R/W	R/C	R/C
LW-10008	(16bit): var8 - station no variable (usage: var8#address)	R/W	R/C	R/C
LW-10009	(16bit): var9 - station no variable (usage: var9#address)	R/W	R/C	R/C
LW-10010	(16bit): var10 - station no variable (usage: var10#address)	R/W	R/C	R/C
LW-10011	(16bit): var11 - station no variable (usage: var11#address)	R/W	R/C	R/C
LW-10012	(16bit): var12 - station no variable (usage: var12#address)	R/W	R/C	R/C
LW-10013	(16bit): var13 - station no variable (usage: var13#address)	R/W	R/C	R/C
LW-10014	(16bit): var14 - station no variable (usage: var14#address)	R/W	R/C	R/C
LW-10015	(16bit): var15 - station no variable (usage: var15#address)	R/W	R/C	R/C

Download
Click the icon to download the demo project. Please confirm your internet connection.

22.3.11. Index Registers

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-9200	(16bit): address index 0	R/W	R/C	R/C
LW-9201	(16bit): address index 1	R/W	R/C	R/C
LW-9202	(16bit): address index 2	R/W	R/C	R/C
LW-9203	(16bit): address index 3	R/W	R/C	R/C
LW-9204	(16bit): address index 4	R/W	R/C	R/C
LW-9205	(16bit): address index 5	R/W	R/C	R/C
LW-9206	(16bit): address index 6	R/W	R/C	R/C
LW-9207	(16bit): address index 7	R/W	R/C	R/C
LW-9208	(16bit): address index 8	R/W	R/C	R/C
LW-9209	(16bit): address index 9	R/W	R/C	R/C
LW-9210	(16bit): address index 10	R/W	R/C	R/C
LW-9211	(16bit): address index 11	R/W	R/C	R/C
LW-9212	(16bit): address index 12	R/W	R/C	R/C
LW-9213	(16bit): address index 13	R/W	R/C	R/C
LW-9214	(16bit): address index 14	R/W	R/C	R/C
LW-9215	(16bit): address index 15	R/W	R/C	R/C
LW-9230	(32bit): address index 16	R/W	R/C	R/C
LW-9232	(32bit): address index 17	R/W	R/C	R/C
LW-9234	(32bit): address index 18	R/W	R/C	R/C
LW-9236	(32bit): address index 19	R/W	R/C	R/C
LW-9238	(32bit): address index 20	R/W	R/C	R/C
LW-9240	(32bit): address index 21	R/W	R/C	R/C
LW-9242	(32bit): address index 22	R/W	R/C	R/C
LW-9244	(32bit): address index 23	R/W	R/C	R/C
LW-9246	(32bit): address index 24	R/W	R/C	R/C
LW-9248	(32bit): address index 25	R/W	R/C	R/C
LW-9250	(32bit): address index 26	R/W	R/C	R/C
LW-9252	(32bit): address index 27	R/W	R/C	R/C
LW-9254	(32bit): address index 28	R/W	R/C	R/C
LW-9256	(32bit): address index 29	R/W	R/C	R/C
LW-9258	(32bit): address index 30	R/W	R/C	R/C
LW-9260	(32bit): address index 31	R/W	R/C	R/C

Download
Click the icon to download the demo project. Please confirm your internet connection.

22.3.12. Modbus Server Communication

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9055	Modbus server (COM 1) receives a request (when ON)	R	R	R
LB-9056	Modbus server (COM 2) receives a request (when ON)	R	R	R
LB-9057	Modbus server (COM 3) receives a request (when ON)	R	R	R
LB-9058	Modbus server (ethernet) receives a request (when ON)	R	R	R
LB-12052	Modbus server status (disabled when ON)	R/W	R/C	R/C
LW-9270	(16bit): request's function code - Modbus server (COM 1)	R	R	R
LW-9271	(16bit): request's starting address - Modbus server (COM 1)	R	R	R
LW-9272	(16bit): request's quantity of registers - Modbus server (COM 1)	R	R	R
LW-9275	(16bit): request's function code - Modbus server (COM 2)	R	R	R
LW-9276	(16bit): request's starting address - Modbus server (COM 2)	R	R	R
LW-9277	(16bit): request's quantity of registers - Modbus server (COM 2)	R	R	R
LW-9280	(16bit): request's function code - Modbus server (COM 3)	R	R	R

LW-9281	(16bit): request's starting address - Modbus server (COM 3)	R	R	R
LW-9282	(16bit): request's quantity of registers - Modbus server (COM 3)	R	R	R
LW-9285	(16bit): request's function code - Modbus server (ethernet)	R	R	R
LW-9286	(16bit): request's starting address - Modbus server (ethernet)	R	R	R
LW-9287	(16bit): request's quantity of registers - Modbus server (ethernet)	R	R	R
LW-9288	(16bit): last error code - Modbus server (ethernet)	R	R	R
LW-9541	(16bit): Modbus/ASCII server station no. (COM 1)	R/W	R/C	R/C
LW-9542	(16bit): Modbus/ASCII server station no. (COM 2)	R/W	R/C	R/C
LW-9543	(16bit): Modbus/ASCII server station no. (COM 3)	R/W	R/C	R/C
LW-9544	(16bit): Modbus/ASCII server station no. (ethernet)	R/W	R/C	R/C
LW-9570	(32bit): received data count (bytes) (COM 1 Modbus server)	R	R	R
LW-9572	(32bit): received data count (bytes) (COM 2 Modbus server)	R	R	R
LW-9574	(32bit): received data count (bytes) (COM 3 Modbus server)	R	R	R
LW-9576	(32bit): received data count (bytes) (Ethernet Modbus server)	R	R	R

22.3.13. Communication Parameter Settings

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9030	update COM 1 communication parameters (set ON)	R/W	R/C	R/C
LB-9031	update COM 2 communication parameters (set ON)	R/W	R/C	R/C
LB-9032	update COM 3 communication parameters (set ON)	R/W	R/C	R/C
LB-9065	disable/enable COM 1 broadcast station no.	R/W	R/C	R/C
LB-9066	disable/enable COM 2 broadcast station no.	R/W	R/C	R/C
LB-9067	disable/enable COM 3 broadcast station no.	R/W	R/C	R/C
LW-9550	(16bit): COM 1 mode(0:RS232,1:RS485 2W,2:RS485 4W)	R/W	R/C	R/C
LW-9551	(16bit): COM 1 baud rate (7:1200,8:2400,0:4800,1:9600,10:14400, 2:19200,11:28800,3:38400,4:57600,..)	R/W	R/C	R/C
LW-9552	(16bit): COM 1 databits (7: 7 bits, 8: 8 bits)	R/W	R/C	R/C
LW-9553	(16bit): COM 1 parity (0:none, 1:even, 2:odd, 3:mark, 4:space)	R/W	R/C	R/C
LW-9554	(16bit): COM 1 stop bits (1: 1 bit, 2: 2 bits)	R/W	R/C	R/C
LW-9555	(16bit): COM 2 mode(0:RS232,1:RS485 2W,2:RS485 4W)	R/W	R/C	R/C
LW-9556	(16bit): COM 2 baud rate (7:1200,8:2400,0:4800,1:9600,10:14400, 2:19200,11:28800,3:38400,4:57600,..)	R/W	R/C	R/C
LW-9557	(16bit): COM 2 databits (7: 7 bits, 8: 8 bits)	R/W	R/C	R/C
LW-9558	(16bit): COM 2 parity (0:none, 1:even, 2:odd, 3:mark, 4:space)	R/W	R/C	R/C
LW-9559	(16bit): COM 2 stop bits (1: 1 bit, 2: 2 bits)	R/W	R/C	R/C
LW-9560	(16bit): COM 3 mode(0:RS232,1:RS485 2W)	R/W	R/C	R/C
LW-9561	(16bit): COM 3 baud rate (7:1200,8:2400,0:4800,1:9600,10:14400, 2:19200,11:28800,3:38400,4:57600,..)	R/W	R/C	R/C
LW-9562	(16bit): COM 3 databits (7: 7 bits, 8: 8 bits)	R/W	R/C	R/C
LW-9563	(16bit): COM 3 parity (0:none, 1:even, 2:odd, 3:mark, 4:space)	R/W	R/C	R/C
LW-9564	(16bit): COM 3 stop bits (1: 1 bit, 2: 2 bits)	R/W	R/C	R/C
LW-9565	(16bit): COM 1 broadcast station no.	R/W	R/C	R/C
LW-9566	(16bit): COM 2 broadcast station no.	R/W	R/C	R/C
LW-9567	(16bit): COM 3 broadcast station no.	R/W	R/C	R/C
LW-10500	(16bit): PLC 1 timeout (unit: 100ms)	R/W	R/C	R/C
LW-10501	(16bit): PLC 1 turn around delay (unit: ms)	R/W	R/C	R/C
LW-10502	(16bit): PLC 1 send ACK delay (unit: ms)	R/W	R/C	R/C
LW-10503	(16bit): PLC 1 parameter 1	R/W	R/C	R/C
LW-10504	(16bit): PLC 1 parameter 2	R/W	R/C	R/C
LW-10505	(16bit): PLC 2 timeout (unit: 100ms)	R/W	R/C	R/C
LW-10506	(16bit): PLC 2 turn around delay (unit: ms)	R/W	R/C	R/C
LW-10507	(16bit): PLC 2 send ACK delay (unit: ms)	R/W	R/C	R/C
LW-10508	(16bit): PLC 2 parameter 1	R/W	R/C	R/C
LW-10509	(16bit): PLC 2 parameter 2	R/W	R/C	R/C
LW-10510	(16bit): PLC 3 timeout (unit: 100ms)	R/W	R/C	R/C
LW-10511	(16bit): PLC 3 turn around delay (unit: ms)	R/W	R/C	R/C
LW-10512	(16bit): PLC 3 send ACK delay (unit: ms)	R/W	R/C	R/C
LW-10513	(16bit): PLC 3 parameter 1	R/W	R/C	R/C
LW-10514	(16bit): PLC 3 parameter 2	R/W	R/C	R/C
LW-10515	(16bit): PLC 4 timeout (unit: 100ms)	R/W	R/C	R/C
LW-10516	(16bit): PLC 4 turn around delay (unit: ms)	R/W	R/C	R/C
LW-10517	(16bit): PLC 4 send ACK delay (unit: ms) (SIEMENS S7/400 Link type)	R/W	R/C	R/C

LW-10518	(16bit): PLC 4 parameter 1 (SIEMENS S7/400 rack)	R/W	R/C	R/C
LW-10519	(16bit): PLC 4 parameter 2 (SIEMENS S7/400 CPU slot)	R/W	R/C	R/C
LW-10520	(16bit): PLC 5 timeout (unit: 100ms)	R/W	R/C	R/C
LW-10521	(16bit): PLC 5 turn around delay (unit: ms)	R/W	R/C	R/C
LW-10522	(16bit): PLC 5 send ACK delay (unit: ms) (SIEMENS S7/400 Link type)	R/W	R/C	R/C
LW-10523	(16bit): PLC 5 parameter 1 (SIEMENS S7/400 rack)	R/W	R/C	R/C
LW-10524	(16bit): PLC 5 parameter 2 (SIEMENS S7/400 CPU slot)	R/W	R/C	R/C
LW-10525	(16bit): PLC 6 timeout (unit: 100ms)	R/W	R/C	R/C
LW-10526	(16bit): PLC 6 turn around delay (unit: ms)	R/W	R/C	R/C
LW-10527	(16bit): PLC 6 send ACK delay (unit: ms) (SIEMENS S7/400 Link type)	R/W	R/C	R/C
LW-10528	(16bit): PLC 6 parameter 1 (SIEMENS S7/400 rack)	R/W	R/C	R/C
LW-10529	(16bit): PLC 6 parameter 2 (SIEMENS S7/400 CPU slot)	R/W	R/C	R/C
LW-10530	(16bit): PLC 7 timeout (unit: 100ms)	R/W	R/C	R/C
LW-10531	(16bit): PLC 7 turn around delay (unit: ms)	R/W	R/C	R/C
LW-10532	(16bit): PLC 7 send ACK delay (unit: ms) (SIEMENS S7/400 Link type)	R/W	R/C	R/C
LW-10533	(16bit): PLC 7 parameter 1 (SIEMENS S7/400 rack)	R/W	R/C	R/C
LW-10534	(16bit): PLC 7 parameter 2 (SIEMENS S7/400 CPU slot)	R/W	R/C	R/C
LW-10535	(16bit): PLC 8 timeout (unit: 100ms)	R/W	R/C	R/C
LW-10536	(16bit): PLC 8 turn around delay (unit: ms)	R/W	R/C	R/C
LW-10537	(16bit): PLC 8 send ACK delay (unit: ms) (SIEMENS S7/400 Link type)	R/W	R/C	R/C
LW-10538	(16bit): PLC 8 parameter 1 (SIEMENS S7/400 rack)	R/W	R/C	R/C
LW-10539	(16bit): PLC 8 parameter 2 (SIEMENS S7/400 CPU slot)	R/W	R/C	R/C
LW-10655	(16bit): PLC 32 timeout (unit: 100ms)	R/W	R/C	R/C
LW-10656	(16bit): PLC 32 turn around delay (unit: ms)	R/W	R/C	R/C
LW-10657	(16bit): PLC 32 send ACK delay (unit: ms)	R/W	R/C	R/C
LW-10658	(16bit): PLC 32 parameter 1	R/W	R/C	R/C
LW-10659	(16bit): PLC 32 parameter 2	R/W	R/C	R/C

22.3.14. Communication Status and Control with PLC (COM)

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9150	auto. connection for PLC 1 (COM 1) (when ON)	R/W	R/C	R/C
LB-9151	auto. connection for PLC 2 (COM 2) (when ON)	R/W	R/C	R/C
LB-9152	auto. connection for PLC 3 (COM 3) (when ON)	R/W	R/C	R/C
LB-9200	PLC 1 status (SN0, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9201	PLC 1 status (SN1, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9202	PLC 1 status (SN2, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9203	PLC 1 status (SN3, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9204	PLC 1 status (SN4, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9205	PLC 1 status (SN5, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9206	PLC 1 status (SN6, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9207	PLC 1 status (SN7, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9455	PLC 1 status (SN255, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9500	PLC 2 status (SN0, COM 2), set on to retry connection.	R/W	R/C	R/C
LB-9501	PLC 2 status (SN1, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9502	PLC 2 status (SN2, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9503	PLC 2 status (SN3, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9504	PLC 2 status (SN4, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9505	PLC 2 status (SN5, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9506	PLC 2 status (SN6, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9507	PLC 2 status (SN7, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9755	PLC 2 status (SN255, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9800	PLC 3 status (SN0, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9801	PLC 3 status (SN1, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9802	PLC 3 status (SN2, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9803	PLC 3 status (SN3, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9804	PLC 3 status (SN4, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9805	PLC 3 status (SN5, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9806	PLC 3 status (SN6, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9807	PLC 3 status (SN7, COM 3), set on to retry connection	R/W	R/C	R/C
LB-10055	PLC 3 status (SN255, COM 3), set on to retry connection	R/W	R/C	R/C
LB-12030	COM 1 status (OFF: normal, ON: open failed) *Note 1	R	R	R
LB-12031	COM 2 status (OFF: normal, ON: open failed)	R	R	R

LB-12032	COM 3 status (OFF: normal, ON: open failed)	R	R	R
LB-12033	COM 4 status (OFF: normal, ON: open failed)	R	R	R
LB-12034	COM 5 status (OFF: normal, ON: open failed)	R	R	R
LB-12035	COM 6 status (OFF: normal, ON: open failed)	R	R	R
LB-12036	COM 7 status (OFF: normal, ON: open failed)	R	R	R
LB-12037	COM 8 status (OFF: normal, ON: open failed)	R	R	R
LB-12038	COM 9 status (OFF: normal, ON: open failed)	R	R	R
LW-9351	(16bit): pending command no. in PLC 1 (COM 1)	R	R	R
LW-9352	(16bit): pending command no. in PLC 2 (COM 2)	R	R	R
LW-9353	(16bit): pending command no. in PLC 3 (COM 3)	R	R	R

Note
1. The ON state of COM is for checking if COM is occupied by other program during simulation on PC.

22.3.15. Communication Status and Control with PLC (Ethernet)

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9153	auto. connection for PLC 4 (ethernet) (when ON)	R/W	R/C	R/C
LB-9154	auto. connection for PLC 5 (ethernet) (when ON)	R/W	R/C	R/C
LB-9155	auto. connection for PLC 6 (ethernet) (when ON)	R/W	R/C	R/C
LB-9156	auto. connection for PLC 7 (ethernet) (when ON)	R/W	R/C	R/C
LB-9157	auto. connection for PLC 8 (ethernet) (when ON)	R/W	R/C	R/C
LB-9158	auto. connection for PLC 9 (ethernet) (when ON)	R/W	R/C	R/C
LB-9189	auto. connection for PLC 40 (ethernet) (when ON)	R/W	R/C	R/C
LB-10070	forced to reconnect PLC 4 (ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10071	forced to reconnect PLC 5 (ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10072	forced to reconnect PLC 6 (ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10073	forced to reconnect PLC 7 (ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10074	forced to reconnect PLC 8 (ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10075	forced to reconnect PLC 9 (ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10099	forced to reconnect PLC 33 (ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10100	PLC 4 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-10400	PLC 5 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-10700	PLC 6 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11000	PLC 7 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11300	PLC 8 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11600	PLC 9 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11900	PLC 10 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11901	PLC 11 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11902	PLC 12 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11903	PLC 13 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11904	PLC 14 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11905	PLC 15 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11906	PLC 16 status (ethernet), set on to retry connection	R/W	R/C	R/C
LB-11939	PLC 49 status (ethernet), set on to retry connection	R/W	R/C	R/C
LW-9354	(16bit): pending command no. in PLC 4 (ethernet)	R	R	R
LW-9355	(16bit): pending command no. in PLC 5 (ethernet)	R	R	R
LW-9356	(16bit): pending command no. in PLC 6 (ethernet)	R	R	R
LW-9357	(16bit): pending command no. in PLC 7 (ethernet)	R	R	R
LW-9389	(16bit): pending command no. in PLC 39 (ethernet)	R	R	R
LW-9600	(16bit): PLC 4's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9601	(16bit): PLC 4's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9602	(16bit): PLC 4's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C

LW-9603	(16bit): PLC 4's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9604	(16bit): PLC 4's port no.	R/W	R/C	R/C
LW-9605	(16bit): PLC 5's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9606	(16bit): PLC 5's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9607	(16bit): PLC 5's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9608	(16bit): PLC 5's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9609	(16bit): PLC 5's port no.	R/W	R/C	R/C
LW-9610	(16bit): PLC 6's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9611	(16bit): PLC 6's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9612	(16bit): PLC 6's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9613	(16bit): PLC 6's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9614	(16bit): PLC 6's port no.	R/W	R/C	R/C
LW-9615	(16bit): PLC 7's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9616	(16bit): PLC 7's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9617	(16bit): PLC 7's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9618	(16bit): PLC 7's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9619	(16bit): PLC 7's port no.	R/W	R/C	R/C
LW-9620	(16bit): PLC 8's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9621	(16bit): PLC 8's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9622	(16bit): PLC 8's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9623	(16bit): PLC 8's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9624	(16bit): PLC 8's port no.	R/W	R/C	R/C
LW-9625	(16bit): PLC 9's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9626	(16bit): PLC 9's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9627	(16bit): PLC 9's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9628	(16bit): PLC 9's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9629	(16bit): PLC 9's port no.	R/W	R/C	R/C
LW-9765	(16bit): PLC 37's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9766	(16bit): PLC 37's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9767	(16bit): PLC 37's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9768	(16bit): PLC 37's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9769	(16bit): PLC 37's port no.	R/W	R/C	R/C

22.3.16. Communication Status and Control with PLC (USB)

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9190	auto. connection for PLC (USB) (when ON)	R/W	R/C	R/C
LB-9191	PLC status (USB), set on to retry connection	R/W	R/C	R/C
LW-9390	(16bit): pending command no. in PLC (USB)	R	R	R

22.3.17. Communication Status and Control with PLC (CAN Bus)

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-12080	auto. connection for PLC (CAN Bus) (when ON)	R/W	R/C	R/C
LB-12081	PLC status (CAN Bus) set on to retry connection	R/W	R/C	R/C
LB-12100	pause CAN Bus device 1 communication (when ON)	R/W	R/C	R/C
LB-12101	pause CAN Bus device 2 communication (when ON)	R/W	R/C	R/C
LB-12102	pause CAN Bus device 3 communication (when ON)	R/W	R/C	R/C
LB-12103	pause CAN Bus device 4 communication (when ON)	R/W	R/C	R/C
LB-12104	pause CAN Bus device 5 communication (when ON)	R/W	R/C	R/C
LB-12105	pause CAN Bus device 6 communication (when ON)	R/W	R/C	R/C
LB-12106	pause CAN Bus device 7 communication (when ON)	R/W	R/C	R/C
LB-12107	pause CAN Bus device 8 communication (when ON)	R/W	R/C	R/C
LB-12108	pause CAN Bus device 9 communication (when ON)	R/W	R/C	R/C
LB-12109	pause CAN Bus device 10 communication (when ON)	R/W	R/C	R/C
LB-12354	pause CAN Bus device 255 communication (when ON)	R/W	R/C	R/C
LW-9392	(16bit): pending command no. in PLC (CAN Bus)	R	R	R

22.3.18. Communication Status and Control with Remote HMI

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9068	auto. connection for remote HMI 1 (when ON)	R/W	R/C	R/C
LB-9069	auto. connection for remote HMI 2 (when ON)	R/W	R/C	R/C
LB-9070	auto. connection for remote HMI 3 (when ON)	R/W	R/C	R/C
LB-9071	auto. connection for remote HMI 4 (when ON)	R/W	R/C	R/C
LB-9072	auto. connection for remote HMI 5 (when ON)	R/W	R/C	R/C
LB-9073	auto. connection for remote HMI 6 (when ON)	R/W	R/C	R/C
LB-9074	auto. connection for remote HMI 7 (when ON)	R/W	R/C	R/C
LB-9075	auto. connection for remote HMI 8 (when ON)	R/W	R/C	R/C
LB-9099	auto. connection for remote HMI 32 (when ON)	R/W	R/C	R/C
LB-9100	remote HMI 1 status (set on to retry connection)	R/W	R/C	R/C
LB-9101	remote HMI 2 status (set on to retry connection)	R/W	R/C	R/C
LB-9102	remote HMI 3 status (set on to retry connection)	R/W	R/C	R/C
LB-9103	remote HMI 4 status (set on to retry connection)	R/W	R/C	R/C
LB-9104	remote HMI 5 status (set on to retry connection)	R/W	R/C	R/C
LB-9105	remote HMI 6 status (set on to retry connection)	R/W	R/C	R/C
LB-9106	remote HMI 7 status (set on to retry connection)	R/W	R/C	R/C
LB-9107	remote HMI 8 status (set on to retry connection)	R/W	R/C	R/C
LB-9148	remote HMI 49 status (set on to retry connection)	R/W	R/C	R/C
LB-9149	forced to reconnect remote HMI when IP changed on-line (set ON)	R/W	R/C	R/C
LW-9800	(16bit): remote HMI 1's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9801	(16bit): remote HMI 1's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9802	(16bit): remote HMI 1's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9803	(16bit): remote HMI 1's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9804	(16bit): remote HMI 1's port no.	R/W	R/C	R/C
LW-9805	(16bit): remote HMI 2's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9806	(16bit): remote HMI 2's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9807	(16bit): remote HMI 2's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9808	(16bit): remote HMI 2's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9809	(16bit): remote HMI 2's port no.	R/W	R/C	R/C
LW-9810	(16bit): remote HMI 3's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9811	(16bit): remote HMI 3's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9812	(16bit): remote HMI 3's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9813	(16bit): remote HMI 3's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9814	(16bit): remote HMI 3's port no.	R/W	R/C	R/C
LW-9815	(16bit): remote HMI 4's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9816	(16bit): remote HMI 4's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9817	(16bit): remote HMI 4's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9818	(16bit): remote HMI 4's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9819	(16bit): remote HMI 4's port no.	R/W	R/C	R/C
LW-9820	(16bit): remote HMI 5's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9821	(16bit): remote HMI 5's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9822	(16bit): remote HMI 5's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9823	(16bit): remote HMI 5's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9824	(16bit): remote HMI 5's port no.	R/W	R/C	R/C
LW-9825	(16bit): remote HMI 6's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9826	(16bit): remote HMI 6's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9827	(16bit): remote HMI 6's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9828	(16bit): remote HMI 6's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9829	(16bit): remote HMI 6's port no.	R/W	R/C	R/C
LW-9830	(16bit): remote HMI 7's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9831	(16bit): remote HMI 7's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9832	(16bit): remote HMI 7's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9833	(16bit): remote HMI 7's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9834	(16bit): remote HMI 7's port no.	R/W	R/C	R/C
LW-9835	(16bit): remote HMI 8's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9836	(16bit): remote HMI 8's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9837	(16bit): remote HMI 8's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9838	(16bit): remote HMI 8's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9839	(16bit): remote HMI 8's port no.	R/W	R/C	R/C
LW-9895	(16bit): remote HMI 20's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C

LW-9896	(16bit): remote HMI 20's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9897	(16bit): remote HMI 20's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9898	(16bit): remote HMI 20's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9899	(16bit): remote HMI 20's port no.	R/W	R/C	R/C
LW-9905	(16bit): remote HMI 21's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9906	(16bit): remote HMI 21's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9907	(16bit): remote HMI 21's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9908	(16bit): remote HMI 21's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9909	(16bit): remote HMI 21's port no.	R/W	R/C	R/C
LW-9910	(16bit): remote HMI 22's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9911	(16bit): remote HMI 22's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9912	(16bit): remote HMI 22's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9913	(16bit): remote HMI 22's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9914	(16bit): remote HMI 22's port no.	R/W	R/C	R/C
LW-9915	(16bit): remote HMI 23's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9916	(16bit): remote HMI 23's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9917	(16bit): remote HMI 23's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9918	(16bit): remote HMI 23's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9919	(16bit): remote HMI 23's port no.	R/W	R/C	R/C
LW-9920	(16bit): remote HMI 24's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9921	(16bit): remote HMI 24's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9922	(16bit): remote HMI 24's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9923	(16bit): remote HMI 24's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9924	(16bit): remote HMI 24's port no.	R/W	R/C	R/C
LW-9925	(16bit): remote HMI 25's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9926	(16bit): remote HMI 25's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9927	(16bit): remote HMI 25's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9928	(16bit): remote HMI 25's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9929	(16bit): remote HMI 25's port no.	R/W	R/C	R/C
LW-9930	(16bit): remote HMI 26's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9931	(16bit): remote HMI 26's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9932	(16bit): remote HMI 26's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9933	(16bit): remote HMI 26's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9934	(16bit): remote HMI 26's port no.	R/W	R/C	R/C
LW-9935	(16bit): remote HMI 27's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9936	(16bit): remote HMI 27's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9937	(16bit): remote HMI 27's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9938	(16bit): remote HMI 27's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9939	(16bit): remote HMI 27's port no.	R/W	R/C	R/C
LW-9940	(16bit): remote HMI 28's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9941	(16bit): remote HMI 28's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9942	(16bit): remote HMI 28's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9943	(16bit): remote HMI 28's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9944	(16bit): remote HMI 28's port no.	R/W	R/C	R/C
LW-9945	(16bit): remote HMI 29's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9946	(16bit): remote HMI 29's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9947	(16bit): remote HMI 29's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9948	(16bit): remote HMI 29's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9949	(16bit): remote HMI 29's port no.	R/W	R/C	R/C
LW-9950	(16bit): remote HMI 30's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9951	(16bit): remote HMI 30's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9952	(16bit): remote HMI 30's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9953	(16bit): remote HMI 30's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9954	(16bit): remote HMI 30's port no.	R/W	R/C	R/C
LW-9955	(16bit): remote HMI 31's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9956	(16bit): remote HMI 31's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9957	(16bit): remote HMI 31's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9958	(16bit): remote HMI 31's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9959	(16bit): remote HMI 31's port no.	R/W	R/C	R/C
LW-9960	(16bit): remote HMI 32's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9961	(16bit): remote HMI 32's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9962	(16bit): remote HMI 32's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9963	(16bit): remote HMI 32's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9964	(16bit): remote HMI 32's port no.	R/W	R/C	R/C
LW-9995	(16bit): remote HMI 39's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C

LW-9996	(16bit): remote HMI 39's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9997	(16bit): remote HMI 39's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9998	(16bit): remote HMI 39's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9999	(16bit): remote HMI 39's port no.	R/W	R/C	R/C

22.3.19. Communication Status and Control with Remote PLC

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-10050	(16bit): IP0 of the HMI connecting to remote PLC 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10051	(16bit): IP1 of the HMI connecting to remote PLC 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10052	(16bit): IP2 of the HMI connecting to remote PLC 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10053	(16bit): IP3 of the HMI connecting to remote PLC 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10054	(16bit): port no. of the HMI connecting to remote PLC 1	R/W	R/C	R/C
LW-10055	(16bit): IP0 of the HMI connecting to remote PLC 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10056	(16bit): IP1 of the HMI connecting to remote PLC 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10057	(16bit): IP2 of the HMI connecting to remote PLC 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10058	(16bit): IP3 of the HMI connecting to remote PLC 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10059	(16bit): port no. of the HMI connecting to remote PLC 2	R/W	R/C	R/C
LW-10060	(16bit): IP0 of the HMI connecting to remote PLC 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10061	(16bit): IP1 of the HMI connecting to remote PLC 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10062	(16bit): IP2 of the HMI connecting to remote PLC 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10063	(16bit): IP3 of the HMI connecting to remote PLC 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10064	(16bit): port no. of the HMI connecting to remote PLC 3	R/W	R/C	R/C
LW-10065	(16bit): IP0 of the HMI connecting to remote PLC 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10066	(16bit): IP1 of the HMI connecting to remote PLC 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10067	(16bit): IP2 of the HMI connecting to remote PLC 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10068	(16bit): IP3 of the HMI connecting to remote PLC 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10069	(16bit): port no. of the HMI connecting to remote PLC 4	R/W	R/C	R/C
LW-10205	(16bit): IP0 of the HMI connecting to remote PLC 32 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10206	(16bit): IP1 of the HMI connecting to remote PLC 32 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10207	(16bit): IP2 of the HMI connecting to remote PLC 32 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10208	(16bit): IP3 of the HMI connecting to remote PLC 32 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10209	(16bit): port no. of the HMI connecting to remote PLC 32	R/W	R/C	R/C
LW-10300	(16bit): remote PLC 1's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10301	(16bit): remote PLC 1's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10302	(16bit): remote PLC 1's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10303	(16bit): remote PLC 1's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10304	(16bit): remote PLC 1's port no.	R/W	R/C	R/C
LW-10305	(16bit): remote PLC 2's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10306	(16bit): remote PLC 2's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10307	(16bit): remote PLC 2's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10308	(16bit): remote PLC 2's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10309	(16bit): remote PLC 2's port no.	R/W	R/C	R/C
LW-10310	(16bit): remote PLC 3's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10311	(16bit): remote PLC 3's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10312	(16bit): remote PLC 3's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C

LW-10313	(16bit): remote PLC 3's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10314	(16bit): remote PLC 3's port no.	R/W	R/C	R/C
LW-10315	(16bit): remote PLC 4's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10316	(16bit): remote PLC 4's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10317	(16bit): remote PLC 4's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10318	(16bit): remote PLC 4's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10319	(16bit): remote PLC 4's port no.	R/W	R/C	R/C
LW-10455	(16bit): remote PLC 32's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10456	(16bit): remote PLC 32's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10457	(16bit): remote PLC 32's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10458	(16bit): remote PLC 32's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10459	(16bit): remote PLC 32's port no.	R/W	R/C	R/C

22.3.20. Local/Remote Operation Restrictions

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9044	disable remote control (when ON)	R/W	R/C	R/C
LB-9053	prohibit password remote-read operation (when ON)	R/W	R/C	R/C
LB-9054	prohibit password remote-write operation (when ON)	R/W	R/C	R/C
LB-9196	local HMI supports monitor function only (when ON)	R/W	R/C	R/C
LB-9197	support monitor function only for remote HMIs (when ON)	R/W	R/C	R/C
LB-9198	disable local HMI to trigger a MACRO (when ON)	R/W	R/C	R/C
LB-9199	disable remote HMI to trigger a MACRO (when ON)	R/W	R/C	R/C

22.3.21. Communication Error Codes

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-9400	(16bit): error code for PLC 1	R	R	R
LW-9401	(16bit): error code for PLC 2	R	R	R
LW-9402	(16bit): error code for PLC 3	R	R	R
LW-9403	(16bit): error code for PLC 4	R	R	R
LW-9404	(16bit): error code for PLC 5	R	R	R
LW-9405	(16bit): error code for PLC 6	R	R	R
LW-9406	(16bit): error code for PLC 7	R	R	R
LW-9407	(16bit): error code for PLC 8	R	R	R
LW-9449	(16bit): error code for PLC 50	R	R	R
LW-9490	(16bit): error code for USB PLC	R	R	R

22.3.22. Driver ID

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-9300	(16bit): driver ID of local PLC 1	R	R	R
LW-9301	(16bit): driver ID of local PLC 2	R	R	R
LW-9302	(16bit): driver ID of local PLC 3	R	R	R
LW-9303	(16bit): driver ID of local PLC 4	R	R	R
LW-9331	(16bit): driver ID of local PLC 32	R	R	R

22.3.23. DLT645 Controller

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-10700	(4 words): DLT_645 operator (COM 1)	R/W	R/C	R/C
LW-10704	(4 words): DLT_645 password (COM 1)	R/W	R/C	R/C
LW-10708	(6 words): DLT_645 address (COM 1)	R/W	R/C	R/C
LW-10715	(4 words): DLT_645 operator (COM 2)	R/W	R/C	R/C
LW-10719	(4 words): DLT_645 password (COM 2)	R/W	R/C	R/C
LW-10723	(6 words): DLT_645 address (COM 2)	R/W	R/C	R/C
LW-10730	(4 words): DLT_645 operator (COM 3)	R/W	R/C	R/C
LW-10734	(4 words): DLT_645 password (COM 3)	R/W	R/C	R/C
LW-10738	(6 words): DLT_645 address (COM 3)	R/W	R/C	R/C

22.3.24. (PLC No Response) Window Control

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9192	disable USB PLC's “PLC No Response” dialog (when ON)	R/W	R/C	R/C
LB-11960	disable PLC 1's “PLC No Response” dialog (when ON)	R/W	R/C	R/C
LB-11961	disable PLC 2's “PLC No Response” dialog (when ON)	R/W	R/C	R/C
LB-11962	disable PLC 3's “PLC No Response” dialog (when ON)	R/W	R/C	R/C
LB-11963	disable PLC 4's “PLC No Response” dialog (when ON)	R/W	R/C	R/C
LB-11964	disable PLC 5's “PLC No Response” dialog (when ON)	R/W	R/C	R/C
LB-11965	disable PLC 6's “PLC No Response” dialog (when ON)	R/W	R/C	R/C
LB-11966	disable PLC 7's “PLC No Response” dialog (when ON)	R/W	R/C	R/C
LB-11967	disable PLC 8's “PLC No Response” dialog (when ON)	R/W	R/C	R/C
LB-12023	disable PLC 64's “PLC No Response” dialog (when ON)	R/W	R/C	R/C
LB-12082	disable CAN Bus device's “PLC No Response” dialog (when ON)	R/W	R/C	R/C

22.3.25. (Fast Selection) Window Control

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9013	FS window control(hide(ON)/show(OFF))	R/W	R/C	R/C
LB-9014	FS button control(hide(ON)/show(OFF))	R/W	R/C	R/C
LB-9015	FS window/button control(hide(ON)/show(OFF))	R/W	R/C	R/C

22.3.26. EasyAccess

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9051	disconnect (set OFF)/connect (set ON) EasyAccess server	R/W	R/C	R/C
LB-9052	status of connecting to EasyAccess server	R	R	R

Download
Click the icon to download the demo project. Please confirm your internet connection.

22.3.27. EasyAccess 2.0

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-10820	(16bit): disable (set 0)/enable (set 1) (EasyAccess 2.0)	R/W	R/C	R/C
LW-10821	(5 words): session ID (EasyAccess 2.0)	R/W	R/C	R/C
LW-10826	(2 words): password (EasyAccess 2.0)	R/W	R/C	R/C
LW-10828	(16bit): execution status (EasyAccess 2.0)	R	R	R
LW-10829	(16bit): the last error code (EasyAccess 2.0)	R	R	R
LW-11170	(16bit): Proxy Disable/Enable (0:disable, 1:enable) (EasyAccess 2.0)	R/W	R/C	R/C
LW-11171	(16bit): Proxy Type (0:HTTP, 1:SOCKSv4, 2:SOCKSv5) (EasyAccess 2.0)	R/W	R/C	R/C
LW-11172	(16bit): Proxy Server IP0 (EasyAccess 2.0)	R/W	R/C	R/C
LW-11173	(16bit): Proxy Server IP1 (EasyAccess 2.0)	R/W	R/C	R/C
LW-11174	(16bit): Proxy Server IP2 (EasyAccess 2.0)	R/W	R/C	R/C
LW-11175	(16bit): Proxy Server IP3 (EasyAccess 2.0)	R/W	R/C	R/C
LW-11176	(16bit): Proxy Server Port (EasyAccess 2.0)	R/W	R/C	R/C
LW-11177	(16bit): Proxy authentication (0:disable, 1:enable) (EasyAccess 2.0)	R/W	R/C	R/C
LW-11178	(16 words): Proxy username (EasyAccess 2.0)	R/W	R/C	R/C
LW-11194	(16 words): Proxy password (EasyAccess 2.0)	R/W	R/C	R/C

22.3.28. Remote Print/Backup Server

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-10069	forced to reconnect remote printer/backup server when IP changed on-line (set ON)	R/W	R/C	R/C
LB-12040	remote printer/backup server disconnection alarm (when ON)	R	R	R
LW-9770	(16bit): remote printer/backup server IP0 (IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9771	(16bit): remote printer/backup server IP1 (IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9772	(16bit): remote printer/backup server IP2 (IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9773	(16bit): remote printer/backup server IP3 (IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9774	(6 words): remote printer/backup server user name *Note 1	R/W	R/C	R/C
LW-9780	(6 words): remote printer/backup server password *Note 1	R/W	R/C	R/C

Note
1. When change settings using LW-9774 and LW-9780, please reboot HMI for the new settings to take effect.

Download
Click the icon to download the demo project. Please confirm your internet connection.

22.3.29. Pass-Through Settings

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-9901	(16bit): pass-through source COM port (1~3: COM 1~COM 3)	R/W	R/C	R/C
LW-9902	(16bit): pass-through destination COM port (1~3: COM 1~COM 3)	R/W	R/C	R/C
LW-9903	(16bit): pass-through control (0: normal, 1: pause, 2: stop communications between HMI and PLC when executing pass-through)	R/W	R/C	R/C
LW-9904	(16bit): pass-through server port no. (2000~2100)	R/W	R/C	R/C
LW-10850	(16bit): disable/enable (0: disable, 1: normal, 2: IP limited) (siemens pass-through)	R/W	R/C	R/C
LW-10851	(16bit): destination COM port (siemens pass-through)	R/W	R/C	R/C
LW-10852	(16bit): destination PLC station no. (siemens pass-through)	R/W	R/C	R/C
LW-10853	(16bit): communication protocol (0: invalid, 1: PPI, 2: MPI) (siemens pass-through)	R/W	R/C	R/C
LW-10854	(16bit): IP0 of connecting client (IP address = IP0:IP1:IP2:IP3) (siemens pass-through)	R/W	R/C	R/C
LW-10855	(16bit): IP1 of connecting client (IP address = IP0:IP1:IP2:IP3) (siemens pass-through)	R/W	R/C	R/C

LW-10856	(16bit): IP2 of connecting client (IP address = IP0:IP1:IP2:IP3) (siemens pass-through)	R/W	R/C	R/C
LW-10857	(16bit): IP3 of connecting client (IP address = IP0:IP1:IP2:IP3) (siemens pass-through)	R/W	R/C	R/C
LW-10858	(16bit): IP0 of designated client (IP address = IP0:IP1:IP2:IP3) (siemens pass-through)	R/W	R/C	R/C
LW-10859	(16bit): IP1 of designated client (IP address = IP0:IP1:IP2:IP3) (siemens pass-through)	R/W	R/C	R/C
LW-10860	(16bit): IP2 of designated client (IP address = IP0:IP1:IP2:IP3) (siemens pass-through)	R/W	R/C	R/C
LW-10861	(16bit): IP3 of designated client (IP address = IP0:IP1:IP2:IP3) (siemens pass-through)	R/W	R/C	R/C
LW-10862	(16bit): connection status (0: ready, 1: client connecting) (siemens pass-through)	R	R	R
LW-10863	(16bit): execution status (0: normal, 1: error) (siemens pass-through)	R	R	R
LW-10864	(16bit): the last error (siemens pass-through)	R	R	R

☞ For more information about Siemens pass-through feature, see “29 Pass-through”.

Download
Click the icon to download the demo project. Please confirm your internet connection.

22.3.30. VNC Control

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-12088	enable VNC monitor mode (when ON) *Note 1	R/W	R/C	R/C
LB-12089	VNC pass word free (when ON) *Note 1	R/W	R/C	R/C
LB-12090	a VNC client connecting to HMI (when ON)(OS version 20120621 or later supports only)	R	R	R
LB-12091	disable auto-logout function when a VNC client connecting to HMI (when ON)(OS version 20120621 or later supports only)	R/W	R/C	R/C
LB-12092	enable VNC (set ON), disable VNC (set OFF)	R/W	R/C	R/C
LB-12093	VNC connection mode (OFF: single connection, ON: multi connection) (OS version 2013.05.09 or later support)*Note1	R/W	R/C	R/C
LW-9530	(8 words): VNC server password	R/W	R/C	R/C

Note
1. To change VNC mode, use LB-12092 to stop and then restart VNC to update the setting.

22.3.31. Project Key and HMI Key

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9046	project key is different from HMI key (when ON)	R	R	R
LW-9046	(32bit): HMI key *Note 1	R/W	R/C	R

Note
1. When change HMI Key using LW-9046, please reboot HMI for the new settings to take effect.

Download
Click the icon to download the demo project. Please confirm your internet connection.

22.3.32. USB Security Key

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-11160	(16bit): start time of USB security key - year	R	R	R
LW-11161	(16bit): start time of USB security key - month	R	R	R
LW-11162	(16bit): start time of USB security key - day	R	R	R
LW-11163	(16bit): start time of USB security key - hour	R	R	R
LW-11164	(16bit): start time of USB security key - minute	R	R	R
LW-11165	(16bit): expiration time of USB security key - year	R	R	R
LW-11166	(16bit): expiration time of USB security key - month	R	R	R
LW-11167	(16bit): expiration time of USB security key - day	R	R	R
LW-11168	(16bit): expiration time of USB security key - hour	R	R	R
LW-11169	(16bit): expiration time of USB security key - minute	R	R	R

22.3.33. User Name and Password

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9050	user logout	W	C	C
LB-9060	password error	R	R	R
LB-9061	update password (set ON)	W	C	C
LW-9082	(16bit): auto logout time (unit: minute, 0: disable the function)	R/W	R/C	R/C
LW-9219	(16bit): user no. (1~12)	R/W	R/C	R/C
LW-9220	(32bit): password	R/W	R/C	R/C
LW-9222	(16bit): classes can be operated for current user (bit 0:A, bit 1:B,bit 2:C, ...)	R	R	R
PLW-9222	(16bit): classes can be operated for current user (bit 0:A, bit 1:B,bit 2:C, ...)	R	R	R
LW-9500	(32bit): user 1's password	R/W	R/C	R/C
LW-9502	(32bit): user 2's password	R/W	R/C	R/C
LW-9504	(32bit): user 3's password	R/W	R/C	R/C
LW-9506	(32bit): user 4's password	R/W	R/C	R/C
LW-9508	(32bit): user 5's password	R/W	R/C	R/C
LW-9510	(32bit): user 6's password	R/W	R/C	R/C
LW-9512	(32bit): user 7's password	R/W	R/C	R/C
LW-9514	(32bit): user 8's password	R/W	R/C	R/C
LW-9516	(32bit): user 9's password	R/W	R/C	R/C
LW-9518	(32bit): user 10's password	R/W	R/C	R/C
LW-9520	(32bit): user 11's password	R/W	R/C	R/C
LW-9522	(32bit): user 12's password	R/W	R/C	R/C
LW-10754	(8 words): current user name *Note 1	R	R	R
PLW-10754	(8 words): current user name *Note 1	R	R	R

Note
1. Only for (Security) » (Enhanced security mode).

Download
Click the icon to download the demo project. Please confirm your internet connection.

22.3.34. Macro

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9059	disable macro TRACE function (when ON)*Note1	R/W	R/C	R/C
LW-10900	(16bit): macro 0 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R
LW-10901	(16bit): macro 1 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R
LW-10902	(16bit): macro 2 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R
LW-10903	(16bit): macro 3 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R
LW-10904	(16bit): macro 4 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R
LW-10905	(16bit): macro 5 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R
LW-10906	(16bit): macro 6 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R
LW-10907	(16bit): macro 7 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R
LW-10908	(16bit): macro 8 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R
LW-10909	(16bit): macro 9 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R
LW-11154	(16bit): macro 254 status (0:ready, 3:executing, 5:waiting response, 9:waiting sync, 17:delay, 32:abnormal end (exceed array size))	R	R	R

Note
1. LB-9059: Disable macro trace function.

Download
Click the icon to download the demo project. Please confirm your internet connection.

22.3.35. Input Object Function

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-9002	(32bit-float): input high limit	R	R	R
LW-9004	(32bit-float): input low limit	R	R	R
LW-9052	(32bit-float): the previous input value of the numeric input object	R	R	R
PLW-9052	(32bit-float): the previous input value of the numeric input object	R	R	R
LW-9150	(32 words): keyboard's input data (ASCII)	R	R	R
LW-9540	(16bit): reserved for caps lock	R/W	R/C	R/C

22.3.36. Miscellaneous

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9000~ LB-9009	initialized as ON	R/W	R/C	R/C
LB-9010	data download indicator	R	R	R
LB-9011	data upload indicator	R	R	R
LB-9012	data download/upload indicator	R	R	R
LB-9016	status is on when a client connects to this HMI	R	R	R
LB-9017	disable write-back in PLC control's (change window)	R/W	R/C	R/C
LB-9039	status of file backup activity (backup in process if ON)	R	R	R
LB-9045	memory-map communication fails (when ON)	R	R	R
LB-9049	enable (set ON)/disable (set OFF) watch dog *Note 1	R/W	R/C	R/C

LB-12053	Failed to send an (Event Log) e-Mail (when ON)	R	R	R
LB-12054	Failed to send an (Backup Object) e-Mail (when ON)	R	R	R
LW-9006	(16bit): connected client no.	R	R	R
LW-9024	(16bit): memory link system register	R/W	R/C	R/C
LW-9032	(8 words): folder name of backup history files to SD, USB memory *Note 3	R/W	R/C	R/C
LW-9050	(16bit): current base window ID	R	R	R
PLW-9050	(16bit): current base window ID	R	R	R
LW-9134	(16bit): language mode *Note 2	R/W	R/C	R/C
PLW-9134	(16bit): language mode *Note 2	R/W	R/C	R/C
LW-9216	(16bit): the result of importing email data	R	R	R
LW-9900	(16bit): HMI run mode (0: normal mode, 1-3: test mode (COM 1-COM 3)	R/W	R/C	R/C

Notes

1. When LB-9049 watch dog function is enabled, if there's a failure in the communication, HMI will reboot 10 seconds later.
2. To display texts on objects in multiple languages, except for using Label Library, the system reserved register (LW-9134: language mode) is needed. The value range in LW-9134 is 0 ~ 23 (PLW-9134 is 0~7). The values in LW-9134 relates to the languages downloaded to HMI. LW-9134 value and language correspondence vary according to the languages selected during project compilation and download.
   For example: If 5 languages are defined by user in Label Library as Language 1 (Traditional Chinese), Language 2 (Simplified Chinese), Language 3 (English), Language 4 (French), and Language 5 (Japanese). If only Language 1, 3, 5 are downloaded, the corresponding language of the value in LW-9134 will be 0 → Language 1 (Traditional Chinese), 1 → Language 3 (English), 2 → Language 5 (Japanese). The following demo project explains how to switch languages using Option List Object and LW-9134.
3. The default name of the backup data folder will be the HMI name.

Download

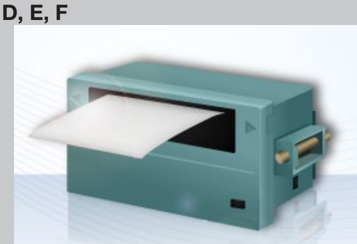
Click the icon to download the demo project. Please confirm your internet connection.

23. HMI SUPPORTED PRINTERS

This chapter describes the printers supported by HMI and the setup steps.

23.1. THE SUPPORTED PRINTER TYPES

HMI supported printer drivers include the following types:

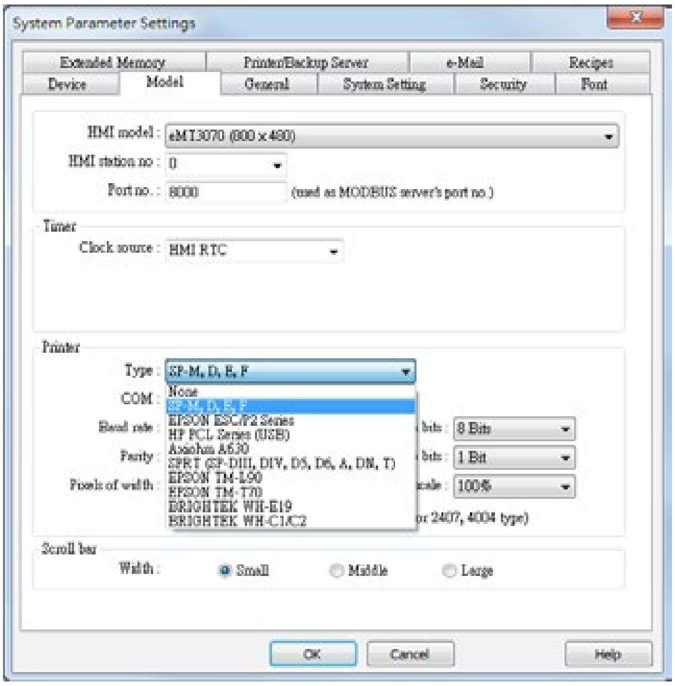
Printer type	Description
SP-M, D, E, F 	Serial printers, please configure communication parameters to match the printer. (Pixels of width) must be correctly set and can't exceed printer default setting: 100 pixels for 1610 series printers. 220 pixels for 2407, 4004 series printers. The driver uses EPSON ESC Protocol for Serial Micro Printer.
EPSON ESC/P2 Series 	Serial printers, please configure communication parameters to match the printer. The ESPON ESC/P2 printer protocol is used. Impact Printer: LQ-300, LQ-300+, LQ-300K+ (RS232), LQ-300+II (RS232) Inkjet Printer: Stylus Photo 750 Laser Printer: EPL-5800
HP PCL Series (USB) 	HP compatible USB printers that support HP PCL5 level 3 protocol. PCL 5 was released on HP LaserJet III in March 1990, added Intellifont font scaling (developed by Compugraphic, now part of Agfa), outline fonts and HP-GL/2 (vector) graphics.
Axiohm A630 	Micro printer from France connects via serial port; please configure communication parameters to match the printer.
SPRT 	Serial printers, please configure communication parameters to match the printer. (Pixels of width) must be correctly set and can't exceed printer default setting "100".
EPSON TM-L90 	Serial printers, please configure communication parameters to match the printer. (Pixels of width) must be correctly set and can't exceed printer default setting "576".

EPSON TM-T70 	Serial printers, please configure communication parameters to match the printer. (Pixels of width) must be correctly set and can't exceed printer default setting "576". The paper cutting mode can be selected: (No cut) / (Partial cut).
BRIGHTEK WH-A19 	Supported models: A92R10-00E72A 72 in model number represents hexadecimal printer, and A represents wide voltage 5~9 V. This is the same as the A6 16 impact printer.
BRIGHTEK WH-E19 	Serial printers, please configure the same communication parameters as the printer.
BRIGHTEK WH-E22 	Supported models: E22R10-00E725: same as A7 16 impact printer. A7 represents A72R90-31E72A. E221R90-00E11740GA: serial printer, connects through RS485 port, please use a RS232-to-RS485 converter.
BRIGHTEK WH-C1/C2 	Serial printers, please configure communication parameters to match the printer. The paper cutting mode can be selected: (No cut) / (Half cut) / Full cut).
Remote Printer Server 	Use EasyPrinter to start printing by the printers connected with PC via Ethernet. This works under MS Windows so most printers on the market are supported.

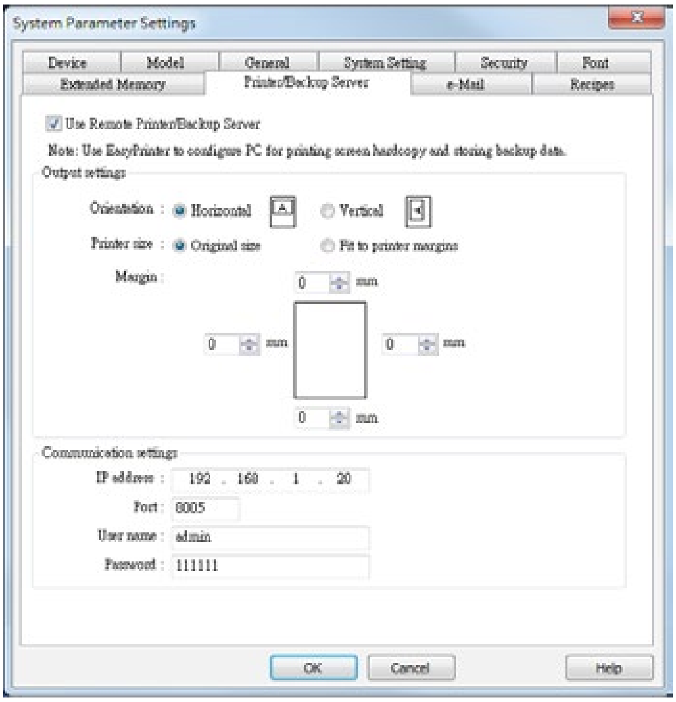
23.2. STEPS TO ADD A NEW PRINTER AND START PRINTING

1. Add printer type.

- In (System Parameter Settings) » (Model) select the printer type and set the relevant parameters.

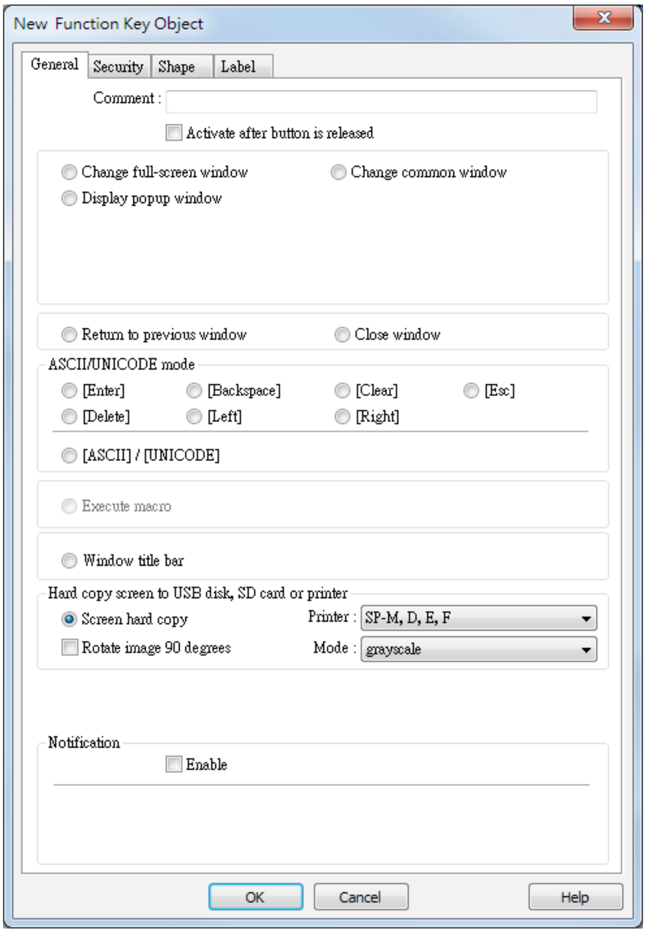


- To connect Remote Printer Server, set the parameters in (System Parameter Settings) » (Printer/Backup Server)

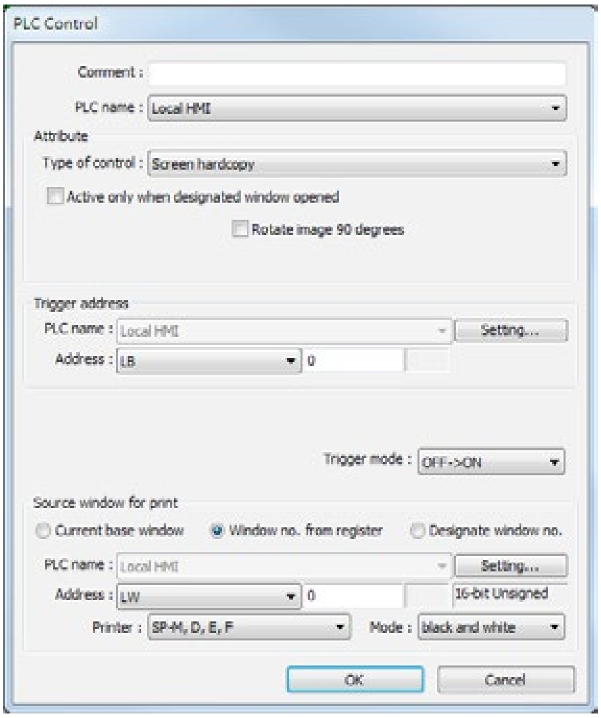


2. Start printing.

▪ Start printing with Function Key



▪ Or, use PLC Control (Screen hardcopy) to start printing with a designated bit address



24. RECIPE EDITOR

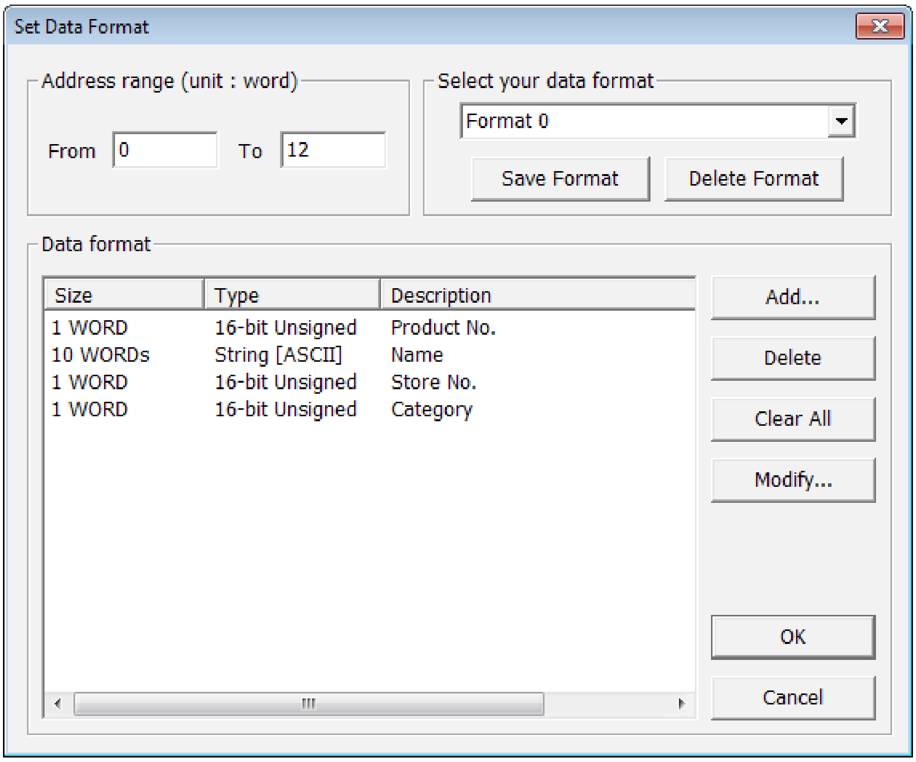
This chapter explains how to use Recipe Editor.

24.1. OVERVIEW

Recipe Editor is used to create, view, and edit recipe data.
EasyBuilder Pro also provides another tool for editing recipe: recipe records. To use this tool, first define a recipe in EasyBuilder Pro (System Parameter Settings) » (Recipe) tab, and then use (Recipe View Object) to display the content. The following introduces the usage of these two editing tools.

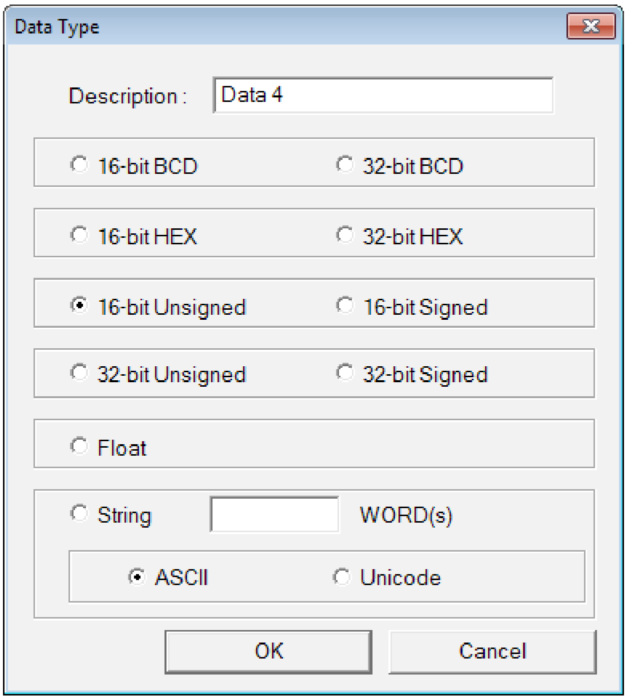
24.2. RECIPE / EXTENDED MEMORY EDITOR SETTING

1. Open Utility Manager and click (Recipe/Extended Memory Editor).
2. To add new .rcp or .emi files, click (File) » (New).
3. Set address range and select data format.

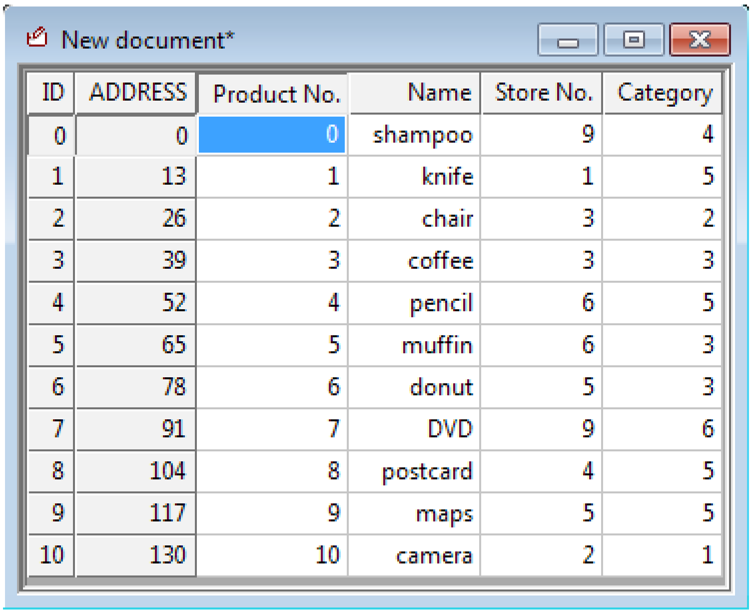


Setting	Description
Address range	Fill in address range, the unit of which is word.
Select your data format	Save the specified data format for loading next time. The saved file name is "dataEX.fmt" under EasyBuilder Pro's installation directory.
Data format	Edit new data format in this field.

4. Click (Add) to enter a description of the data type, and select data format. When selecting (String), please enter the length (words) and select (ASCII) or (Unicode).



5. After setup, click (OK) to start editing recipe data.


New document*

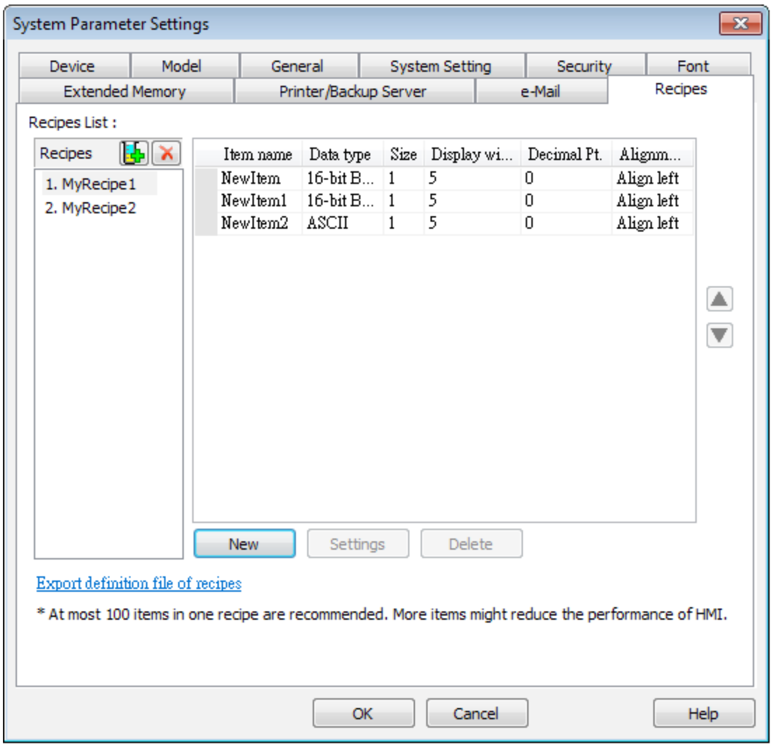
ID	ADDRESS	Product No.	Name	Store No.	Category
0	0	0	shampoo	9	4
1	13	1	knife	1	5
2	26	2	chair	3	2
3	39	3	coffee	3	3
4	52	4	pencil	6	5
5	65	5	muffin	6	3
6	78	6	donut	5	3
7	91	7	DVD	9	6
8	104	8	postcard	4	5
9	117	9	maps	5	5
10	130	10	camera	2	1

In this example, the total length of data format is 13 words. Each 13 words will be one set of recipe data.
The first set: “product no.” = address 0, “Name” = address 1 ~ 10, “Store No.”= address 11 , “Category” = address 12;
The second set: “product no.” = address 13, “Name” = address 14 ~ 23, “Store No.”= address 24, “Category” = address 25;…and so on.

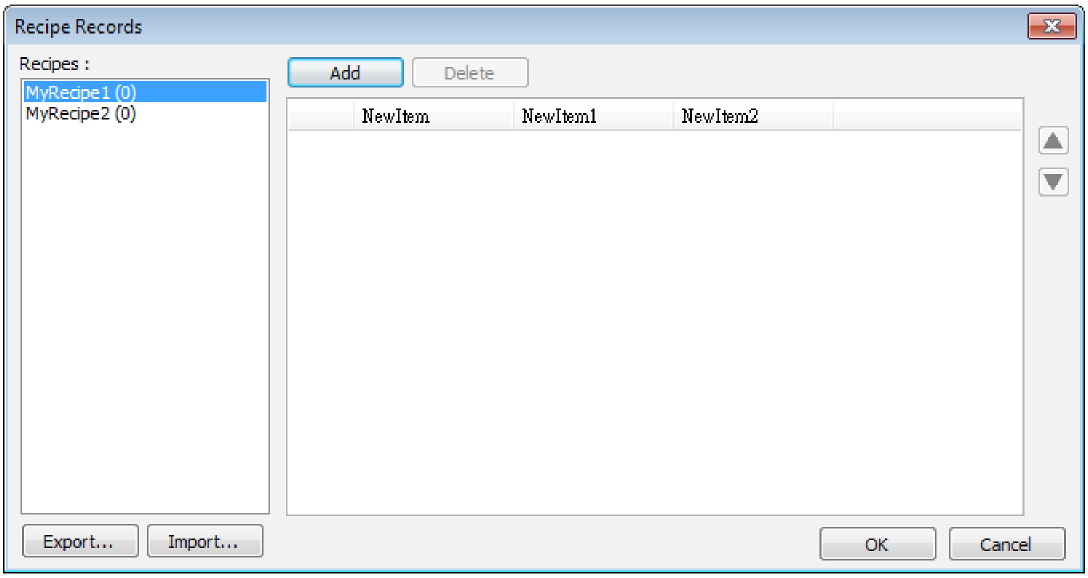
	Note After editing recipe data, it can be saved as .rcp, .emi, or .csv files. The .rcp files can be downloaded to HMI using Utility Manager or external devices (USB drive or SD card). The .emi files can be saved directly to the external device which is inserted to HMI as extended memory (EM).

24.3. RECIPE RECORDS SETTING

1. Before using Recipe Records, first enable it in EasyBuilder Pro (System Parameter Settings) » (Recipes). Please see “5 System Parameter Settings” for more detail.

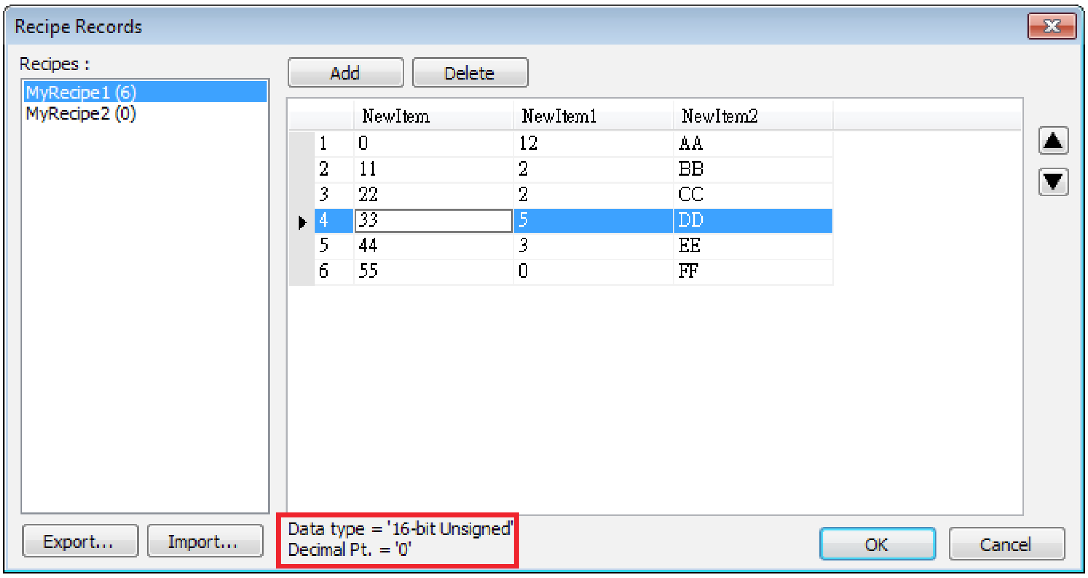


2. When finished, Recipe Records can be opened in main menu » (Library) » (Recipe Records). In the example shown below, there are Recipe1 and Recipe2. Three items are shown on the right hand side. The names of recipe come from System Parameter Settings.



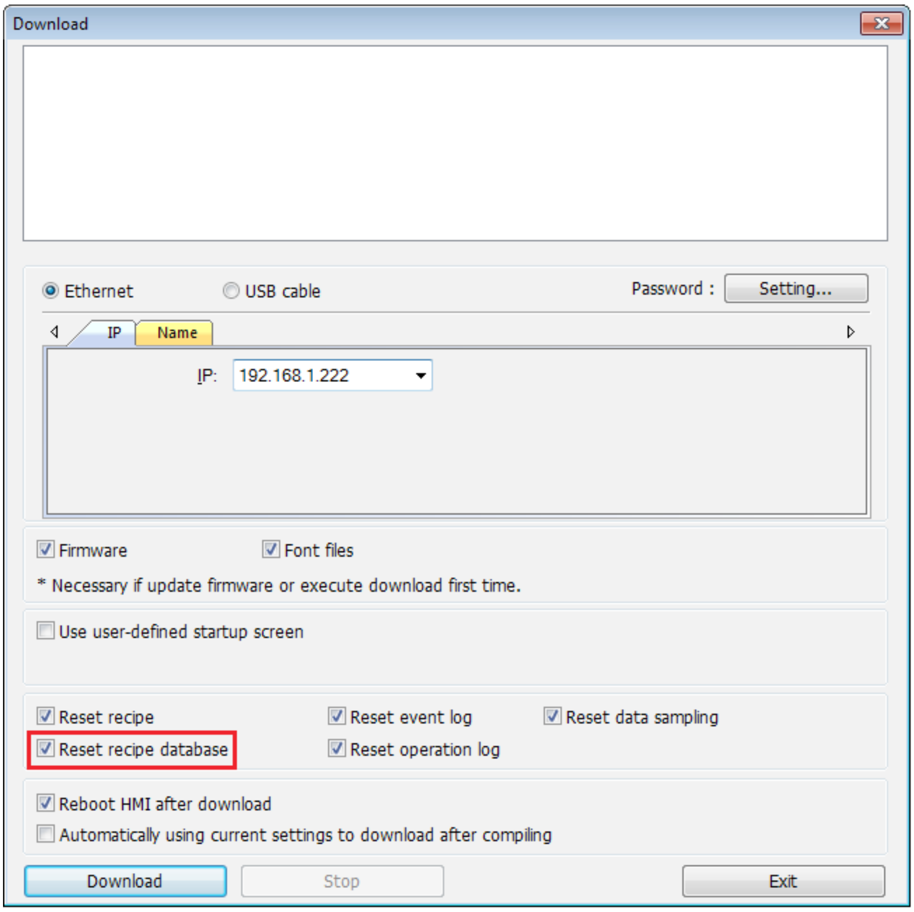
Setting	Description
Recipes	The recipes created in System Parameter Settings. The number enclosed in brackets shows the total number of records in the corresponding recipe.
Add	Inserts records into the recipe according to the item format.
Delete	Deletes the edited content.
Up / Down Arrows	Moves the selected record upward / downward.

3. To define recipes according to the specified format, click (Add) button above the record list to insert a new record and start editing each item. When click on the item, the item format will be shown under the record list. This helps users to fill in each item with legal value. Click (OK) to confirm and save the records.



Note

- Each recipe can hold a maximum of 10,000 records.
- The recipe records will be stored in the .exob file after compilation and will be downloaded to HMI. These recipes cannot be shared with other project files. If users need to modify the recipe contents and download them to the HMI, make sure that (Reset recipe database) check box is selected during download. If not, the recipe database in the HMI will not be updated.



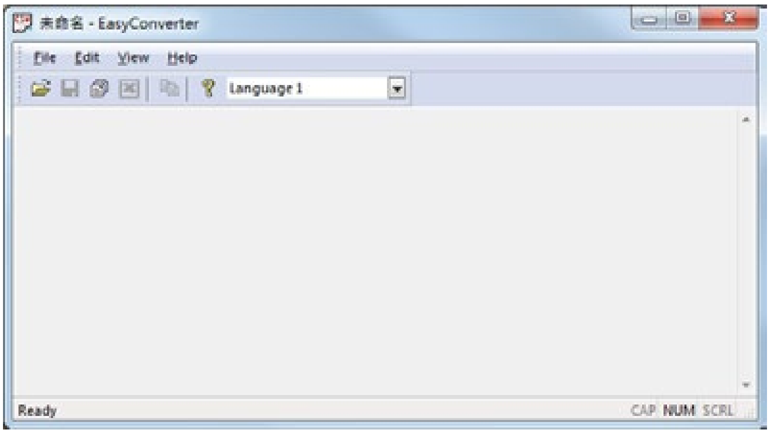
25. EASYCONVERTER

This Chapter explains how to use EasyConverter.

25.1. OVERVIEW

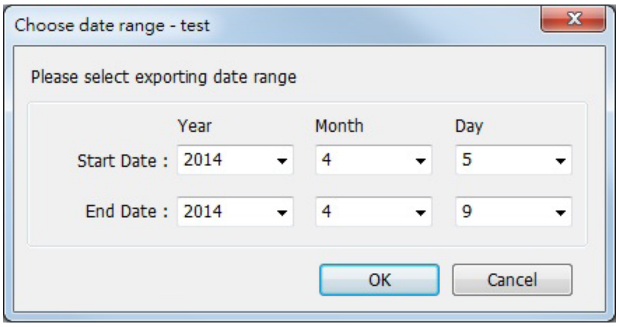
EasyConverter reads the Data Sampling file, Event Log file, and Operation Log file in HMI and convert the files to Excel format.

- From Utility Manager click (EasyConverter)
- From EasyBuilder Pro menu select (Tool) » (Data/Event Log Converter)

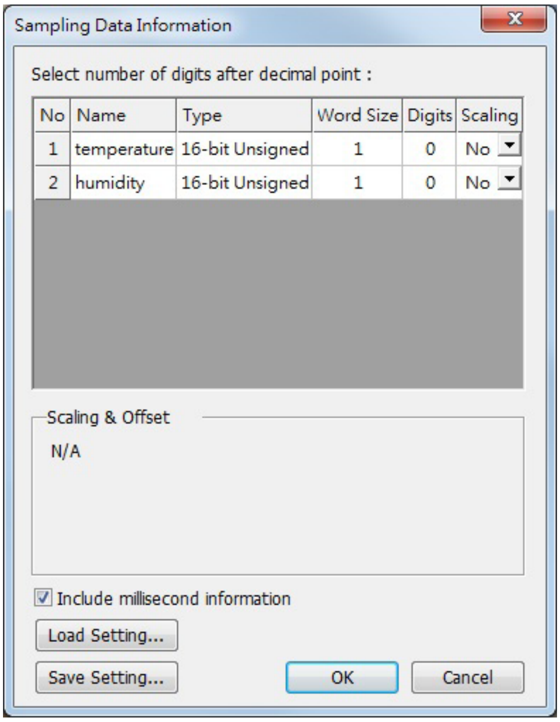


25.2. CONVERTING DATA SAMPLING FILE TO EXCEL FILE

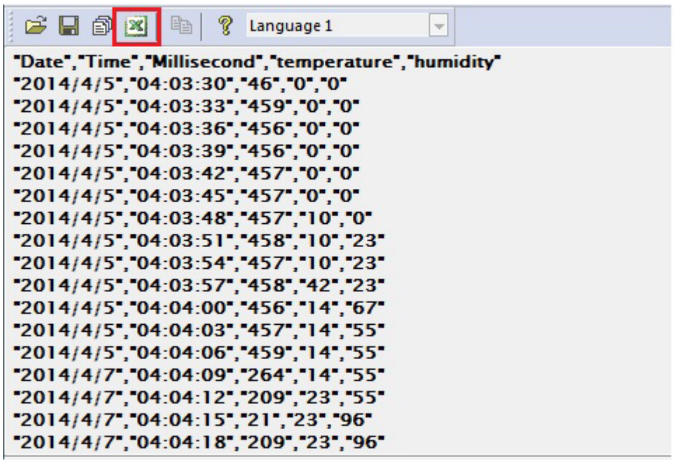
1. If the Data Sampling file format is .db, and the file includes data of more than one day, the data to be viewed can be specified by selecting a date range. (If the file format is .dtl, please skip this step).



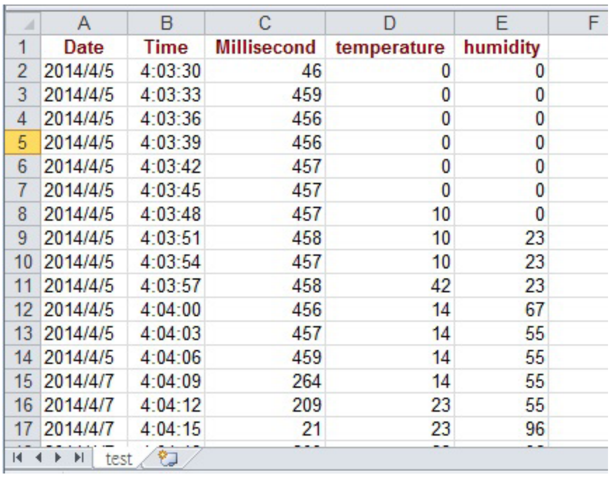
2. The following is the setting dialog box, please set based on actual needs.



3. Click (OK), the Data Sampling layout is shown in the following figure. Click (Export to Excel). The file will be converted to Excel format.



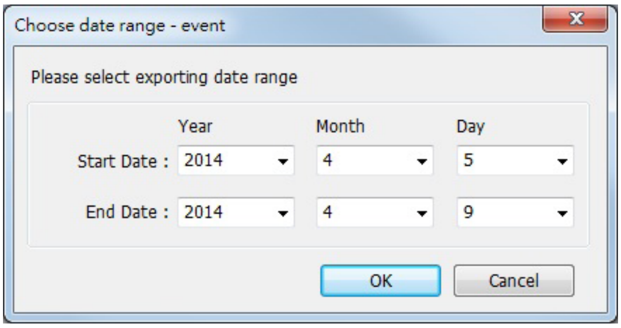
4. The Excel layout is shown in the following figure.



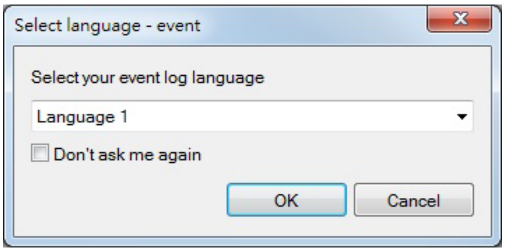
	A	B	C	D	E	F
1	Date	Time	Millisecond	temperature	humidity	
2	2014/4/5	4:03:30	46	0	0	
3	2014/4/5	4:03:33	459	0	0	
4	2014/4/5	4:03:36	456	0	0	
5	2014/4/5	4:03:39	456	0	0	
6	2014/4/5	4:03:42	457	0	0	
7	2014/4/5	4:03:45	457	0	0	
8	2014/4/5	4:03:48	457	10	0	
9	2014/4/5	4:03:51	458	10	23	
10	2014/4/5	4:03:54	457	10	23	
11	2014/4/5	4:03:57	458	42	23	
12	2014/4/5	4:04:00	456	14	67	
13	2014/4/5	4:04:03	457	14	55	
14	2014/4/5	4:04:06	459	14	55	
15	2014/4/7	4:04:09	264	14	55	
16	2014/4/7	4:04:12	209	23	55	
17	2014/4/7	4:04:15	21	23	96	

25.3. CONVERTING EVENT LOG FILE TO EXCEL FILE

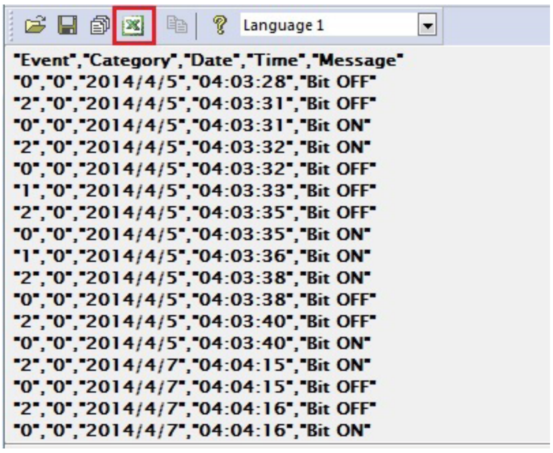
1. If the Event Log file format is .db, and the file includes data of more than one day, the data to be viewed can be specified by selecting a date range (If the file format is .evt, please skip this step).



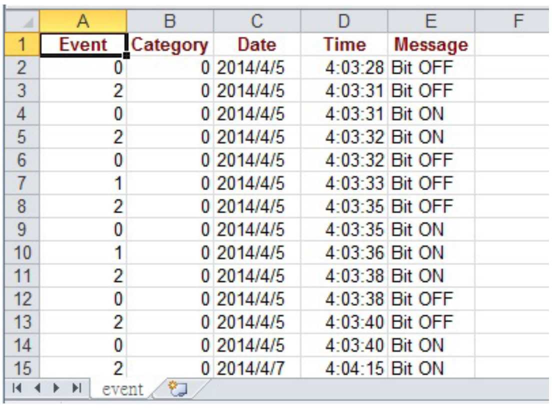
2. If the .db file of Event Log contains multiple languages, the language to be viewed can be specified (If the file format is .evt, please skip this step).



3. Click (OK), the Event Log layout is shown in the following figure. Click (Export to Excel). The file will be converted to Excel format.



4. The Excel layout is shown in the following figure.



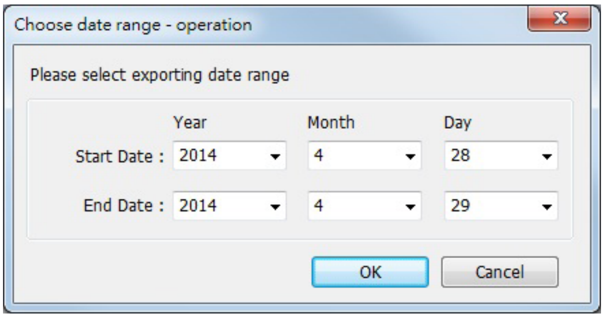
	A	B	C	D	E	F
1	Event	Category	Date	Time	Message	
2	0	0	2014/4/5	4:03:28	Bit OFF	
3	2	0	2014/4/5	4:03:31	Bit OFF	
4	0	0	2014/4/5	4:03:31	Bit ON	
5	2	0	2014/4/5	4:03:32	Bit ON	
6	0	0	2014/4/5	4:03:32	Bit OFF	
7	1	0	2014/4/5	4:03:33	Bit OFF	
8	2	0	2014/4/5	4:03:35	Bit OFF	
9	0	0	2014/4/5	4:03:35	Bit ON	
10	1	0	2014/4/5	4:03:36	Bit ON	
11	2	0	2014/4/5	4:03:38	Bit ON	
12	0	0	2014/4/5	4:03:38	Bit OFF	
13	2	0	2014/4/5	4:03:40	Bit OFF	
14	0	0	2014/4/5	4:03:40	Bit ON	
15	2	0	2014/4/7	4:04:15	Bit ON	

Note

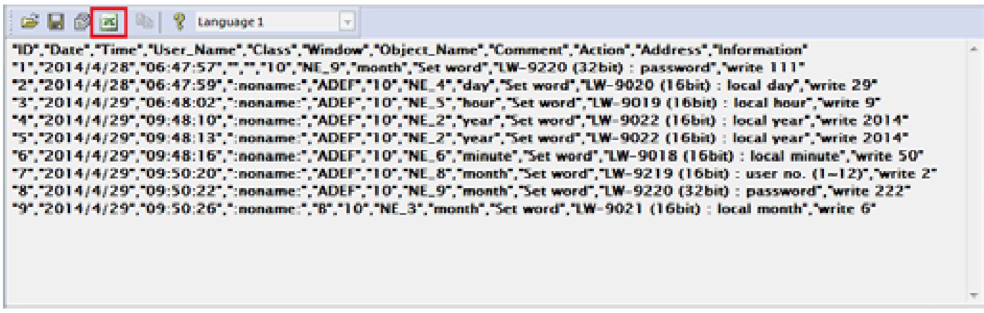
The “Event” column can be found. 0-> Event triggered; 1-> Event acknowledged; 2-> Event returns to normal.

25.4. CONVERTING OPERATION LOG FILE TO EXCEL FILE

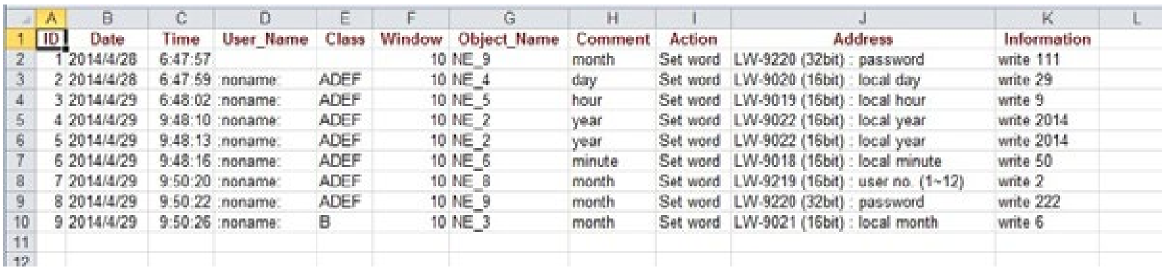
1. If the Operation Log file includes data of more than one day, the data to be viewed can be specified by selecting a date range.



2. Click (OK), the Operation Log layout is shown in the following figure. Click (Export to Excel). The file will be converted to Excel format.



3. The Excel layout is shown in the following figure.

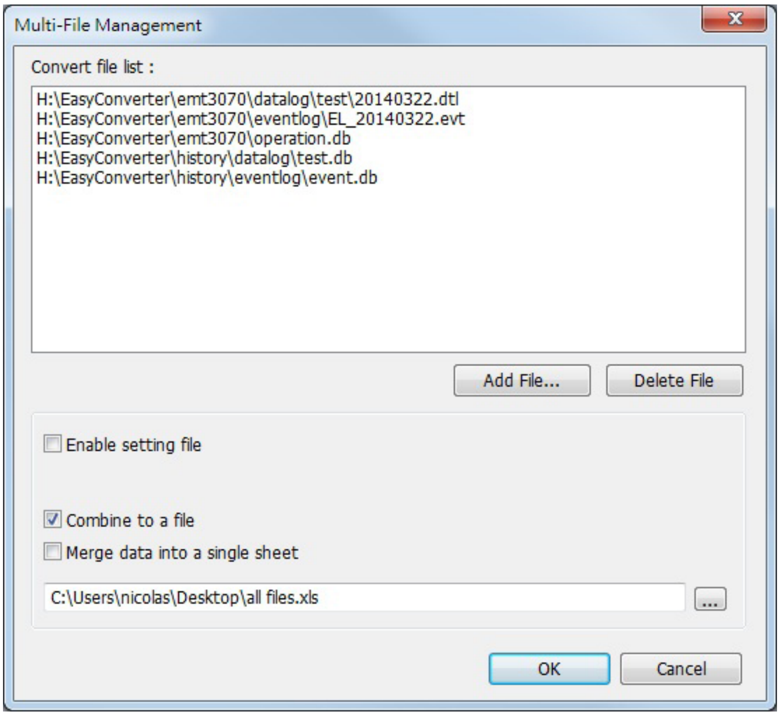


ID	Date	Time	User_Name	Class	Window	Object_Name	Comment	Action	Address	Information
1	2014/4/28	6:47:57			10	NE_9	month	Set word	LW-9220 (32bit) : password	write 111
2	2014/4/28	6:47:59	:noname:	ADEF	10	NE_4	day	Set word	LW-9020 (16bit) : local day	write 29
3	2014/4/29	6:48:02	:noname:	ADEF	10	NE_5	hour	Set word	LW-9019 (16bit) : local hour	write 9
4	2014/4/29	9:48:10	:noname:	ADEF	10	NE_2	year	Set word	LW-9022 (16bit) : local year	write 2014
5	2014/4/29	9:48:13	:noname:	ADEF	10	NE_2	year	Set word	LW-9022 (16bit) : local year	write 2014
6	2014/4/29	9:48:16	:noname:	ADEF	10	NE_6	minute	Set word	LW-9018 (16bit) : local minute	write 50
7	2014/4/29	9:50:20	:noname:	ADEF	10	NE_8	month	Set word	LW-9219 (16bit) : user no. (1~12)	write 2
8	2014/4/29	9:50:22	:noname:	ADEF	10	NE_9	month	Set word	LW-9220 (32bit) : password	write 222
9	2014/4/29	9:50:26	:noname:	B	10	NE_3	month	Set word	LW-9021 (16bit) : local month	write 6

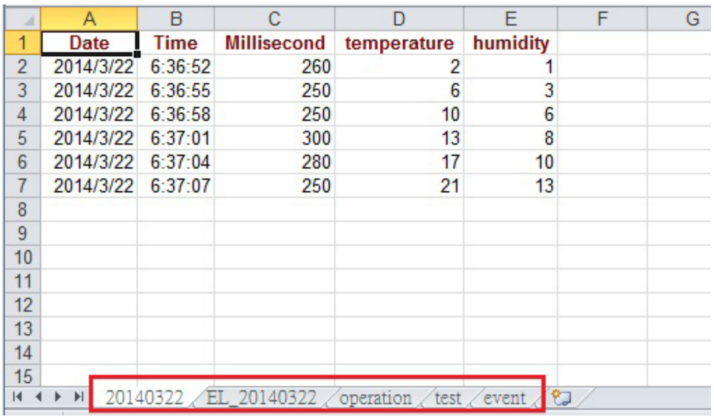
25.5. CONVERTING MULTIPLE FILES

1. Click (Multi-File) to open the following dialog box.

2. Click (Add File) to add the files to be converted. If click (OK) without selecting (Combine to a file), the files will be exported to separate Excel files.

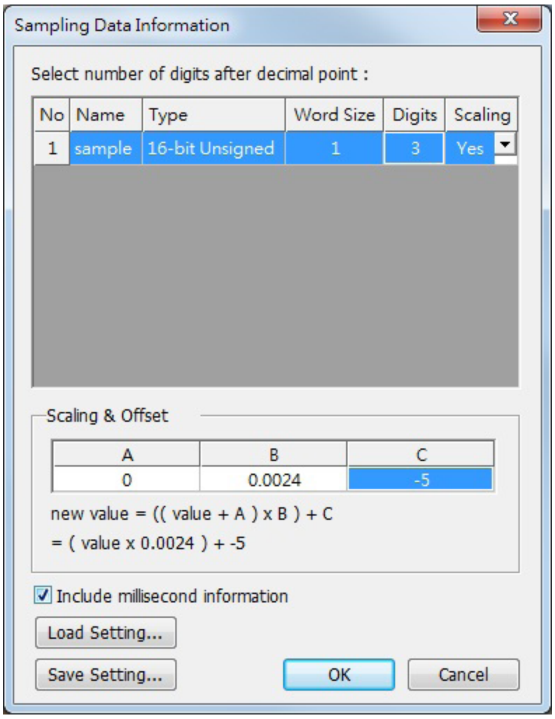


3. If (Combine to a file) is selected, the files will be separated into different sheets of one Excel file as shown in the following figure.

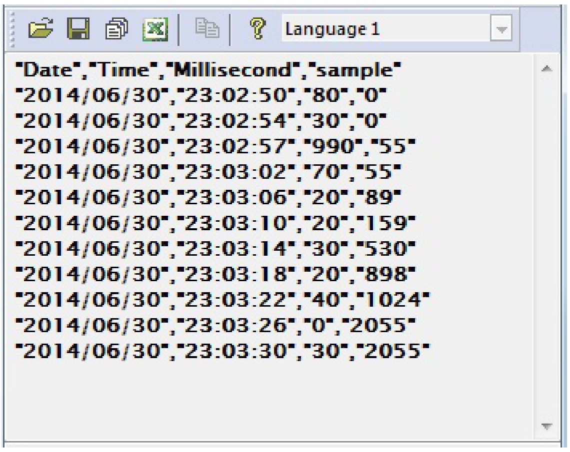


25.6. SCALING FUNCTION

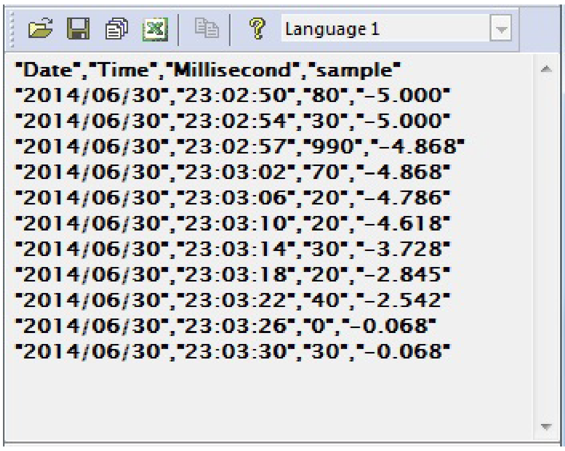
The equation of scaling new value = ((value + A) x B) + C, and users can set the values of A, B, and C.
A -> lower limit of the value ; B -> ((scaled max) - (scaled min) / (upper limit) - (lower limit)) ; C -> scaled min.
For example, here is a voltage data with a format of 16-bit unsigned (range: 0 ~ 4096).
To convert the data to volt, range form -5V to +5V, the new value = ((value + 0) x 0.0024) + (-5).



- Before scaling:



- After scaling:



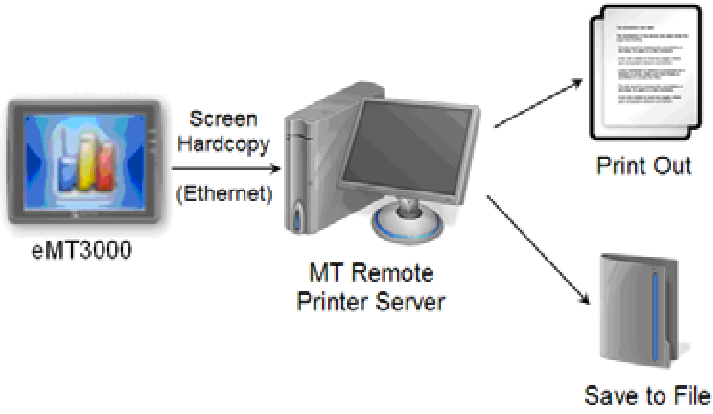
The settings described earlier can be saved and loaded next time. The extension name of the setting file is *.lgs.

26. EASYPRINTER

This chapter explains the setup steps of EasyPrinter.

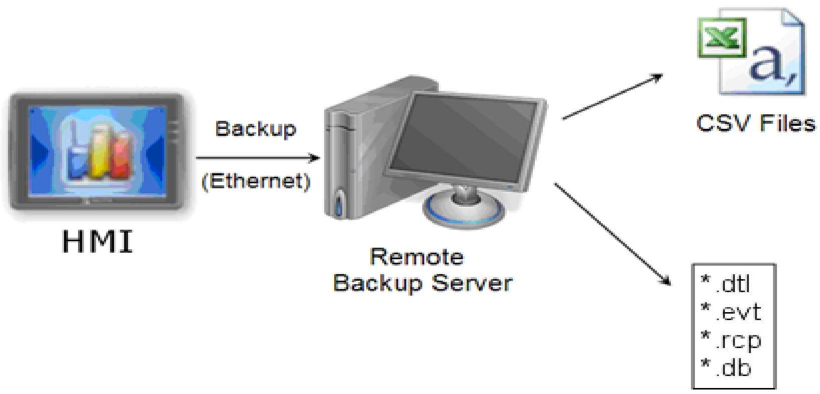
26.1. OVERVIEW

EasyPrinter is a Win32 application and can only run on MS Windows 2000 / XP / Vista / 7 / 8. It enables HMI to output screen hardcopies to a remote PC via Ethernet. The following explains how to use EasyPrinter.

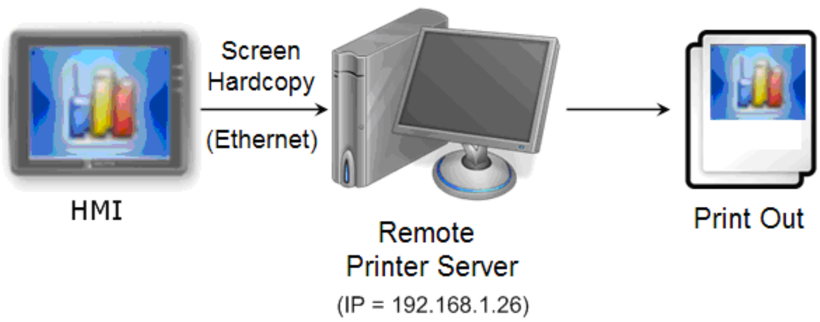


Here are some advantages of using EasyPrinter:
- EasyPrinter provides two modes of hardcopy output: (Print Out) and (Save to File). Users can use either or both modes
- Since EasyPrinter runs on MS Windows system, it supports most of the printers available on the market
- Multiple HMIs can share one printer so users don't have to prepare printers for each HMI

Additionally, EasyPrinter can also be a backup server. Users can use Backup objects on HMI to copy history files such as Data Sampling records and Event Log to a remote PC via Ethernet. Please see the following illustration:



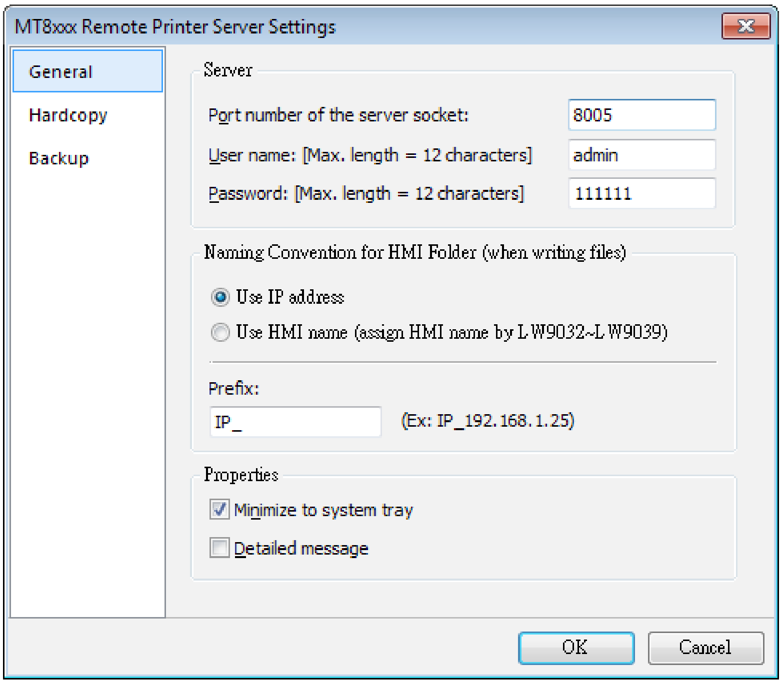
26.2. USING EASYPRINTER AS A PRINTER SERVER



Users can make screen hardcopies with a Function Key object. The hardcopies will be transferred to the Remote Printer Server via Ethernet and then printed out.

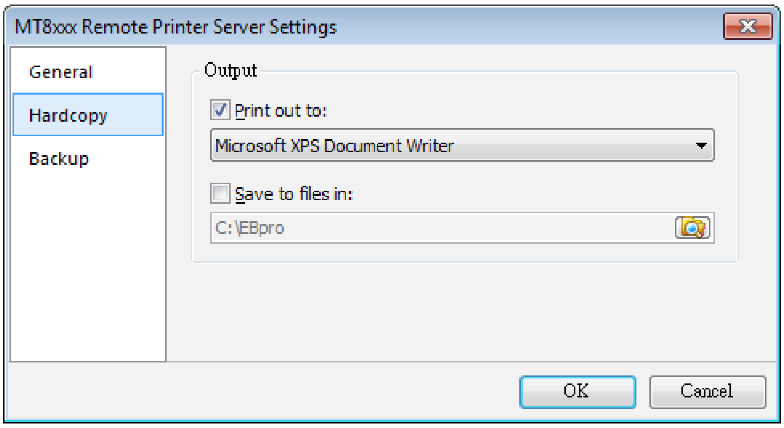
26.2.1. Setup Procedure in EasyPrinter

In EasyPrinter's main menu, select (Options) » (Settings) and the following dialog box appears:



1. Select (General) on the left hand side.
2. In (Server), set (Port number of the server socket) to “8005”, (User name) to “admin” and (Password) to “111111”. (These are default values.)
3. In (Naming Convention for HMI Folder), select (Use IP address) and enter “IP_” in the (Prefix) field.
4. In (Properties), select (Minimize to system tray) check box.

Set the print out location.

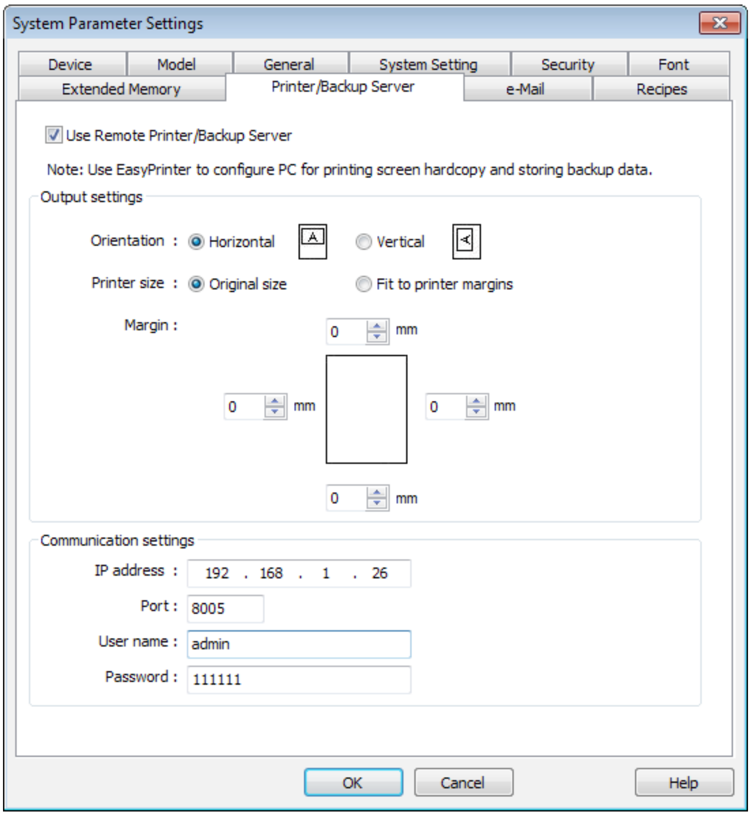


1. Select (Hardcopy) on the left hand side.
2. Under (Output) select (Print out to) and choose a printer as the output device for screen hardcopies. (The printer shown in the image above is an example; please select an actual printer located in your network environment.)
3. Click (OK) to confirm the settings.
4. In EasyPrinter main menu select (File) » (Enable Output) to output any incoming print request.

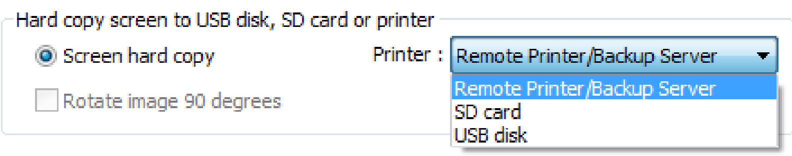
26.2.2. Setup Procedure in EasyBuilder Pro

The setting procedure of EasyPrinter in EasyBuilder Pro:

1. Open a new project or an existing project in EasyBuilder Pro.
2. In EasyBuilder Pro main menu select (Edit) » (System Parameter Settings) » (Printer/Backup Server) and select (Use Remote Printer/Backup Server) check box.



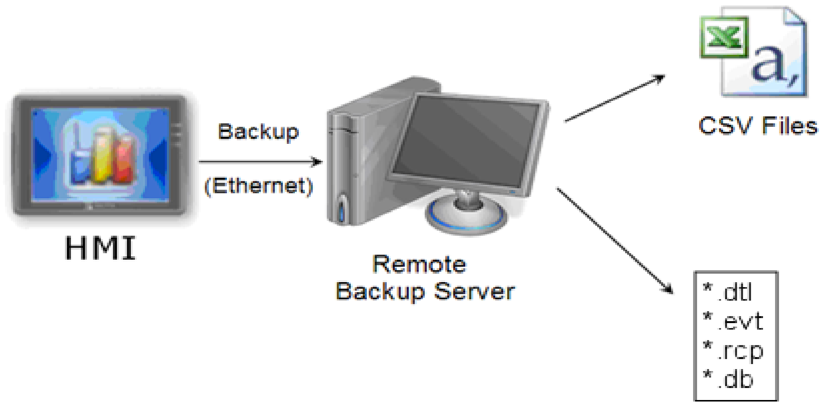
3. Under (Output settings) set appropriate values for left / top / right / bottom margins. (The margins are all set to 15mm in the example.)
4. Under (Communication settings) fill in the (IP address) of the printer server according to the settings in EasyPrinter. Set (Port) to “8005”, (User name) to “admin” and (Password) to “111111”.
5. Click (OK).
6. In EasyBuilder Pro main menu select (Objects) » (Button), select (Function Key), select (Screen hardcopy) and set (Printer) to (MT Remote Printer/Backup Server).



7. Place the Function Key object in the common window (window no. 4) so that screen hardcopies can be captured anytime when needed.
8. Compile and download the project to HMI. Press the Function Key object on the screen to make a screen hardcopy.

Note
- A PLC Control object can also be used to make screen hardcopies.
- Alarm information cannot be printed via EasyPrinter.
- EasyPrinter can only communicate with HMI via Ethernet. Please check that the HMI has appropriate network settings.

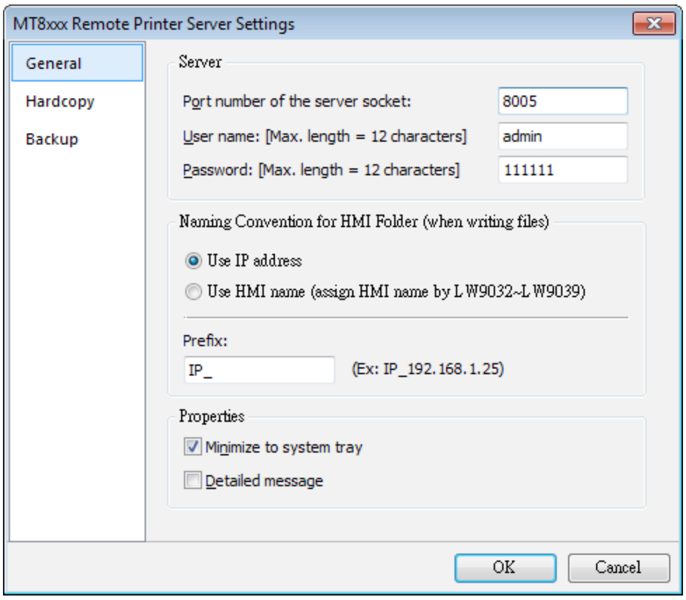
26.3. USING EASYPRINTER AS A BACKUP SERVER



Backup objects can upload historical data and Operation Log to remote backup server.

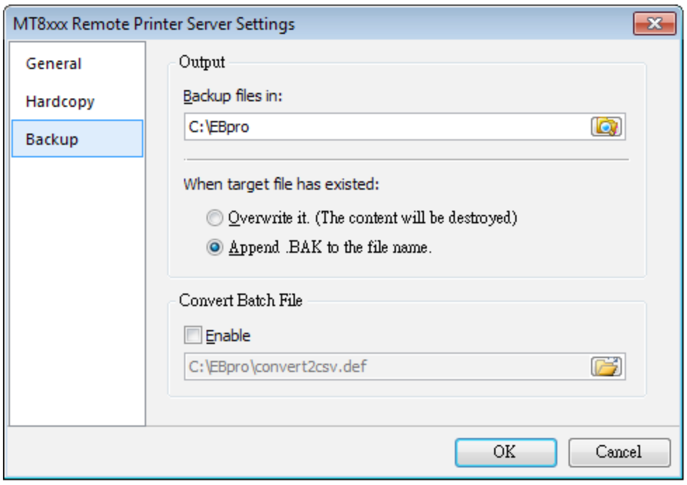
26.3.1. Setup Procedure in EasyPrinter

In EasyPrinter's main menu, select (Objects) » (Settings) and the following dialog box will appear:



1. Select (General) on the left hand side.
2. Under (Server) set (Port number of the server socket) to “8005”, (User name) to “admin” and (Password) to “111111”. (These are default values.)
3. Under (Naming Convention for HMI Folder) select (Use IP address) and enter “IP_” in the (Prefix) field.
4. Under (Properties) select (Minimize to system tray).

Set the backup location.

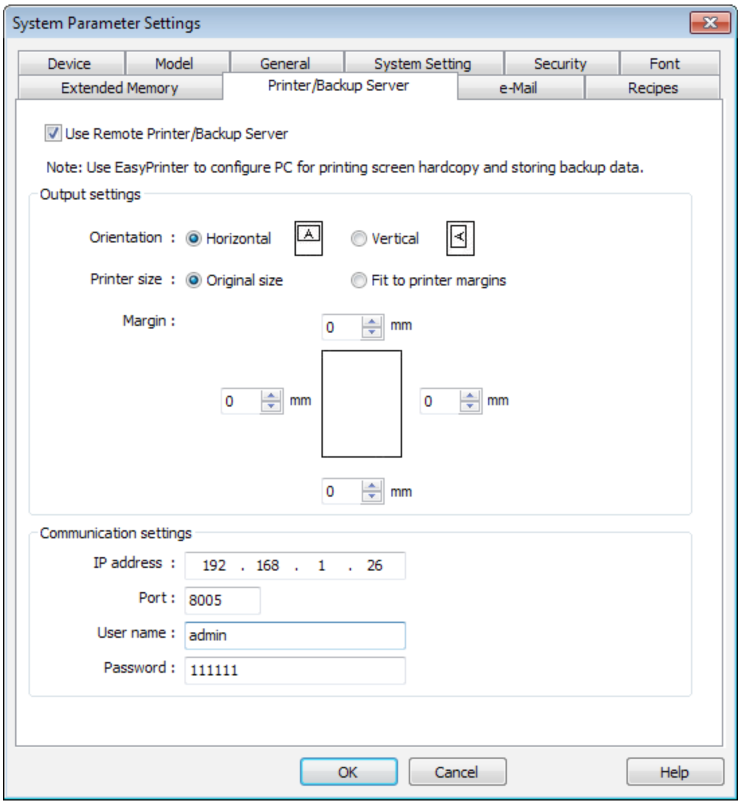


1. Select (Backup) on the left.
2. Under (Output) click the button to browse and select a storage directory of the incoming history files.
3. Click (OK) to confirm the settings.
4. In the main menu, select (File) » (Enable Output) to backup data in the selected directory.

26.3.2. Setup Procedure in EasyBuilder Pro

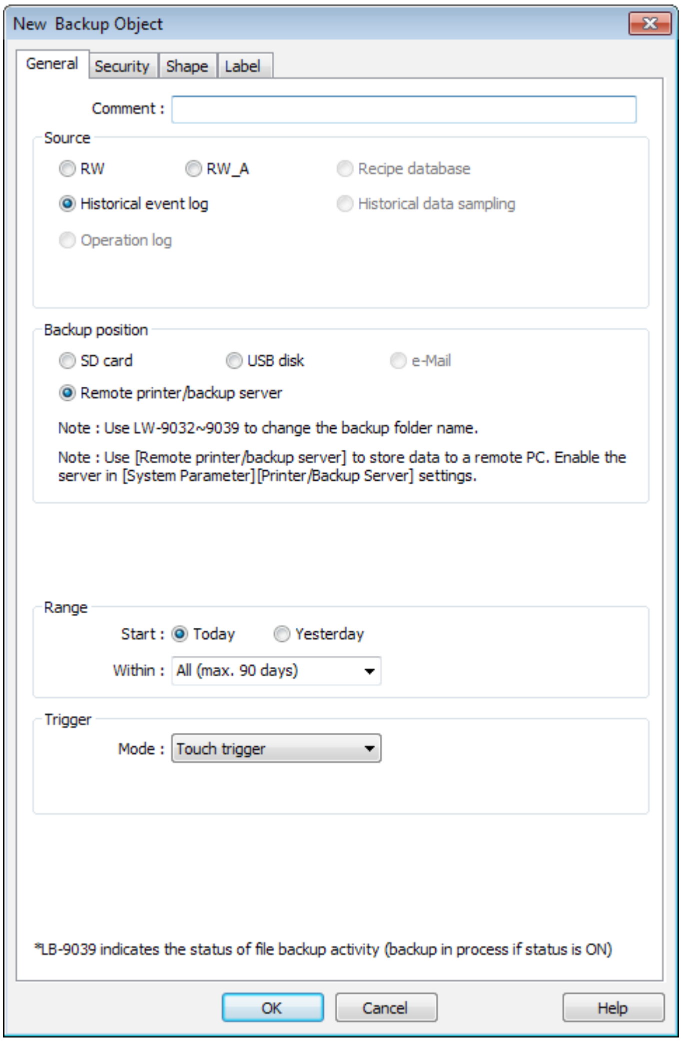
The setup procedure of EasyPrinter in EasyBuilder Pro:

1. Open a new project or an existing project in EasyBuilder Pro.
2. In EasyBuilder Pro’s main menu, select (Edit) » (System Parameter Settings) » (Printer/Backup Server) and select the (Use Remote Printer/Backup Server) check box.



3. Under (Communication settings) fill in the (IP address) of the printer server according to the settings in EasyPrinter. Set the (Port) to “8005”, (User name) to “admin” and (Password) to “111111”. (Note: these are default values.)
4. Click (OK).

Create a Backup object.
1. In EasyBuilder Pro’s main menu, select (Objects) » (Backup) and the following dialog box appears:



2. Under (Source) select (Historical event log) (or (RW), (RW_A) if needed.)
3. Under (Backup position) select (Remote printer/backup server).
4. Under (Range) select (Today) and (All) (or other options if needed.)
5. Under (Trigger) select (Touch trigger).
6. Click (OK).
7. Place the (Backup) object in the common window (window no. 4), and users will be able to make backups anytime when needed.
8. Compile and download the project to HMI. Press the Backup object on the screen to make a backup of the history data.

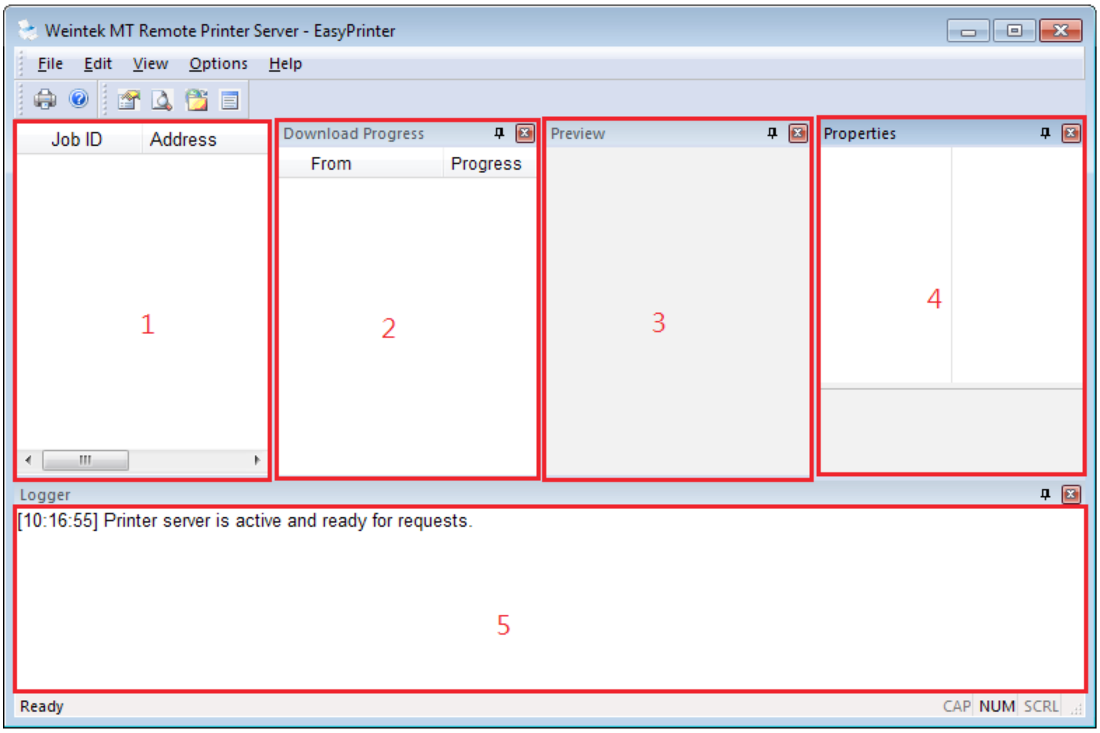
Note
- The Backup object can also be triggered by a bit address.
- Users can arrange a Scheduler object, which turns a bit ON at the end of a week, to trigger the Backup object to automatically back up all history data.

26.4. EASYPRINTER OPERATION GUIDE

The following introduces the interface and operation of EasyPrinter.

26.4.1. EasyPrinter Managing Window

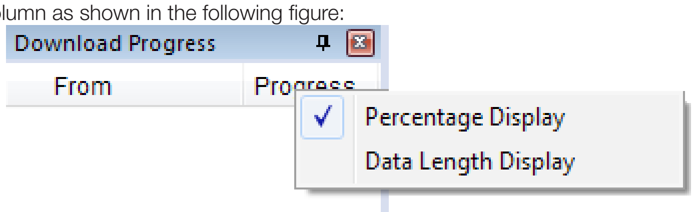
EasyPrinter main menu is divided into 5 parts as shown in the following figure:



Area	Name	Description
1	Job List	Lists all incoming tasks, such as screen hardcopy and backup requests.
2	Download Progress	Shows the download progress of incoming requests.
3	Preview	Shows the preview image of the screen hardcopy task selected from (Job List).
4	Properties	Shows the information about the task selected from (Job List).
5	Logger	Shows the time and message information of events such as incoming request, incorrect password, etc.

26.4.2. Operation Guide

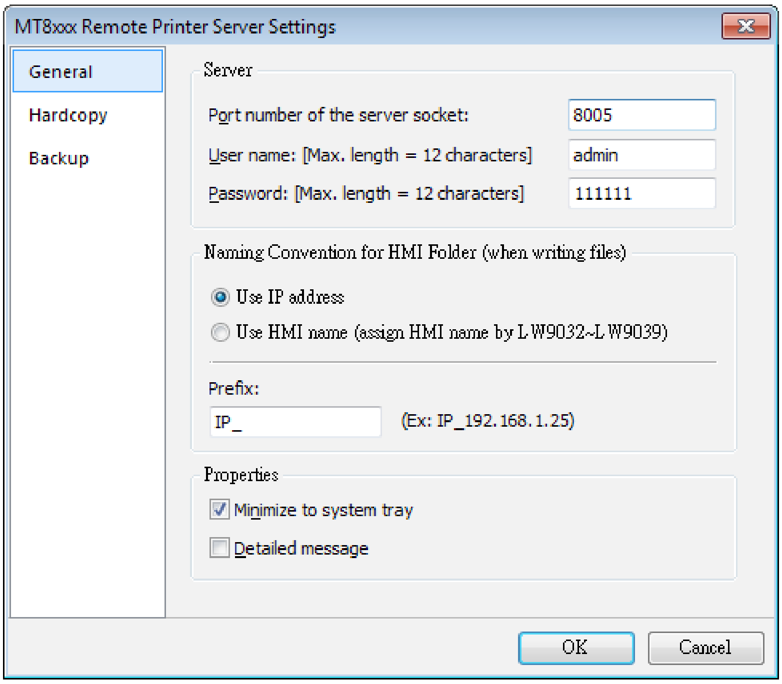
The following describes the function of EasyPrinter menu items.

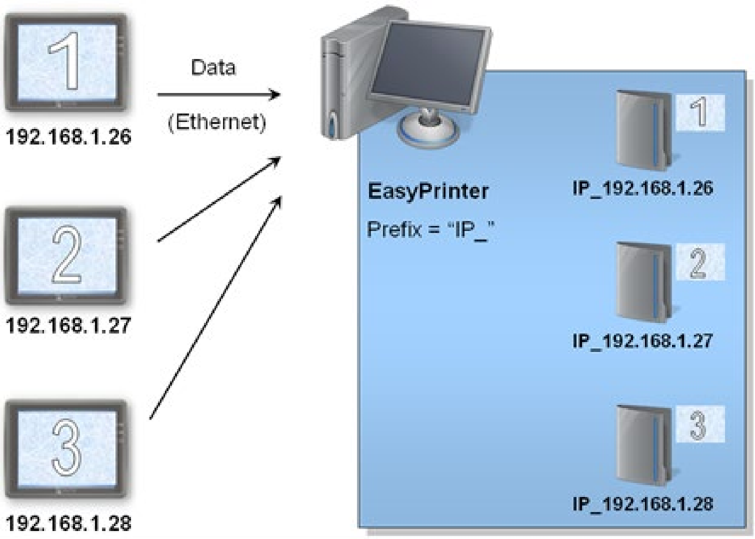
Menu	Description
File	**Enable Output** If selected, EasyPrinter processes the tasks one by one, otherwise, EasyPrinter stores the tasks in memory.
Edit	**Edit** Edits screen hardcopy by setting (Orientation), (Scaling) and (Margins). **Delete** Deletes the selected tasks permanently. **Select All** Selects all tasks from (Job List).
View	**Properties Bar** Shows or hide the Property Window. **Preview Bar** Shows or hide the Preview Window. **Download Bar** In (Download Progress) Window, the mode to display download progress can be set by clicking the header of the (progress) column as shown in the following figure:  **Logger Bar** EasyPrinter can reserve up to 10,000 messages in Message Window. If a new message comes in, the oldest message will be deleted.
Options	Please see the following page.

	Note ▪ EasyPrinter can only reserve up to 128 MB of task data in memory. If the memory is full, any request coming in afterwards will be rejected. Users must either operate (Enable Output) or delete some tasks to make room for new tasks. ▪ The backup task is not editable. ▪ (Edit) is available only when a task is selected. ▪ (Delete) is available when at least one task is selected.

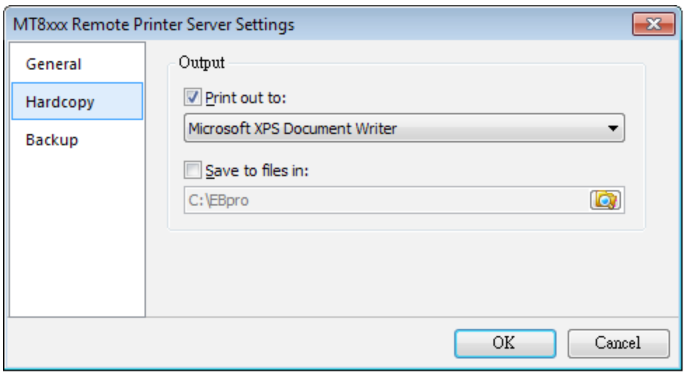
The following is the detail for (Options) » (Settings)

▪ In General Tab:



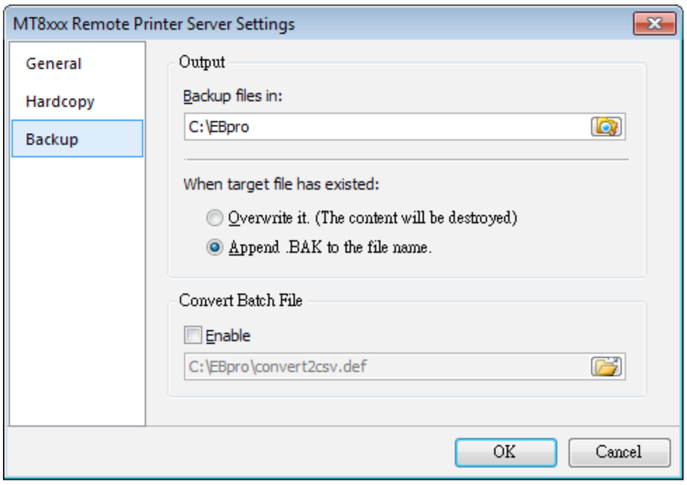
Setting	Description
Server	**Port number of the server socket** Sets the Ethernet port number to connect the HMI. Range: 1 ~ 65535. Default: 8005. **User name / Password** Sets the user name and password to let only authorized HMIs send requests to EasyPrinter.
Naming Convention for HMI Folder	EasyPrinter uses different folders to store files (e.g. hardcopy bitmap files, backup files) from different HMI. There are two ways to name the folders: **Use IP address** EasyPrinter names the folder as (Prefix) + (IP address) after the HMI at this IP address sends request.  **Use HMI name** EasyPrinter names the folder in (Prefix) + (HMI name) after the HMI this name indicates sends request.
Properties	Minimize to system tray If this check box is selected, the EasyPrinter shortcut icon will be placed in the system tray in PC. Double click the on icon in system tray to open EasyPrinter. **Detailed message** Select this check box to display more detailed messages about events in the message window.

▪ In Hardcopy Tab:



Setting	Description
Output	**Print out to** EasyPrinter prints out the hardcopy result with the specified printers. **Save to files in** EasyPrinter converts the hardcopy result into a bitmap file and saves it in the specified directory. The bitmap files are found at: (Specified Path) \ (HMI Folder) \ yymmdd_hhmm.bmp For example, when a hardcopy request is given at 17:35:00, 12/Jan/2009, the bitmap file will be named “090112_1735.bmp”. And if there is another bitmap file generated within the same minute, it will be named “090112_1735_01.bmp” and so on.

▪ In Backup Tab:



Setting	Description

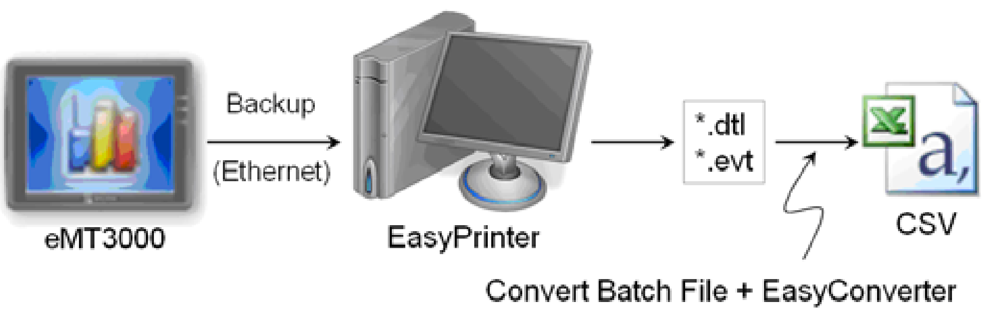
Output	Backup files in:
	EasyPrinter stores the backup files to the specified path.
	The upper directory is the same:
	(Specified Path) \ (HMI Name) or (IP address)
	The lower directory:
	▪ For Event Log files:
	  ▪ \ eventlog\ EL_yyyymmdd.evt
	▪ For Data Sampling files:
	  ▪ \ datalog \ (File name of the Data Sampling) \ yyyymmdd.dtl
	▪ For Recipe files:
	  ▪ \ recipe \ recipe.rcp or recipe_a.rcp
	▪ For Recipe Database:
	  ▪ \ recipe \ recipe.db
	▪ For Operation Log:
	  ▪ \ operationlog\ operationlog.db

Convert batch file	Select (Enable) to convert the selected history file to .csv or .xls (Excel) format of Convert Batch Files.

Note
System registers LW-9032 to LW-9039 can be used to specify HMI name.

26.5. CONVERT BATCH FILE

EasyPrinter provides a conversion tool to convert the uploaded Data Sampling and Event Log history files to .csv files automatically. To do so, please select (Enable) under (Convert Batch File) to make EasyPrinter convert the history files.
In the following illustration, the conversion is actually executed by EasyConverter. EasyPrinter simply follows the criteria in Convert Batch File and activates EasyConverter with proper arguments to achieve the conversion.



Note
▪ EasyConverter is another Win32 application that converts history data into .csv or MS Excel .xls files. Users can find it in the EasyBuilder Pro installation directory.
▪ Users requesting this function must ensure EasyPrinter and EasyConverter are placed in the same directory.

26.5.1. The Default Value of Convert Batch File

The following is the default Convert Batch File: convert2csv.def
Listing 1. Default Convert Batch File

1: “dtl”, “EasyConverter /c $(Pathname)”
2: “evt”, “EasyConverter /c $(Pathname)”

There are two lines in the file. Each line has two arguments separated by a comma and forms a criterion of how to process a specific type of files. The first argument stands for the extension name of the file type to be processed. The second argument stands for the command to be executed in console mode. Please note that “$(Pathname)” is a key word to inform EasyPrinter to replace it with the real name of the converted backup file. For example, if a Data Sampling history file named 20090112.dtl is uploaded and stored, EasyPrinter will send out the following command to a console window:

1: EasyConverter /c 20090112.dtl

A file named 20090112.csv is created.
The criteria of the default Convert Batch File:

1. Convert all Data Sampling history files (.dtl) into .csv files.
2. Convert all Event Log history files (.evt) into .csv files.

Note

- “$(Pathname)” in the second argument stands for the full path name of the file. In the previous case, EasyPrinter replaces it with:
  - (Specified Path) \ (HMI Folder) \ (datalog) \ (Folder name of the Data-Sampling object) \ 20090112.dtl
- EasyPrinter interprets the Convert Batch File in line basis, that is, each line forms a criterion.
- Any two arguments should be separated by a comma.
- Every argument should be put in double quotes.
- Do not put any comma inside an argument.

For more information, see “25 Easy Converter”.

26.5.2. Specialized Criteria

The specialized criterion are needed when:
- Upload file to a specific HMI, see listing 2
- Identify the HMI by HMI name, see listing 3
- Process differently to different Data Sampling, see listing

(This can only be used for Data Sampling file with the file name “voltage”.)
The 3rd argument (“*”) indicates this criterion accepts the Data Sampling files that meet the criterion from any HMI. Users can also change the 3rd argument to “192.168.1.26”, “192.168.1.*”, or HMI name, etc. for narrowing the range of the target HMI.

Listing 2. Specialized Criterion for the HMI IP: 192.168.1.26
1: “dtl”, “EasyConverter /c $(Pathname)”, “192.168.1.26”

Listing 3. Specialized Criterion for HMI name: Weg_01
1: “dtl”, “EasyConverter /c $(Pathname)”, “Weg_01”

Listing 4. Specialized Creterion for Data Sampling file name: Voltage
1: “dtl”, “EasyConverter /s Voltage.lgs $(Pathname)”, “*”, “Voltage”

26.5.3. The Format of a Convert Batch File

The following explains the arguments in a criterion.

File Type	Command (line)	HMI IP / Name	Condition 1	Condition 2

- File Type
  This argument specifies the extension name of the uploaded file in this criterion
  (e.g. “.dtl” for Data Sampling history files, “.evt” for Event Log history files)
- Command (line)
  The command EasyPrinter sends to a console window if the uploaded file meets the criterion

- HMI IP / Name
  This argument specifies the HMI that meets the criterion
- Condition 1
  This argument specifies the folder name of the Data Sampling files that meet the criterion.
  This is not effective to other format of files
- Condition 2
  Not used (Reserved for future use)

26.5.4. The Order of Examining Criterion

EasyPrinter examines criterion in descending order every time a file is uploaded. Once the file meets a criterion, it stops the examination and starts over for the next file. Therefore, users should place the criterion with a wider range downward in the Convert Batch File and place the more specific criteria upward. For example:
"evt", "EasyConverter /c $(Pathname)"
"dtl", "EasyConverter /c $(Pathname)"
"dtl", "EasyConverter /c $(Pathname)", "192.168.1.26"
"dtl", "EasyConverter /c $(Pathname)", "my_HMI_01"
"dtl", "EasyConverter /c $(Pathname)", "my_HMI_02"
"dtl", "EasyConverter /s Voltage.lgs $(Pathname)", "*", "Voltage"

The correct order of examination would be: (from bottom to top)
"dtl", "EasyConverter /s Voltage.lgs $(Pathname)", "*", "Voltage"
"dtl", "EasyConverter /c $(Pathname)", "my_HMI_02"
"dtl", "EasyConverter /c $(Pathname)", "my_HMI_01"
"dtl", "EasyConverter /c $(Pathname)", "192.168.1.26"
"dtl", "EasyConverter /c $(Pathname)"
"evt", "EasyConverter /c $(Pathname)"

27. EASYSIMULATOR

This chapter explains how to use EasySimulator.

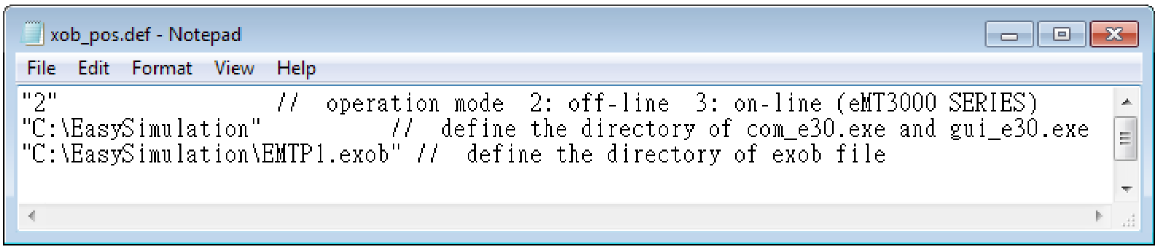
27.1. OVERVIEW

EasySimulator allows a project to be run in the On- or Off-line simulator without having to start the simulator from EasyBuilder Pro. To do this, please prepare the required files and follow the steps to setup EasySimulator.

27.2. STEPS TO SETUP EASYSIMULATOR

1. Prepare the following required files.
   - (driver) → (win32)
   - com_e30.exe
   - EasySimulator.exe
   - gui_e30.exe
   - sqlite3.dll
   - xob_pos.def

2. Open xob_pos.def by using a text editing tool (e.g. Notepad) and edit the contents.



Line number	Description
1	“2” run an Off-line Simulation; “3” run an On-line Simulation.
2	The directories of the relevant files. (e.g. com_e30.exe, gui_e30.exe, EasySimulator.exe…etc.)
3	The full path of the .exob file.

3. Double click on EasySimulator.exe to start a simulation.
4. On-line / Off-line Simulation is displayed on the screen.

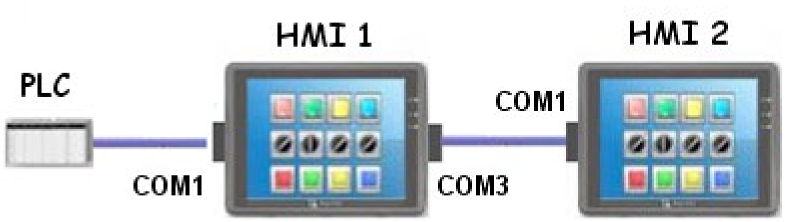
Note
- The required files can be found in the EasyBuilder Pro installation directory. Please install EasyBuilder Pro first then copy the required files to your PC.
- If EasySimulator.exe is not activated, please check if the installation directory is correct.
- If the “Failed to open project file: no such file or directory” dialog box appears, this indicates that there is an error of the .exob file path, please check again.

28. MULTI-HMI COMMUNICATION (MASTER SLAVE MODE)

This chapter explains how to connect multiple HMIs.

28.1. OVERVIEW

Multi-HMI Communication means that a HMI connects with a remote HMI via COM port, and read the data in the PLC connected to the remote HMI as shown in the following figure.



The PLC is connected with HMI 1, and HMI 1 is connected with HMI 2 via COM port, so that HMI 2 can read the data in PLC through HMI 1.

The following parts explain how to create the projects used in HMI 1 and HMI 2 by using EasyBuilder Pro.

28.2. STEPS TO CREATE A PROJECT OF MASTER HMI

The following is the settings of HMI 1 in (System Parameter Settings) » (Device List).

Device list :

No.	Name	Location	Device type	Interface
Local HMI	Local HMI	Local	eMT/MT SERIES	-
Local PLC 1	FATEK FB Series	Local	FATEK FB Series	COM 1 (9600,E,...
Local Server	Master-Slave S...	Local	Master-Slave S...	COM 3 (115200,...

1. Since COM 1 of HMI 1 connects to PLC; the device list must include (Local PLC 1), and set the correct parameters. In this example the connected PLC is “FATEK FB Series”.
2. COM 3 of HMI 1 is used to receive commands from HMI 2; a new device must be added– (Master-Slave Server) for setting communication properties of COM 3.
   The parameters of COM 3 in the example are set to “115200, E, 8, 1”, and uses RS232. These parameters are not required to be the same as PLC settings, but the (Data bits) must be set to 8. In general, a higher baud rate is recommended for HMI 2 to efficiently read PLC data.

28.3. STEPS TO CREATE A PROJECT OF SLAVE HMI

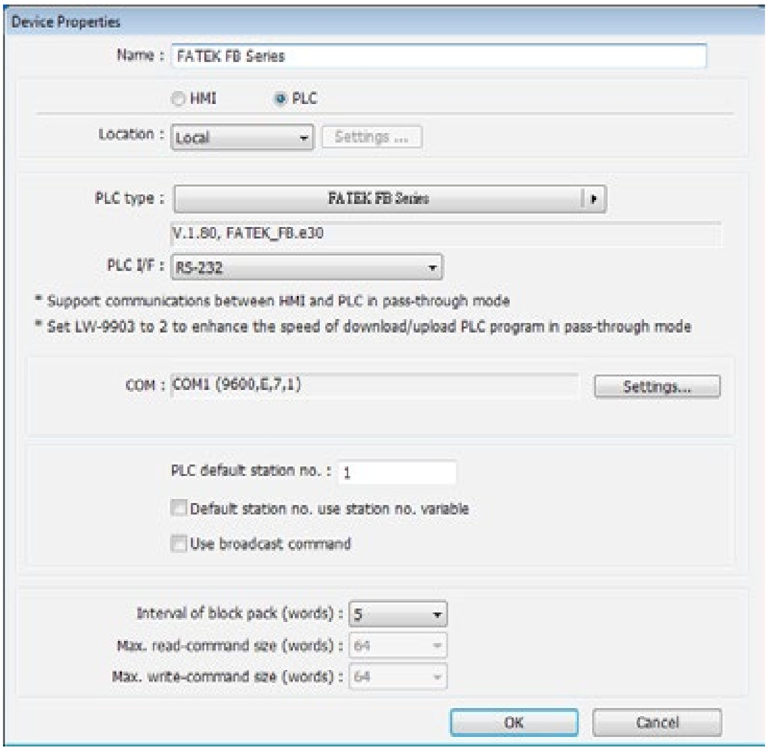
The following is the settings of HMI 2 in (System Parameter Settings) » (Device List).

Device list :

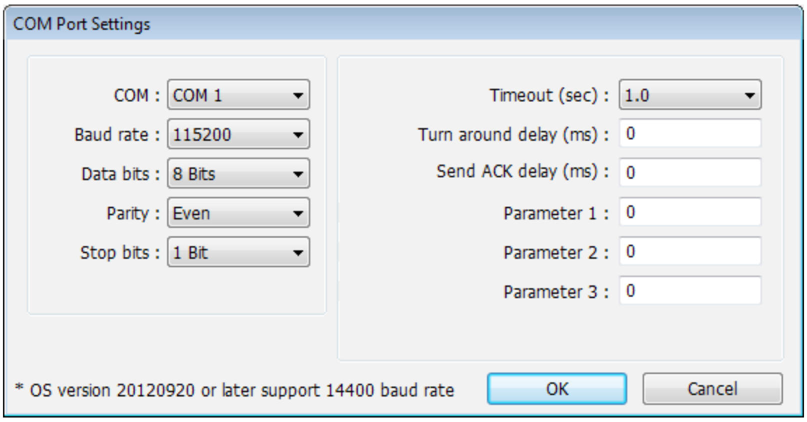
No.	Name	Location	Device type	Interface
Local HMI	Local HMI	Local	eMT/MT SERIES	-
*Remote PL...	FATEK FB Series	COM 1 (...	FATEK FB Series	COM 1 (115200,...

Since the PLC that HMI 2 reads is connected with HMI 1, thus for HMI 2, PLC is a remote device. Therefore, it is necessary to add a (Remote PLC) into the device list. In this example the connected PLC is “FATEK FB Series”. The way to create (*Remote PLC 1) is described in the following steps.

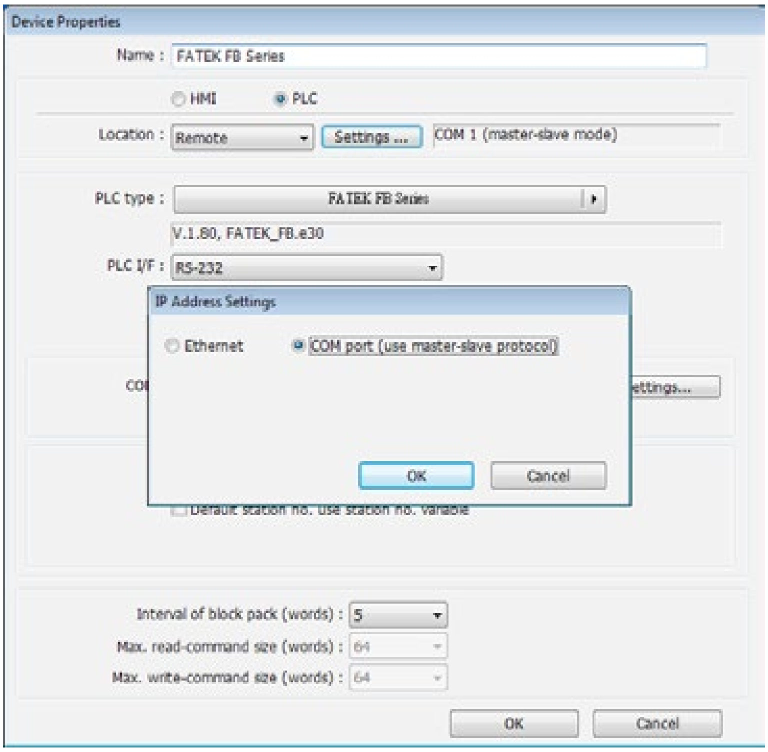
1. Add a new device. Set (PLC type) to (FATEK FB Series) and (PLC default station no.) must be set in accordance with the connected PLC.

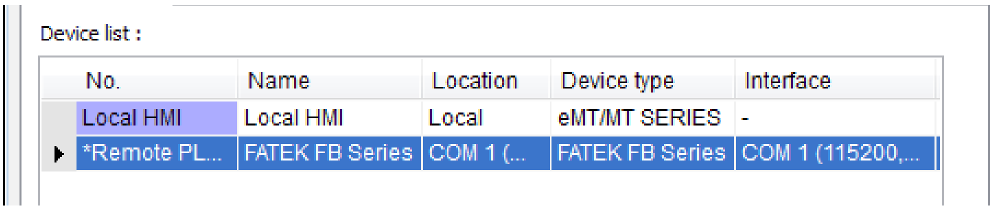


2. Correctly set the parameters. Since COM 1 of HMI 2 connects with COM 3 of HMI 1 instead of directly connect with PLC, the settings of PLC will be ignored. HMI 2 COM 1 and HMI 1 COM 3 must set to the same communication parameters and interfaces. As shown in the following figure, use RS232, and set parameters to (115200, E, 8, 1).



3. For HMI 2, PLC is a remote device, change (Location) to (Remote), and select (COM port) to connect remote HMI (HMI 1).





4. When finished, a new device (Remote PLC) can be found in the (Device List). This device has a "*" symbol, which means, even if it contains "Remote" in the name, it actually gives commands and gets replies through a local COM port, and therefore the connection with PLC can be checked from a local system register. (*Remote PLC 1), (*Remote PLC 2), (*Remote PLC 3) and (Local PLC 1), (Local PLC 2), (Local PLC 3) use the same system registers from the listed below.

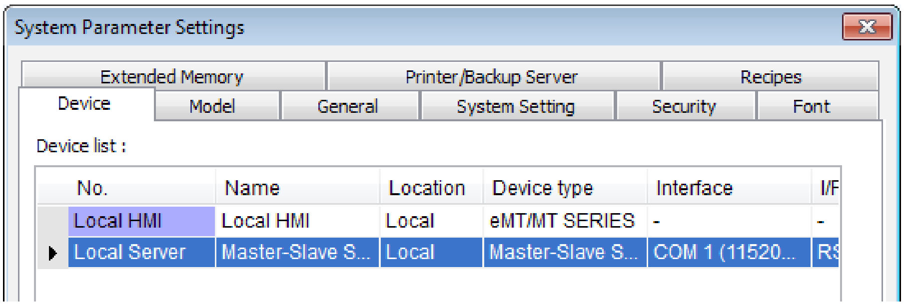
Register	Description
LB-9150	When ON, automatically connects with PLC (COM 1) when disconnected. When OFF, ignores disconnection with PLC.
LB-9151	When ON, automatically connects with PLC (COM 2) when disconnected. When OFF, ignores disconnection with PLC.
LB-9152	When ON, automatically connects with PLC (COM 3) when disconnected. When OFF, ignores disconnection with PLC.
LB-9200~ LB-9455	These local registers indicate the connection states with PLC (through COM1). LB9200 indicates the connection state with PLC (station no. 0), and LB9201 indicates the connection state with PLC (station no. 1) and so on. When ON, indicates the connection state is normal. When OFF, indicates disconnection with PLC. Set ON again, the system will then try to connect with PLC.
LB-9500~ LB-9755	These local registers indicate the connection states with PLC (through COM2). LB9500 indicates the connection state with PLC (station no. 0), and LB9501 indicates the connection state with PLC (station no. 1) and so on. When ON, indicates the connection state is normal. When OFF, indicates disconnection with PLC. Set ON again, the system will then try to connect with PLC.
LB-9800~ LB-10055	These local registers indicate the connection states with PLC (through COM3). LB9800 indicates the connection state with PLC (station no. 0), and LB9801 indicates the connection state with PLC (station no. 1) and so on. When ON, indicates the connection state is normal. When OFF, indicates disconnection with PLC. Set ON again, the system will then try to connect with PLC.

28.4. STEPS TO CONNECT WITH MT500 SLAVE HMI

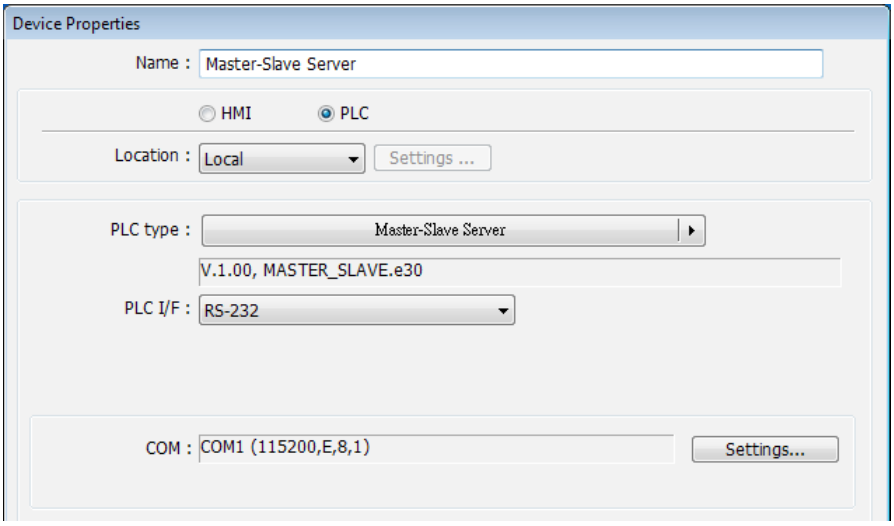
EasyBuilder Master-Slave Protocol enables MT500 to exchange data with eMT3000 local data via the connected PLC.

28.4.1. Settings in EasyBuilder Pro

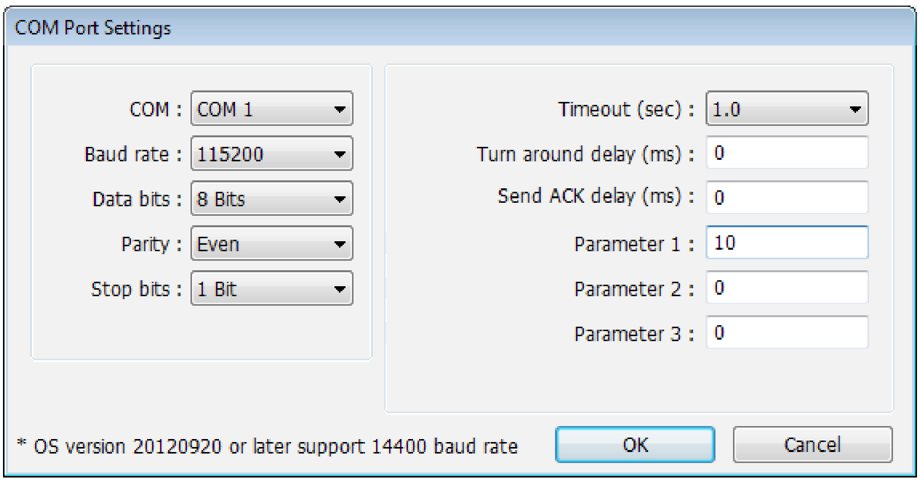
1. Select (Master-Slave Server) and click (Settings). If a PLC is connected, follow the original settings.



2. Select (RS232), click (Settings).



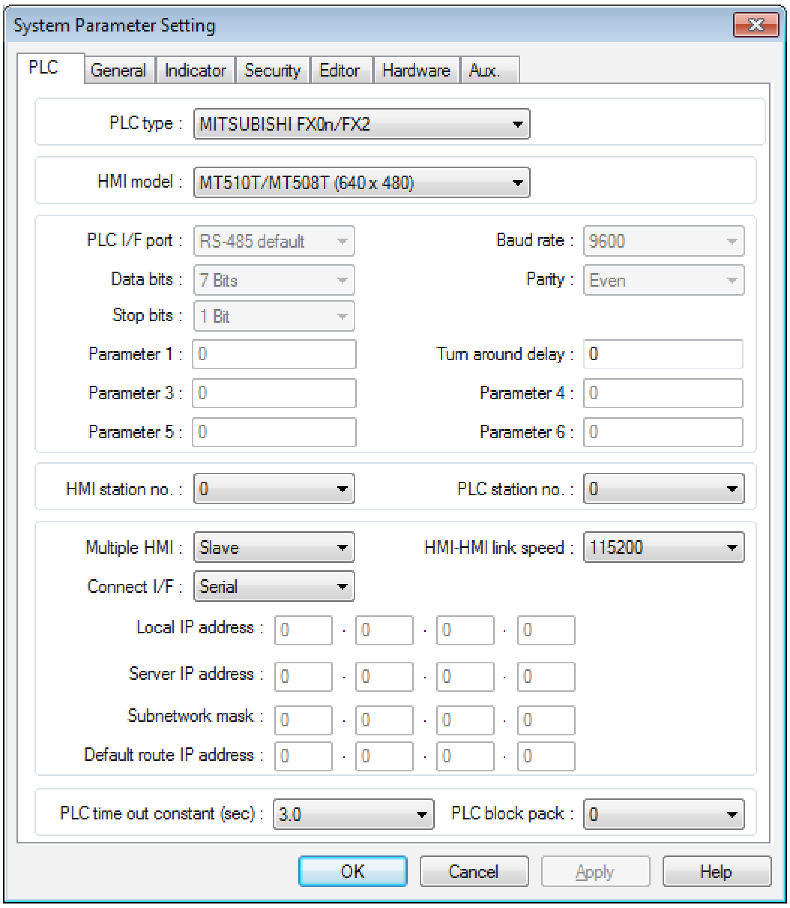
3. Fill in MT500 PLC ID No. in (Parameter 1) (Refer to MT500 settings).



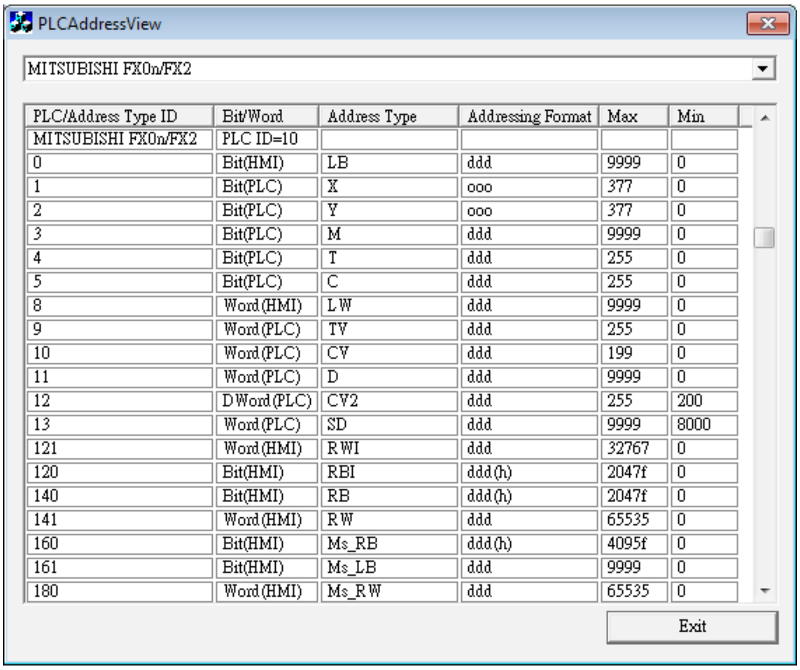
28.4.2. Settings in EasyBuilder500

1. In (System Parameter Settings), set (Multiple HMI) to Slave, set (HMI-HMI link speed) to 115200.

Note
(Baud rate) must be identical in EasyBuilder500 and EasyBuilder Pro.



2. Double click PLC Address View.exe to check PLC ID No. and fill in (Parameter 1) of EasyBuilder.



PLC/Address Type ID	Bit/Word	Address Type	Addressing Format	Max	Min
MITSUBISHI FX0n/FX2	PLC ID=10				
0	Bit(HMI)	LB	ddd	9999	0
1	Bit(PLC)	X	ooo	377	0
2	Bit(PLC)	Y	ooo	377	0
3	Bit(PLC)	M	ddd	9999	0
4	Bit(PLC)	T	ddd	255	0
5	Bit(PLC)	C	ddd	255	0
8	Word (HMI)	LW	ddd	9999	0
9	Word (PLC)	TV	ddd	255	0
10	Word (PLC)	CV	ddd	199	0
11	Word (PLC)	D	ddd	9999	0
12	DWord (PLC)	CV2	ddd	255	200
13	Word (PLC)	SD	ddd	9999	8000
121	Word (HMI)	RWI	ddd	32767	0
120	Bit(HMI)	RBI	ddd (h)	2047f	0
140	Bit(HMI)	RB	ddd (h)	2047f	0
141	Word (HMI)	RW	ddd	65535	0
160	Bit(HMI)	Ms_RB	ddd (h)	4095f	0
161	Bit(HMI)	Ms_LB	ddd	9999	0
180	Word (HMI)	Ms_RW	ddd	65535	0

3. Connect HMIs via Com Port RS232, the communication is then enabled.

Note

- There will always be a PLC selected in MT500 system parameter settings, in this case, even to read/write eMT3000 local data, the ID of the selected PLC of MT500 system parameters must also be filled in EasyBuilder (Parameter 1).
- When using S7-200, S7-300 drivers, since MT500 reverses the high bytes and the low bytes, this will cause MT500 to misread eMT3000 local data, therefore this way is not available in Master-Slave Mode.

The Comparison between MT500 and eMT3000:

Bit/Word	MT500	eMT3000	Range
B	Ms_RB	RW_Bit	dddd: 0~4095 (h): 0~f
B	Ms_LB	LB	dddd: 0~9999
W	Ms_RW	RW	ddddd: 0~65535
W	Ms_LW	LW	dddd: 0~9999

29. PASS-THROUGH

This chapter explains how to set up Pass-through mode.

29.1. OVERVIEW

The Pass-through feature allows PC applications to control PLC via HMI. In this case the HMI is an adaptor.
The Pass-through feature provides two modes:

- Ethernet
- COM port

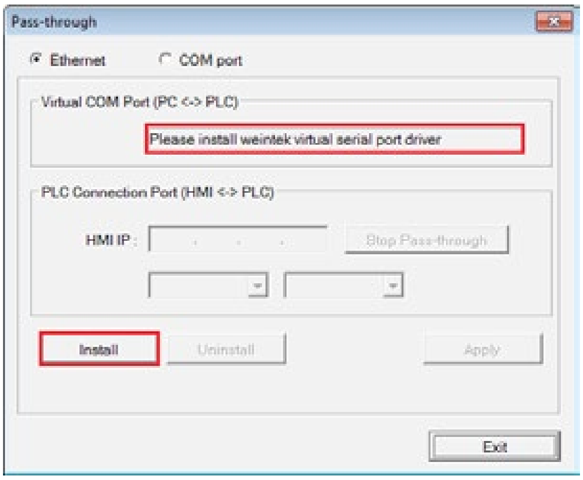
Click (Pass-through) in Utility Manager to open the setting dialog box.

29.2. ETHERNET MODE

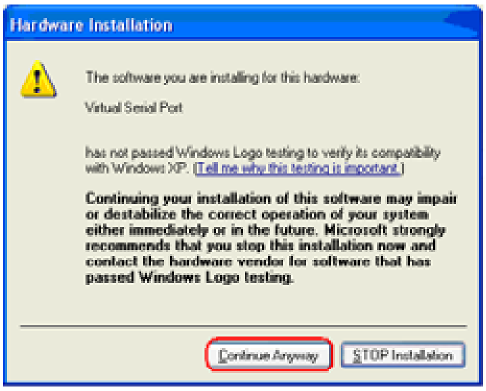
29.2.1. Steps to Install Virtual Serial Port Driver

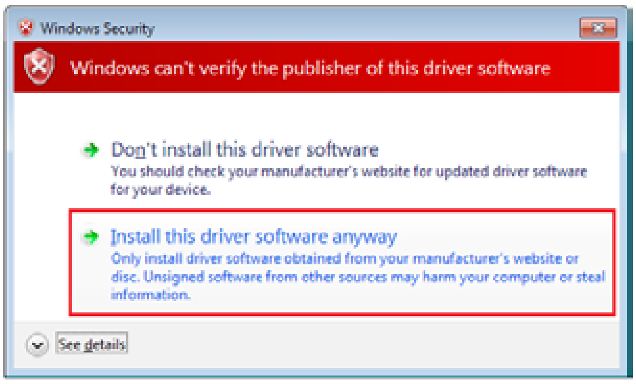
Before using (Ethernet) mode, please check if Weintek virtual serial port driver has been installed.

1. Open Utility Manager to check if the driver has been installed. If it shows (Please install weintek virtual serial port driver), please click (Install).



2. If the dialog below pops up during installation asking for verification, please click (Continue Anyway).

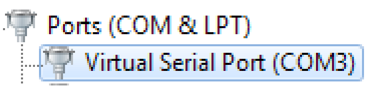




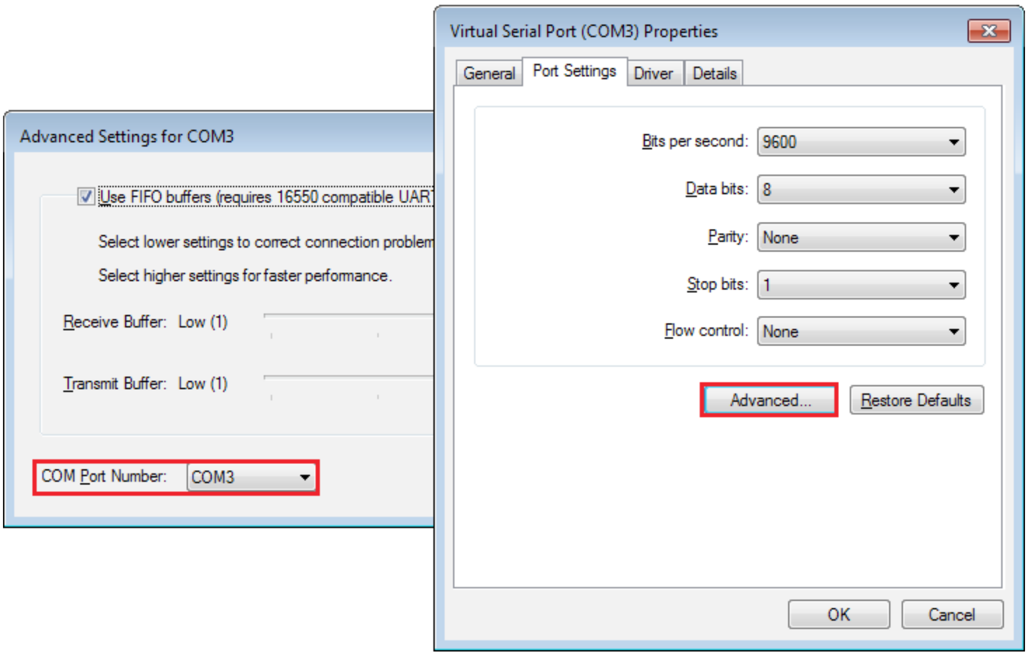
3. When finished, the (Virtual COM Port (PC <-> PLC)) field displays the virtual COM port used.

29.2.2. Steps to Change the Virtual Serial Port

1. Open (Device Manager) to check the installed (Virtual Serial Port).

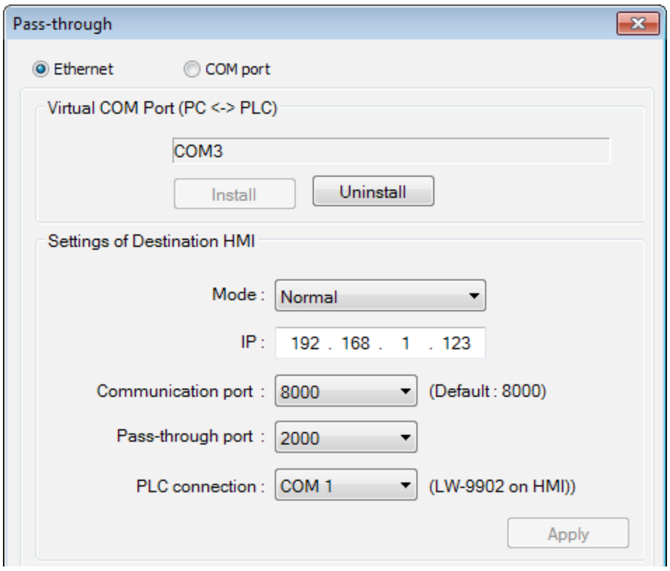


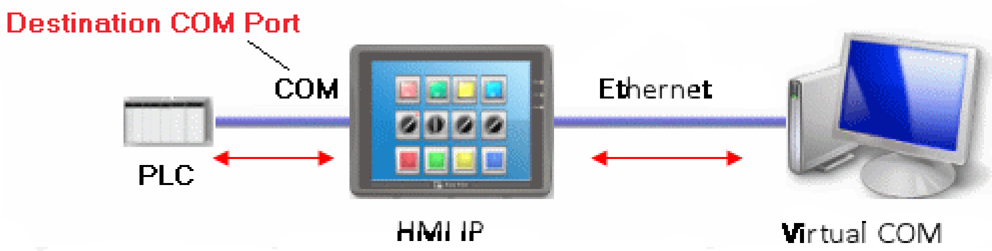
2. To change the number of virtual serial port, click (Virtual Serial Port) to open (Port Settings) » (Advanced).



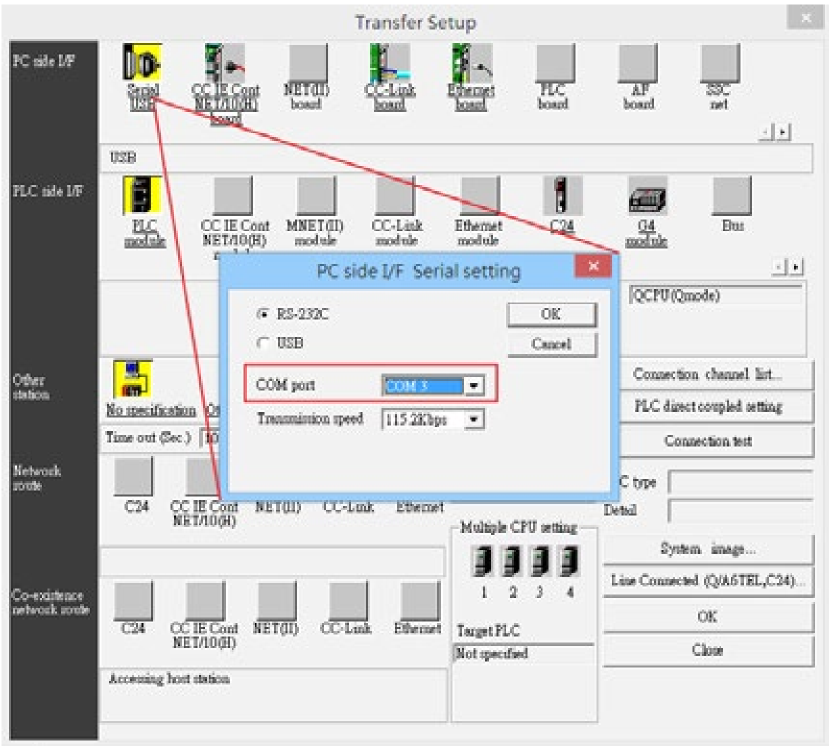
29.2.3. Settings of Ethernet Mode

After installing the virtual serial port driver, follow the steps to use Ethernet mode of pass-through feature.
1. Set the IP address of the HMI connected with PLC.
2. Set the communication port and the serial port that connects HMI with PLC.
3. Click (Apply), to apply the settings.



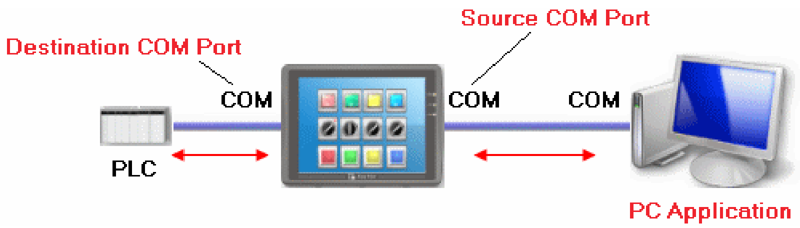


4. When running PC application, set COM port to the used virtual serial port. For example, in Mitsubishi application, if the virtual serial port is COM 3, set (PC side I/F Serial setting) » (COM port) to COM 3.



5. With the correct configurations, upon execution of PLC application on PC, HMI will be automatically switched to Pass-through mode. During Pass-through, the PLC is controlled by PC via the virtual serial port. Pass-through mode will be turned off when the application ends.

29.3. COM PORT MODE



(Source COM Port) The port connects HMI with PC.
(Destination COM Port) The port connects HMI with PLC.
To use (COM port) mode of Pass-through, please set the properties of Source COM Port and Destination COM Port correctly.

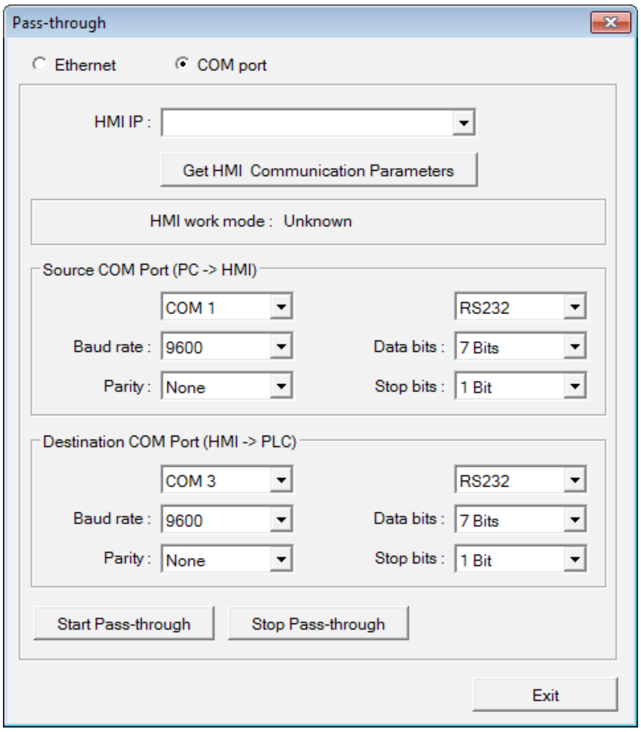
29.3.1. Settings of COM Port Mode

There are two ways to enable (COM port) mode of Pass-through feature.

- Using Utility Manager.
- Using system registers.
  - LW-9901: pass-through source COM port (1 ~ 3: COM 1 ~ COM 3)
  - LW-9902: pass-through destination COM port (1 ~ 3: COM 1 ~ COM 3)

29.3.2. Using Utility Manager

1. Click (Pass-through) button in Utility Manager to set the communication parameters as shown in the following figure.



Setting	Description
HMI IP	HMI IP address.
Get HMI Communication Parameters	Reads the settings of Source and Destination COM port. Click this button to update the communication parameters.
Source COM port (PC->HMI) / Destination COM port (HMI->PLC)	The communication parameters of Source and Destination COM Port are displayed. The settings will be applied when (Start Pass-through) is clicked.
Baud rate / Data bits / Parity / Stop bits	Source and Destination COM Port parameters should be set to be same. Since (Source COM Port) connects PC, select RS232 mode in most situations; (Destination COM Port) connects PLC, so the setting depends on the PLC type, and can be one of RS232, RS485 2W, or RS485 4W.

Note
When pass-through feature is no longer needed, click (Stop Pass-through) to stop it. HMI will then resume communication with PLC.

There are three work modes of HMI.

Mode	Description
Unknown	The work mode before reading the settings of HMI.
Normal	The work mode after reading the settings of HMI. The HMI does not accept any data form the Source COM Port.
Pass-through	The work mode is “Pass-through.” the PC connected via Source COM Port can control the PLC connected via Destination COM Port.

29.3.3. Using System Registers

Another way of enabling pass-through is by writing to LW-9901(Source COM port) and LW-9902 (Destination COM port).When the values of LW-9901 and LW-9902 match the conditions below, HMI will start Pass-through automatically:

- The values of LW-9901 and LW-9902 are 1 to 3 (1 to 3: COM 1 to COM 3)
- The values of LW-9901 and LW-9902 are different

To change the communication parameters, just change the value in the related registers and set ON the appropriate registers: (LB-9030: update COM 1 communication parameters), (LB-9031: update COM 2 communication parameters) and (LB-9032: update COM 3 communication parameters). HMI will then update the settings.

Note
To stop Pass-through, change the values of LW-9901 and LW-9902 to 0.

29.4. PASS-THROUGH CONTROL

Generally speaking, during pass-through, HMI closes its connection with the PLC until the pass-through mode ends. However, certain PLC drivers allow communications between HMI and PLC in pass-through mode.

To see whether a driver supports concurrent communication, see “PLC Connection Guide”.
Pass-through control is controlled by LW-9903. The following table shows valid LW-9903 values and their features.

LW-9903	Description
0 (Default)	Normal Mode. Communications between HMI and PLC in pass-through mode is allowed.
2	Stop Mode. No communications between HMI and PLC in pass-through mode

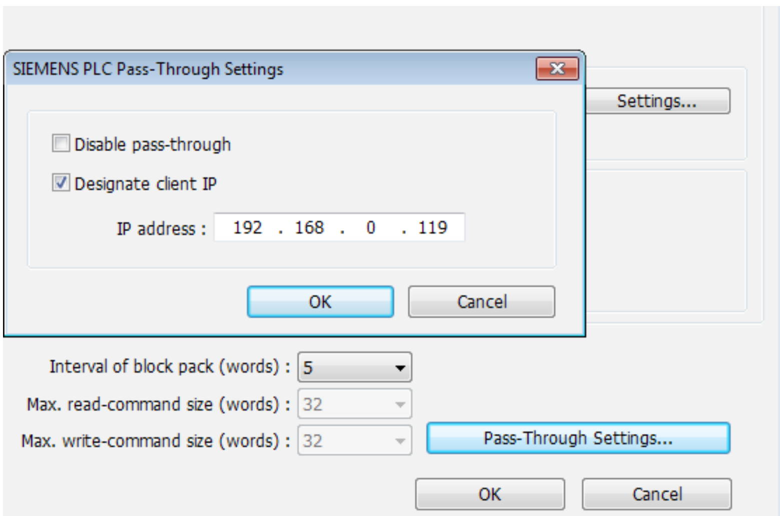
Note
Due to speed limitation, users may wish to set LW-9903 to 2 to enhance the speed of program download/upload in pass-through mode.

29.5. SIEMENS S7-200 PPI AND S7-300 MPI PASS-THROUGH SETTINGS

EasyBuilder Pro supports SIEMENS S7-200 PPI and S7-300 MPI pass-through feature.

29.5.1. EasyBuilder Pro Settings

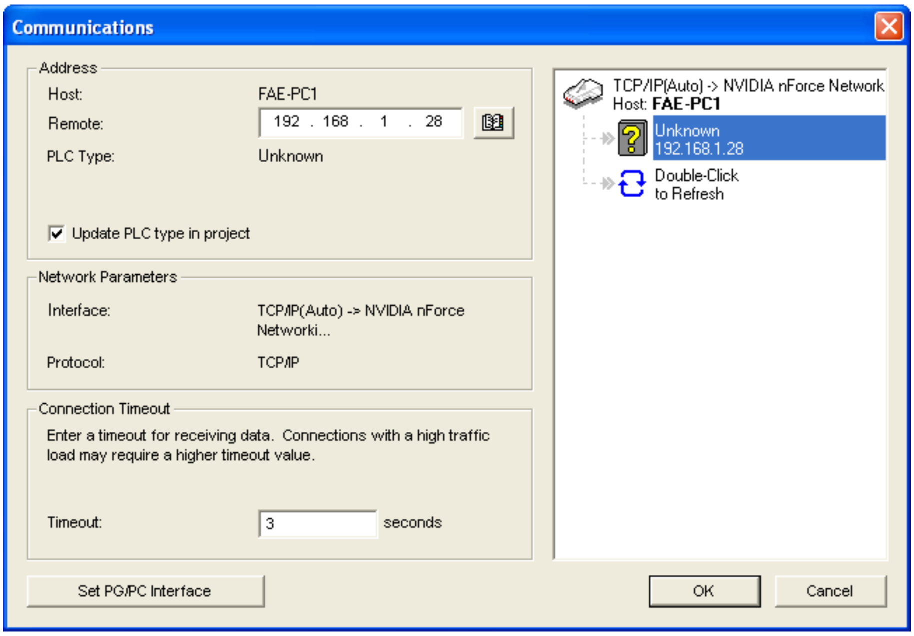
Launch EasyBuilder Pro, go to (System Parameter Settings) » (Device list), and then add SIEMENS S7-200 PPI or S7-300 MPI device. Click (Pass-Through Settings) and the following dialog box appears.



Setting	Description
Disable pass-through	Select this check box to disable pass-through mode. By default this check box is not selected.
Designate client IP	Designate client HMI IP address used in pass-through mode.

29.5.2. S7-200 PPI Connection

Confirm that the HMI used in pass-through communication is started and connected to the network. Launch STEP 7 Micro/Win, open (Communications) dialog box, and then search for the HMI IP address. Connect the HMI to communicate.

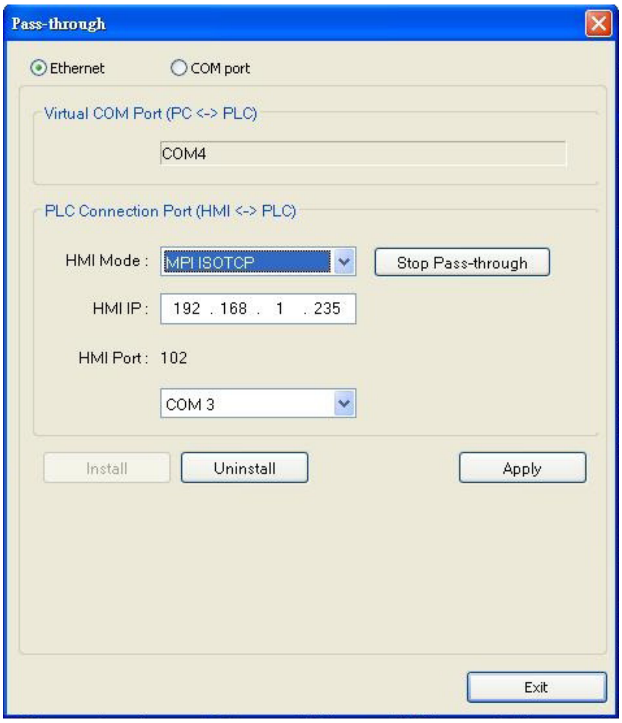


29.5.3. S7-300 MPI Connection

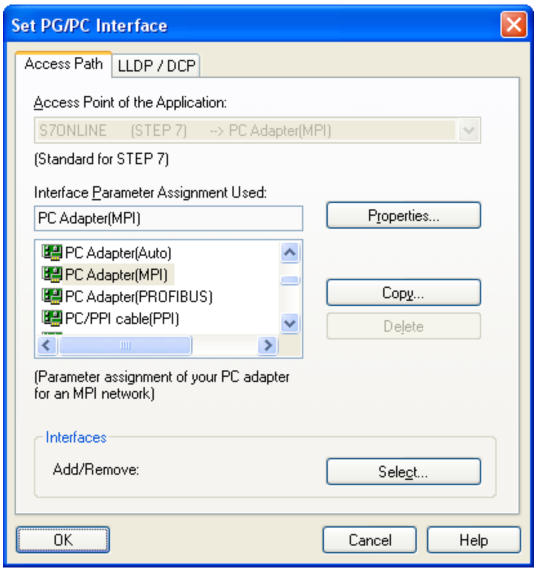
Connect via virtual COM port or Ethernet.

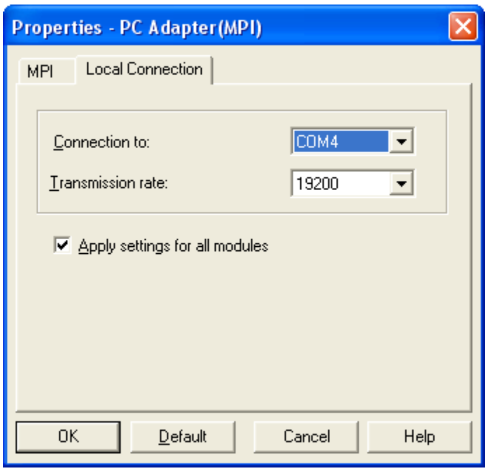
29.5.3.1. Virtual COM Port

1. In Utility Manager run (Pass-Through), in (HMI Mode) select “MPI ISOTCP” to install virtual serial port driver. Set the HMI IP address and the COM port that connects PLC, and then start Pass-through.



2. In STEP 7, go to (Option) » (Set PG/PC Interface). Confirm that the interface used is “PC Adapter(MPI)”, and then click (Properties). Select the same COM port as the virtuel serial port. In the example COM 4 is used.

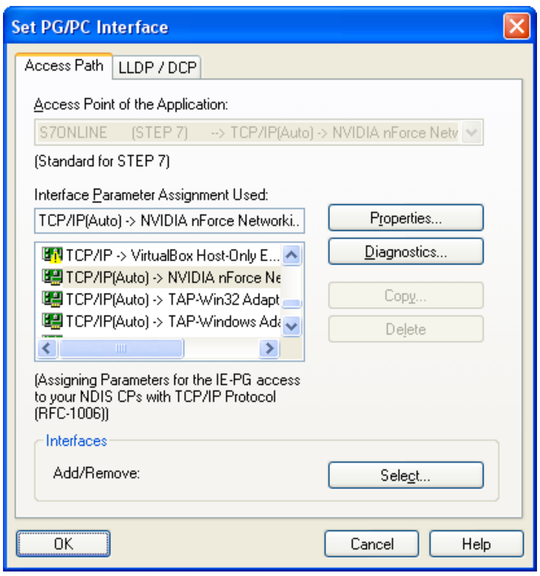




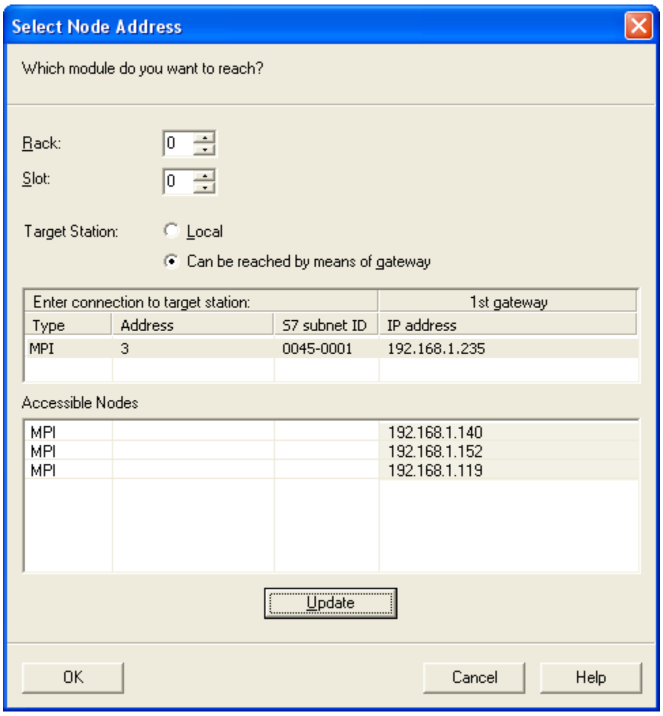
3. When finished, STEP 7 can be used to Upload / Download PLC program via HMI.

29.5.3.2. Ethernet

1. In STEP 7 go to (Option) » (Set PG/PC Interface). As shown in the following figure, select “TCP/IP(Auto) -> the name of the network interface card”.



2. Go to (PLC) » (Update station to PG), in (Target Station) select (Can be reached by means of gateway). From left to right columns enter MPI, PLC station number, S7 Subnet ID, and HMI IP address. When finished, S7 can upload PLC program to STEP 7 via HMI.



29.5.4. Registers of SIEMENS Pass-Through

System registers from LW-10850 to LW-10864 are used to set or indicate pass-through status of SIEMENS devices.

☞ For more information see “22 System Registers”.

During pass-through mode, LW-10864 indicates errors and LW-10865 displays error code. The following table lists the error codes, the description of each code, and the possible reason.
(The client usually refers to STEP 7 PLC program)

Error code	Description	Possible reason
0	Successfully executed	
1	Prohibit client from connecting HMI	HMI is already running pass-through and won’t accept any request from other client.
2	Prohibit client from connecting HMI	When LW-10850 is set to 1, the client IP for connecting HMI is different from the IP specified in LW-10858 ~ LW-10861.
3	Invalid communication protocol	Incorrect setting in LW-10853.
4	Invalid PLC station number	The PLC station number specified in LW-10852 does not exist.
5	Delayed communication	PLC connection failure.
6	Busy communication	PLC does not accept pass-through request, please confirm PLC settings.
7	Invalid pass-through request	Environment setup failure.

30. PROJECT PROTECTION

This chapter explains the settings relevant to project protection.

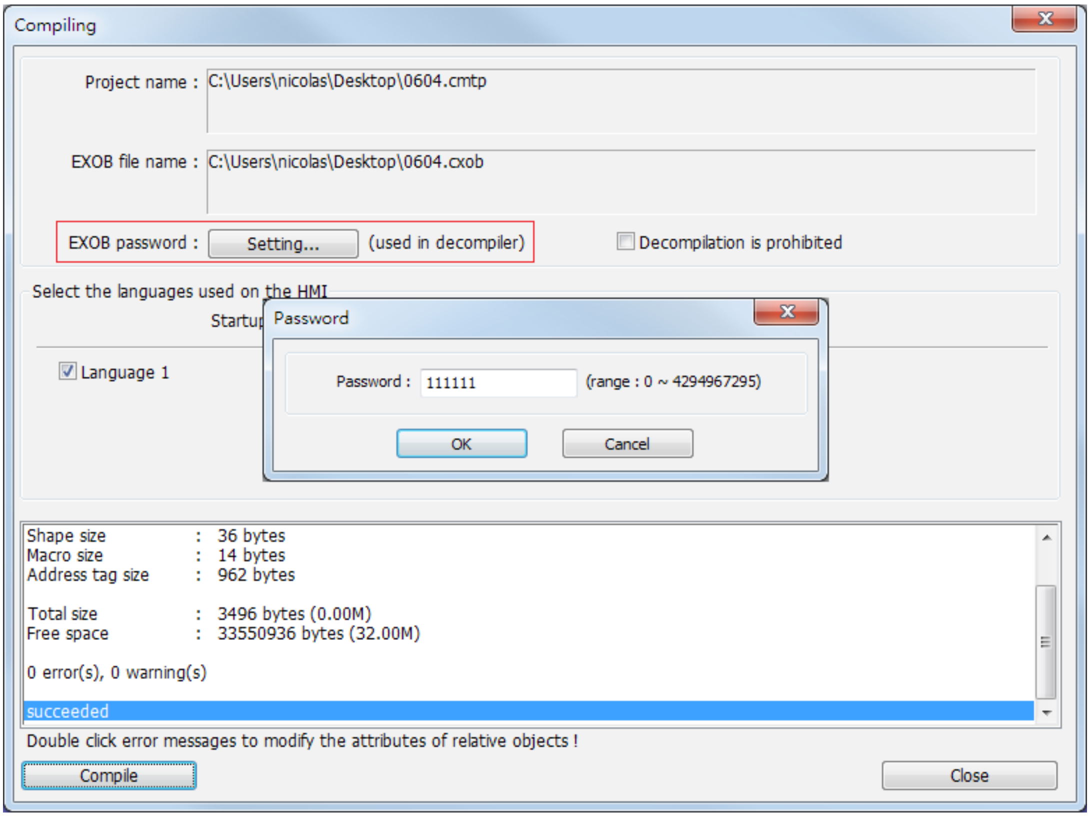
30.1. OVERVIEW

The copyright of program design must be protected. This chapter discusses how to protect the projects by settings in EasyBuilder Pro.

	Note The protected projects cannot be decrypted by the factory since they are encrypted by users, therefore, please remember your password.

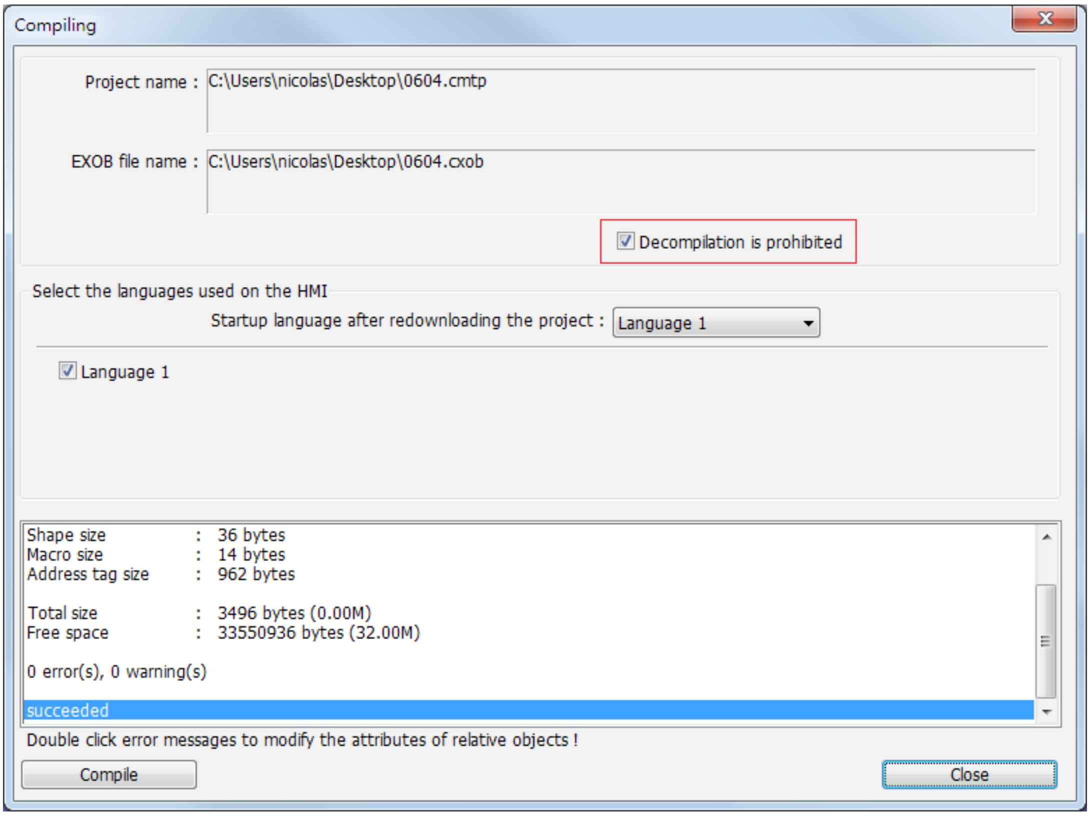
30.2. EXOB PASSWORD

After editing a project (.emtp), users can compile the project to .exob format. The .exob file can be downloaded to HMI. Password can be set to protect the .exob file in (EXOB password) when compiling. (Password range: 0 ~ 4294967295)
A password will be required when attempting to decompile the .exob file back to .emtp file. If the password is entered incorrectly for three times, please restart EasyBuilder Pro.



30.3. DECOMPILATION IS PROHIBITED

After a project (.emtp) is done editing, users can compile the project to .exob format. The .exob file can be downloaded to HMI. If (Decompilation is prohibited) check box is selected when compiling, the setting in (EXOB password) will be ignored. Furthermore, the .exob file cannot be decompiled to .emtp file.

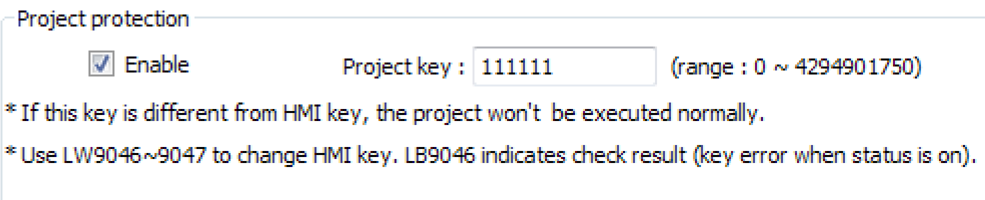


30.4. DISABLE EXOB UPLOAD FUNCTION

EasyBuilder Pro provides a system reserved register (LB-9033). When this register is set ON, the .exob file cannot be uploaded. If attempting to upload an .exob file with this register set ON, the file obtained after uploading is 0 byte, and cannot be decompiled. Please reboot HMI for the changed setting to take effect.

30.5. PROJECT KEY

Projects can be restricted to run on a specific HMI.
The setting is in (System Parameters Settings) » (General) » (Project protection).



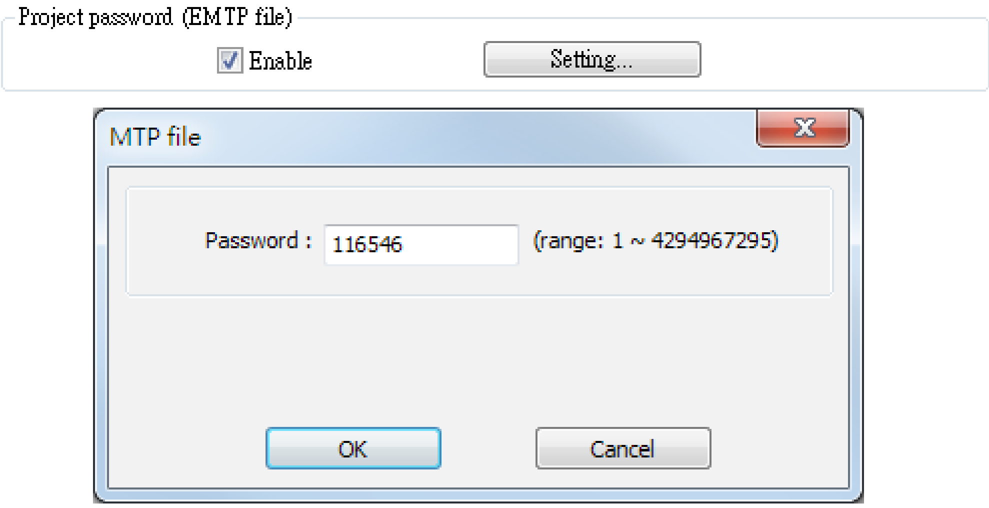
If the (Enable) check box is selected under (Project protection), please set the (Project key) (password range: 0 ~ 4294901750). System registers LW-9046 ~ LW-9047 (32-bit) can be used to set the (HMI key) for HMI. The values in LW-9046 and LW-9047 cannot be read or written by a remote device. The .exob file obtained after compiling can only be executed on HMI when (HMI key) and (Project key) match. If the keys don't match, LB-9046 is set ON. To change (HMI key), please reboot HMI.

	Note When (HMI key) and (Project key) don't match, HMI and PLC cannot communicate.

Download
Click the icon to download the demo project. Please confirm your internet connection.

30.6. EMTP PASSWORD

After a project (.emtp) is done editing, a password can be set to protect the .emtp file. In (System parameter) » (Security) tab, select (Enable) check box under (Project password) and click (Settings) (password range: 1 ~ 4294967295).
The password will be required when attempting to open the .emtp file.



Note
When using “Window Copy” function, if the source file is protected by EMTP password, please enter the correct password for the system to execute window copy.

31. MEMORY MAP

This chapter explains the settings relevant to Memory Map.

31.1. OVERVIEW

Memory Map communication protocol is similar to IBM 3764R, and it is used when the memory data transferred seldom between two devices. When setting the two devices, one is set as Master, and another is Slave. Generally, Master and Slave do not communicate unless the data in the assigned address has changed. Once the data is synchronized, the communication will stop. The purpose of Memory Map is to keep the consistency of the assigned part of data between two devices (Master and Slave).
The corresponding addresses of Master and Slave devices should have the same property as MW (MB) address type. The size of MW (MB) in HMI is 10,000 words.
MB and MW indicate the same area of memory, for example, MB0~MBf correspond to the bits of MW0, MB10~MB1f correspond to MW1, as shown in the following table:

Device type	Format	Range
MB	DDDDh	DDDD:0~4095 h:0~f(hex)
MW	DDDD	DDDD:0~9999

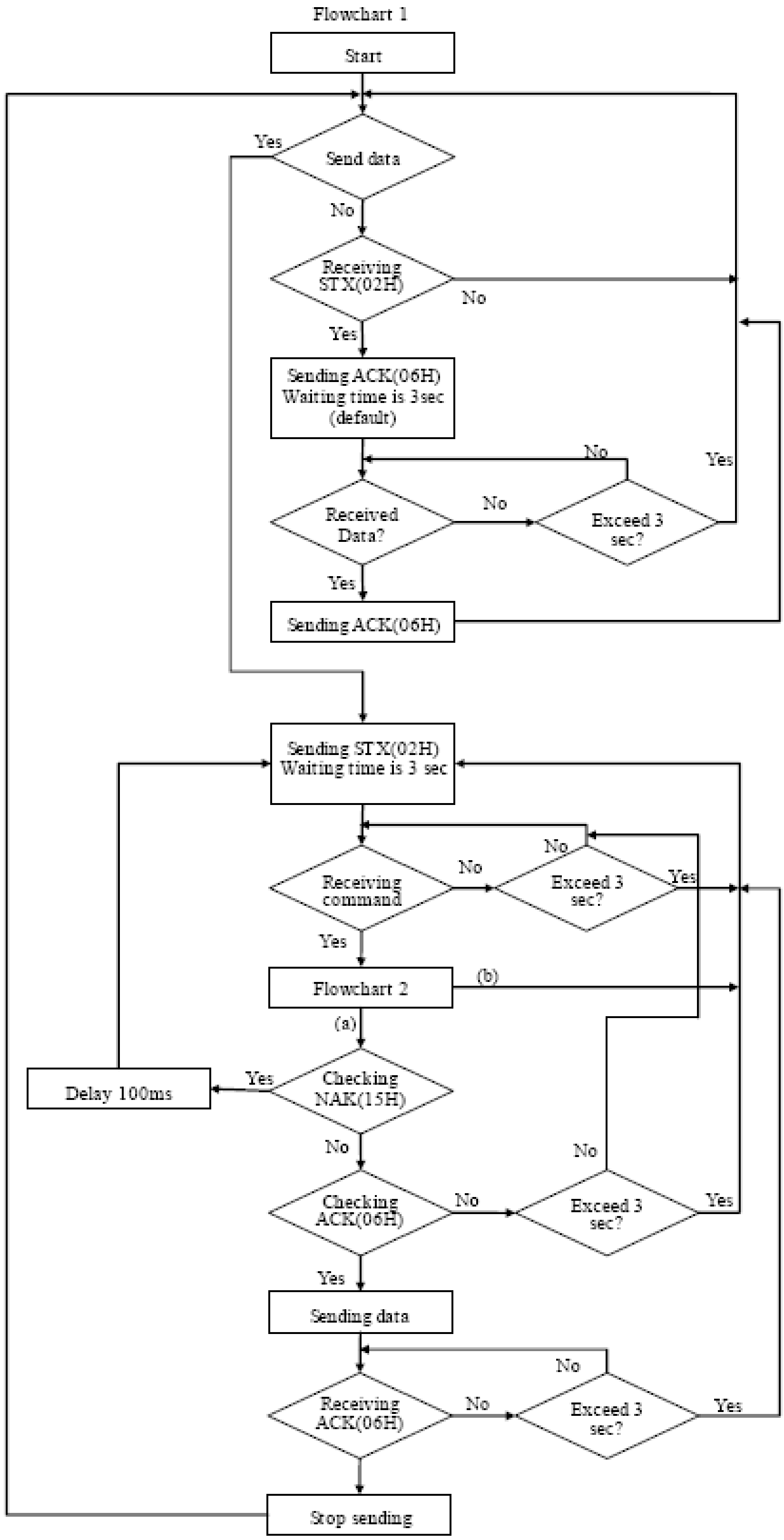
31.2. PIN SETTINGS

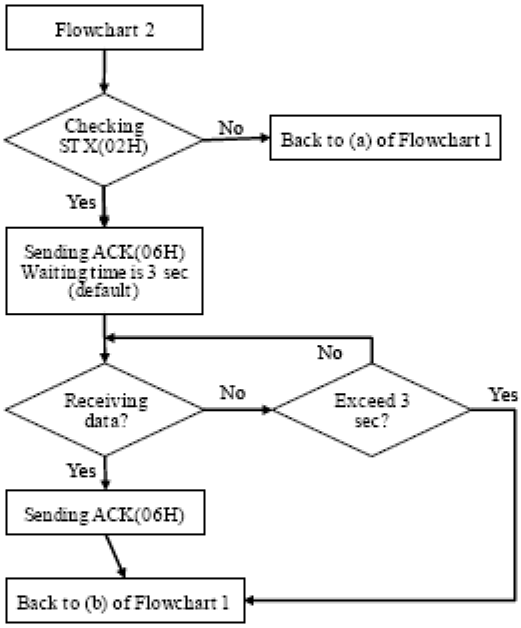
When using Memory Map communication protocol, the Master and Slave must have the same communication parameters. The wiring is shown in the following table:
(the # will be distinct depends on the type of PLC or controller.)

COM Port	RS232	
Device	Master	Slave
Pin mapping	TX(#)	RX(#)
	RX(#)	TX(#)
	GND(#)	GND(#)

COM Port	RS485 (4W)	
Device	Master	Slave
Pin mapping	TX+(#)	RX+(#)
	TX-(#)	RX-(#)
	RX+(#)	TX+(#)
	RX-(#)	TX-(#)
	GND(#)	GND(#)

31.3. COMMUNICATION FLOWCHART





Note

- Flowchart 2 works for slave but not master.
- STX: start of text, ACK: Acknowledge, NAK: negative Acknowledge.

31.4. ADDRESS TYPES

There are two address types, MB and MW.
The format of the commands that controls MB are listed in the following table:

MB Commands		
Offset (byte)	**Format**	**Description**
0	0x02	The operating sign to MB
1 **2**	0x## 0x##	Address (Low byte) Bit Address (High byte) For example: MB-18 = 1*16 + 2 = 18 = 0x12 and 0x00
3	0x00 (or 0x01)	The data in MB address. (Bit type, must be 0 or 1)
4, 5	0x10, 0x03	Stop sign
6	0x##	The checksum. Calculate XOR from offset 0 to 5.

The format of the commands that controls MW are listed in the following table:

MW Commands		
Offset (byte)	**Format**	**Description**
0	0x01	The operating sign to MW
1 **2**	0x## 0x##	Address (Low byte) Bit Address (High byte) If the address includes 0x10, insert another 0x10 after it and all offsets after that are increased by 1. For example: 0x10, 0x04 will become 0x10,0x10,0x04
3	0x##	Number of sending bytes (To control a word, the number of bytes must be even). If the number of bytes is 0x10, insert another 0x10 after it and all offsets after that are increased by 1.
4 to 4+n-1	0x##(L),0x##(H) 0x##(L),0x##(H) …	The address that the first and second bytes correspond to is the initial address. “n” is the number of bytes. If the data includes 0x10, insert another 0x10 after it and the “Number of sending bytes” (offset 3) remains the same, but n = n + 1. Same thing applies to other 0x10 data.
4+n, **4+n+1**	0x10 0x03	End sign
4+n+2	0x##	The checksum. Calculate XOR from all above.

31.4.1. Communication Examples

Example 1

If Master sets the data of MW-3 to 0x0a, Master will build communication with Slave immediately due to the data changed, so Slave will update its MW-3 to 0x0a, the procedure is:

1. Master sends STX(0x02h).
2. Slave receives STX(0x02h) from Master, and sends ACK(0x06h) to Master.
3. Master receives ACK(0x06h) from Slave.
4. Master sends 0x01,0x03,0x00,0x02,0x0a,0x00,0x10,0x03,0x19, as shown in the following table:

Offset (byte)	Format	Description
0	0x01	The operating sign for MW
1 **2**	0x03 0x00	Address(Low byte) Bit Address (High byte)
3	0x02	The number of bytes sent (MW-3= two bytes).
4, 5	0x0a, 0x00	Data in MW-3 is 0x0a and 0x00
6, 7	0x10, 0x03	End sign
8	0x19	The checksum 0x01^0x03^0x00^0x02^0x0a^0x00^0x10^0x03=0x19

5. Slave receives data from Master and then sends ACK(0x06h).
6. Master receives ACK(0x06h) from Slave.

When finish communicating, Master sends the updated data in MW to Slave, and Slave synchronizes its MW data with Master.

Example 2

If the data includes 0x10; please notice the change in data format.
If MW-10 of Slave is set to 0x10, Slave will build communication with Master immediately, and Master will update its MW-10 to 0x10, the procedure is:

1. Slave sends STX(0x02h)
2. Master receives STX(0x02h) from Slave, and sends ACK(0x06h) to Slave.
3. Slave receives ACK(0x06h) from Master
4. Slave sends 0x01,0x10,0x10,0x00,0x02,0x10,0x10,0x00,0x10,0x03,0x10 as shown in the following table:

Offset (byte)	Format	Description
0	0x01	The operating sign to MW
1 **2** **3**	0x10 0x10 0x00	Address(Low byte) Insert 0x10 Bit Address (High byte)
4	0x02	The number of bytes sent (MW-10= two bytes).
5 **6** **7**	0x10 0x10 0x00	0x10 is the low byte in MW-10 Insert 0x10 0x00 is the high byte
8 **9**	0x10 0x03	End sign
10	0x10	The checksum, 0x01^0x10^0x10^0x00^0x02^0x10^0x 10^0x00^0x10^0x03=0x10

5. Master receives data from Slave and sends ACK(0x06h) to Slave.
6. Slave receives ACK(0x06h) from Master.

Slave sends the updated data in MW to Master, and Master synchronizes its MW data with Slave.

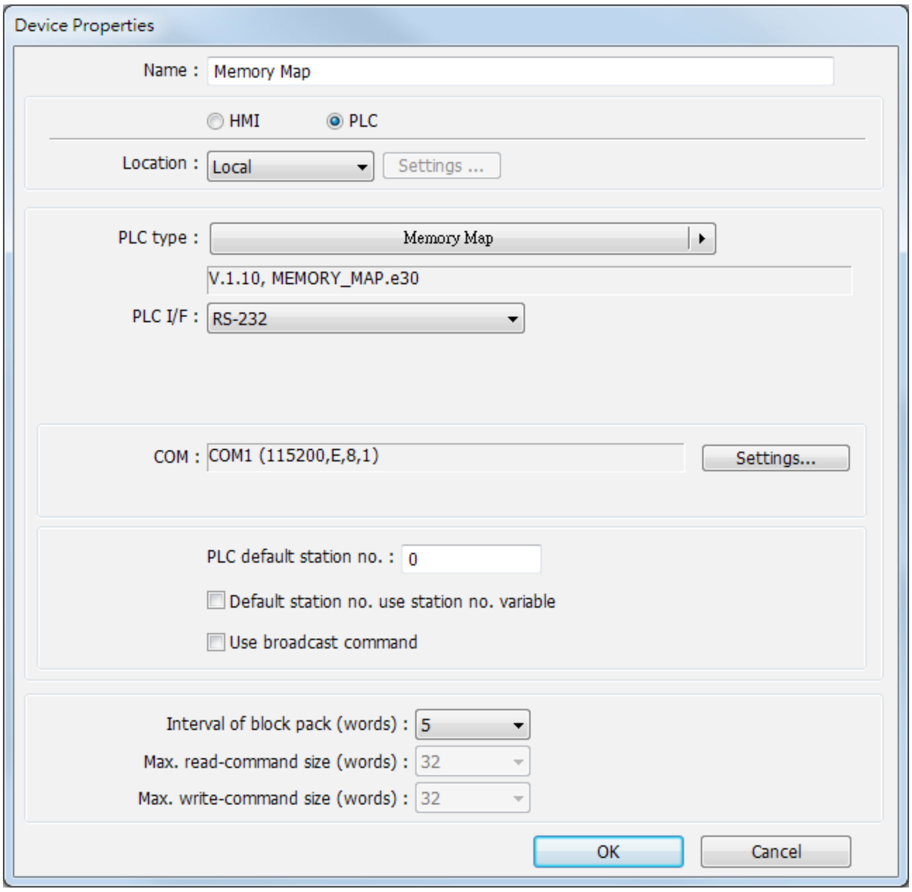
31.5. SETTINGS

The following explains how to connect two HMIs using Memory Map protocol.

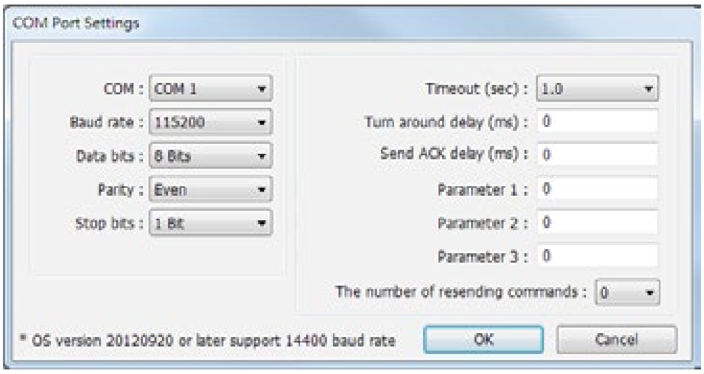
Note
If the type of these two HMIs are different, please create different project files, or, after setting the first HMI, directly change to the type of the second HMI in (Edit) » (System Parameter Settings) » (Model), and then compile and download the project to the second HMI.

31.5.1. Steps to Add a Memory Map Device

1. Launch EasyBuilder Pro, select (New), and the model of HMI.
2. Click (Edit) form the main menu, click (System Parameter Settings), and select (Device) tab, then click (New) to add a new device.
3. In the (Name) field enter “Memory Map”, and then select (PLC), set the (Location) to (Local).
4. Set (PLC type) to (Memory Map), and set (PLC I/F) to (RS232).



5. Click (Settings), and the setting is shown in the following figure.



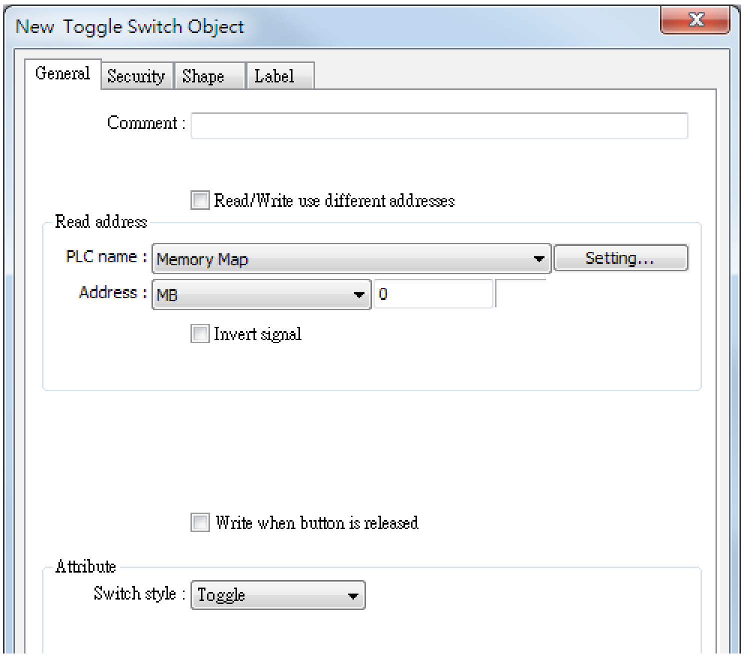
6. After setting the COM port click (OK).
7. Click (OK) to finish setting.

	Note
	▪ Memory Map in MT500 is divided into (Memory Map_Master) and (MemoryMap_Slave); please refer to the relevant manual. ▪ For eMT3000 and MT8000 Series, select (Memory Map) in the PLC type setting. ▪ (Data bit) must set to 8 bits. ▪ All the settings of the two HMIs must be the same.

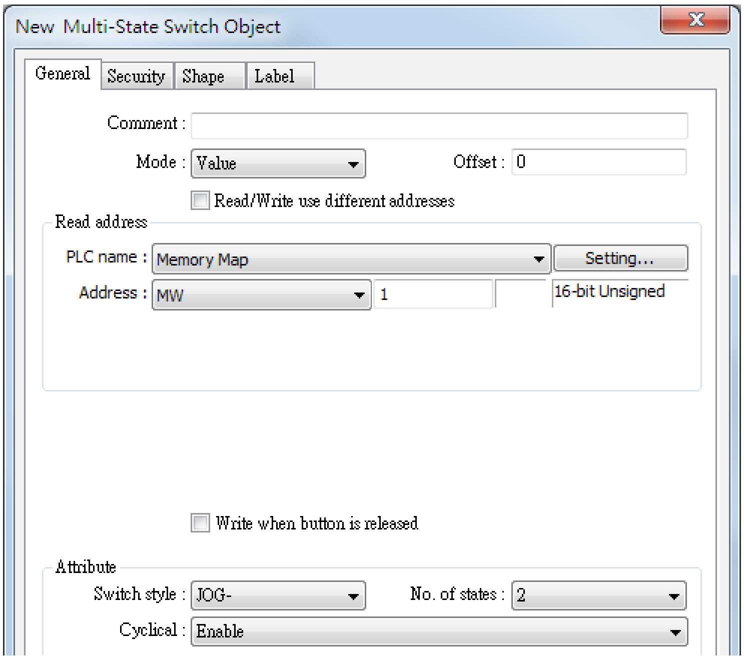
31.5.2. Object Settings

Add two objects in window no. 10, a Toggle Switch and a Multi-state Switch:
Create a Toggle Switch Object as shown in the following steps.

1. Set the (PLC name) of read address and write address to (Memory Map).
2. Set (Address) to MB-0.
3. Set (Switch style) to (Toggle). (The picture and label of the object can be selected).

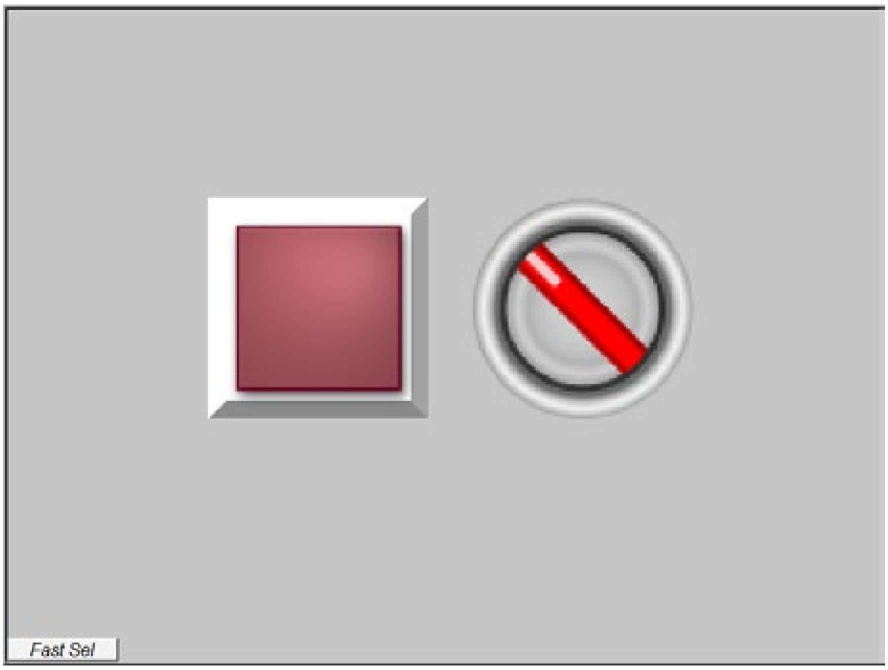


Create a Multi-state Object as shown in the following steps.
1. Set the (PLC name) of read address and write address to (Memory Map).
2. Set (Address) to MW-1.
3. Set (Cyclical) to (Enable). (The picture and label of the object can be selected).



31.5.3. Executing the Settings

Compile and download the same project to HMI 1 and HMI 2.



When pressing the button in one of the HMIs, the status of another one will also be changed.
The way to connect a HMI with a controller is similar to the example above. The data in the same addresses of the two devices are kept identical.

32. FTP SERVER APPLICATION

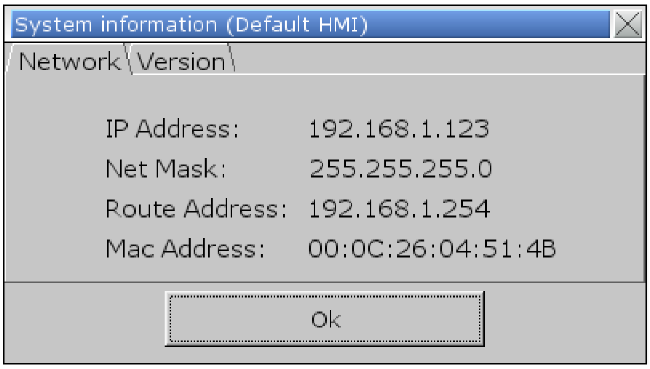
This chapter explains how to use FTP Server.

32.1. OVERVIEW

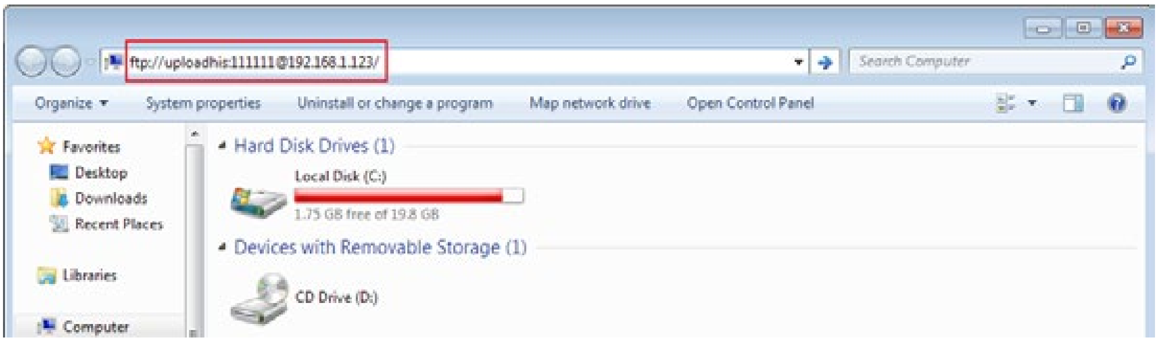
Apart from saving the history data from an HMI to your PC by using SD card, USB disk or EasyPrinter, the FTP Server can also do the backup. After downloading a project to the HMI, the FTP Server can be used to backup or update the history data and the recipe data, but not able to delete those data.

32.2. STEPS TO LOG IN FTP SERVER

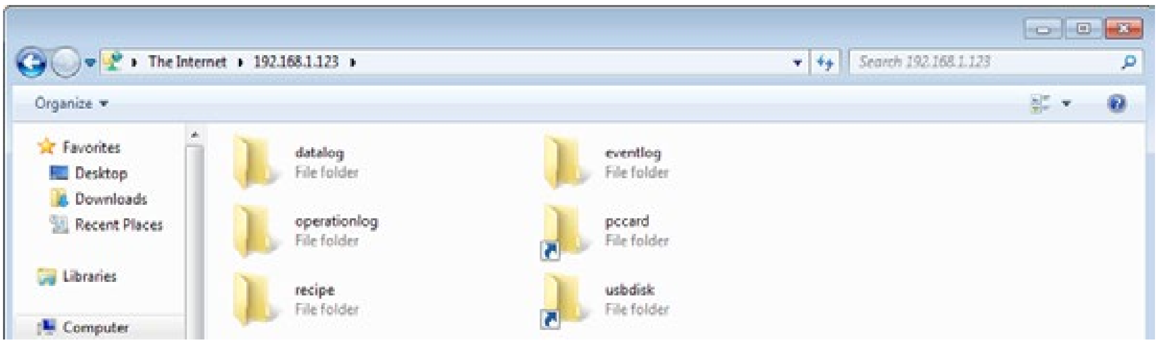
1. Before logging in FTP Server, please check the HMI IP address.



2. On PC, enter the HMI IP address: ftp://192.168.1.123/ (example), then log in by the user name: uploadhis, and enter the HMI (history upload password) (if not changed, the default password is 111111). Or, directly enter "ftp://uploadhis:111111@192.168.1.123/"



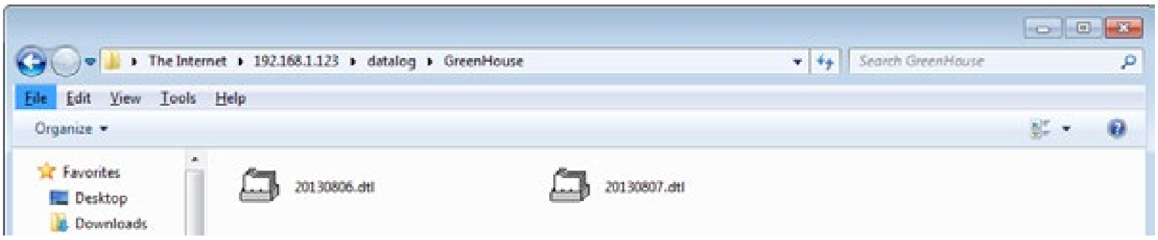
3. After entering the IP address,the ftp address: ftp://192.168.1.123 and its folders are shown as below:



32.3. BACKUP HISTORY DATA AND UPDATE RECIPE DATA

Steps to backup Data Sampling records

1. Click “datalog” folder to view the files.
2. Click the file to check the records.
3. Copy and paste the needed files to your PC.

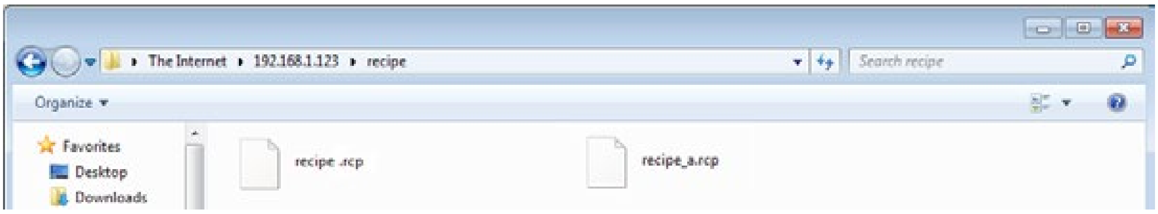


Steps to backup Event (Alarm) Log records

1. Click “eventlog” folder to view the files.
2. Copy and paste the needed files to your PC.

Steps to backup or update Recipe records

1. Click “recipe” folder to view the files.
2. Copy and paste the needed files to your PC.



Note

- Since the recipe data is automatically saved per minute, after updating “recipe.rcp” or “recipe_a.rcp”, the HMI must be rebooted within one minute or the saving will be failed.
- The HMI can be rebooted by the system registers: (LB-9047) (reboot HMI) and (LB9048) (reboot HMI protection). You can set (LB-9048) ON first, and then set (LB-9047) ON to reboot the HMI.

33. EASYDIAGNOSER

This chapter explains how to use EasyDiagnoser.

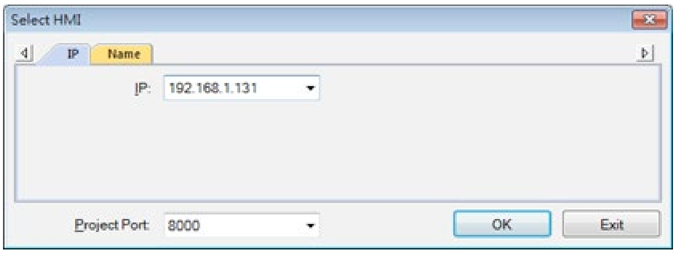
33.1. OVERVIEW

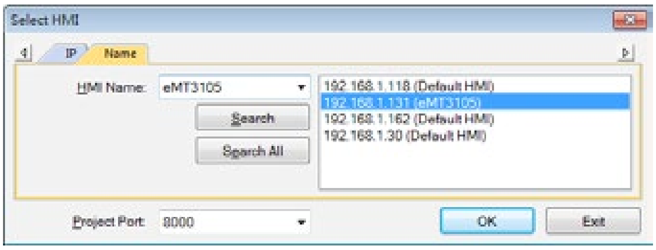
EasyDiagnoser is a tool for detecting the error in the communication of HMI with PLC.

33.2. CONFIGURATION

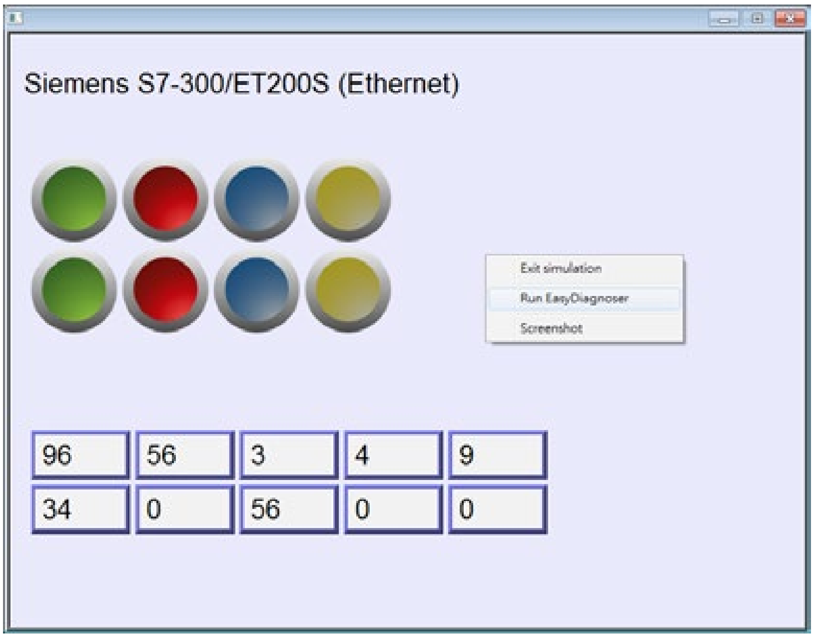
The following steps explain how to configure EasyDiagnoser.

1. Open Utility Manager and click EasyDiagnoser.
2. Set the HMI IP address. Enter IP address or click (Search All), and then enter (Project Port).

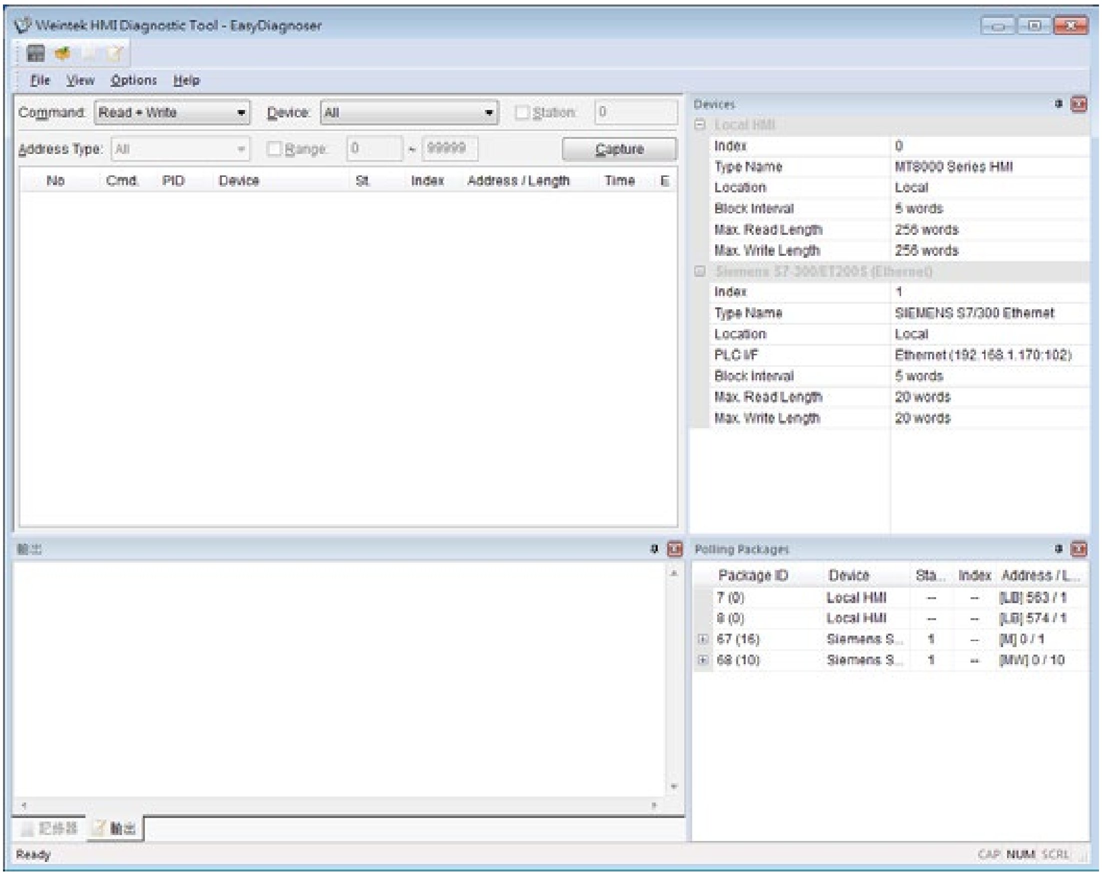




- Or, during On-line simulation, right click and select (Run EasyDiagnoser) to open EasyDiagnoser. EasyDiagnoser will monitor the communication between PC and PLC



3. When finished, click (OK), EasyDiagnoser dialog box is shown in the following figure.



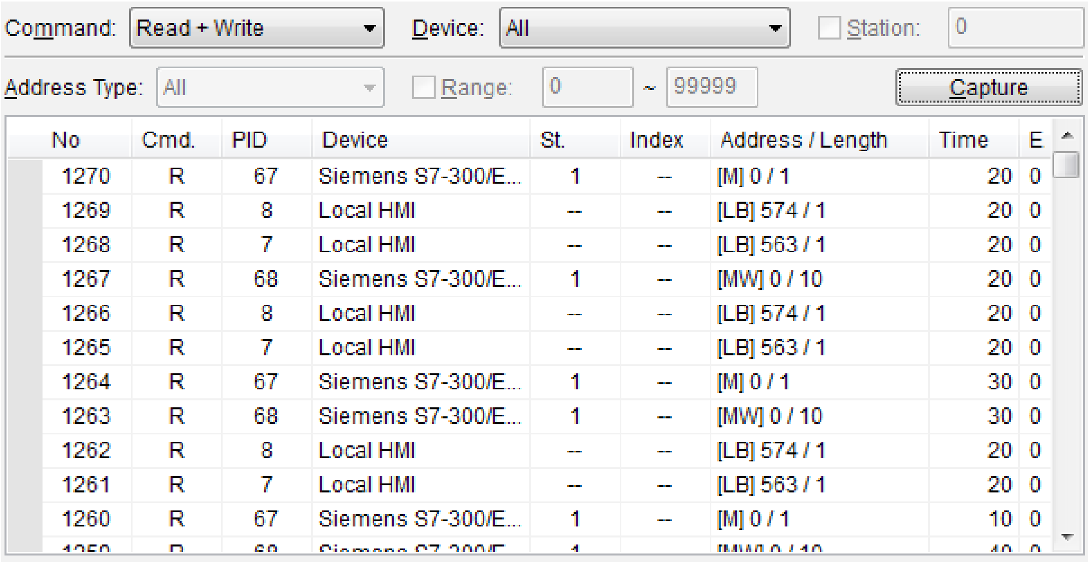
33.3. EASYDIAGNOSER SETTINGS

33.3.1. Main Menu

Item	Description
File	**Save As** The communication data can be saved as .xls file which can be opened by Excel. **Exit** Exits current file.
View	**Device Bar** displays Device window. **Package Bar** displays Package window. **Logger Bar** displays Logger window. **Output Bar** displays Output window.
Options	**Toolbars** displays toolbar icons of Device Bar, Package Bar, Logger Bar, and Output Bar. **Status Bar** displays information of CAP, NUM, and SCRL at the bottom of EasyDiagnoser window. **Update Package List** displays the Polling Package information of current page. **Show Object ID (HMI**) shows the ID of the objects on HMI as shown in the following figure.  **Clear Activity List** clears all the information recorded during communication.
Help	Displays EasyDiagnoser version information.

33.3.2. Activity Area

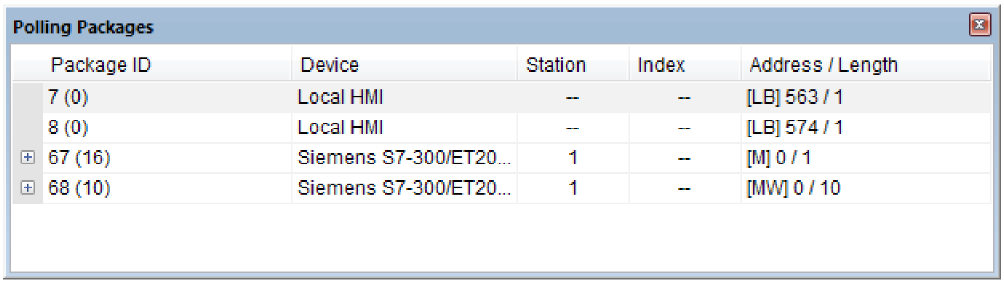
In the activity area, users can observe the communication between HMI and PLC.



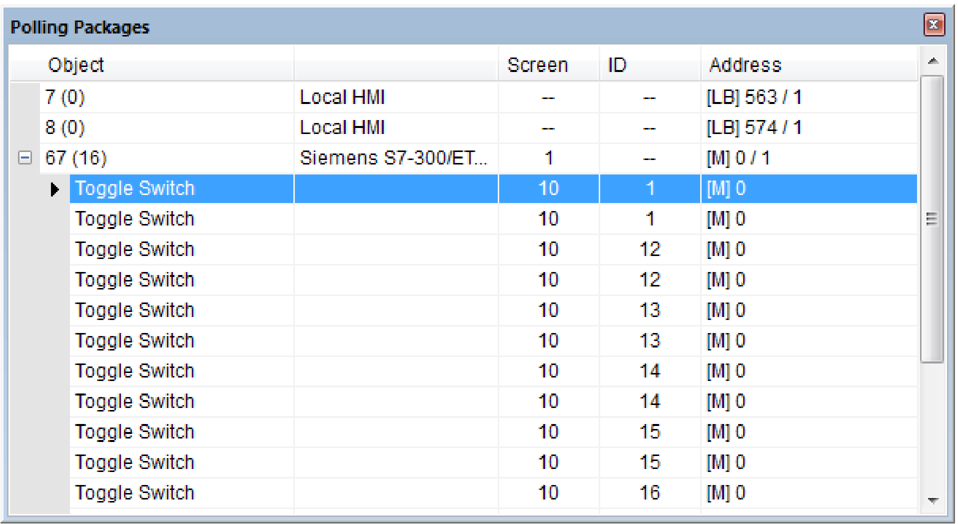
No	Cmd.	PID	Device	St.	Index	Address / Length	Time	E
1270	R	67	Siemens S7-300/E...	1	--	[M] 0 / 1	20	0
1269	R	8	Local HMI	--	--	[LB] 574 / 1	20	0
1268	R	7	Local HMI	--	--	[LB] 563 / 1	20	0
1267	R	68	Siemens S7-300/E...	1	--	[MW] 0 / 10	20	0
1266	R	8	Local HMI	--	--	[LB] 574 / 1	20	0
1265	R	7	Local HMI	--	--	[LB] 563 / 1	20	0
1264	R	67	Siemens S7-300/E...	1	--	[M] 0 / 1	30	0
1263	R	68	Siemens S7-300/E...	1	--	[MW] 0 / 10	30	0
1262	R	8	Local HMI	--	--	[LB] 574 / 1	20	0
1261	R	7	Local HMI	--	--	[LB] 563 / 1	20	0
1260	R	67	Siemens S7-300/E...	1	--	[M] 0 / 1	10	0

Item	Description
Command	**Read + Write** Displays Read and Write information in activity area. **Read** Displays only Read information in activity area. **Write** Displays only Write information in activity area.
Device	**All** Displays the information of Local HMI and PLC. ▪ If command is set to Read + Write, the Read and Write information of Local HMI and PLC will be displayed in the activity area ▪ If command is set to Read, the Read information of Local HMI and PLC will be displayed in the activity area ▪ If command is set to Write, the Write information of Local HMI and PLC will be displayed in the activity area **Local HMI** Displays the information of Local HMI. ▪ If command is set to Read + Write, the Read and Write information of Local HMI will be displayed in the activity area ▪ If command is set to Read, the Read information of Local HMI will be displayed in the activity area ▪ If command is set to Write, the Write information of Local HMI will be displayed in the activity area **PLC** Displays the information of PLC. ▪ If command is set to Read + Write, the Read and Write information of PLC will be displayed in the activity area ▪ If command is set to Read, the Read information of PLC will be displayed in the activity area ▪ If command is set to Write, the Write information of PLC will be displayed in the activity area
Station	Selects the PLC station number to be displayed. (This function is disabled when selecting (All) in (Device)).
Address type	Selects all or a preferred address type to be displayed. (This function is disabled when selecting (All) in (Device)).
Range	Sets the range of address types. (This function is disabled when selecting (All) in (Device)).
Capture	Click to start/stop capturing the communication message.
Error	Please see “33.4 Error Code”.

33.3.3. Polling Packages



Item	Description
Package ID	Uses the Package ID to check the error of the object.
Device	Displays HMI and PLC type.
Station	Displays PLC station number.
Index	Displays the index register numbers of the objects.
Address / Length	Displays the device type and the size of the package (in words).



Item	Description
Object	Check the object in the package.
Screen	The window in the project where the object is placed.
ID	The ID number of the object.
Address	The address of the object.

Note

- Click (Package ID), the device station number will be displayed in the 3rd column.

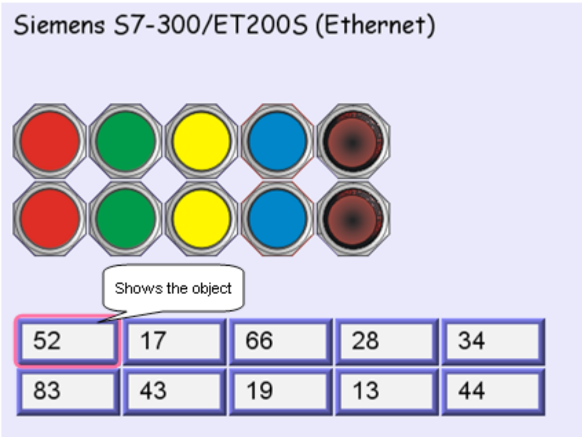
Polling Packages

Package ID	Device	Station	Index	Address / Length
7 (0)	Local HMI	--	--	[LB] 563 / 1
8 (0)	Local HMI	--	--	[LB] 574 / 1
67 (16)	Siemens S7-300/ET2...	1	--	[M] 0 / 1
68 (10)	Siemens S7-300/ET2...	1	--	[MW] 0 / 10

- Double click (Package ID) then select (object) to display the position of the object.
- For example, select (Numeric Input) and the screen no. displays 10.
- This shows that this object is in window no. 10 in the project and will be marked with pink frame on HMI as shown in the following figures

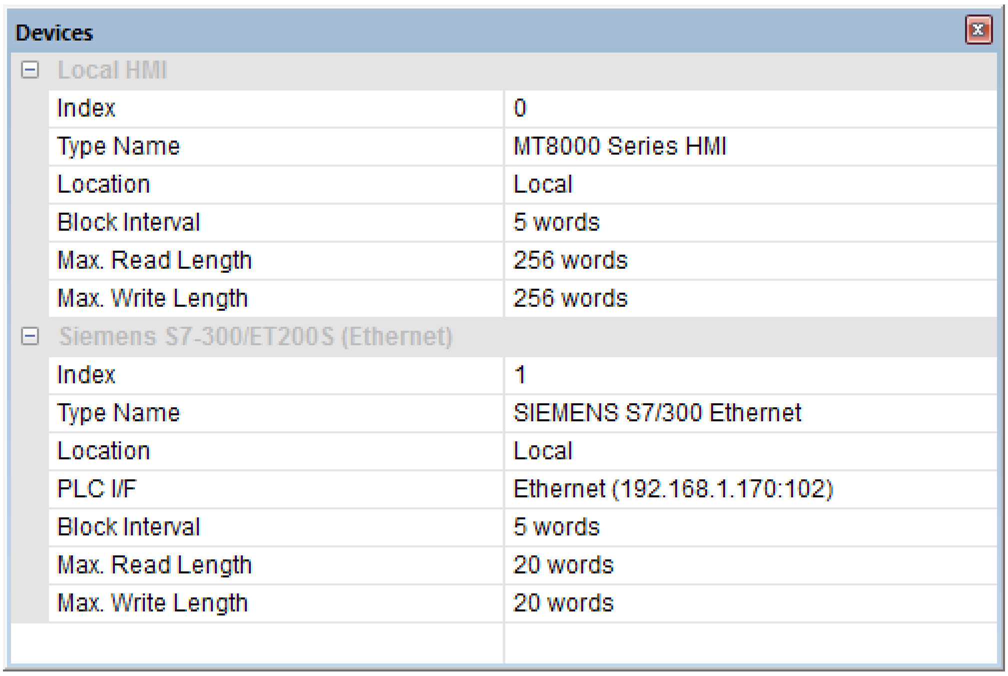
Polling Packages

Object		Screen	ID	Address
7 (0)	Local HMI	--	--	[LB] 563 / 1
8 (0)	Local HMI	--	--	[LB] 574 / 1
67 (16)	Siemens S7-300/ET20...	1	--	[M] 0 / 1
68 (10)	Siemens S7-300/ET20...	1	--	[MW] 0 / 10
Numeric Input		10	2	[MW] 0
Numeric Input		10	3	[MW] 2
Numeric Input		10	4	[MW] 4
Numeric Input		10	5	[MW] 6
Numeric Input		10	6	[MW] 8
Numeric Input		10	7	[MW] 10
Numeric Input		10	8	[MW] 12
Numeric Input		10	9	[MW] 14
Numeric Input		10	10	[MW] 16
Numeric Input		10	11	[MW] 18



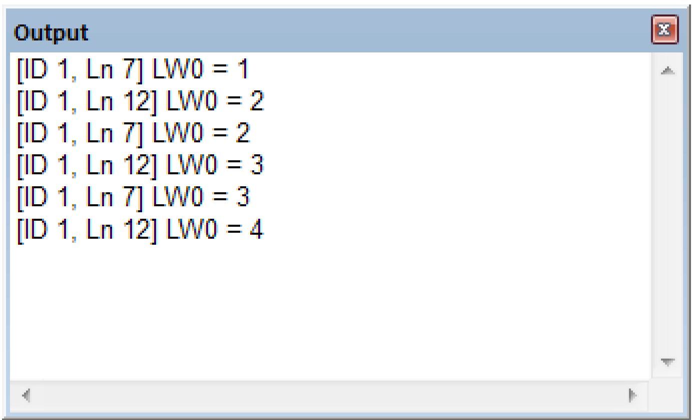
33.3.4. Devices

Displays the information of HMI and PLC.



33.3.5. Output (Macro debug)

With Macro Trace function, the executing status of Macro can be seen.
In the illustration below, for (ID 1, Ln 7) and (ID 1, Ln 12)
ID 1 represents Macro name.
Ln 7 and Ln 12 represent that data are in the 7th and 12th line of Macro.



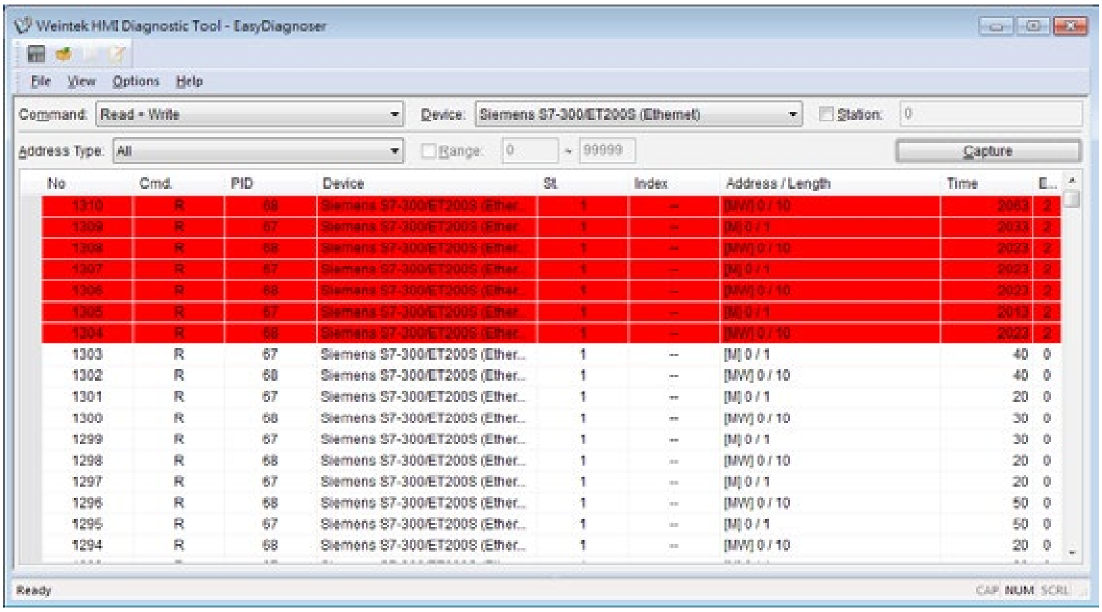
For more information, see “18 Macro Reference”.

33.4. ERROR CODE

In the activity area, users can find the reason of error through the error codes listed below.

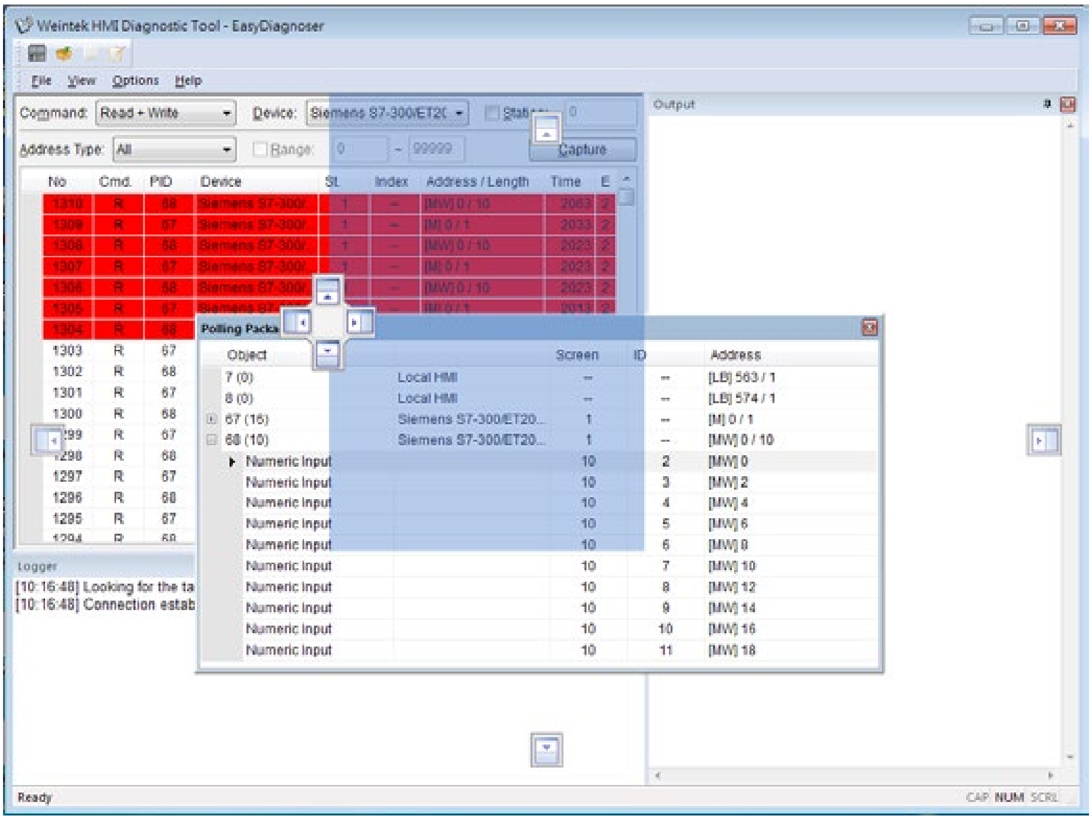
- 0: normal
- 1: time out
- 2: fail error
- 12: ignore

When error occurs, error message will be shaded red as shown in the following figure.
The error code is 1 since the PLC is disconnected with HMI.
The error code is 12 since "PLC No Response" message window is shown.



33.5. WINDOW ADJUSTMENT

Users can drag or use the smart docking icons in editing window to place the windows to a desired position.



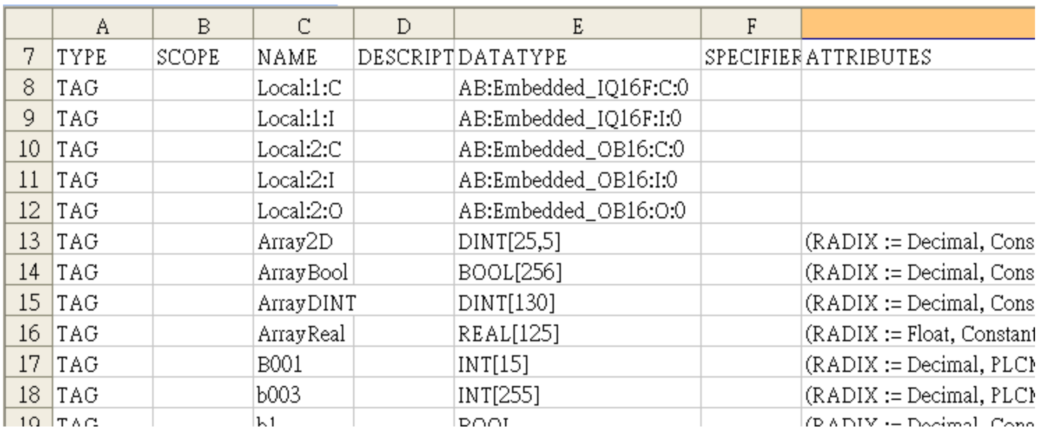
	Note EasyDiagnoser doesn't support Siemens S7/1200 (Ethernet) and Allen-Bradley Ethernet/IP (CompactLogix/ControlLogix) – Free Tag Names since both of the PLCs use tag.

34. ROCKWELL ETHERNET/IP FREE TAG NAMES

This chapter explains how to use Rockwell EtherNet / IP Free Tag Names.

34.1. OVERVIEW

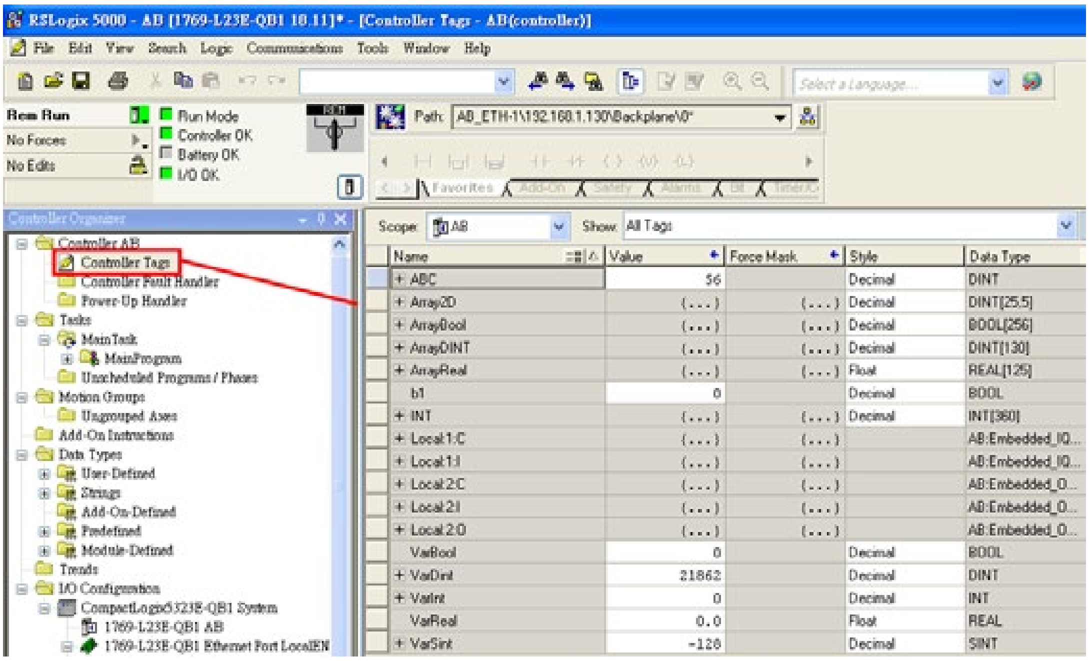
When using Rockwell EtherNet/IP Free Tags (CompactLogix/ControlLogix) driver, the User-defined tag in RSLogix5000 can be exported to .csv file, and then imported to EasyBuilder Pro.
However, the tags in User-Defined, Predefined and Module-Defined will not be exported.
Structure Editor in EasyBuilder Pro is then used for importing and editing tags of data structures in User-Defined, Predefined and Module-Defined.



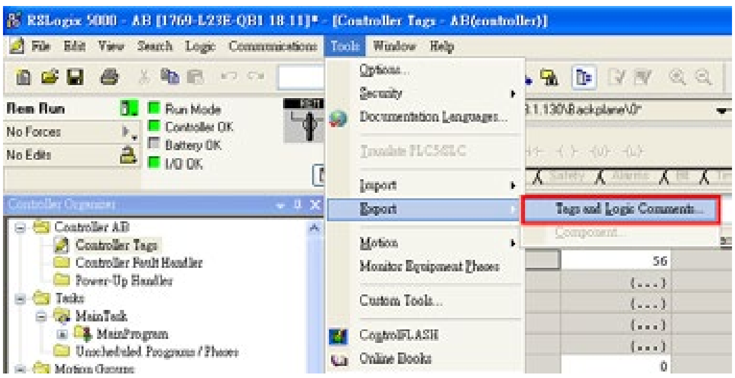
	A	B	C	D	E	F	
7	TYPE	SCOPE	NAME	DESCRIPT	DATATYPE	SPECIFIER	ATTRIBUTES
8	TAG		Local:1:C		AB:Embedded_IQ16F:C:0		
9	TAG		Local:1:I		AB:Embedded_IQ16F:I:0		
10	TAG		Local:2:C		AB:Embedded_OB16:C:0		
11	TAG		Local:2:I		AB:Embedded_OB16:I:0		
12	TAG		Local:2:O		AB:Embedded_OB16:O:0		
13	TAG		Array2D		DINT[25,5]		(RADIX := Decimal, Cons
14	TAG		ArrayBool		BOOL[256]		(RADIX := Decimal, Cons
15	TAG		ArrayDINT		DINT[130]		(RADIX := Decimal, Cons
16	TAG		ArrayReal		REAL[125]		(RADIX := Float, Constant
17	TAG		B001		INT[15]		(RADIX := Decimal, PLCN
18	TAG		b003		INT[255]		(RADIX := Decimal, PLCN
19	TAG		b1		BOOL		(RADIX := Decimal, Cons

34.2. STEPS TO IMPORT USER-DEFINED AB TAG CSV FILE TO EASYBUILDER PRO

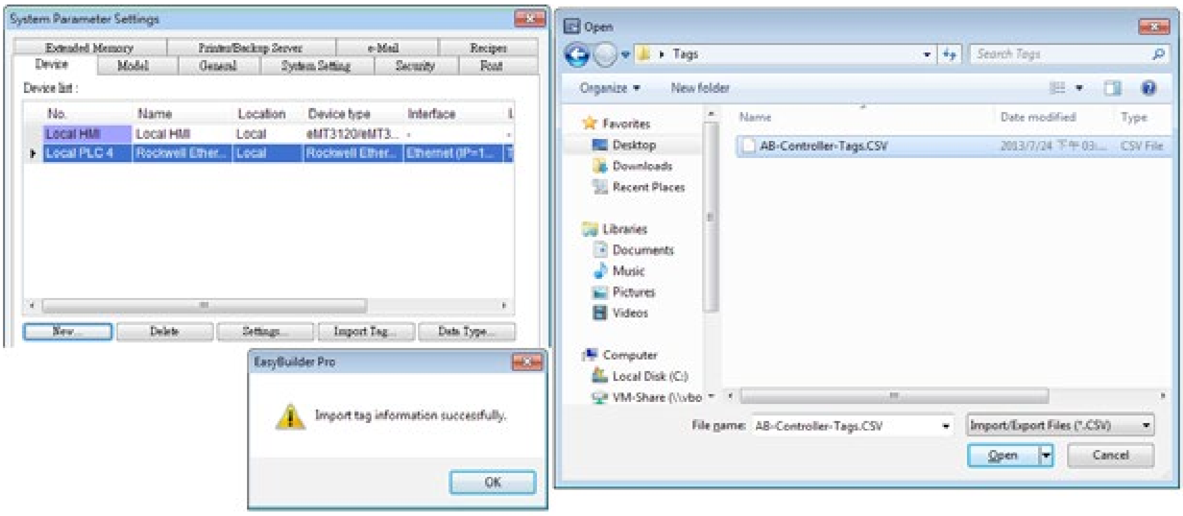
1. Create Tags in RSLogix5000.



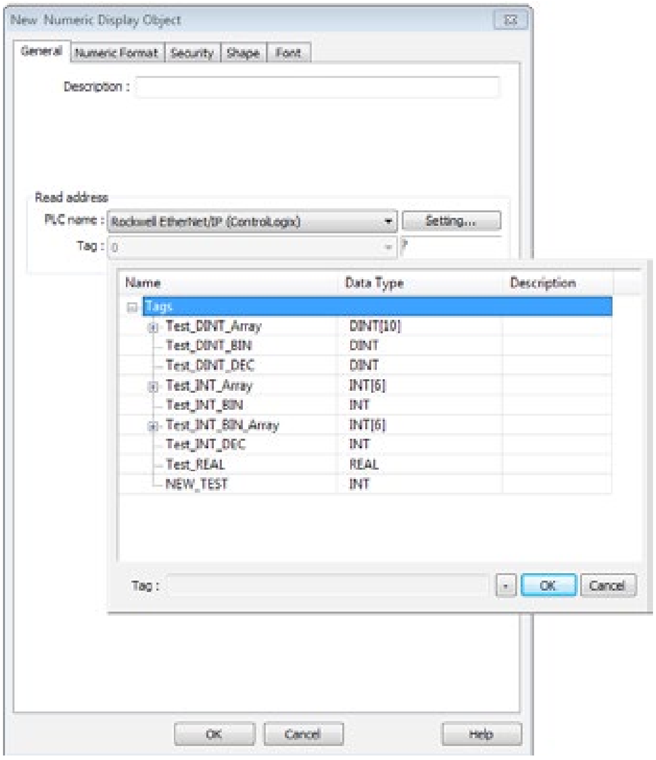
2. Export Tags to .csv file.



3. In EasyBuilder Pro, add Rockwell EtherNet/IP-Tag (CompactLogix/ControlLogix) driver.
   Enter PLC IP address and click (Import Tag).



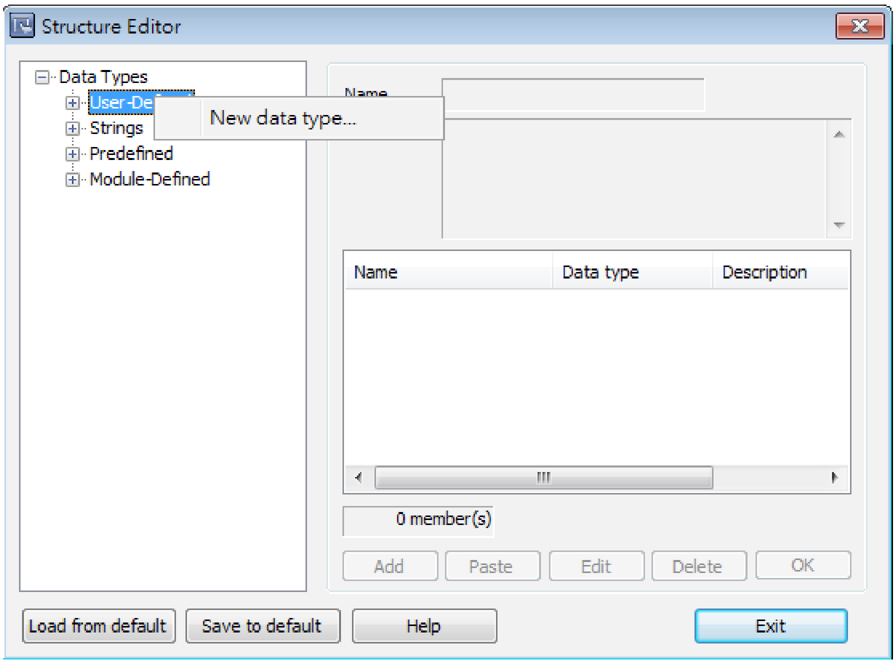
4. In the object setting dialog, select the PLC type, and select a controller tag.



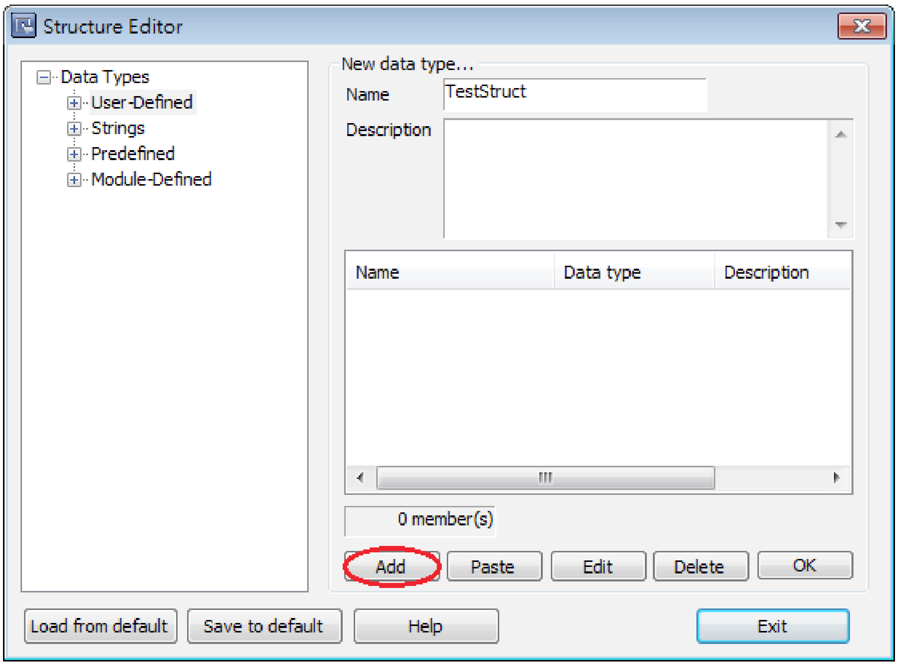
34.3. STEPS TO ADD A NEW DATA TYPE

Structure Editor is in the installation directory of EasyBuilder Pro. Double-click Structure Editor.exe and the editor window will appear. See the following steps.

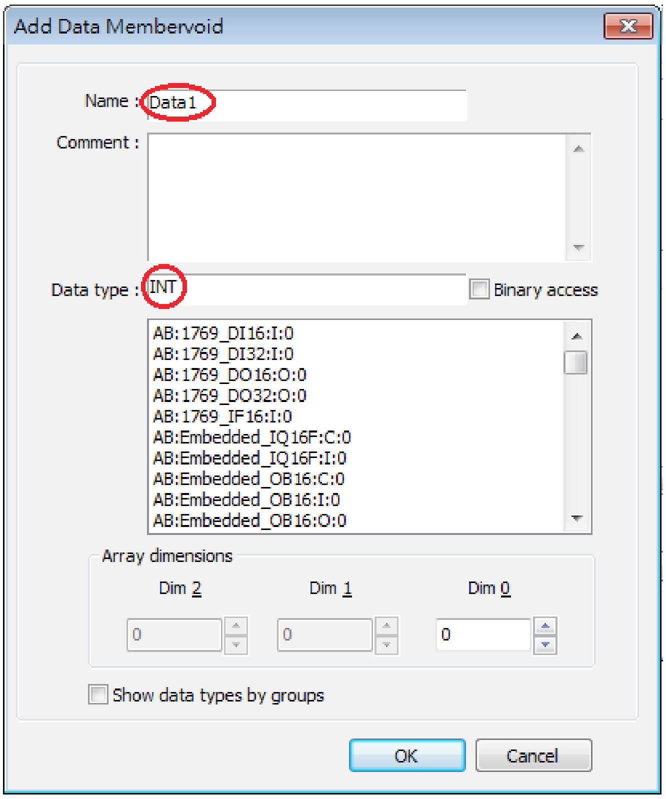
1. Right click the assigned data type (usually labeled as User-Defined), then click (New Data Type) to start editing.



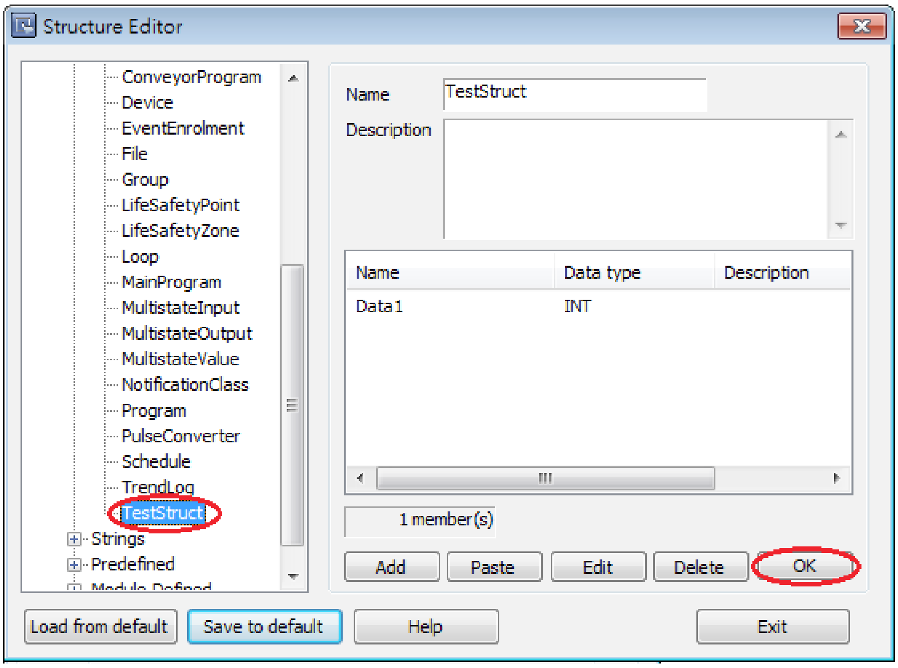
2. Enter the name of the data type. (Description) field can be left blank. To add a member, click (Add).



3. Enter the name and the data type then click (OK).



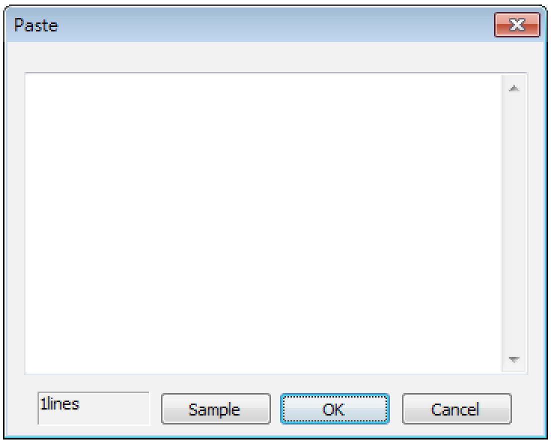
4. After adding all members, click (OK). The new data type will be added to the list of data types.



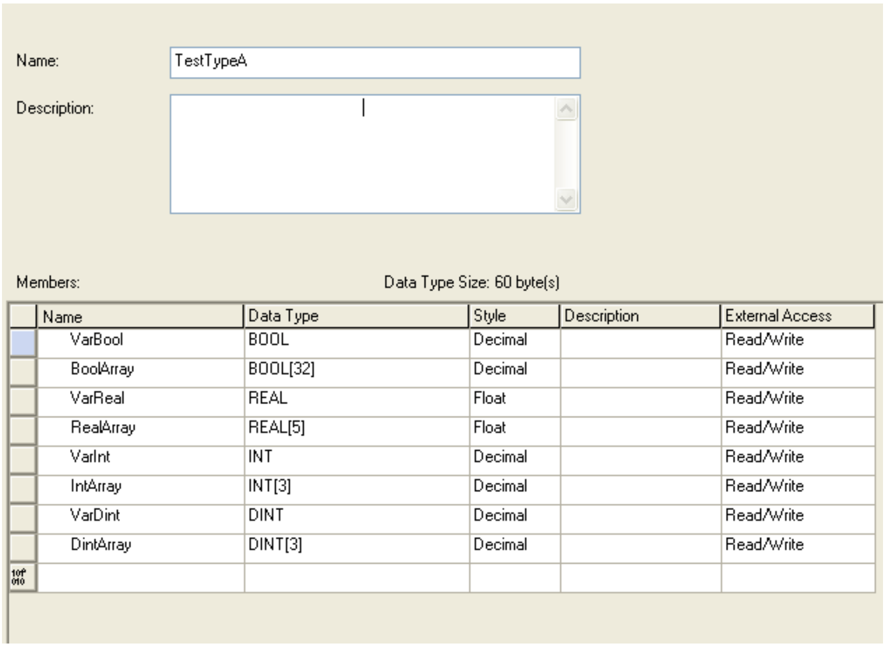
5. After changing the name or description of a data type, click (OK) to update.

34.4. STEPS TO PASTE

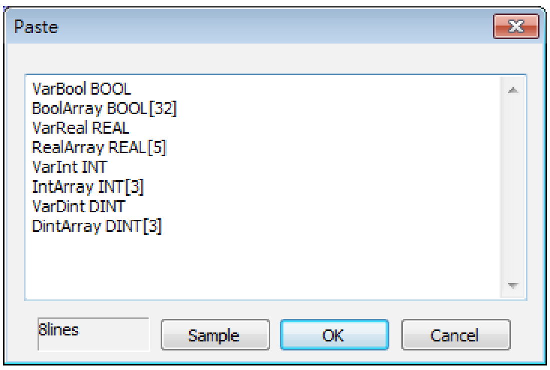
1. When adding a patch of members, this function allows users to add multiple data in one step. First, click the (Paste) button.



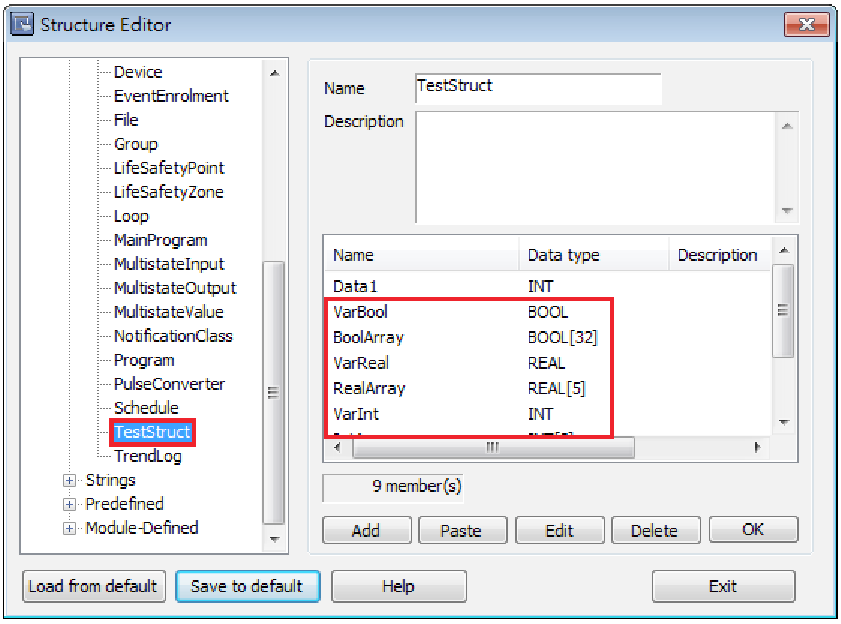
2. Type in data name, and then enter data type, separated by Space key or Tab key. It is recommended that data be directly copied and pasted from RSLogix5000 to avoid errors. Users can click (Sample) for a formatting example.



3. Copy the needed Name and Data defined in RSLogix such as one shown above. Paste the content in the editing window, as shown in the following figure.



4. Click (OK) to finish setting, and return to the main menu to check the added data.

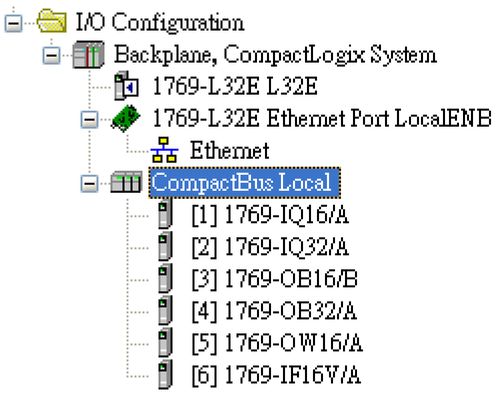


34.5. MISCELLANEOUS FUNCTIONS

- Revising member data
  Double click the member to be revised, or click the member then click (Edit)
- Deleting member data
  Select the data to be deleted then click (Delete). To delete all members of a data type, press and hold the Delete button on the keyboard and then click the (Delete) button in the dialog box
- Deleting a data type
  Select the data type from the list on the left and then press the Delete key on the keyboard
- Load from Default
  To start over from default settings, click (Load from Default) button
- Save to Default
  Saves data type settings to default for use in other projects

34.6. MODULE-DEFINED

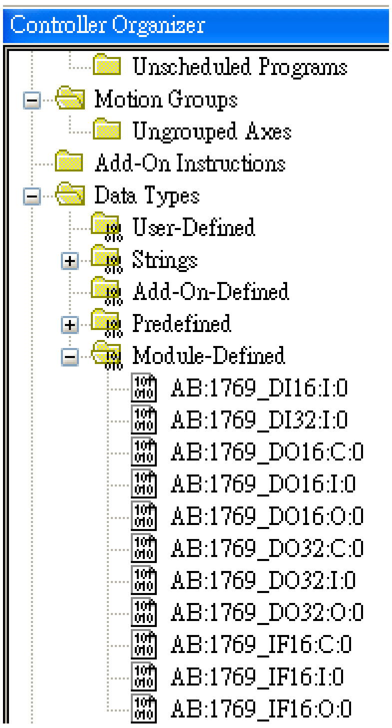
Module-Defined is a default structure of a module.
Here is an example showing how to define the default structure of a module.
In RSLogix5000 (I/O Configuration), the I/O module is set.



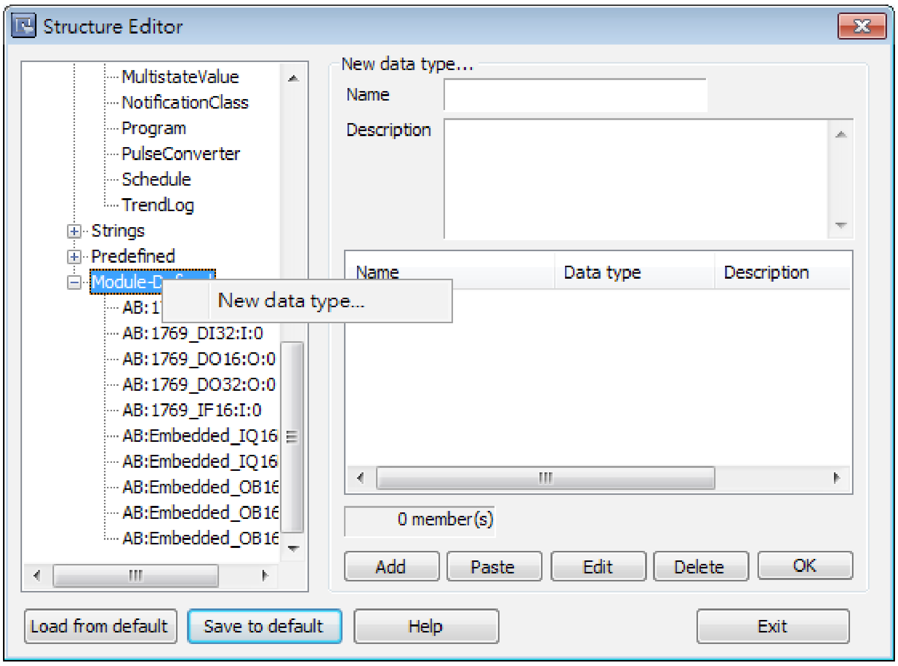
The exported CSV file will not list tags that are associated with module-defined structure. Please define the tags manually by following the steps below.

	A	B	C	D	E	F	G	H
7	TYPE	SCOPE	NAME	DESCRIPT	DATATYPE	SPECIFIER	ATTRIBUTES	
8	TAG		Local:1:I		AB:1769_DI16:I:0			
9	TAG		Local:2:I		AB:1769_DI32:I:0			
10	TAG		Local:3:C		AB:1769_DO16:C:0			
11	TAG		Local:3:I		AB:1769_DO16:I:0			
12	TAG		Local:3:O		AB:1769_DO16:O:0			
13	TAG		Local:4:C		AB:1769_DO32:C:0			
14	TAG		Local:4:I		AB:1769_DO32:I:0			
15	TAG		Local:4:O		AB:1769_DO32:O:0			
16	TAG		Local:5:C		AB:1769_DO16:C:0			
17	TAG		Local:5:I		AB:1769_DO16:I:0			
18	TAG		Local:5:O		AB:1769_DO16:O:0			
19	TAG		Local:6:C		AB:1769_IF16:C:0			
20	TAG		Local:6:I		AB:1769_IF16:I:0			
21	TAG		Local:6:O		AB:1769_IF16:O:0			
22								

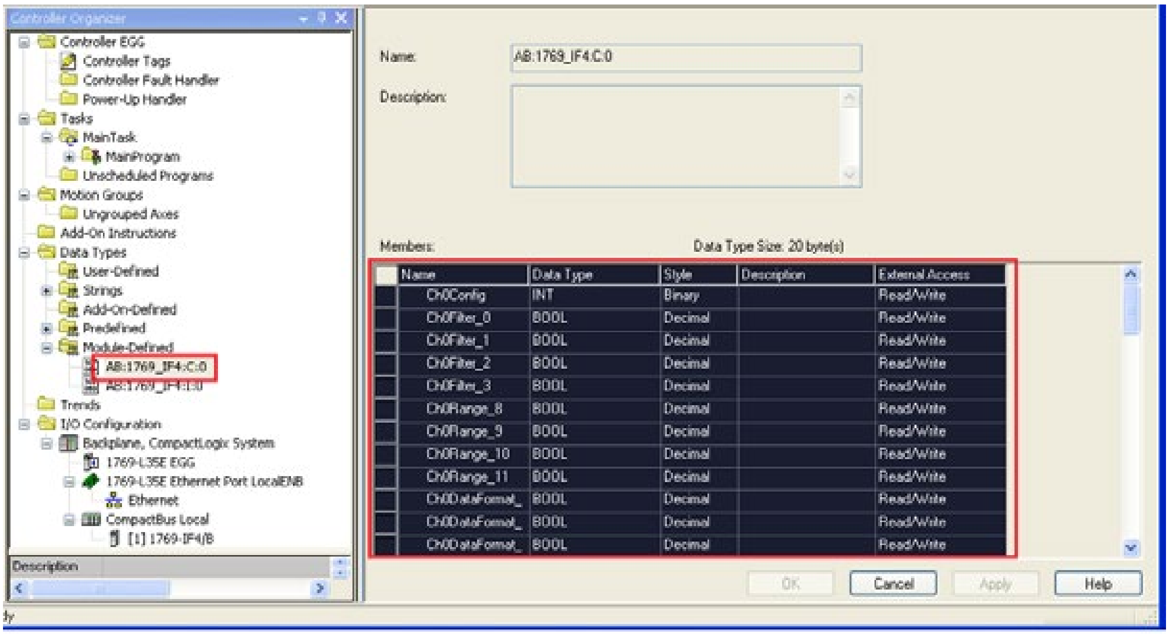
1. In RSLogix5000 (Controller Organizer) » (Data Types) » (Module-Defined), double click Data Type of the module. Members of the module will be shown in a popup dialog. Copy the Name and Data Type of the members.

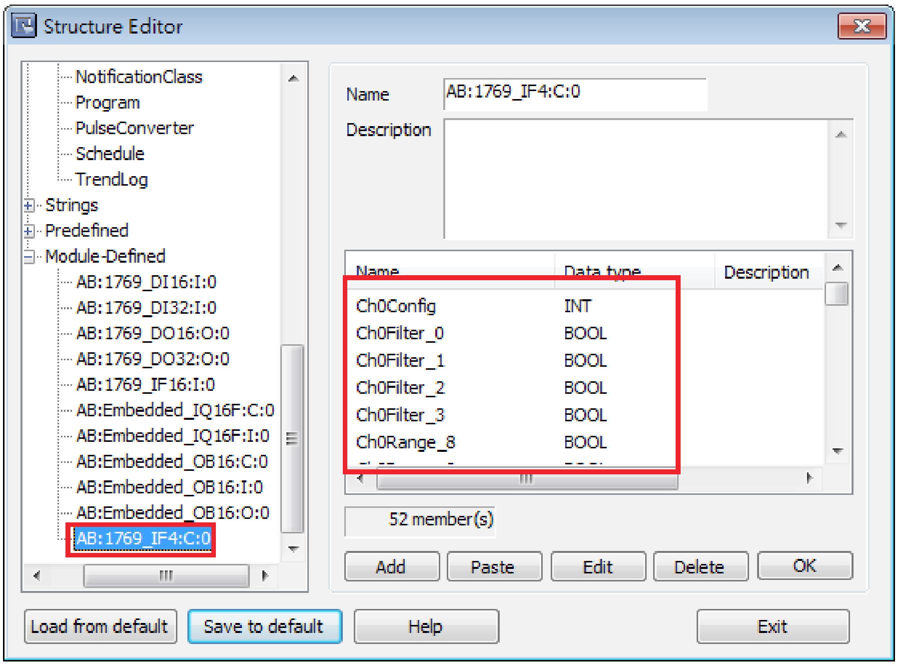


2. In Structure Editor, right click (Module-Defined), and then click (New Data Type). In (New Data Type) » (Name), enter the Module-Defined name.

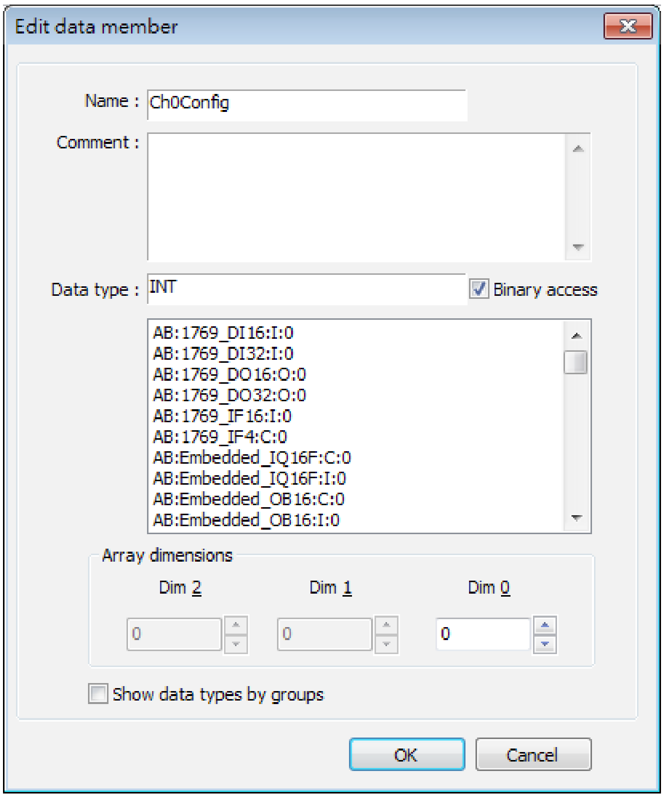


3. Click (Paste), and paste the data type information in the dialog box.





4. Select a member and then click (Edit). Since the data of the modules allows bit-wise operation, (Binary Access) should be selected, then click (OK) to return to Structure Editor.



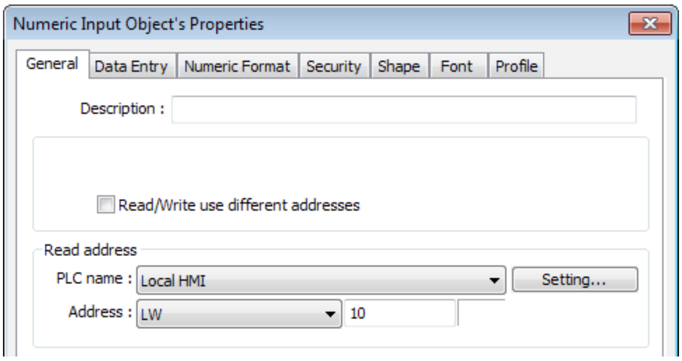
5. Click (OK) to finish setting.

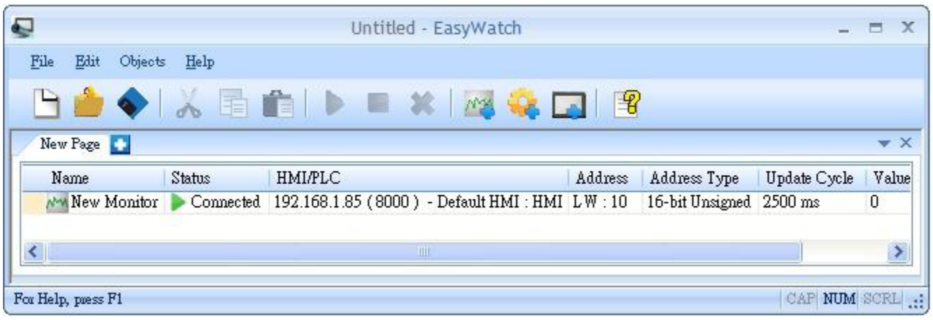
35. EASYWATCH

This chapter explains how to use EasyWatch.

35.1. OVERVIEW

EasyWatch allows users to monitor the HMI or the PLC address values via Ethernet from the PC, or to invoke the Macro for debugging, remote monitoring, and controlling.
For example, In EasyBuilder Pro, after creating a Numeric Input Object, you can set its address to LW-10, and set the same address in EasyWatch. The value of LW-10 will be shown in EasyWatch when successfully connecting.





	Note When the system register (LB-9044 (disable remote control)) or (System Parameter Settings) » (System Setting) » (Prohibit remote HMI connecting to this machine) is enabled, the feature of monitoring in EasyWatch will be unavailable.

35.2. CONFIGURATION

35.2.1. Basic Functions

Item	Description
File	**New:** opens a new EasyWatch file. **Open:** opens an existing EasyWatch file. **Save:** saves an EasyWatch file. **Save As:** saves an EasyWatch file to .ewt format. **Exit:** exits EasyWatch.
Edit	**Cut:** relocates the selected items to the clipboard. **Copy:** copies the selected items to the clipboard. **Paste:** pastes the items in the clipboard at the selected location.
Objects	**Add Object:** adds new Monitor or Macro objects. **Delete Objects:** selects the objects to be deleted, a dialog box appears, click (Yes) to delete. **Modify Object:** changes the settings of the selected object. **HMI Manager:** adds, modifies, or removes HMI settings. **Run:** executes the selected object. **Stop:** stops executing the selected object.
Help	**Help Topics:** reference of how to operate EasyWatch. **About EasyWatch:** easyWatch version information.

35.2.2. Quick Selection Tools

Item	Description
New	Opens a new EasyWatch file.
Open	Opens an existing EasyWatch file.
Save	Saves an EasyWatch file.
Cut	Relocates the selected items to the clipboard.
Copy	Copies the selected items to the clipboard.
Paste	Pastes the items in the clipboard at the selected location.
Run	Executes the selected object.
Stop	Stops executing the selected object.
Delete	Deletes the selected object.
Add Monitor	Adds a new address monitor.
Add Marco	Adds a new macro.
HMI	Opens HMI Manager to add, modify, or remove HMI settings.
Help Topics	Reference of how to operate the EasyWatch.

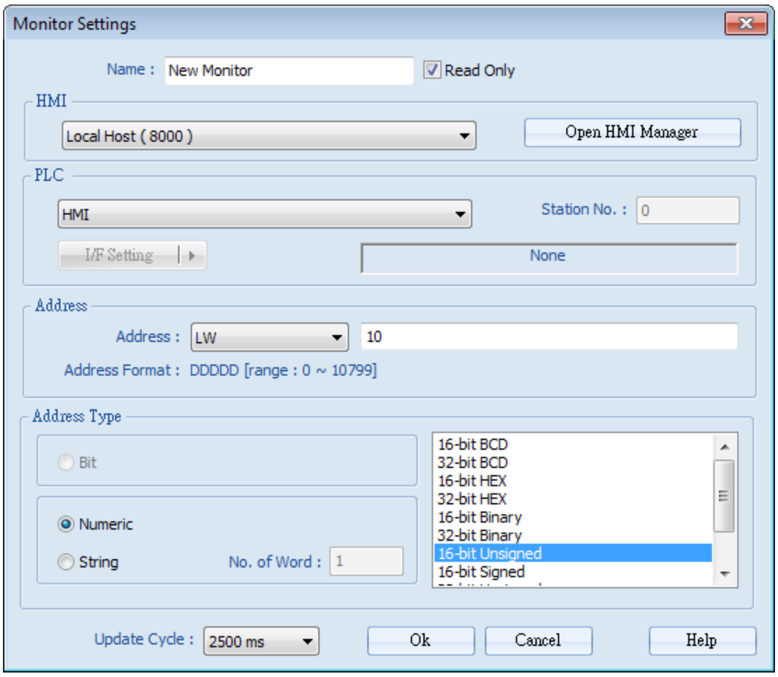
35.3. MONITOR SETTINGS

35.3.1. Adding a Monitor

There are two ways to create a Monitor Object.

- Select from the toolbar: (Objects) » (Add Object) » (Add Monitor)
- Select from the quick selection toolbar: (Add new address monitor)

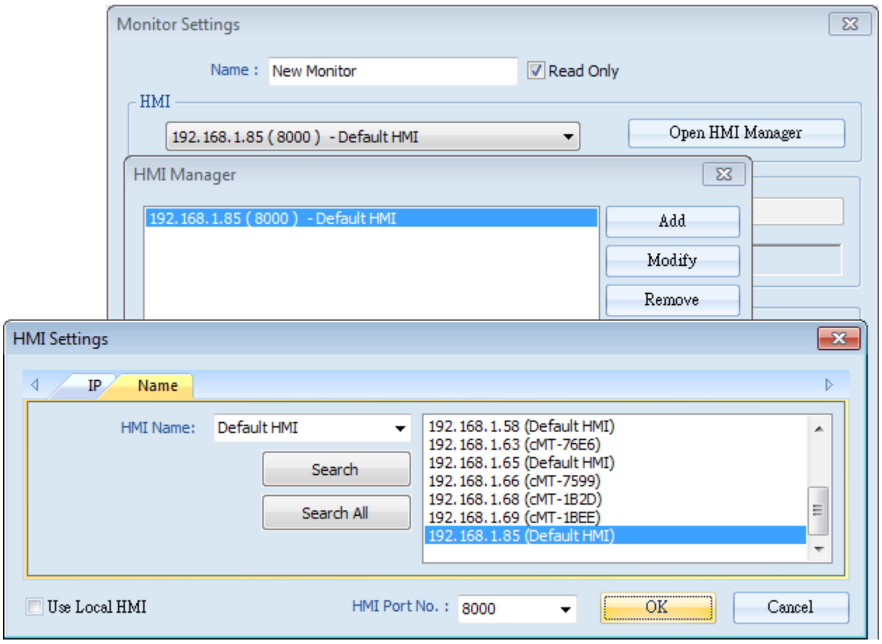
35.3.2. Monitor Settings



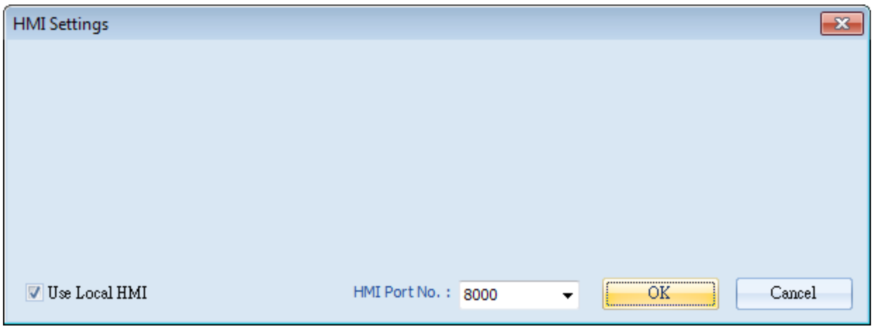
Setting	Description
Name	Enters an object name which is an unique name. **Read Only:** if an object is set to read only, its address value can't be edited.
HMI	Select a HMI to monitor.
PLC	Selects the desired PLC to monitor and sets its type, station number, and connection method.
Address	Selects the desired object address to monitor and sets its address type.
Address type	When the address type is set, the available formats of the address can be selected. When executing, the address will bedisplayed according to the selected format.
Update cycle	Sets the update interval of the monitoring object. If many objects are executed simultaneously, the errors or delays may appear.

35.3.3. Steps to Add a New Monitor

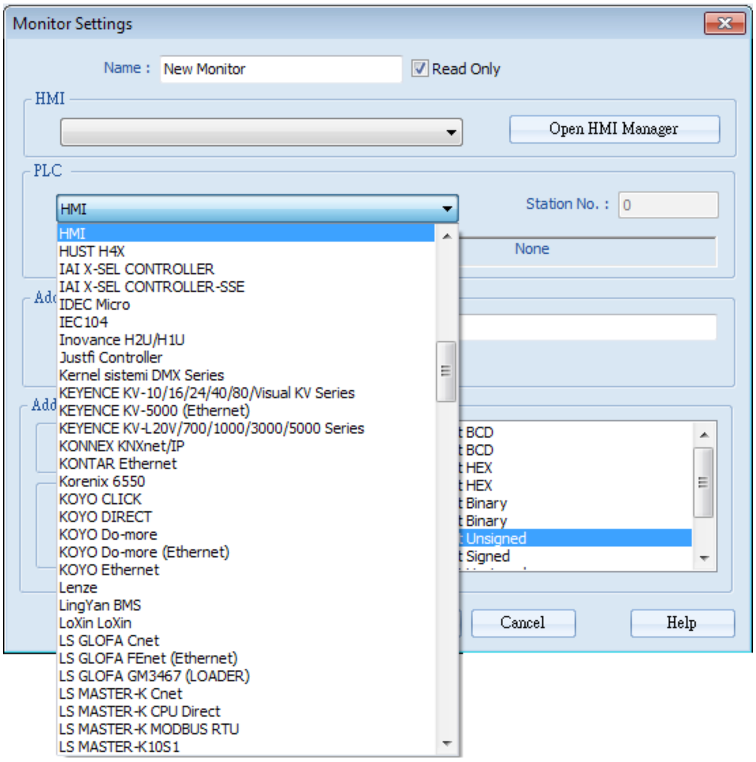
1. Select a target HMI. If the target HMI does not exist, click (Open HMI Manager) and click (Add) to search the HMI for adding.



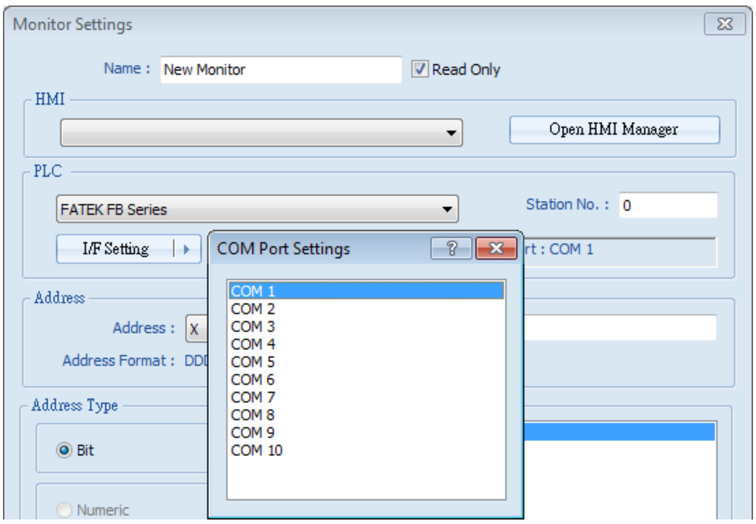
▪ Or, select (Use Local HMI) checkbox to use the project on PC as a monitoring device



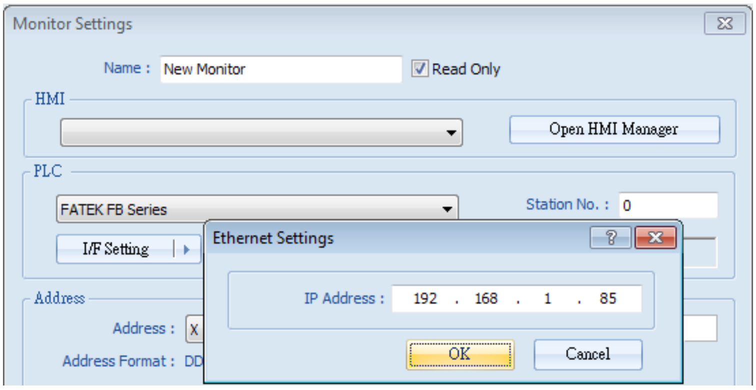
2. Select a target HMI or PLC. If one HMI is selected, you can directly control the local HMI.



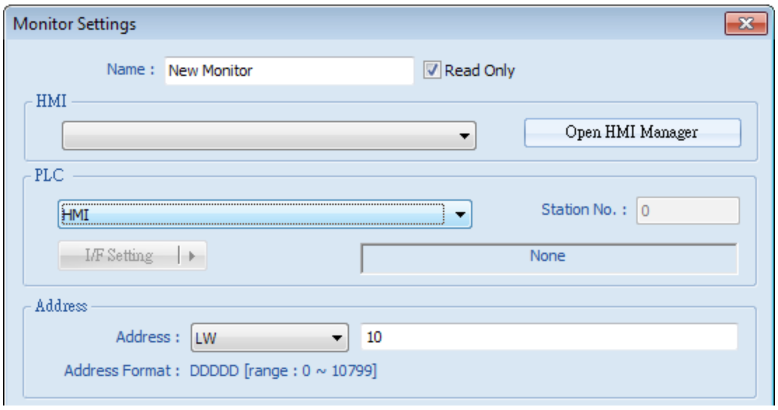
▪ If one PLC is selected, click (I/F Setting) and select (Com Port) to choose a COM port



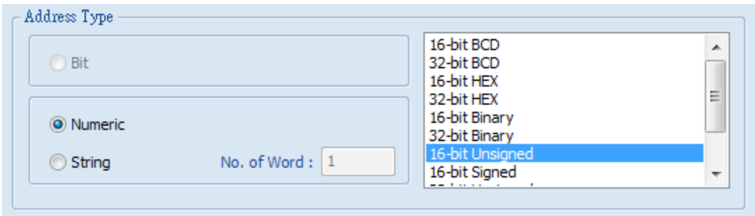
▪ Or, click (I/F Setting) and select (Ethernet) to set the IP address



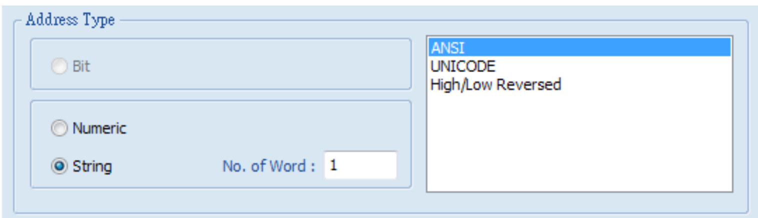
3. Set the object address and its address type for being monitored.



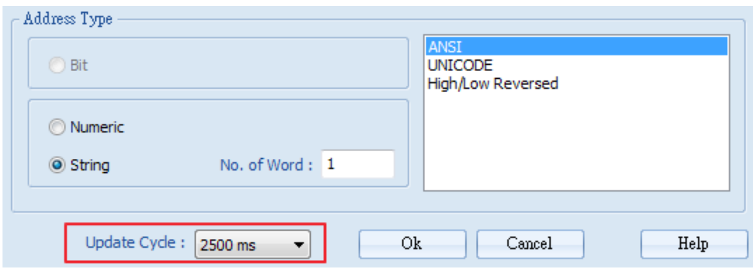
4. When a Word address is selected, set the address type to (Numeric) or (String).
   (Numeric): select the data format of the monitor address.



▪ (String): select the data format from (ANSI), (UNICODE), and (High/Reversed). Set (No. of Word) for reading the number of WORD



5. Set the update interval of the monitor object. The range can be set from 500ms to 5000ms.



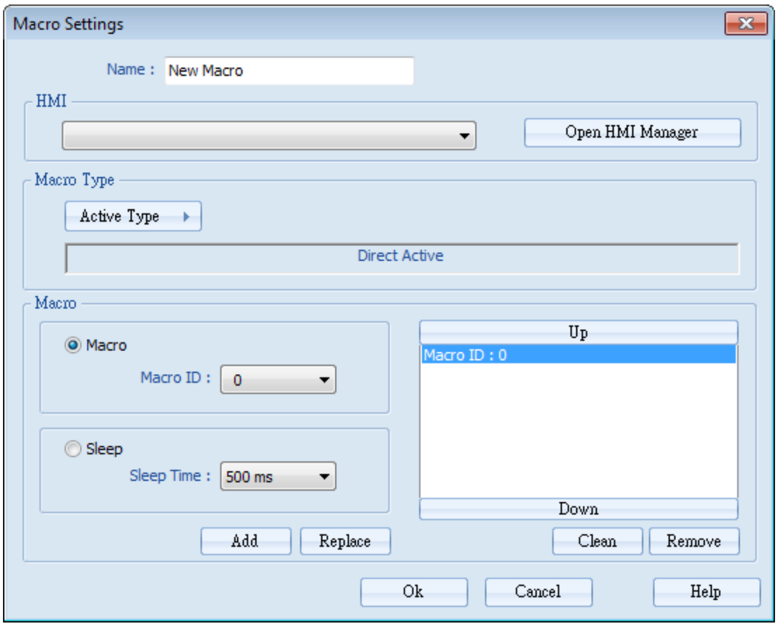
35.4. MACRO SETTINGS

35.4.1. Adding a Macro

There are two ways to create a Macro object.

- Select from the toolbar: (Objects) » (Add Object) » (Add Macro)
- Select from the quick selection toolbar: (Add new macro)

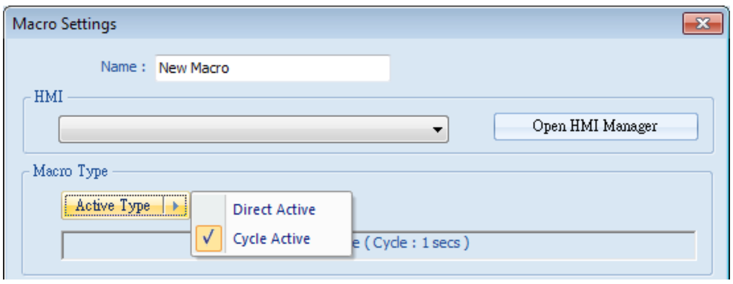
35.4.2. Macro Settings



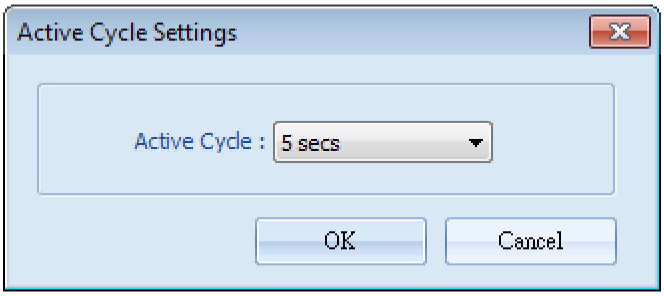
Setting	Description
Name	Enters an object name which is an unique name.
HMI	Selects a HMI to monitor.
Macro type	The ways to execute the Macro included Direct Active and Cycle Active.
Macro	Each Macro Object can execute multiple macros. The time interval between the executions of two macros can be changed.

35.4.3. Steps to Add a New Macro

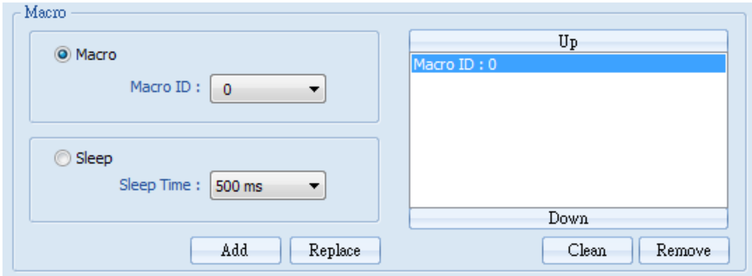
1. Select a target HMI. If the target HMI does not exist, add a new device, see “35.3.3 Steps to Add a New Monitor”.
2. Set Active Type under Macro Type to (Direct Active) or (Cycle Active).
   - (Direct Active): directly executes Macro once.
   - (Cycle Active): set the interval of executing Macro.



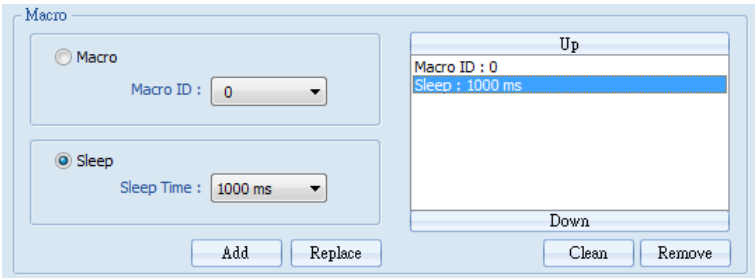
- For example, if (Cycle Active) is set to 5 seconds, when executing a macro, the next time to execute the macro object will be 5 seconds later



3. Set Macro to (Macro) or (Sleep).
   (Macro): select the Macro ID for execution, and click (Add) to add the Macro to the list.



- (Sleep): set the time interval between the executions of two Macros. Click (Add) or (Replace) to add or replace the Macros from the list



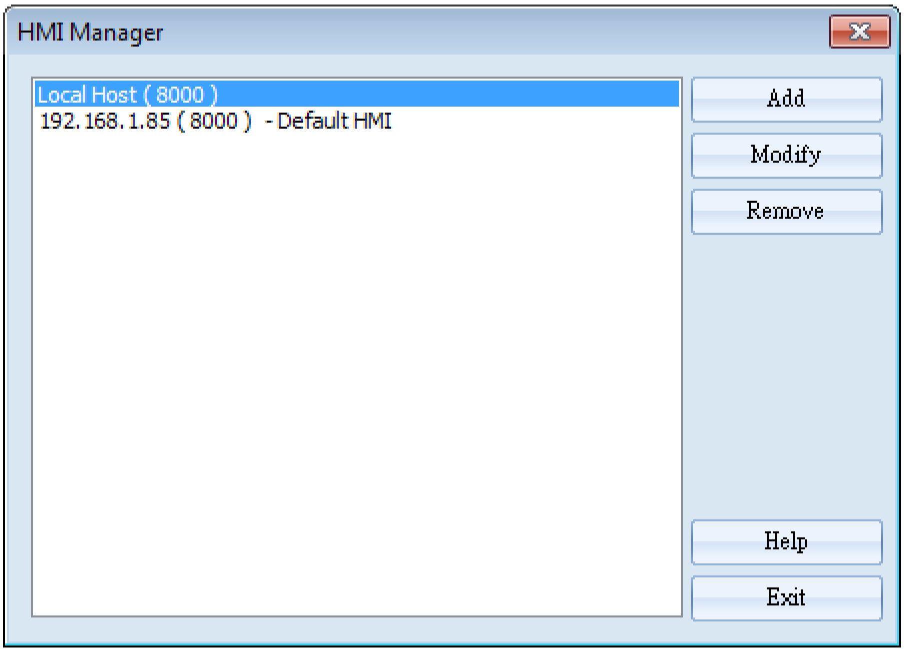
35.5. HMI MANAGER

35.5.1. Opening HMI Setting

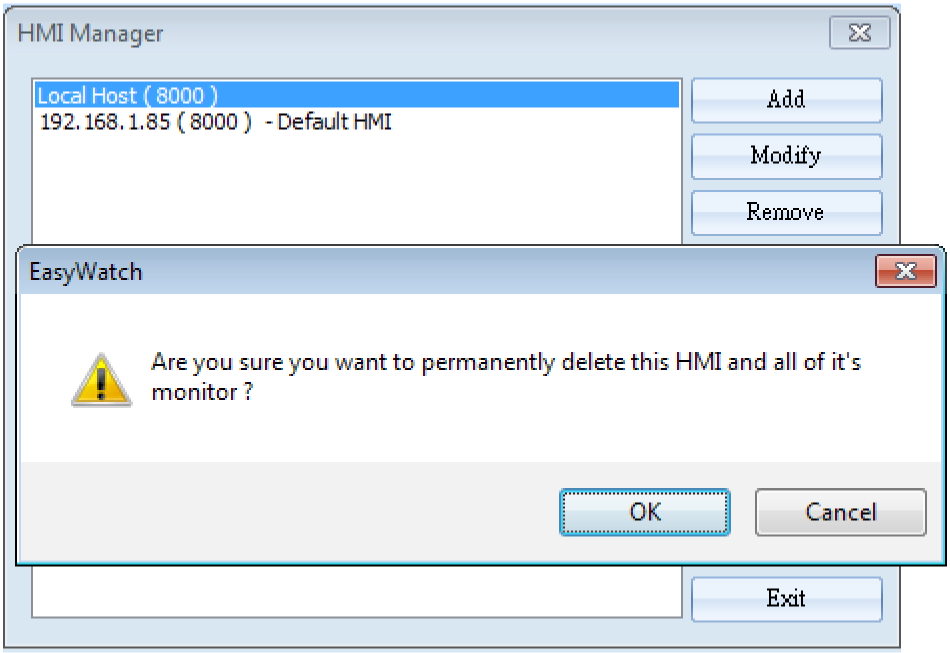
There are two ways to open the HMI Setting.

- Select from the toolbar: (Objects) » (HMI Manager)
- Select from the quick selection toolbar: (Open HMI manager)

35.5.2. HMI Manager

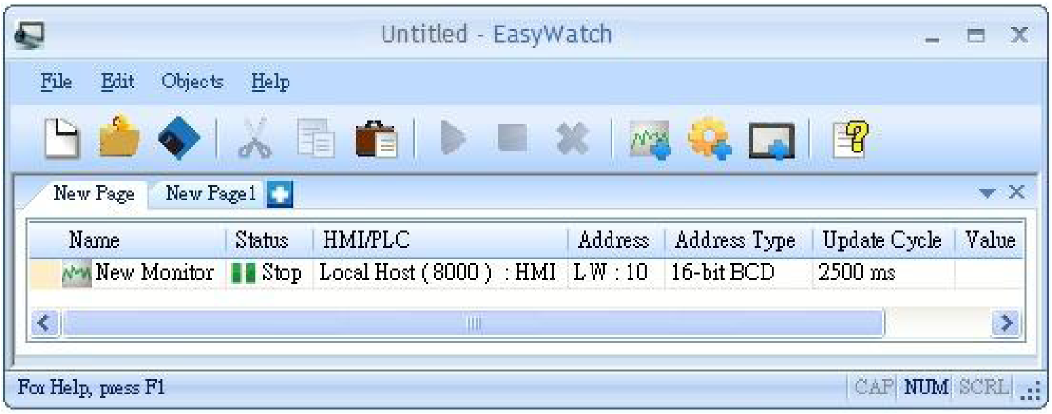


Setting	Description
Add	Select a target HMI. If the target HMI does not exist, add a new device, see "35.3.3 Steps to add a new monitor".
Modify	Modify the settings of the HMI.
Remove	Remove the HMI.



35.6. OBJECT LIST

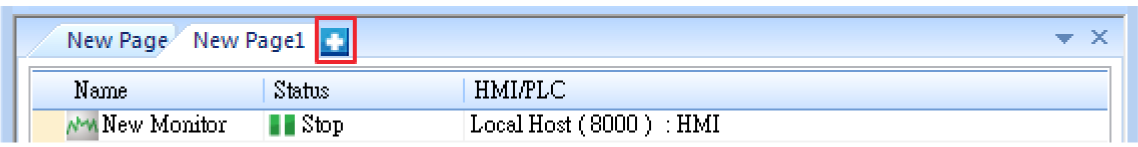
35.6.1. Object List Columns



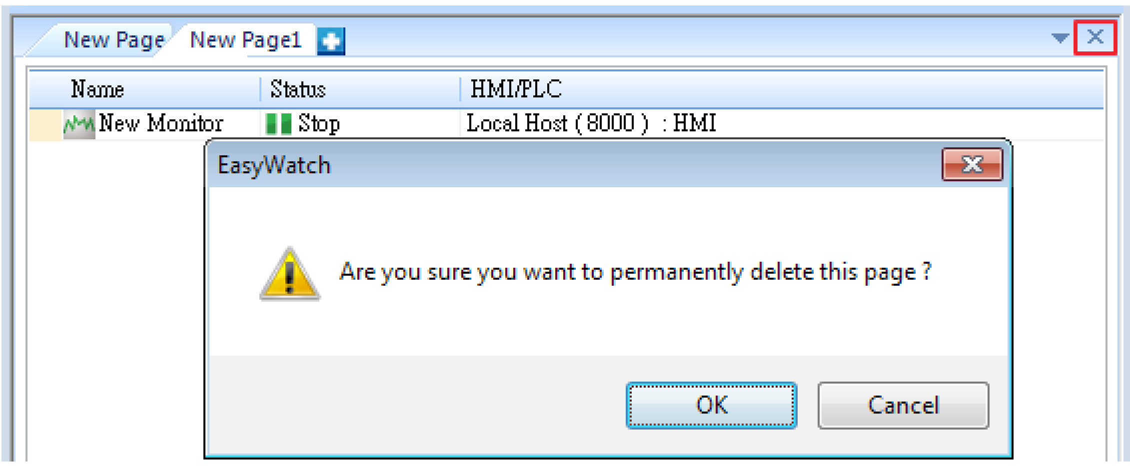
Setting	Description
Name	Displays the object name. The small icons of the names are for users to identify the objects.
Status	Displays the status of the objects: (Connecting), (Connected), and (Stop). If HMI is not connected or Port No. is incorrect, the error message “HMI Not Found” will appear. For Monitor Objects, if the address is incorrect, “Address Error” message will appear.
HMI / PLC	Displays the information of HMI / PLC that is currently operated by the objects.
Address Address type	For Monitor Objects, the relevant address setting is displayed.
Update cycle	Set the update interval of the monitor object.
Value	For Monitor Object, if the status shows (Connected), the current HMI address value will be displayed. Modifying the value is also available when the Read-Only checkbox is unselected. For Macro Object, if set to (Direct Active), there will be an (Active) button in this column for clicking to directly execute a macro.

35.6.2. Editing Object List

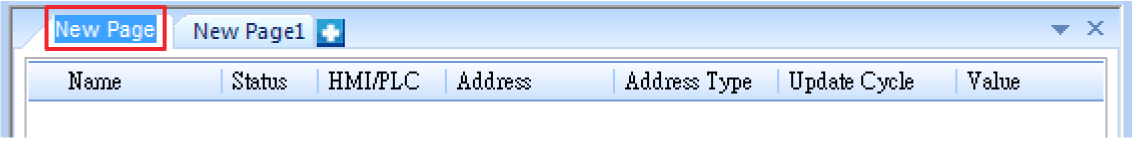
Adding a new page: click the icon to add a new page.



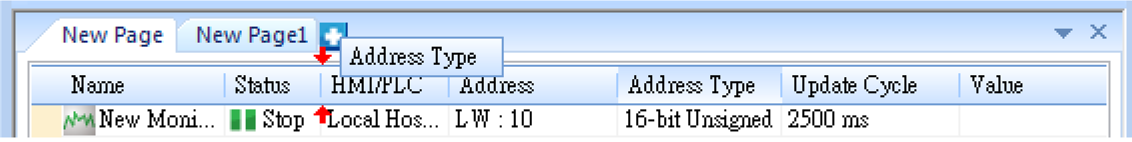
Deleting a page: click the icon and confirm the deletion.



Renaming the page: double-click on the page name and enter a new name.



Positioning the column headers: drag and drop the column headers to the desired location.

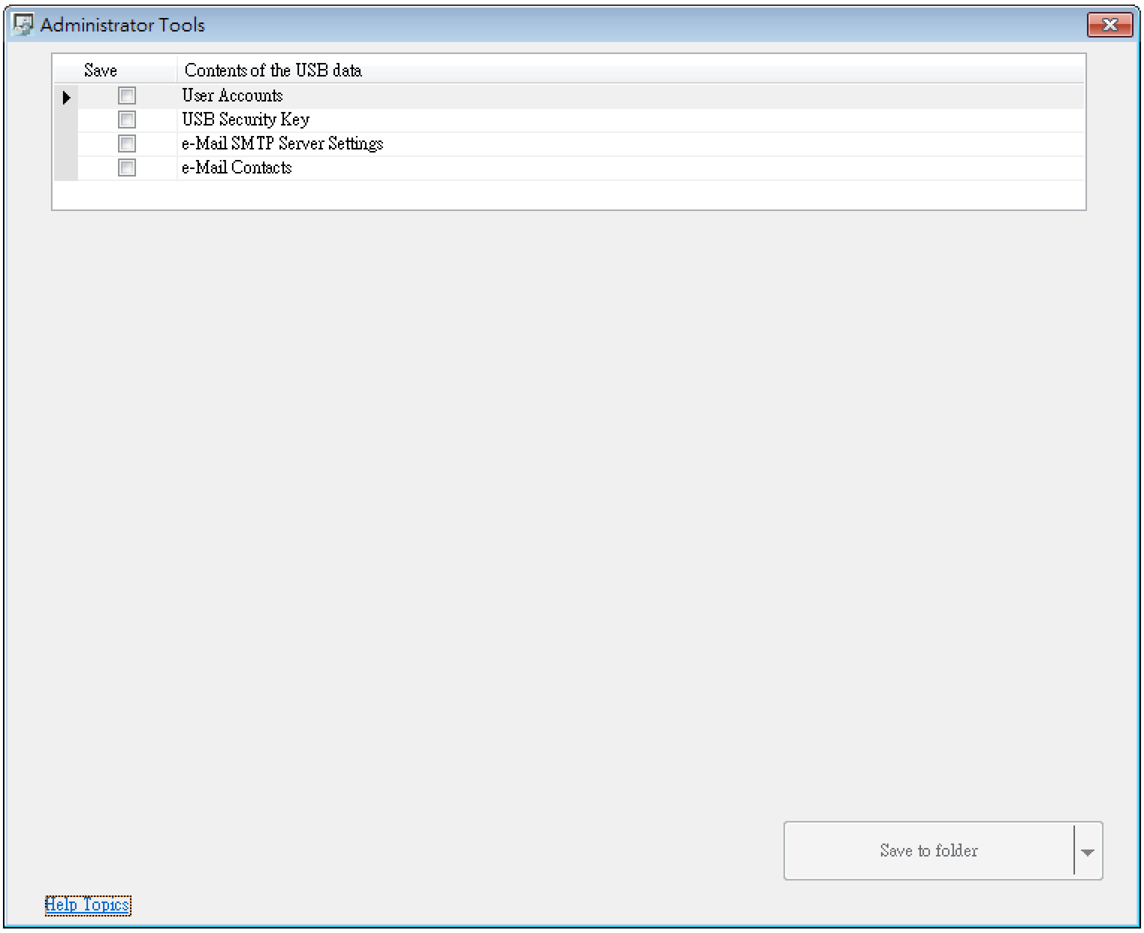


36. ADMINISTRATOR TOOLS

This chapter explains how to setup Administrator Tools.

36.1. OVERVIEW

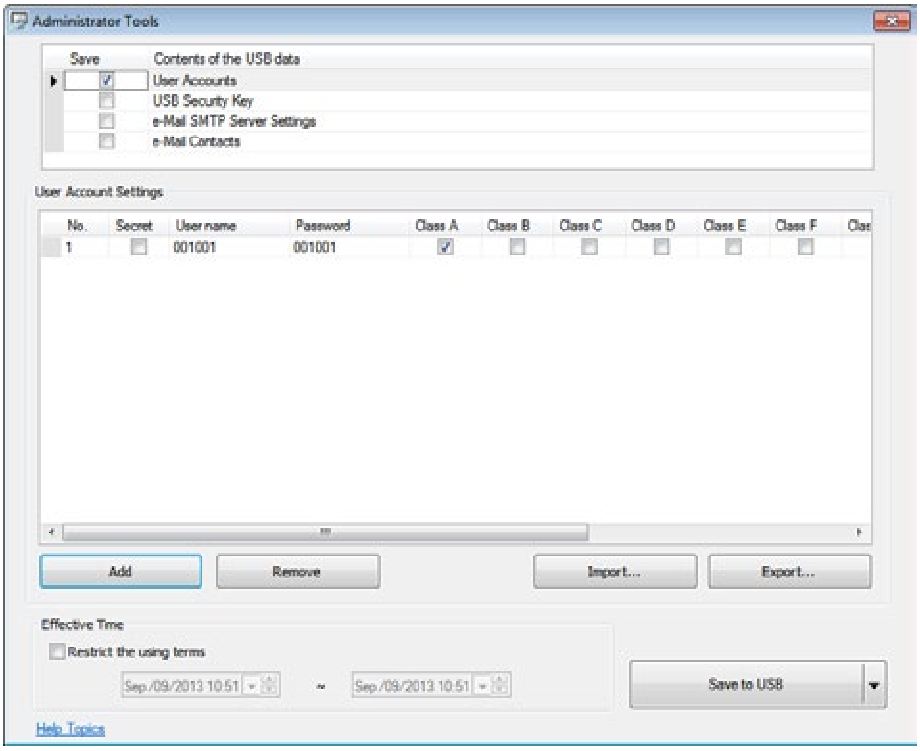
Administrator Tools allows storing the data of (User Accounts), (USB Security Key), (e-Mail SMTP Server Settings), and (e-Mail Contacts) to USB. With EasyBuilder Pro user accounts and e-Mail function, the data built can be imported to HMI by Function Key object set to “Import user data / Use (USB Security Key)”. The portability and convenience is greatly improved.
Launch Administrator Tools, select the check boxes in (Save) column to enable the selected functions introduced in this chapter.

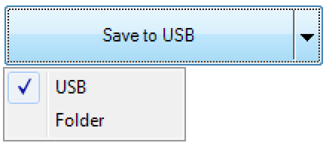


36.2. USER ACCOUNTS

36.2.1. User Accounts Settings

Select (User Accounts) check box and complete the relevant settings.

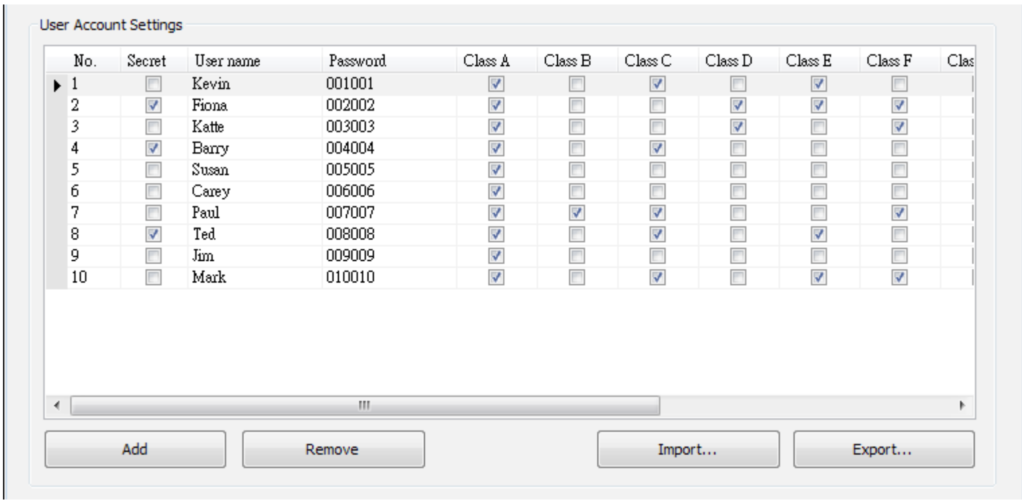


Settings	Description
Secret	Select the check box to set secret accounts.
User Name	Sets User Name. *Note 1
Password	Sets User Password. *Note 1
Class A to L	Sets user privilege.
Add	Adds a new account. *Note 2
Remove	Deletes an existing account.
Import	Imports user account data.
Export	Exports user account data.
Effective time	If (Restrict the using terms) is not selected, it is available to import data anytime. If select (Restrict the using terms) check box, and set an effective time period, the importing of data to HMI can only be done in the time limit specified. When time expired, data cannot be imported, please reset the data with this tool again.
Save to USB	Saves data to USB. To save to a specific folder, click "▼" button and select "Folder". 

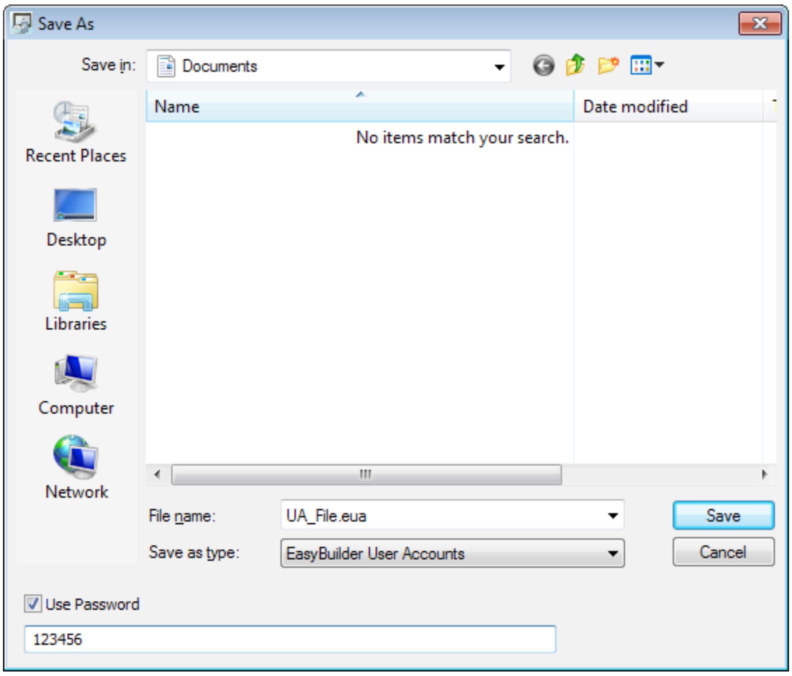
	Note 1. Only allows letters, numbers, "-" or "_", case-sensitive. 2. A maximum of 127 user accounts can be added.

36.2.2. Steps to Set User Accounts

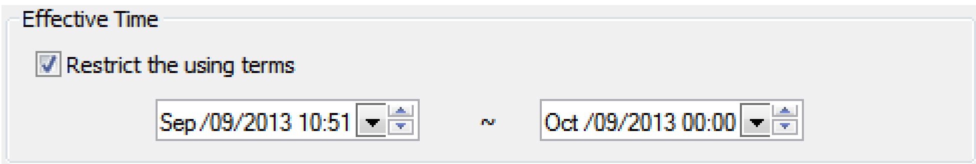
1. Click (Add) to create a new account. Click (Remove) to delete the selected account. Select (Secret) check box to define the account as a secret user. Type in (User name) and (Password) and check the privilege from (Class A) to (Class L) check boxes.



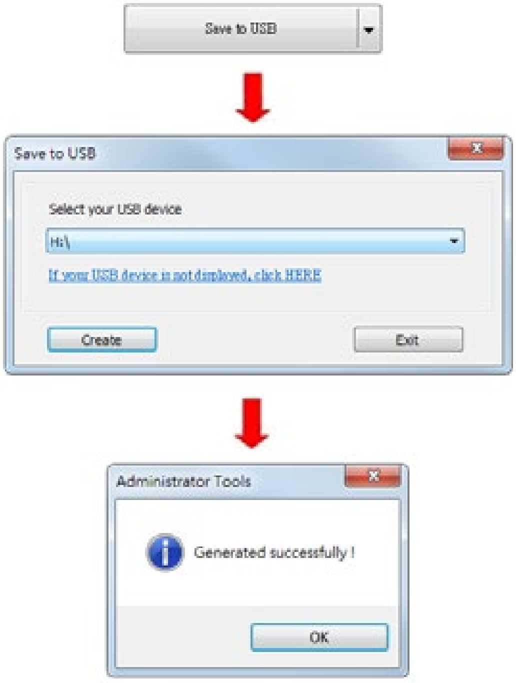
2. To back up the data, click (Export). Click (Use Password) to protect the data, next time when click (Import) to import the backup data, a password is required.



3. If under (Effective Time), the (Restrict the using terms) check box is selected, only during the specified time period can the users import account data to HMI. If not selected, users can import data to HMI at any time.



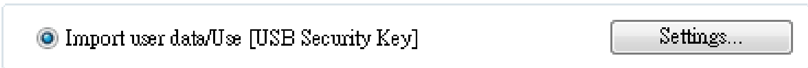
4. When finished, click (Save to USB), select the location of USB and then click (Create). If successful, the "Generated successfully!" message is shown.



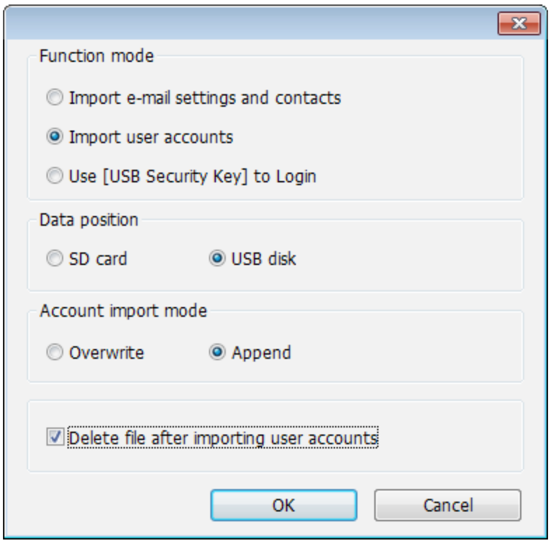
36.2.3. Steps to Import Accounts Using EasyBuilder Pro

The following steps explain how to create a Function Key to import data in EasyBuilder Pro.

1. Select "Import user data/Use (USB Security Key)" in Function Key setting dialog box, and then click (Settings).



2. Under (Function mode) select (Import user accounts). Select the device that stores the data in (Data position). Select (Overwrite) under (Account import mode); HMI will only store the account data imported this time. Select (Append), HMI will store the accounts imported this time and those already exist. Select (Delete file after importing user accounts) check box to delete the source files after importing.

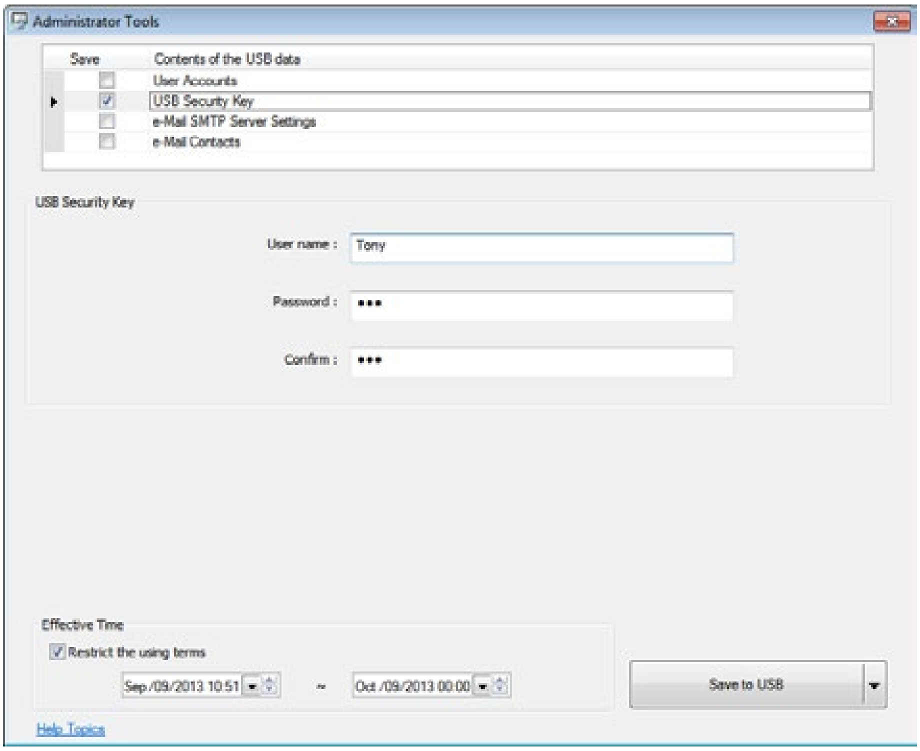


	Download Click the icon to download the demo project that explains how to import user accounts by using Function Key. Please confirm your internet connection.

36.3. USB SECURITY KEY

36.3.1. USB Security Key Settings

With the predefined user login information, the USB Security Key can be used to log in directly.

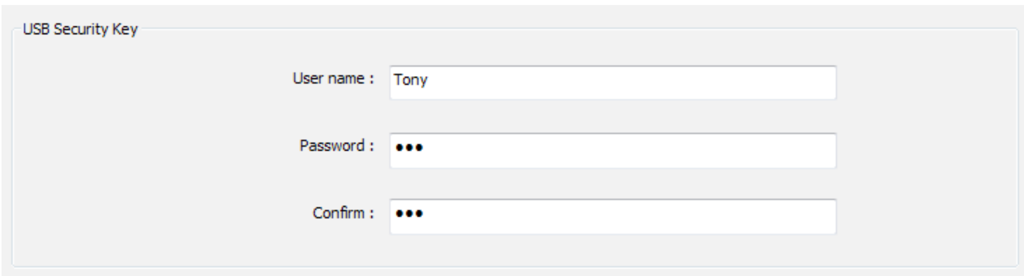


Settings	Description
User name	Sets User Name. *Note 1
Password	Sets User Password. *Note 1
Confirm	Confirms User Password.
Effective time	Logs in using USB Security Key during the specified time period. If not specifying Effective Time, log in at any time.
Save to USB	Saves data to USB.

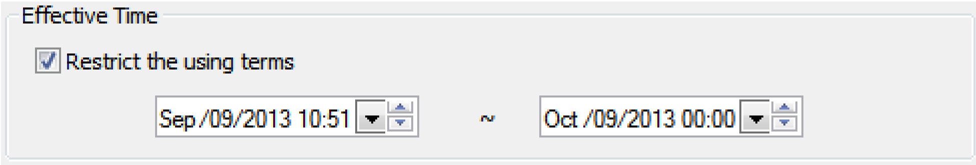
Note
1. Only allows letters, numbers, "-" or "_", case-sensitive.

36.3.2. Steps to Set USB Security Key

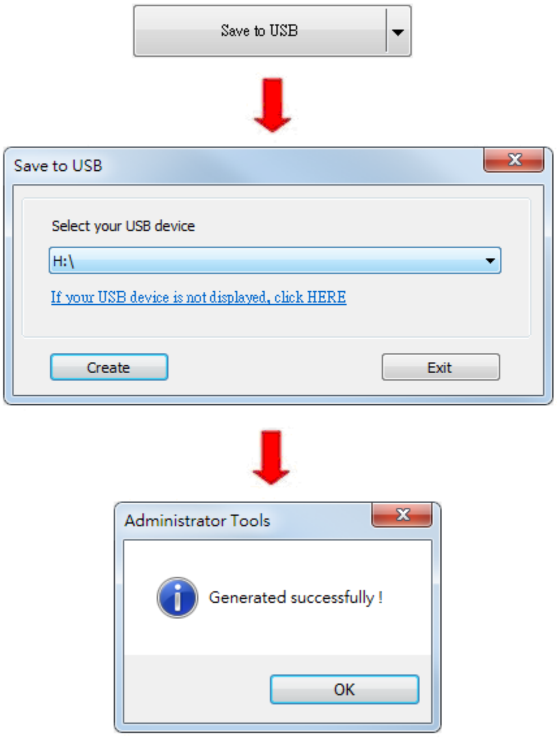
1. Type in the existing user name and password. Type the password again in (Confirm) field for password confirmation.



2. Under (Effective Time) if (Restrict the using terms) check box is selected, only during the specified time period can users log in using USB Security Key. If not selected, users can log in using USB Security Key at any time.



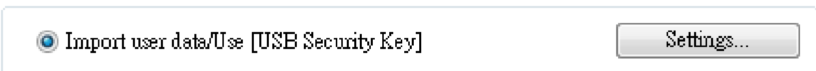
3. When finished, click (Save to USB), select the location of USB and then click (Create). If successful, the "Generated successfully!" massage is shown.



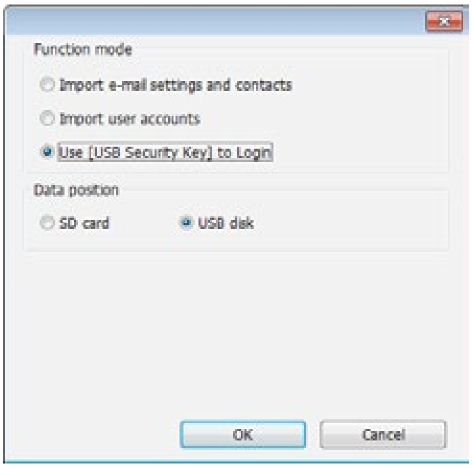
36.3.3. Steps to Set USB Security Key Using EasyBuilder Pro

The following steps explain how to create a Function Key to enable USB Security Key in EasyBuilder Pro. By touching the object, the USB Security Key is enabled for login.

1. Select "Import user data/Use (USB Security Key)" in Function Key setting dialog box, and then click (Settings).



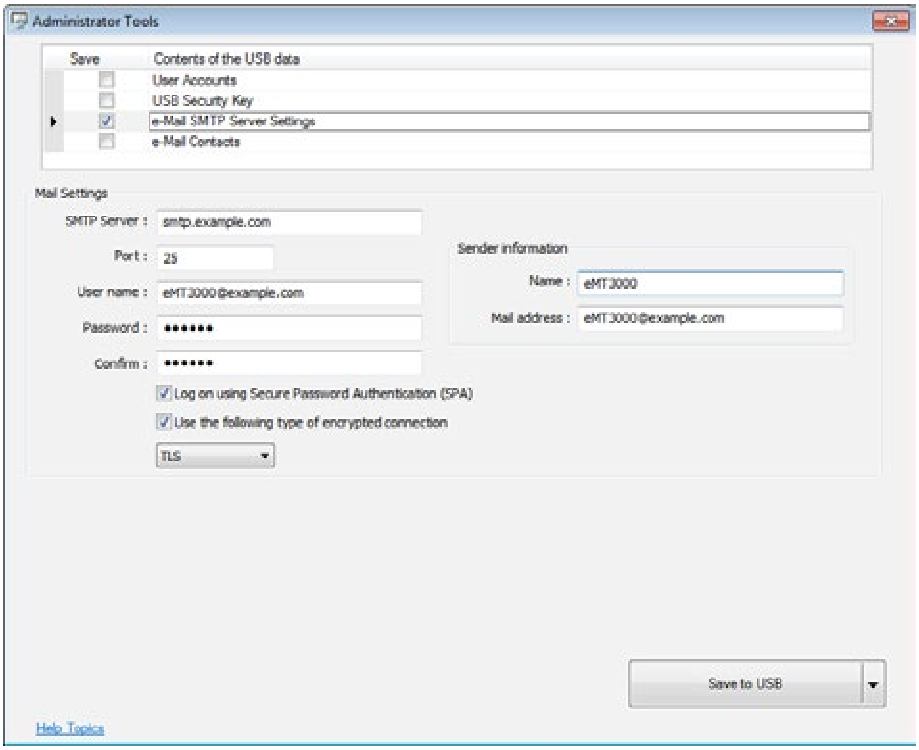
2. Under (Function mode) select (Use USB Security Key to Login). Select the device that stores the data in (Data position).



	**Download** Click the icon to download the demo project that explains how to enable login using USB Security Key by using Function Key. Please confirm your internet connection.

36.4. E-MAIL SMTP SERVER SETTINGS

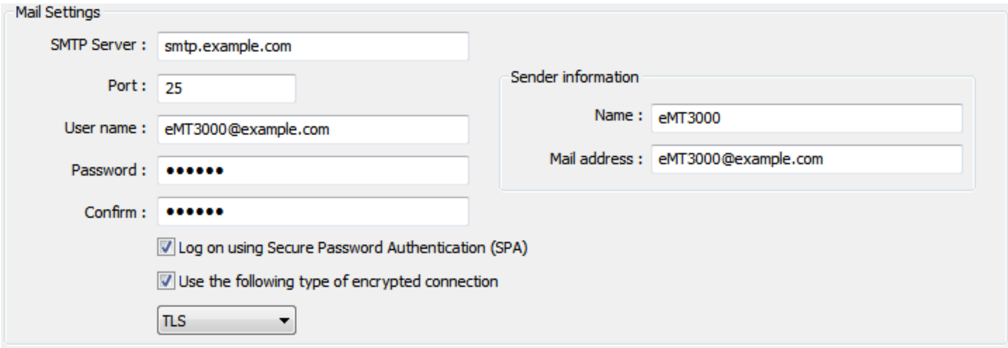
Select (e-Mail SMTP Server Settings) check box to complete the relevant settings.



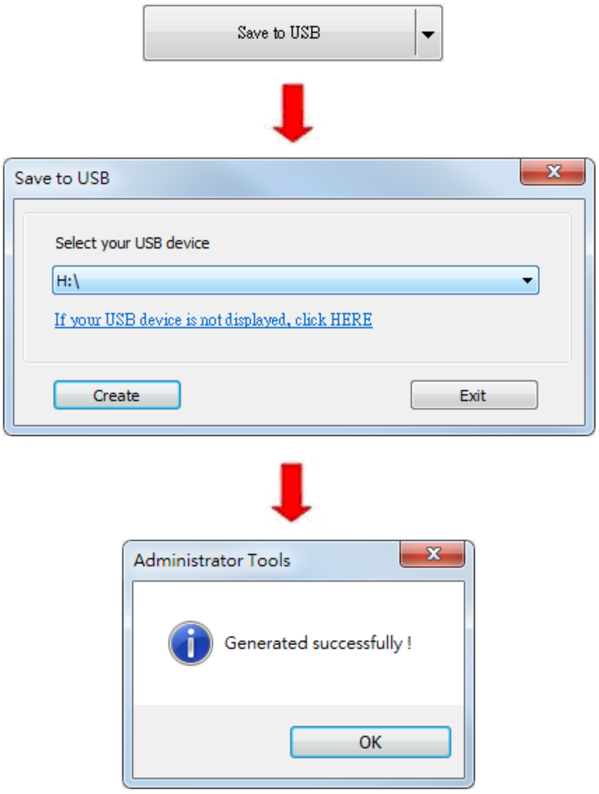
Mail settings	Description
Smtp server	Specifies SMTP Server.
Port	Specifies SMTP Server port number.
User name	User e-mail account name.
Password	User e-mail account password.
Confirm	Confirm user e-mail account password.
Sender information	**Description**
Name	The sender name displayed when mail received.
Mail address	The sender address displayed when mail received.
Save to USB	Saves data to USB.

36.4.1. Steps to set e-Mail SMTP Server Settings

1. Set the settings as shown in the following figure.



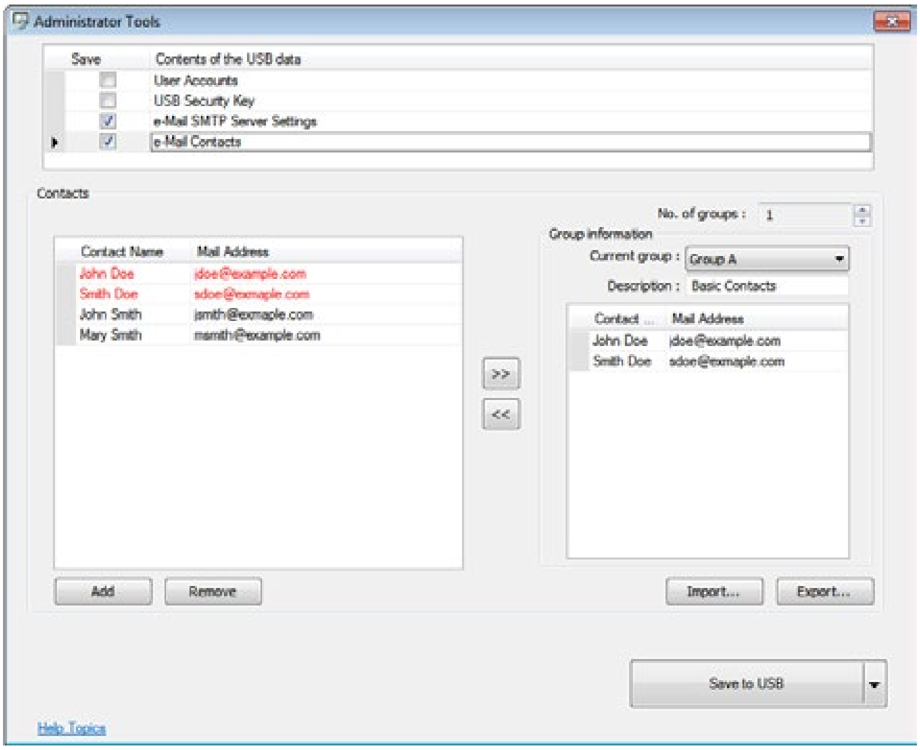
2. When finished, click (Save to USB), select the location of USB and then click (Create). If successful, the “Generated successfully!” massage is shown.



36.5. E-MAIL CONTACTS

36.5.1. e-Mail Contacts Settings

Select (e-Mail Contacts) check box to complete the relevant settings.

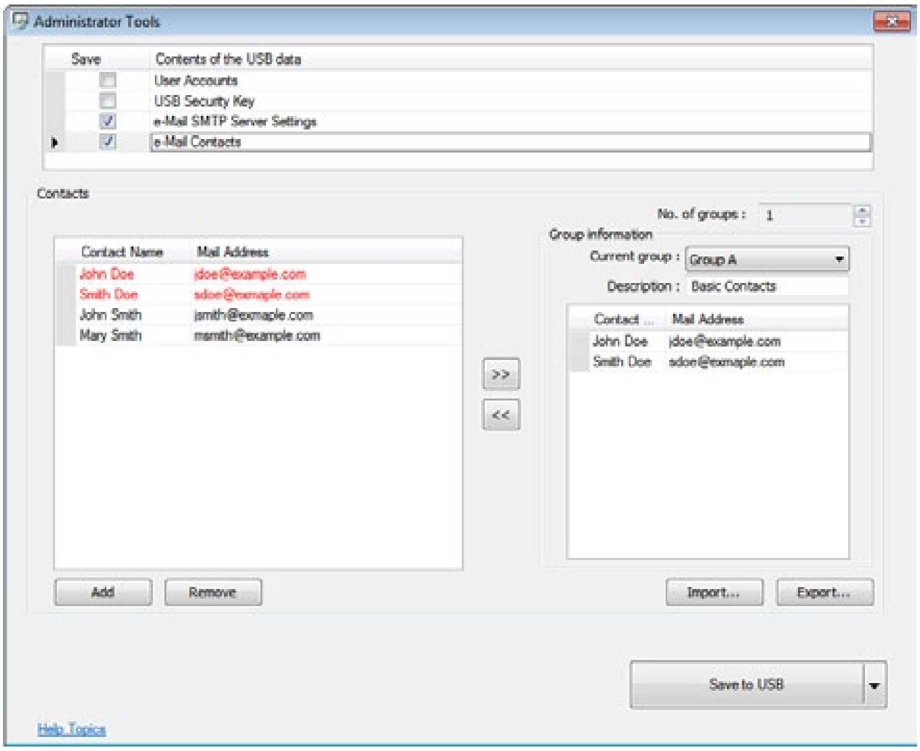


Settings	Description
Add	Adds a new contact. *Note1
Remove	Removes a contact.
No. of groups	The number of groups. *Note2
Current group	The name of current group. *Note3
Description	Group description.
Import	Imports contact information.
Export	Exports contact information.
Save to USB	Saves data to USB.

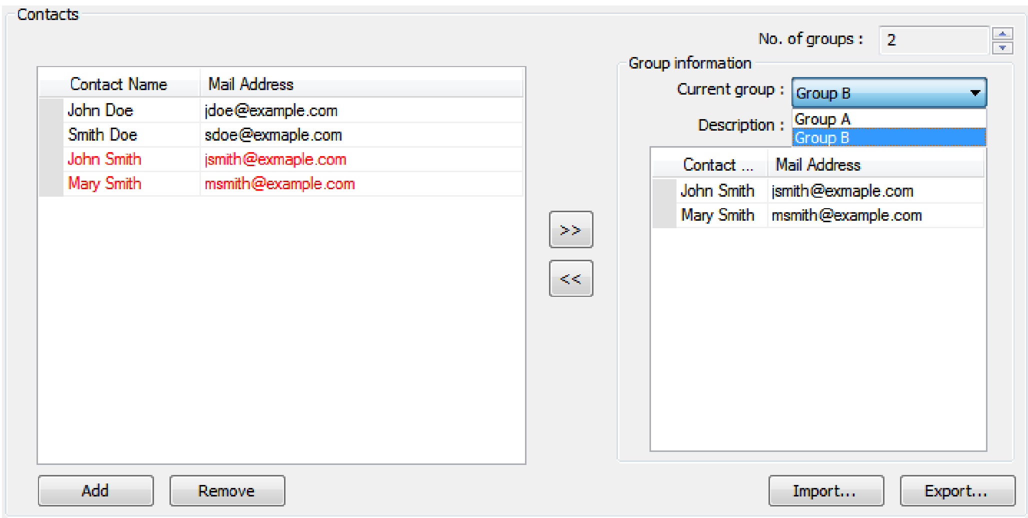
Note
1. A maximum of 256 contacts can be added.
2. A maximum of 16 groups can be added. (Group A to Group P)
3. From Group A to P, when (No. of groups) is “1”, only Group A will exist, when added to “2”, Group A and Group B will exist, and so on.

36.5.2. Steps to Set e-Mail Contacts

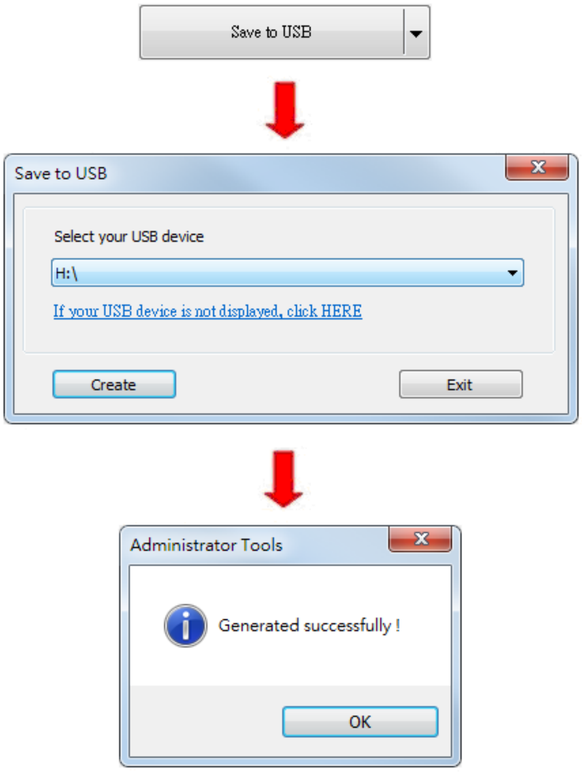
1. Click (Add) to add in all contacts.
2. Add the contacts to Group A, the added contacts are displayed in red font.



3. Press the up or down arrows of the spin box of (No. of groups) to add new groups. If add to “2”, Group B can be found. Repeat step 1 and step 2 to add contacts into groups.



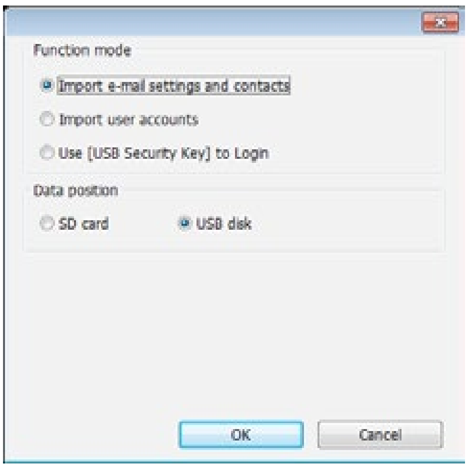
4. After adding all the e-mail contacts, click (Export) to back up the data for future use and modification. Next time when needed, click (Import) to import the backup data.
5. When finished, click (Save to USB), select the location of USB and then click (Create). If successful, the “Generated successfully!” massage is shown.



36.5.3. Steps to Import e-Mail Settings and Contacts Using EasyBuilder Pro

The following steps explain how to create a Function Key to import e-Mail contacts.

1. Select “Import user data/Use (USB Security Key)” in Function Key setting dialog box, and then click (Settings).
2. Under (Function mode) select (Import e-mail settings and contacts). Select the device that stores the data in (Data position).



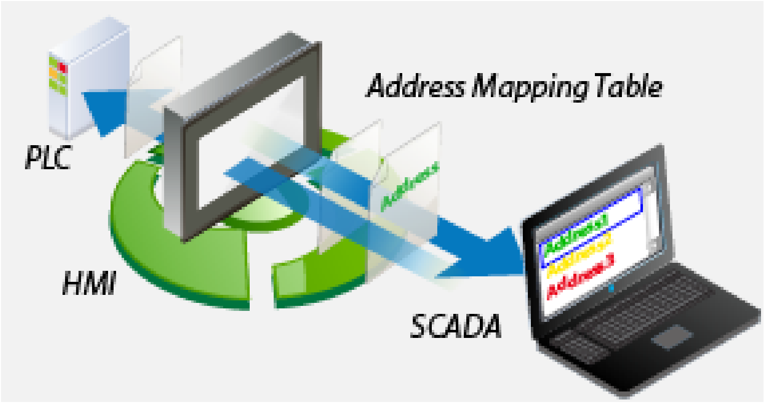
	Download Click the icon to download the demo project that explains how to import e-mail settings and contacts by using Function Key. Please confirm your internet connection.

37. MODBUS TCP/IP GATEWAY

This chapter explains how to use Modbus TCP/IP Gateway and configure address mapping tables.

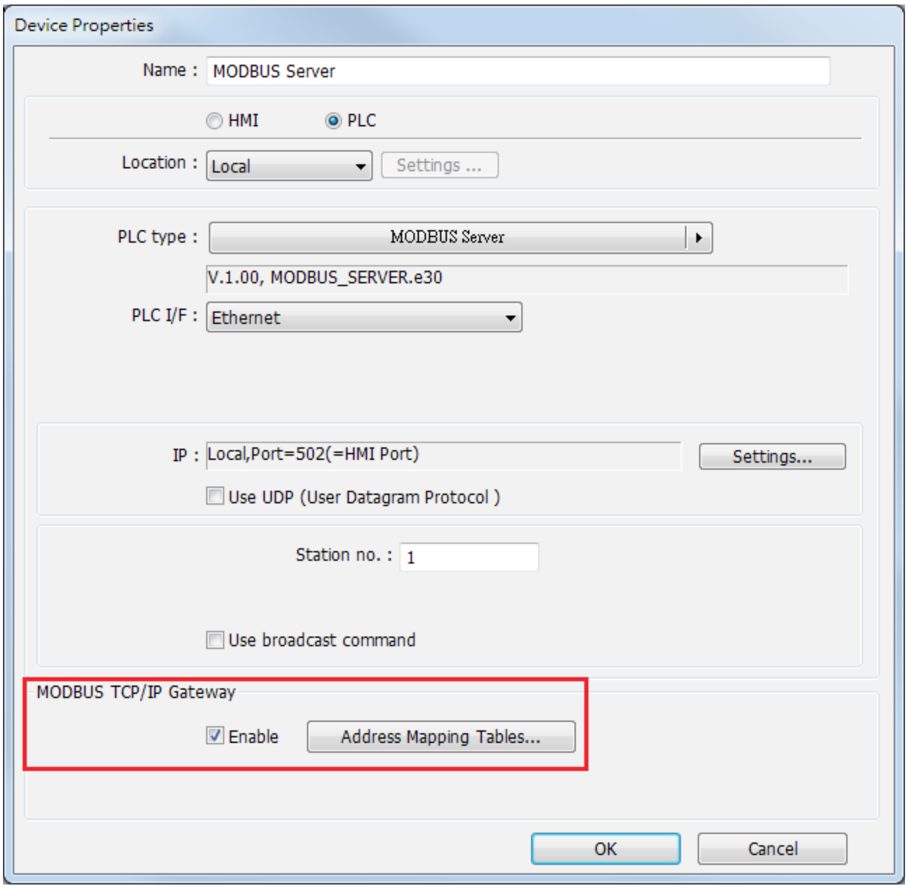
37.1. OVERVIEW

To access the data of the PLC connected to HMI with SCADA software (Supervisory Control and Data Acquisition), the former way was to transfer PLC data to the HMI's local address first, and then use Modbus TCP/IP protocol on PC to read HMI local address to get PLC data. Now by using Modbus TCP/IP Gateway provided by EasyBuilder, the mapping of Modbus address to PLC address can be defined first, and then one can directly use Modbus TCP/IP protocol to access PLC data.

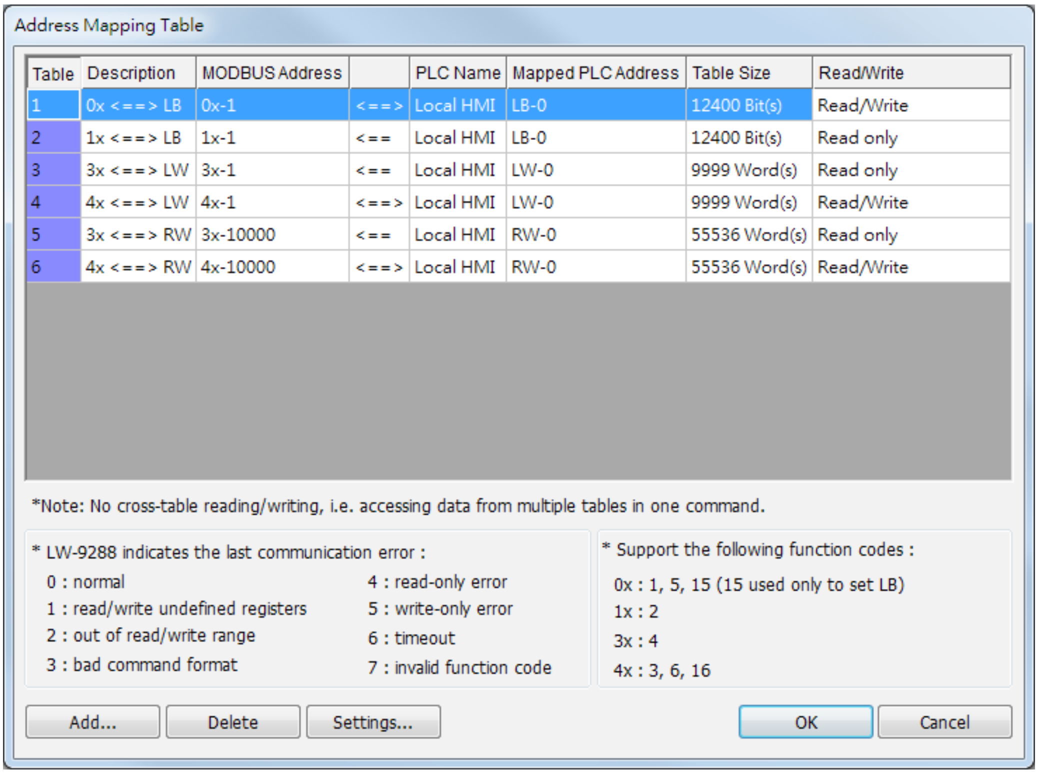


37.2. STEPS TO CREATE AN ADDRESS MAPPING TABLE

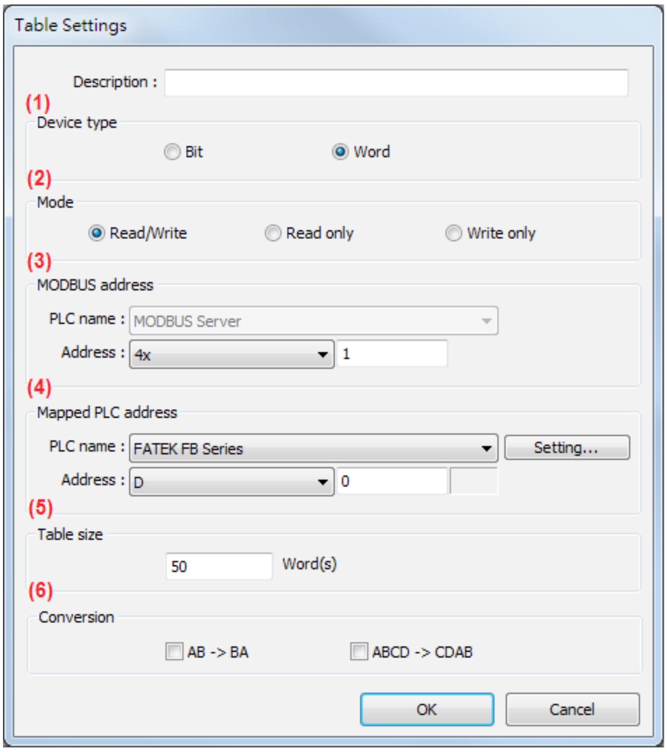
To create an Address Mapping Table, please follow the steps:
1. In (System Parameter Settings) » (Device) tab, add the PLC device. (In the example FATEK FB Series is used).
2. Add Modbus Server (Ethernet), select (Enable) check box under (Modbus TCP/IP Gateway) as shown in the following figure.



3. Click (Address Mapping Tables) button and the following default tables will be displayed. Modify the tables if needed or add new tables.



4. For example, to access the data in the 50 consecutive registers of FATEK FB Series PLC starting from register D-0, configure the settings as shown in the following figure.



- (1) Select the device type of the registers to be mapped, in the example select (Word).
- (2) Select the mode to access the data in the mapped register, in the example set to (Read/Write).
- (3) Set the start address of Modbus, in the example set to “4x-1”.
- (4) Set the start address of the mapped PLC, in the example set to “D-0”.
- (5) Set the range size of address mapping, in the example set to “50”.
- (6) If needed, select high/low byte swap (AB->BA) or high/low word swap (ABCD->CDAB).

Table	Description	MODBUS Address		PLC Name	Mapped PLC Address	Table Size	Read/Write
1	Access D0 ~ D49	4x-1	<==>	FATEK FB Series	D-0	50 Word(s)	Read/Write

The above figure shows that Modbus Server 4x-1 ~ 4x-50 registers are mapped to FATEK FB Series PLC D-0 ~ D-49 registers.

5. When finished, the data of FATEK FB Series PLC D-0 ~ D-49 registers are now accessible by using Modbus TCP/IP protocol to send read / write command to 4x-1 ~ 4x-50 registers.

37.3. NOTES ABOUT CONFIGURING ADDRESS MAPPING

- UDP is not supported when using the Modbus TCP/IP Gateway feature
- This feature is only supported by Modbus Server (Ethernet) interface
- System register LW-9288 is used to indicate if data transfer has been correctly executed

The following error codes represent:

Value	Definition
0	Normal
1	Read or write the register that is not defined in the Address Mapping Table.
2	Read or write a range of registers that is not within the range defined in a single Address Mapping Table. (Or, read / write a register that is defined in other Address Mapping Table.)
3	The command format does not follow Modbus TCP/IP protocol.
4	Modify a read-only register.
5	Read a write-only register.
6	Cannot get the correct reply from PLC within the specified time range.
7	Use a function code that is not supported by Modbus Server.

- The defined register range must not overlap between different mapping tables
- If (Modbus TCP/IP Gateway) is enabled, EasyBuilder will cancel the original mapping between Modbus Server and HMI register. That includes:

1. 0x, 1x mapped to LB
2. 3x, 4x mapped to LW, RW

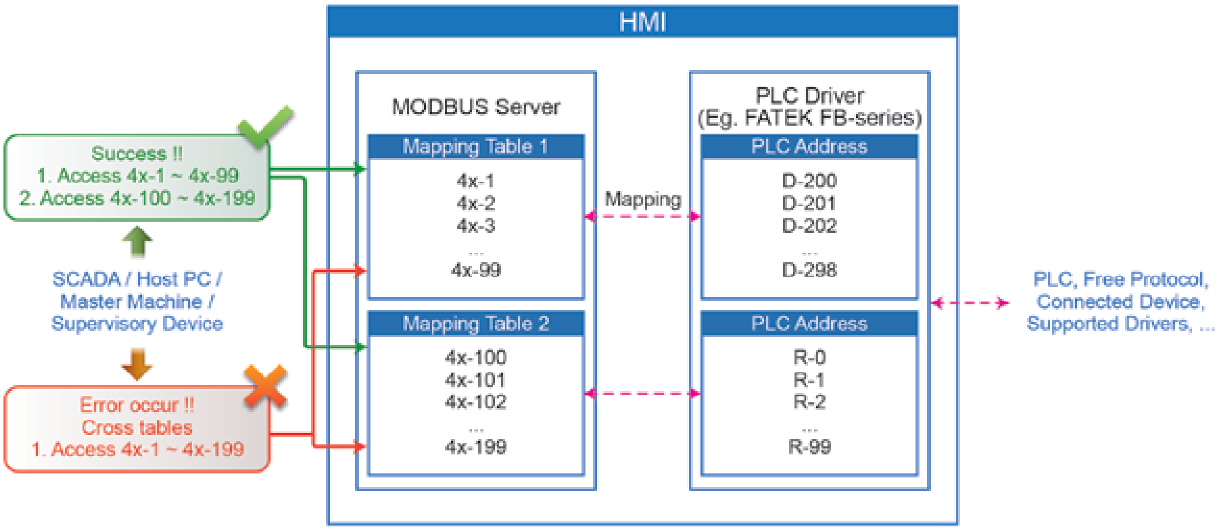
Therefore, to access data in LB or LW register via 0x, 1x, 3x, 4x, configure the Address Mapping Table again. The following figure is an example.

Table	Description	MODBUS Address		PLC Name	Mapped PLC Address	Table Size	Read/Write
1	0x <==> LB	0x-1	<==>	Local HMI	LB-0	12400 Bit(s)	Read/Write
2	1x <==> LB	1x-1	<==	Local HMI	LB-0	12400 Bit(s)	Read only
3	3x <==> LW	3x-1	<==	Local HMI	LW-0	9999 Word(s)	Read only
4	4x <==> LW	4x-1	<==>	Local HMI	LW-0	9999 Word(s)	Read/Write
5	3x <==> RW	3x-10000	<==	Local HMI	RW-0	55536 Word(s)	Read only
6	4x <==> RW	4x-10000	<==>	Local HMI	RW-0	55536 Word(s)	Read/Write

▪ SCADA can only read / write the register defined in one Address Mapping Table at one time, that is, the same Modbus command cannot access the data in the registers defined in different Address Mapping Tables

Table	Description	MODBUS Address		PLC Name	Mapped PLC Address	Table Size	Read/Write
1	Access D200 ~ D298	4x-1	<==>	FATEK FB Series	D-200	99 Word(s)	Read/Write
2	Access R0 ~ R99	4x-100	<==>	FATEK FB Series	R-0	100 Word(s)	Read/Write

As shown in the above figure, in Mapping Table 1 set Modbus 4x-1 to access register D-200, table size 99 words, and in Mapping Table 2 set Modbus 4x-100 to access register R-0, table size 100 words. If using SCADA to send a command to read from 4x-1 to 4x-199, table size 199 words, since the range spans two different tables, the command will not be accepted by HMI. Instead, access the data with two separate commands (4x-1~4x-99 and 4x-100~4x-199), each reading only from one table as shown in the following figure.



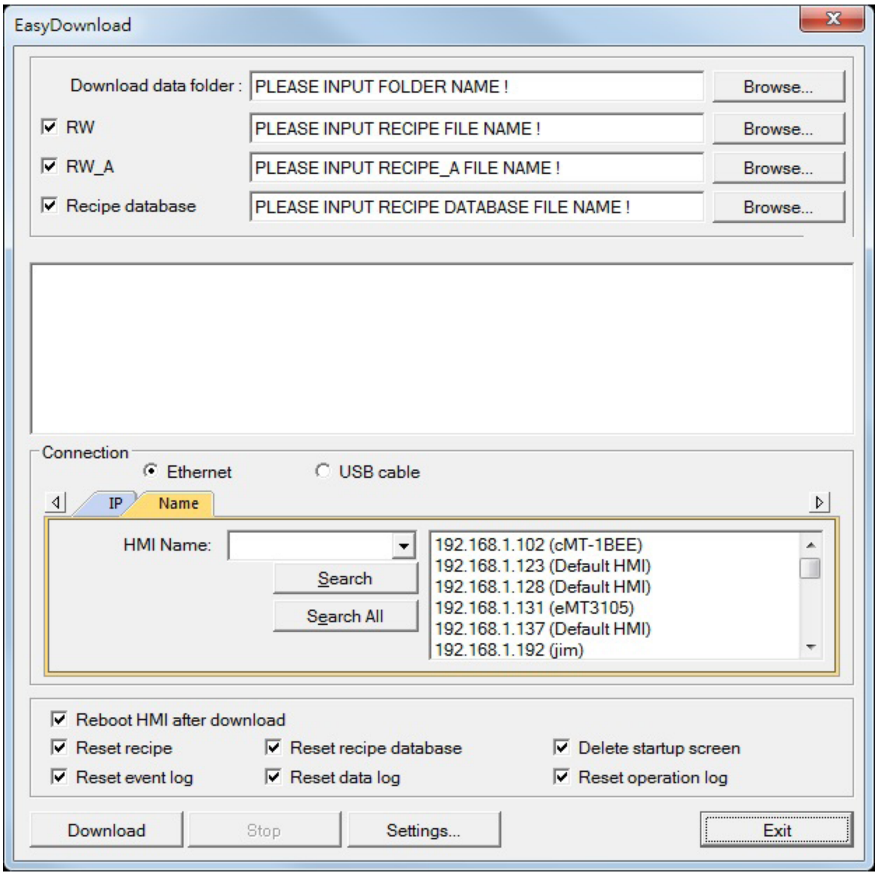
38. EASYDOWNLOAD

This chapter explains how to setup EasyDownload.

38.1. OVERVIEW

EasyDownload allows downloading the project data file built in EasyBuilder Pro via Ethernet or USB cable. In EasyBuilder Pro main menu select (Tools) and then select (Build Data for USB Disk or SD Card Download) to build the data file before running EasyDownload.

38.2. CONFIGURATION



Note

- Take eMT3000 Series as an example, the built download data directory has the following structure. Please select the parent directory of the generated file when downloading.

Parent directory	First subdirectory	Second subdirectory
emt3000	001	
	002	
	Pub	driver
		font

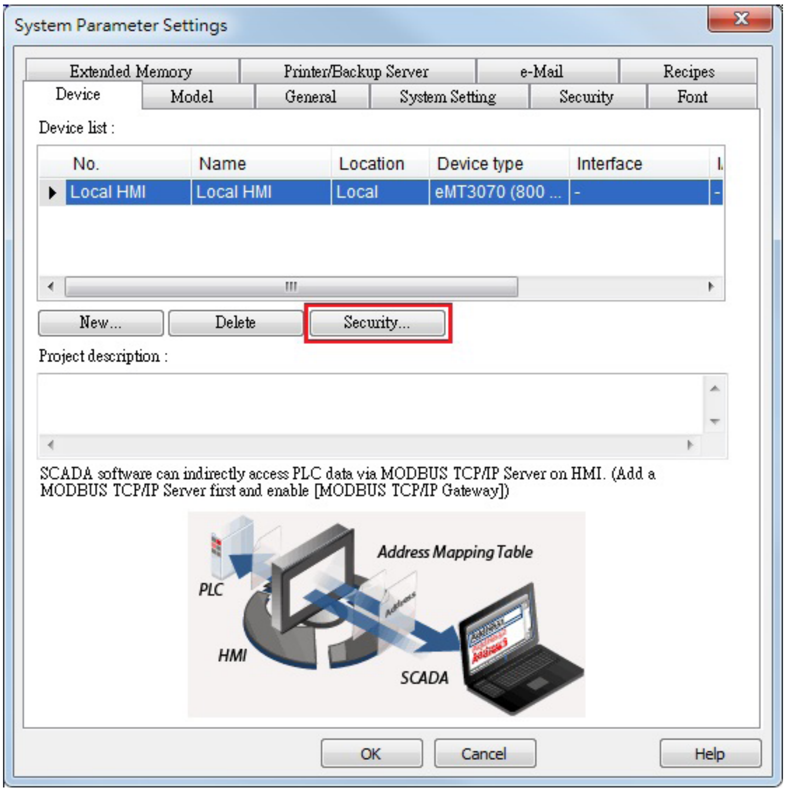
- The parent directory name changes according to the model used.

39. DATA SECURITY

This chapter explains how to setup Data Security.

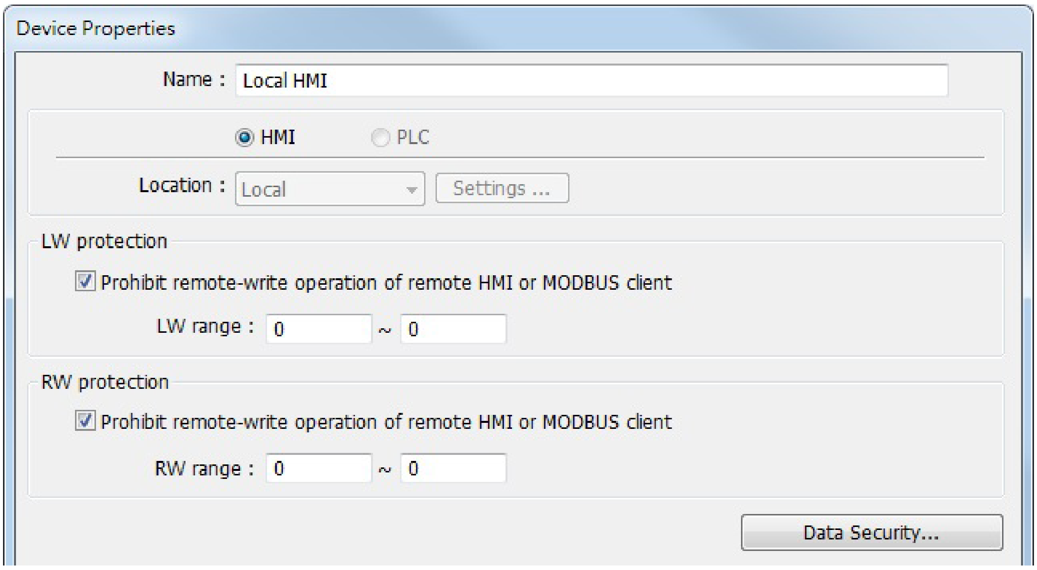
39.1. OVERVIEW

Data Security allows setting restrictions on the write operations that modify local Word or Bit register data. To do so, open (System Parameter Settings) » (Device) tab, select (Local HMI) and then click (Security...) button.



39.2. CONFIGURATION

The following is the settings dialog box:

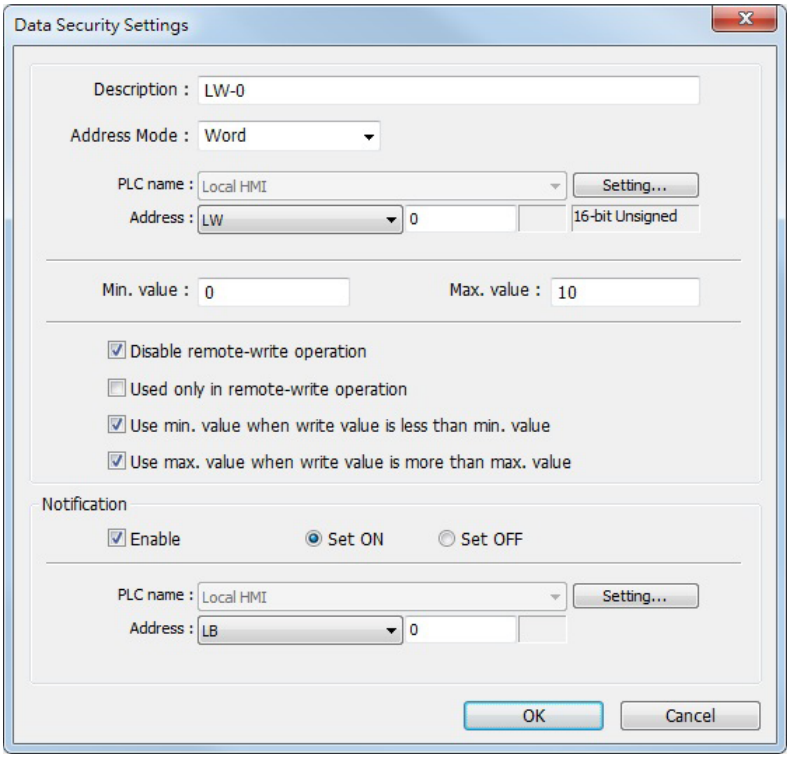


Setting	Description
LW protection \ Prohibit remote-write operation of remote HMI or Modbus client	If selected, a remote HMI or Modbus client will not be able to write to the specified LW addresses.
RW protection \ Prohibit remote-write operation of remote HMI or Modbus client	If selected, a remote HMI or Modbus client will not be able to write to the specified RW addresses.

Click (Data Security) button to set the restrictions on the write operations that modify local Word or Bit register data.

39.2.1. Word Address Settings

Set the restrictions relevant to local Word addresses.

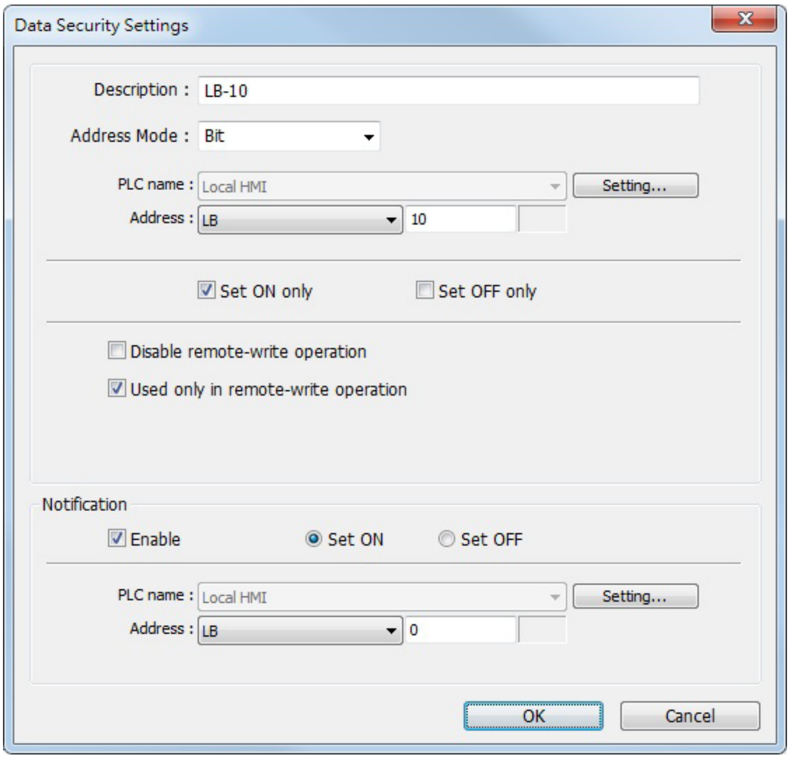


Setting	Description
Description	Enter the description or memo about this setting.
Address mode	Select (Word) to set the relevant attributes.
Min. value	Set the minimum value that can be written to the designated word address.
Max. value	Set the maximum value that can be written to the designated word address.
Disable remote-write operation	If selected, the remote HMI will not be able to write to the protected address.
Used only in remote-write operation	If selected, the range between (Min. value) and (Max. value) is only used to restrict the value written by a remote device.
Use min. value when write value is less than min. value	If selected, when the written value is less than (Min. value), the system will write the specified minimum value instead. If not selected, when the written value is less than (Min. value), the system will keep the original value.
Use max. value when write value is more than max. value	If selected, when the written value is greater than (Max. value), the system will write the specified maximum value instead. If not selected, when the written value is greater than (Max. value), the system will keep the original value.
Notification	When the written value is not within the specified range between (Min. value) and (Max. value), the system will trigger the designated notification bit address.

As shown in the preceding figure, the remote HMI will not be able to write to LW-0, and when the value written to the local address is greater than 10, the value 10 is written instead, and the notification bit LB-0 will be set ON.

39.2.2. Bit Address Settings

Set the restrictions relevant to local Bit addresses.



Setting	Description
Description	Enter the description or memo about this setting.
Address mode	Select (Bit) to set the relevant attributes.
Set ON only	If selected, the designated bit address can only be set ON.
Set OFF only	If selected, the designated bit address can only be set OFF.
Disable remote-write operation	If selected, the remote HMI will not be able to write to the protected address.
Used only in remote-write operation	If selected, the specified condition is only used to restrict the write operation by a remote device.
Notification	When enabled and: ▪ Select (Set ON), the system will trigger the notification bit address when attempting to set OFF the protected bit address. ▪ Select (Set OFF), the system will trigger the notification bit address when attempting to set ON the protected bit address.

As shown in the preceding figure, the remote HMI can only set ON LB-10, while the local HMI is not restricted. If the remote HMI attempts to set LB-10 OFF, the system will trigger notification bit LB-0 ON.

40. COMPARISON OF HMI SOFTWARE FEATURES

- eMT Series: eMT3070A, eMT3105P, eMT3120A, eMT3150A
- cMT Series: cMT-SVR100, cMT-iV5
- mTV Series: mTV-100
- iE Series: MT8050iE, MT8070iE, MT8100iE, MT8121iE, MT8150iE, MT6070iE
- XE Series: MT8121XE, MT8150XE

Series	eMT	cMT	mTV	iE	XE
Project size	64MB	32MB	64MB	16MB	64MB
History data size	64MB	(*3)	64MB	16MB	64MB
Embed pictures in project	√	√	√	√	√
Embed PLC tag information in project	√	√	√	√	√
Project protection	√	(*4)	√	√	√
Enhanced security	√	√	√	√	√
Recipe database / View database	√	√	√	√	√
Operation log / Operation view	√	N/A	N/A	√	√
eMail	√	√	√	√	√
Media player	(*1)	N/A	N/A	N/A	√
Audio output	√	(*5)	N/A	N/A	N/A
Video input	(*2)	N/A	N/A	N/A	N/A
USB CAM	√	N/A	√	N/A	√
Circular trend display	√	√	√	√	√
Combo button	√	√	√	√	√
CAN Bus	√	N/A	N/A	N/A	N/A
VNC	√	N/A	√	√	√
Download project via USB	√	N/A	N/A	(*6)	√
Download project via ethernet	√	√	√	(*7)	√
User-defined boot screen	√	N/A	√	√	√
EasySystemSetting	√	N/A	√	√	√

Note
1. eMT3105P/eMT3120A/eMT3150A support Media Player.
2. eMT3120A/eMT3150A support Video Input.
3. Maximum: 40 Data Samplings, 10,000 records for each. cMT-iV5 is not included.
4. cMT-iV5 does not support Project Protection.
5. cMT-iV5 has built-in mono speaker.
6. MT6070iE/MT8121iE/MT8150iE support downloading project via USB cable.
7. MT6070iE does not support downloading project via Ethernet.

WEG Worldwide Operations

ARGENTINA
San Francisco - Cordoba
Phone: +54 3564 421484
info-ar@weg.net

Cordoba - Cordoba
Phone: +54 351 4641366
weg-morbe@weg.com.ar

Buenos Aires
Phone: +54 11 42998000
ventas@pulverlux.com.ar

AUSTRALIA
Scoresby - Victoria
Phone: +61 3 97654600
info-au@weg.net

AUSTRIA
Markt Piesting - Wiener Neustadt-Land
Phone: +43 2633 4040
watt@wattdrive.com

BELGIUM
Nivelles - Belgium
Phone: +32 67 888420
info-be@weg.net

BRAZIL
Jaraguá do Sul - Santa Catarina
Phone: +55 47 32764000
info-br@weg.net

CHILE
La Reina - Santiago
Phone: +56 2 27848900
info-cl@weg.net

CHINA
Nantong - Jiangsu
Phone: +86 513 85989333
info-cn@weg.net

Changzhou – Jiangsu
Phone: +86 519 88067692
info-cn@weg.net

COLOMBIA
San Cayetano - Bogota
Phone: +57 1 4160166
info-co@weg.net

ECUADOR
El Batan - Quito
Phone: +593 2 5144339
ceccato@weg.net

FRANCE
Saint-Quentin-Fallavier - Isère
Phone: +33 4 74991135
info-fr@weg.net

GERMANY
Türnich - Kerpen
Phone: +49 2237 92910
info-de@weg.net

Balingen - Baden-Württemberg
Phone: +49 7433 90410
info@weg-antriebe.de

Homberg (Efze) - Hesse
Phone: +49 5681 99520
info@akh-antriebstechnik.de

GHANA
Accra
Phone: +233 30 2766490
info@zestghana.com.gh

INDIA
Bangalore - Karnataka
Phone: +91 80 41282007
info-in@weg.net

Hosur - Tamil Nadu
Phone: +91 4344 301577
info-in@weg.net

ITALY
Cinisello Balsamo - Milano
Phone: +39 2 61293535
info-it@weg.net

JAPAN
Yokohama - Kanagawa
Phone: +81 45 5503030
info-jp@weg.net

MALAYSIA
Shah Alam - Selangor
Phone: +60 3 78591626
info@wattdrive.com.my

MEXICO
Huehuetoca - Mexico
Phone: +52 55 53214275
info-mx@weg.net

Tizayuca - Hidalgo
Phone: +52 77 97963790

NETHERLANDS
Oldenzaal - Overijssel
Phone: +31 541 571080
info-nl@weg.net

PERU
La Victoria - Lima
Phone: +51 1 2097600
info-pe@weg.net

PORTUGAL
Maia - Porto
Phone: +351 22 9477700
info-pt@weg.net

RUSSIA and CIS
Saint Petersburg
Phone: +7 812 363 2172
sales-wes@weg.net

SOUTH AFRICA
Johannesburg
Phone: +27 11 7236000
info@zest.co.za

SPAIN
Coslada - Madrid
Phone: +34 91 6553008
wegiberia@wegiberia.es

SINGAPORE
Singapore
Phone: +65 68589081
info-sg@weg.net

Singapore
Phone: +65 68622220
watteuro@watteuro.com.sg

SCANDINAVIA
Mölnlycke - Sweden
Phone: +46 31 888000
info-se@weg.net

UK
Redditch - Worcestershire
Phone: +44 1527 513800
info-uk@weg.net

UNITED ARAB EMIRATES
Jebel Ali - Dubai
Phone: +971 4 8130800
info-ae@weg.net

USA
Duluth - Georgia
Phone: +1 678 2492000
info-us@weg.net

Minneapolis - Minnesota
Phone: +1 612 3788000

VENEZUELA
Valencia - Carabobo
Phone: +58 241 8210582
info-ve@weg.net

For those countries where there is not a WEG own operation, find our local distributor at www.weg.net.

WEG Group - Automation Business Unit
Jaraguá do Sul - SC - Brazil
Phone: +55 47 3276 4000
automacao@weg.net
www.weg.net

Rev: 00 | Date (m/y): 05/2015