HMI Programming Manual

Shenzhen Honyee Technology Co., Ltd
Version 24001

CONTENTS

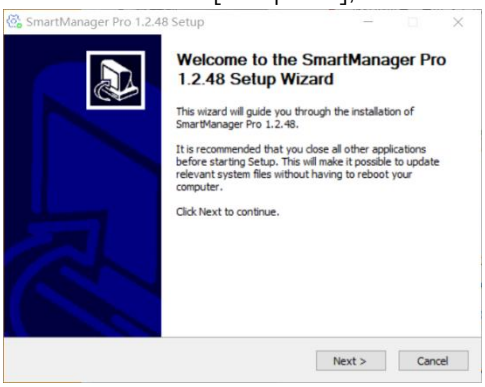
Chapter 1 SmartManager Pro Installation

1.1 Requirement

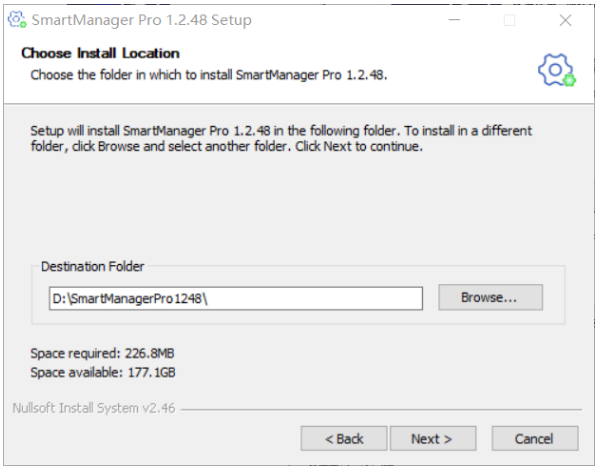
Operation System:
Windows 7 (32bit / 64bit)
Windows 8 (32bit / 64bit)
Windows 8.1 (32bit / 64bit)
Windows 10 (32bit / 64bit)

1.2 Steps to Install SmartManager Pro.

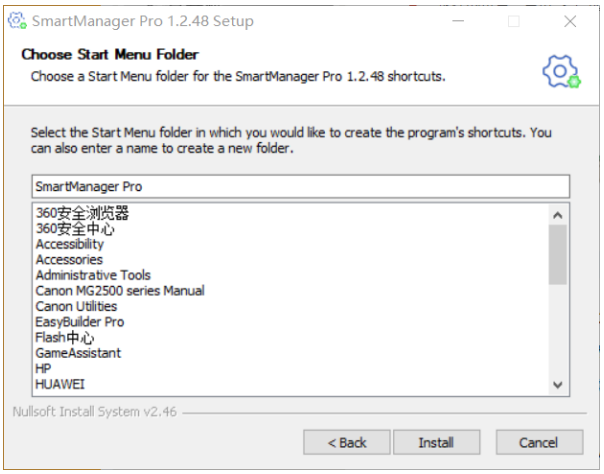
1. Double click [setup.exe], and then click [Next].



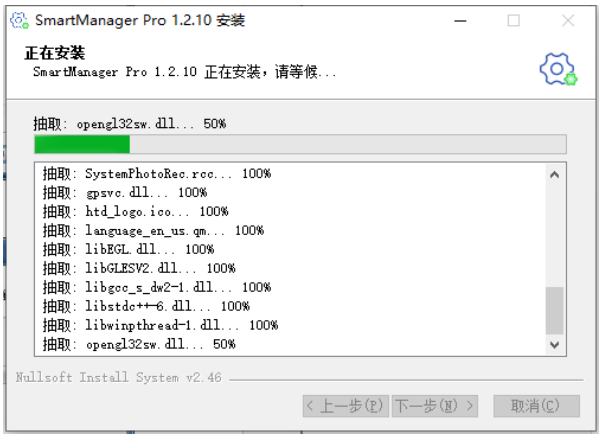
2. Select a or just use the system suggests folder for SmartManager Pro installation, click [Next].



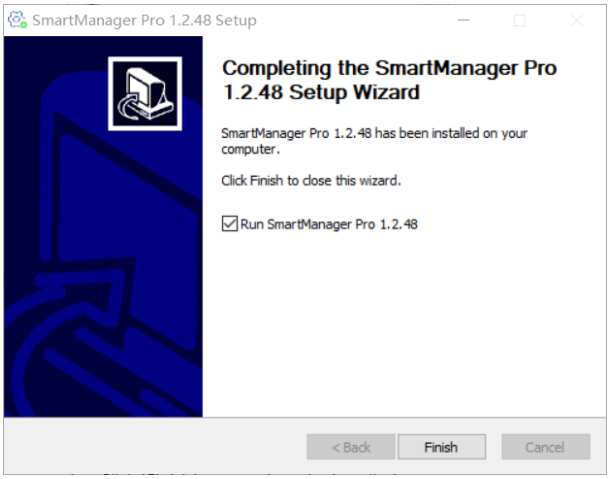
3. Select a start menu folder, or use the default folder. Click [Install] to start installation.



4. In installation.



5. Click [Finish] to complete the installation.



Chapter 2 Creat a SmartManager Pro Project

This chapter explains the basic steps to create an SmartManager Pro (Hereinafter referred to as "this software") project.

2.1 Overview

The following is the process of creating a project.

(1) Create a new project file.

(2) Save and compile the project file.

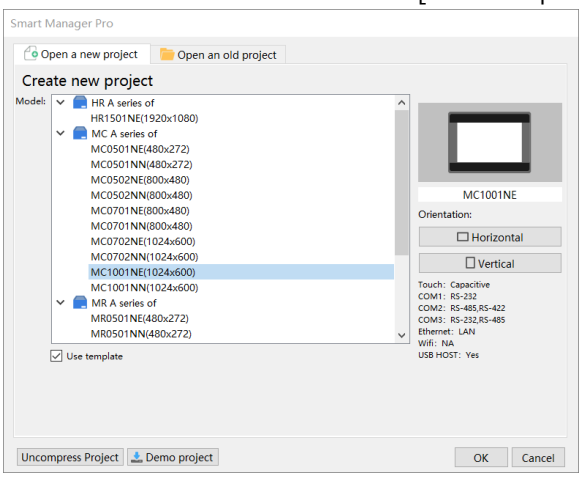
(3) Run On-line or Off-line simulation.

(4) Download the project file to HMI.

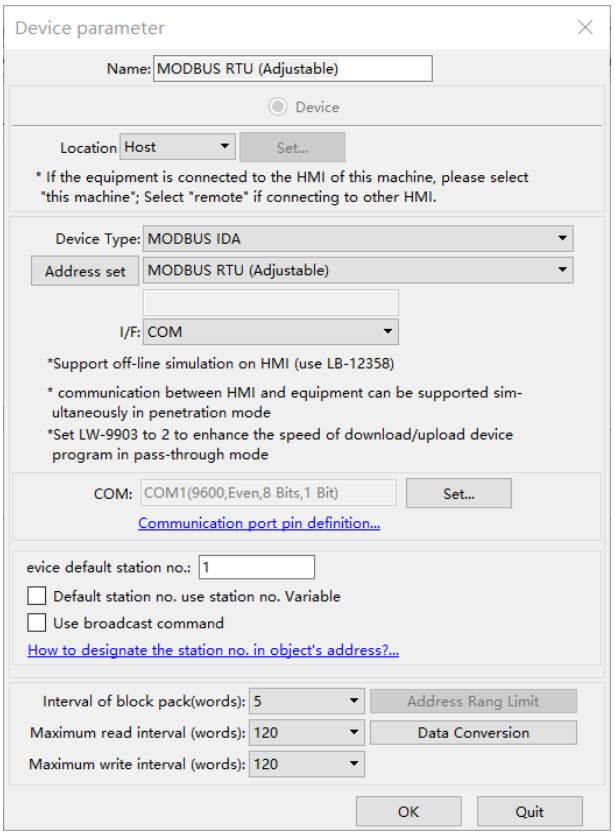
The following describes each process.

2.2 Create a new project file.

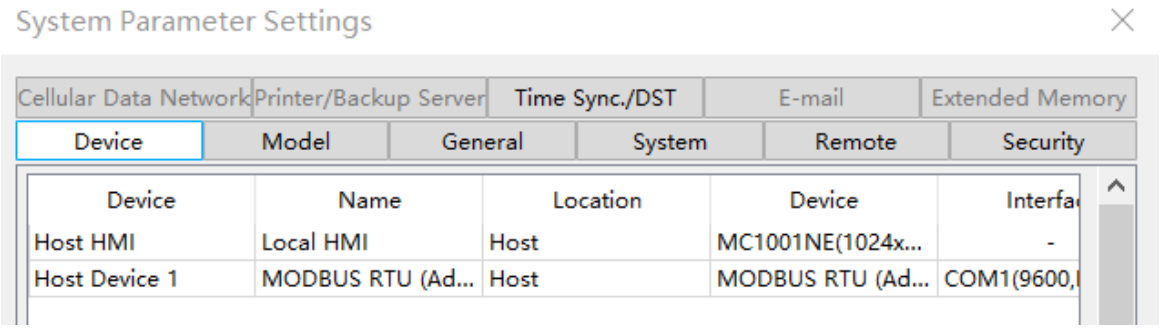
1. Launch SmartManager Pro and open a new project file.
2. Select a model and select [Use template] check box.



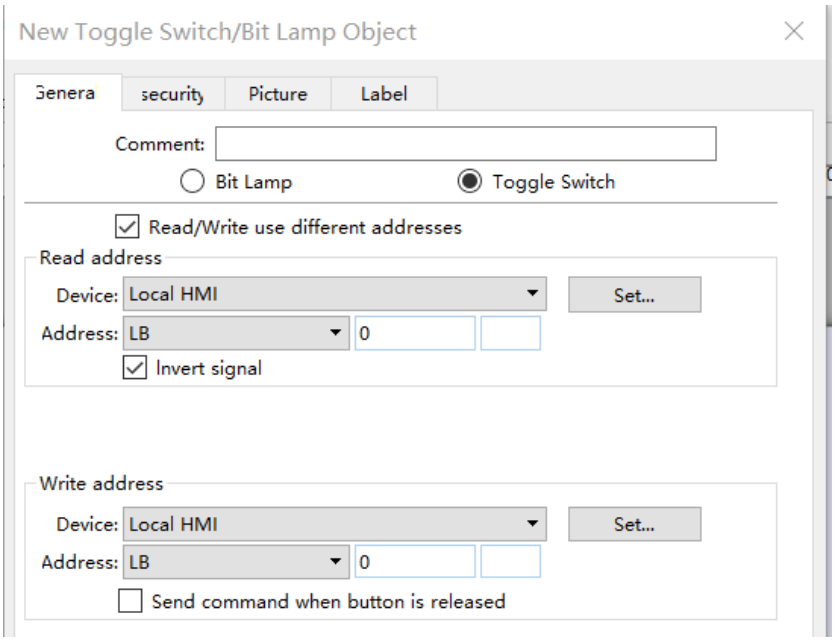
3. Click [New device] in [Home]>>[System Parameter]>>[Device] to select a device.
4. Configure parameters.



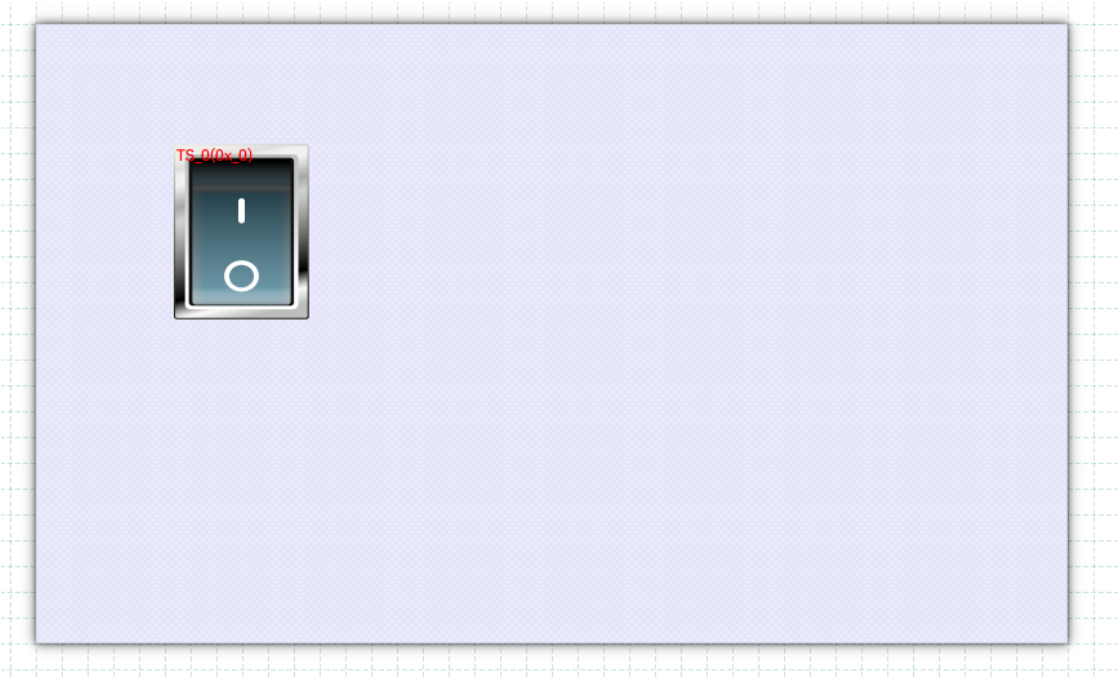
5. A new device is added to the [Device List].



6. Create an object, for example, a Toggle Switch, and then set the address.

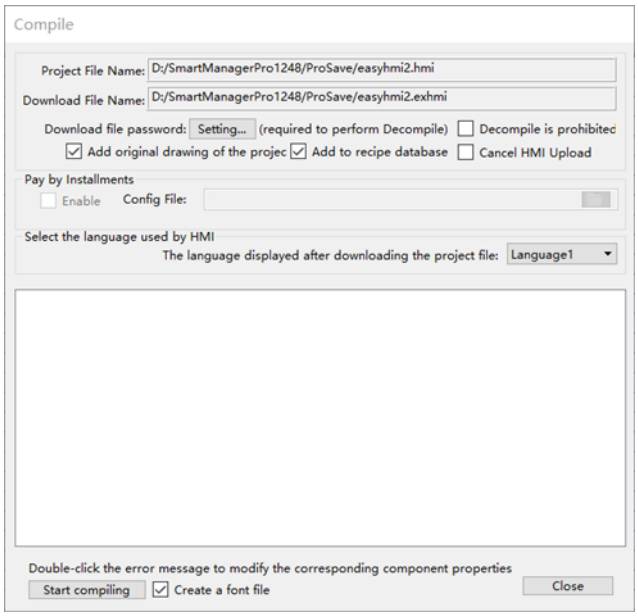


7. Place the object in the edit window. A project is now created.



2.3 Save and Compile the Project File

1. In this software toolbar click [File] » [Save] to save the .hmi file.
2. In this software toolbar click [Project] » [Compile] to compile .hmi file as .exhmi file, which could be downloaded to HMI. This also checks if the project can run correctly.
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.



2.4 Run On-Line or Off-Line Simulation

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

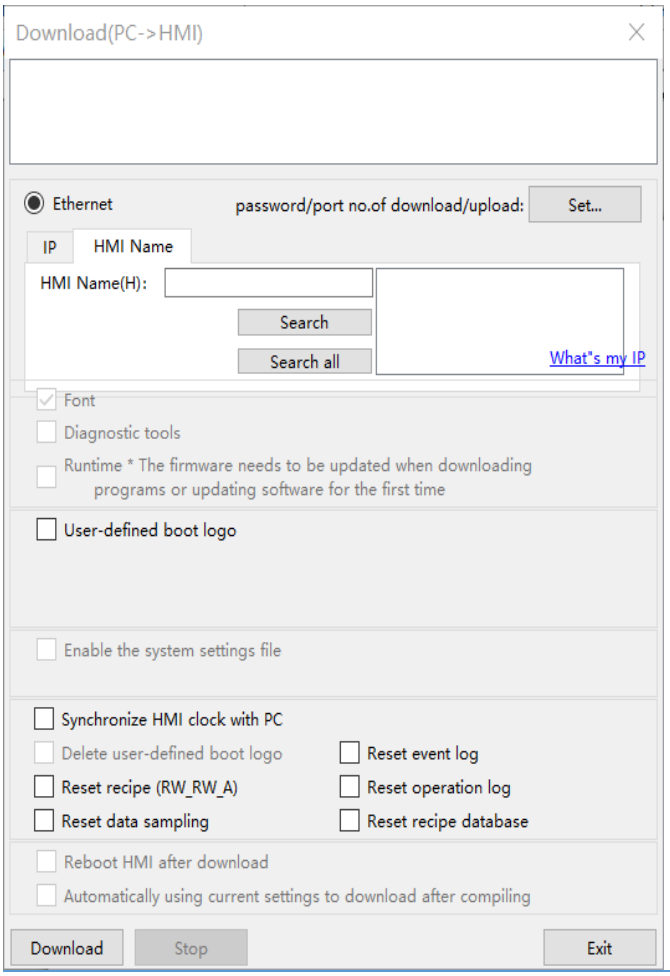
Off-line simulation: Simulate project operation on PC without the need for connecting any device..

2.5 Download the Project File to HMI

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

2.5.1 Download by assigned IP address

1. In this software toolbar, click [Project] » [Download]. Make sure that all the settings are correct.
2. Select [Ethernet], set password and HMI IP.

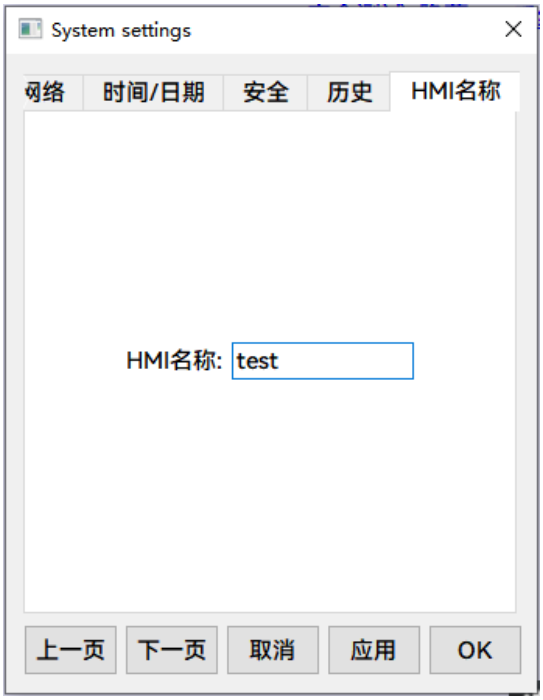


Setting	Description
Font	Download the font used in the project to HMI.
Diagnostic tools	
Runtime	Select the check box to update the HMI kernel programs. If it is the first time downloading file or this software version is updated, please download the firmware before downloading files to HMI
User-defined boot logo	Download the selected .bmp picture to HMI as the startup screen.
Enable the system settings file	Download the system settings file to update hardware settings
Enable installent payment	
Synchronize HMI clock with PC	Synchronize HMI time with PC time when downloading project file.
Reset recipe/ event log/ data sampling/ recipe database/ operation log/ Delete user-defined boot logo	The selected files will be erased before downloading.
Reboot HMI after download	HMI will reboot after the downloading process is done.

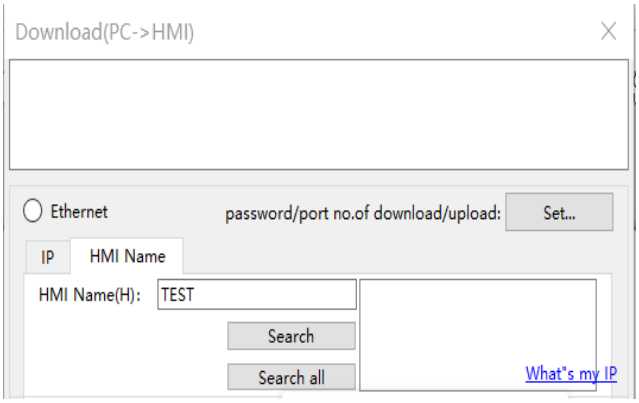
Automatically using current settings to download after compiling	If select this check box, the system will compile the project and download it to the latest target HMI if click [download].

2.5.2 Use HMI Name

1. Go to [System settings] on HMI and then set HMI name first.



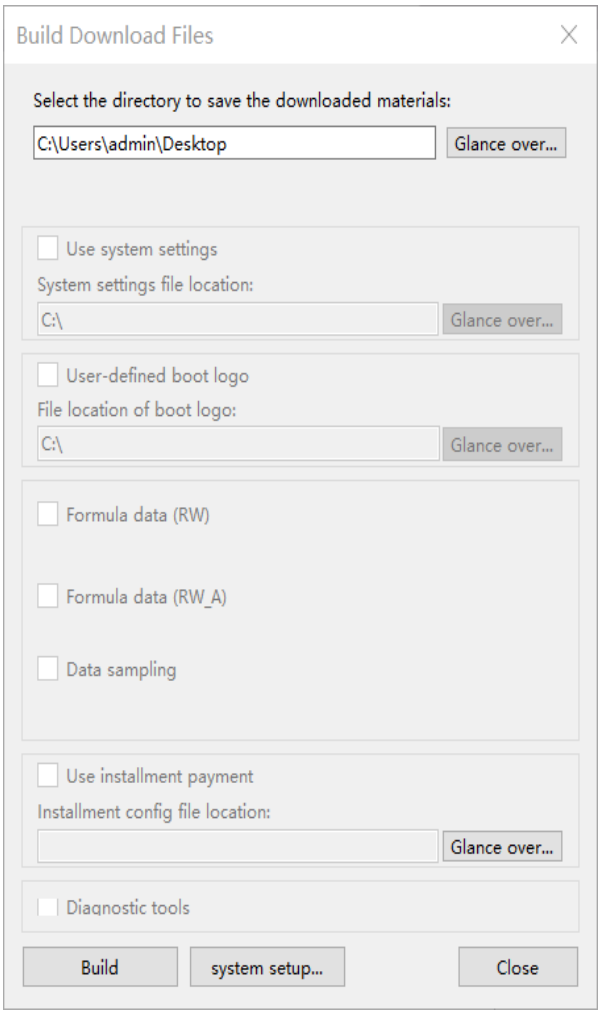
2. In this software toolbar, click [Project]>>[Download(PC->HMI]>>[Ethernet]>>[HMI Name] and start downloading. To use [Search], enter the HMI name first to search for the HMI. [Search all] searches for all HMIs in the same subnet network.



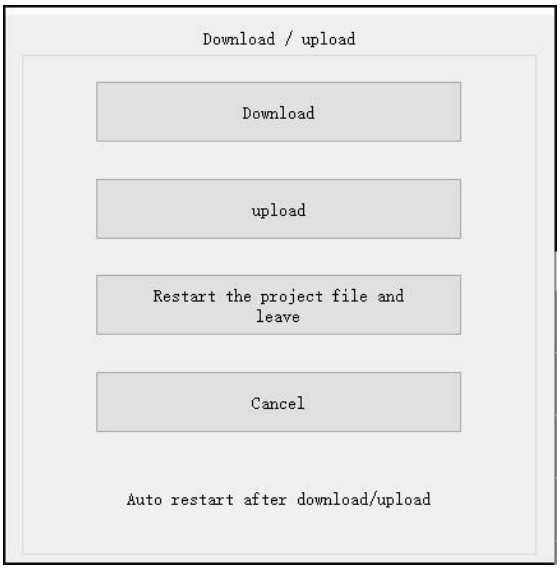
2.5.3 Use USB Drive / SD card

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

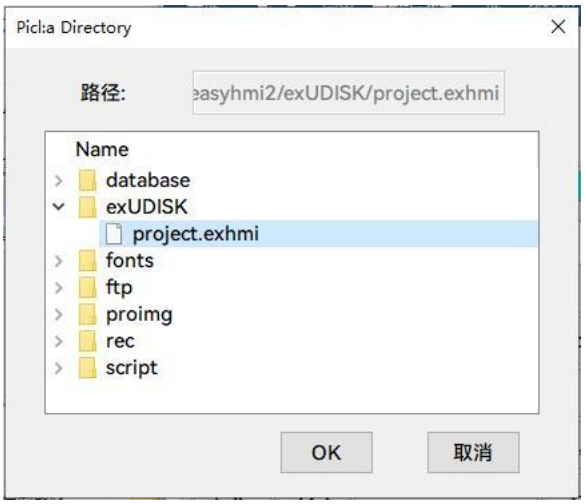
1. On this software toolbar, click [Project] » [Build download file]. Browse for the project file and then click [Build], the file for download will be generated in the connected external device.



2. Insert an external device to HMI.
3. Select [Download] on HMI and enter password.



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



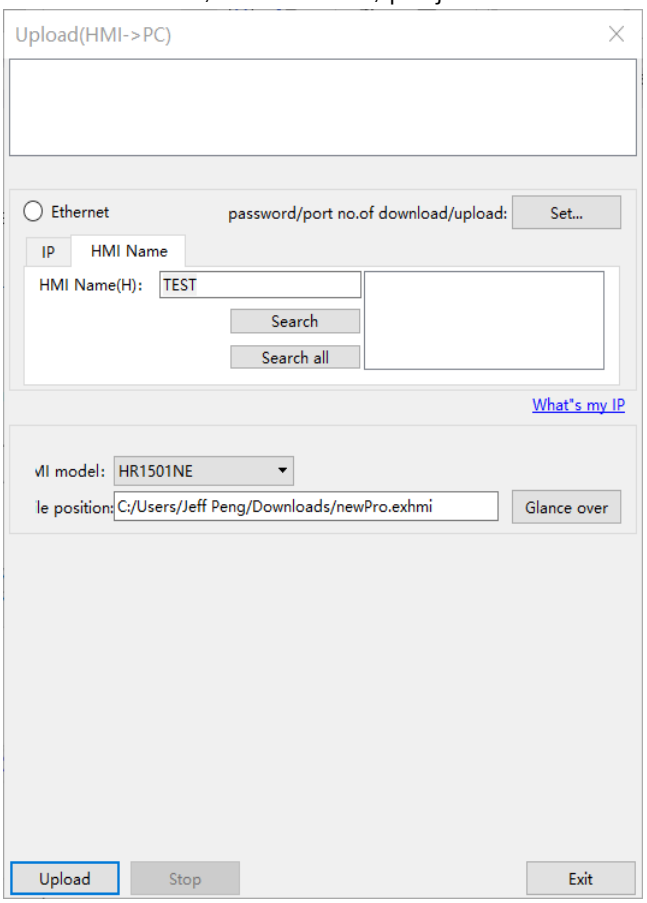
5. Select .exhmi project file, then press [OK] to start downloading.

Note

- It is normal that HMI will restart during downloading. Do not pull out the external device during restart. After HMI restart, unplug the external device when the project display is correct.

2.6 Upload the Project File from HMI

1. On this software toolbar, click [File]>>[Upload].
2. Set HMI IP, HMI model, project location and project name, and then click [Upload].



Chapter 3 Hardware Settings

3.1 I/O Ports

The I/O ports may vary between models; please refer to the datasheet for more information. The I/O Ports include,

- SD card slot: Allows downloading / uploading project file, including recipe data, event log, data log…etc. and also allows recording history data and data backup.
- COM Port: Connects PLC or other peripheral devices. The type of serial port include: RS-232, RS-485 2W, RS-485 4W.
- Ethernet: Allows downloading / uploading project file, including recipe data, event log, data log…etc. Connects to Ethernet devices, such as PLC, laptop.

3.2 Touch Panel Settings

3.2.1 Reset HMI to get back default setting.

Each HMI supports Reset function. If we lost the password of HMI system setting, we can reset the HMI to get back the factory default setting. But all project files will also be cleared. The following are steps to set,

1. Power on HMI, keep pressing on the screen and waiting for 5~8 seconds, then the HMI will enter 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, the system will display two buttons, one is Run Project, the other is Clear Project. Click Clear Project, project files and history records stored in HMI will all be removed. The password is also restored to the factory password: 111111.

3.2.2 Daily Calibration

Daily calibration will improve the sensitive of touch screen. Please refer to the steps in 3.2.1 and only in the third step, click Run Project.

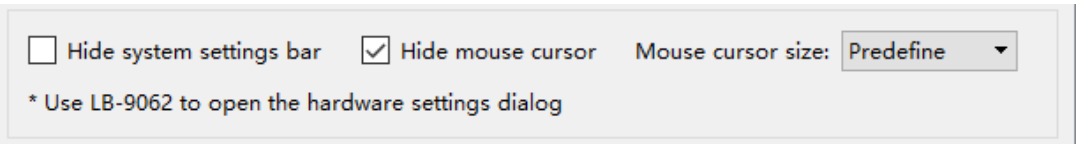
.

3.3 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:

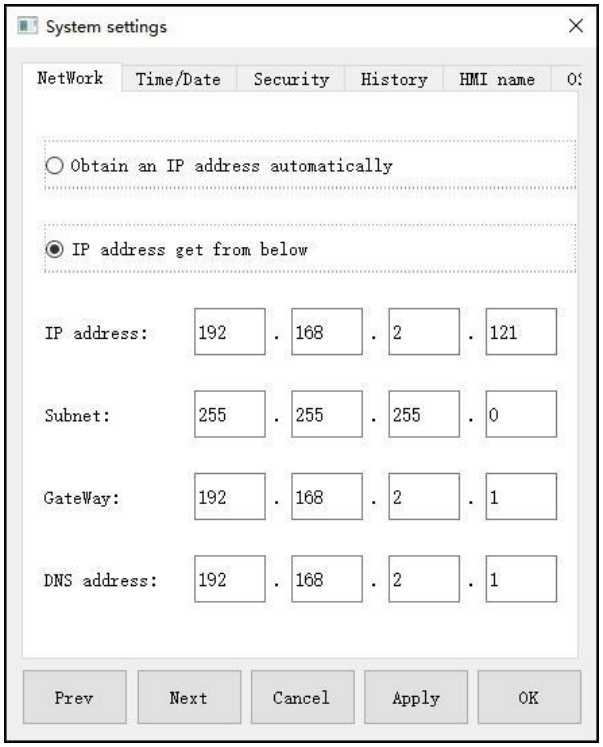
- On this softwar toolbar, click [Home]>>[System Parameters]>>[System Settings], check [Hide System Settings Toolbar], and click [OK]. the system setting toolbar is disabled. When the project is downloaded to HMI, the system setting column of HMI will be hidden



- 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.

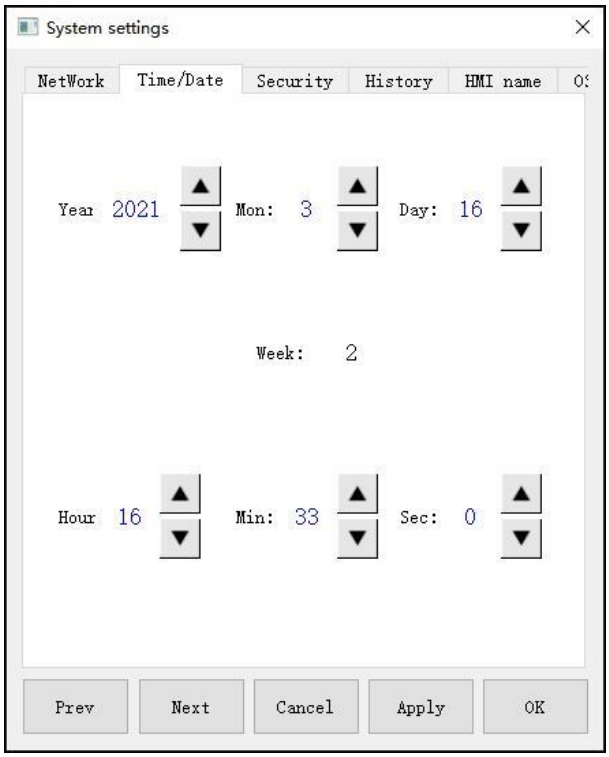
3.3.1 System Settings

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



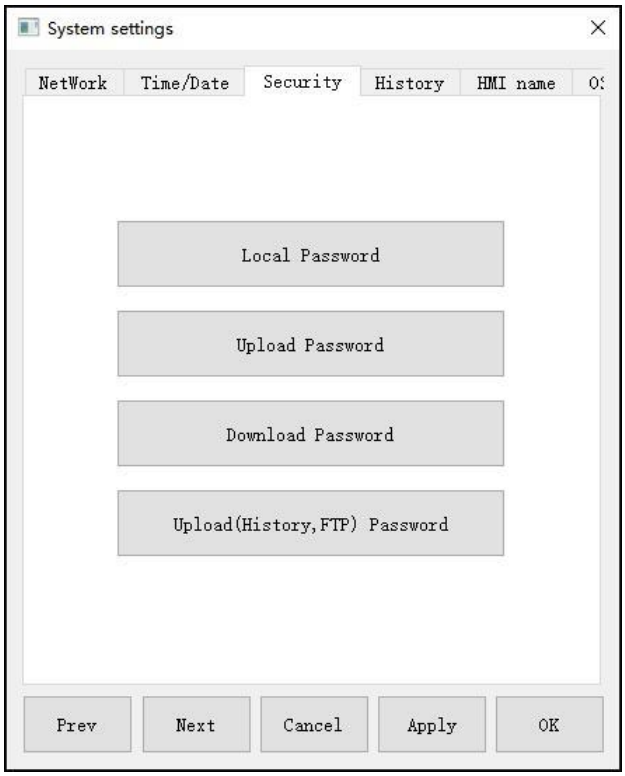
Network

Download project file to HMI via Ethernet. The IP address of the target device (HMI) must be correctly set. Users can choose from obtaining an IP address automatically or entering an IP address manually.



Time/Date

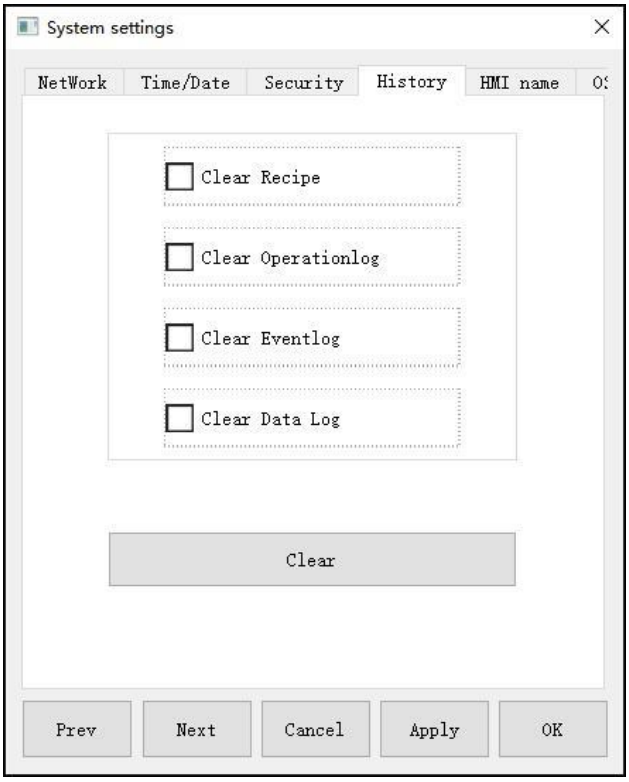
Set HMI local date and time.



Security

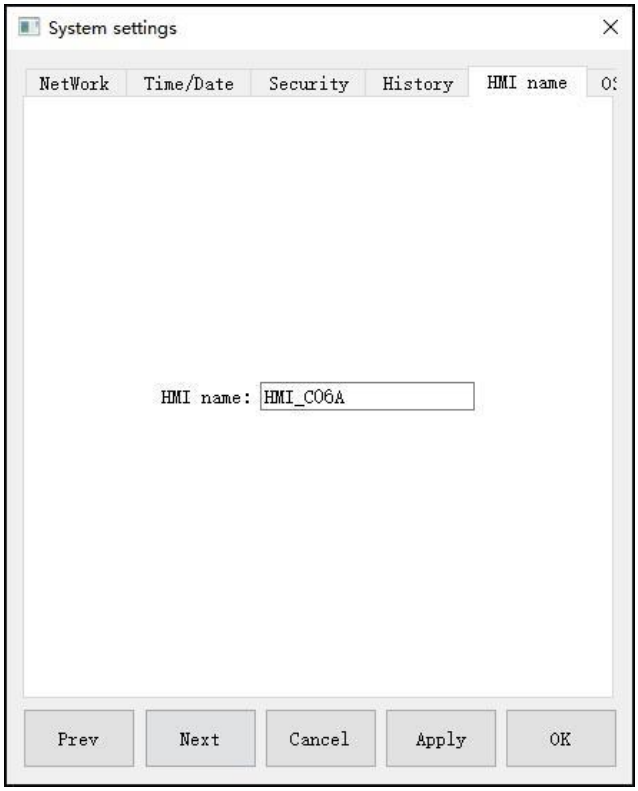
Srt HMI's Password. The factory default password is 111111. User-defined passwords can be used after verification.

[Local Password] Set by system
[Upload Password] for uploading file to PC
[Download Password] for downloading files to HMI
[Upload (History, FTP) Password] for uploading history records to PC.



History

Clear historical data stored in HMI.
[Clear Recipe]
[Clear Operationlog]
[Clear Eventlog]
[Clear Data Log]

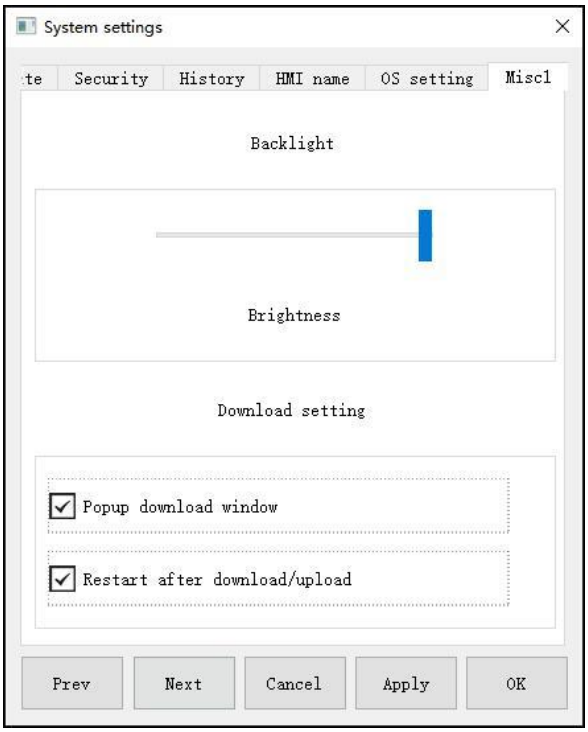


HMI name

Set the HMI name for identification when downloading / uploading a project.

OS setting

[Upgrade OS]
Upgrade firmware. Please do not turn off or unplug HMI during OS upgrade process.
[Portrait Mode]
Set screen orientation mode. After changing the mode, reconnect HMI to power supply, for the setting to take effect. That is, disconnect all power from HMI, and then connect again. If portrait mode is used (90 or 270 degree), the project must be designed for portrait mode, otherwise it cannot be correctly displayed.



Misc 1

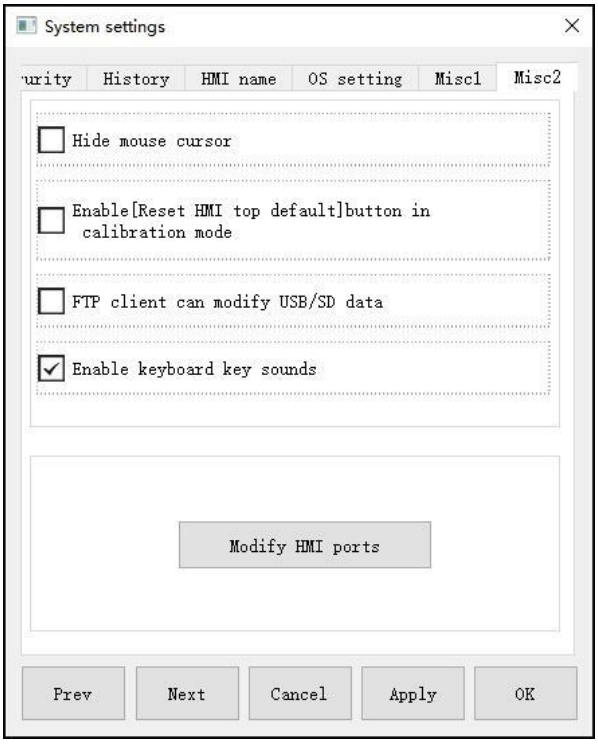
Slide switch for adjusting LCD brightness.

[Popup download window]

If selected, after inserting USB disk or SD card to HMI, the Upload / Download dialog box shows.

[Restart after download/upload]

If selected, restarts HMI automatically after uploading / downloading project.



Misc 2

[Hide mouse cursor]

If selected, the mouse cursor will be hidden.

[Enable [Reset HMI to default] button in calibration mode]

If selected, when the operator presses and holds anywhere on the screen for more than 2 seconds during HMI startup, the touch screen calibration mode will start. After calibration, [Reset HMI to default] option shows.

[FTP client can modify USB/SD data]

If selected, USB/SD data can be modified using FTP

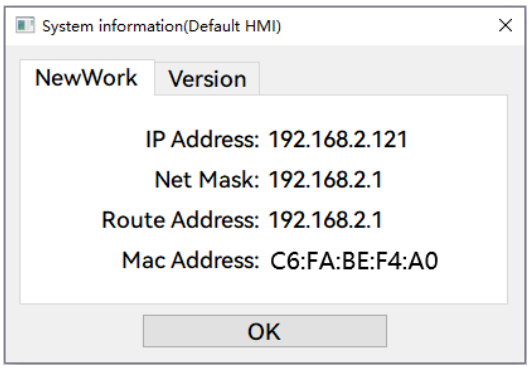
[Enable keyboard key sounds]

[Modify HMI ports]

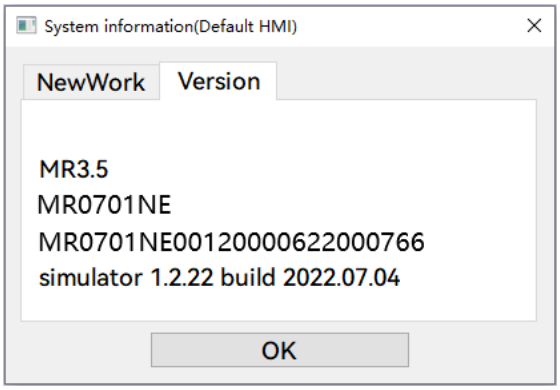
Change the port number for Upload/Download and FTP.

3.2.2 System Information

Network: Displays network information & HMI IP address.



Version: Displays HMI firmware version and model type.



Chapter 4 System Parameters Settings

4.1 Overview

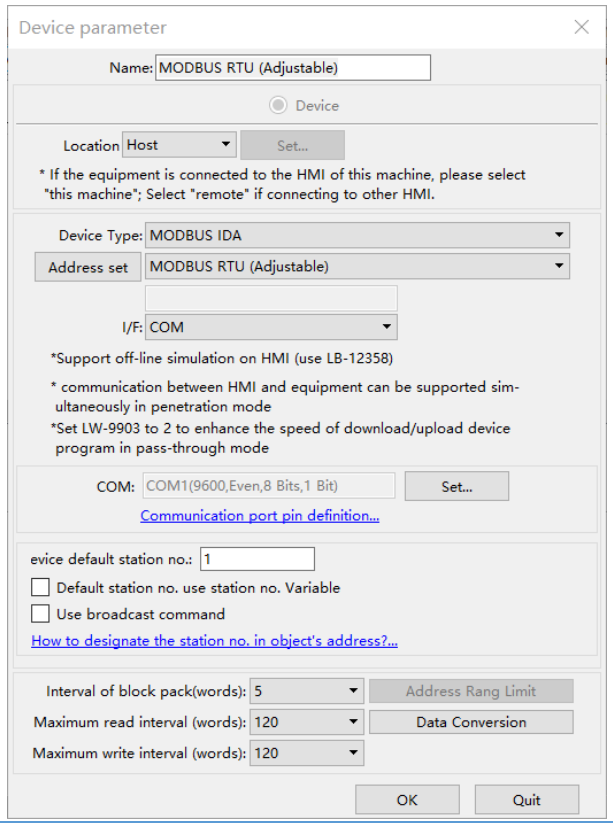
In the main menu of this software, select [Home] » [System Parameters] to open the [System Parameter Settings] dialog box. System Parameter Settings are divided into several tabs, this chapter will introduce these tabs respectively.

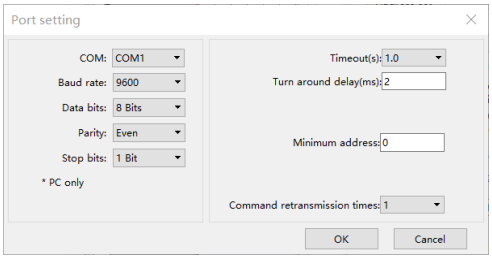
4.2 Device

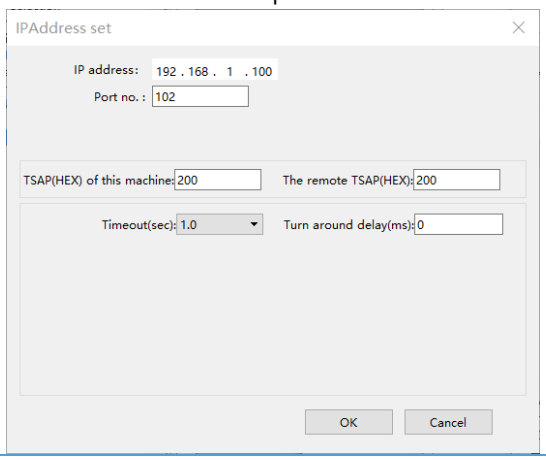
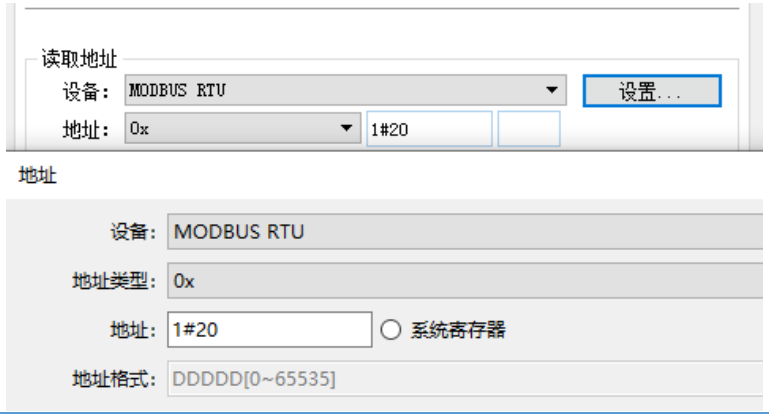
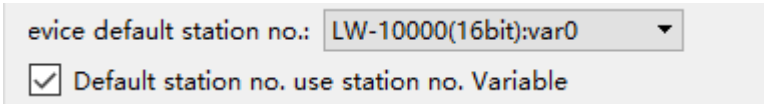
Parameters in this tab determine the attributes of each device connected with HMI. The device can be a Local / Remote HMI / PLC. A default device "Local HMI" exists when creating a new project. This device is the HMI that will be updated and programmed.

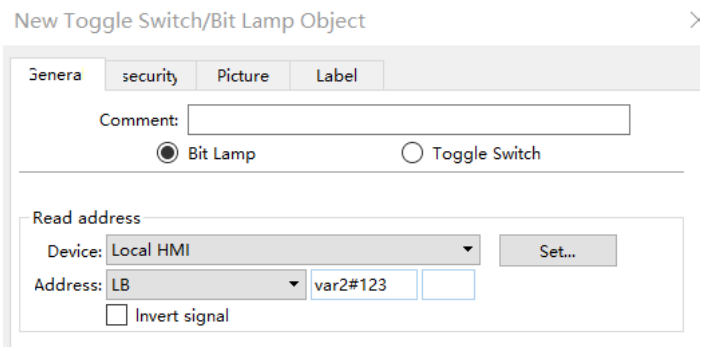
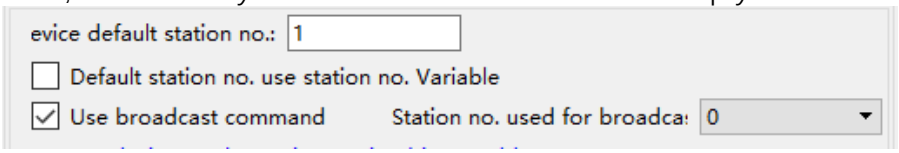
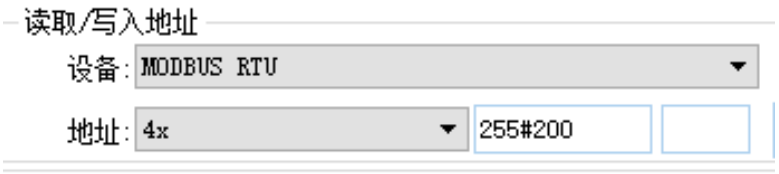
4.2.1 How to Control a Local PLC

“Local Device” is a device connected to the local HMI. To control/connect a Local Device, add this type of device first. Click [System Parameters] » [New] to open [Device Properties] dialog box. For example, when connecting MODBUS RTU as a Local Device:



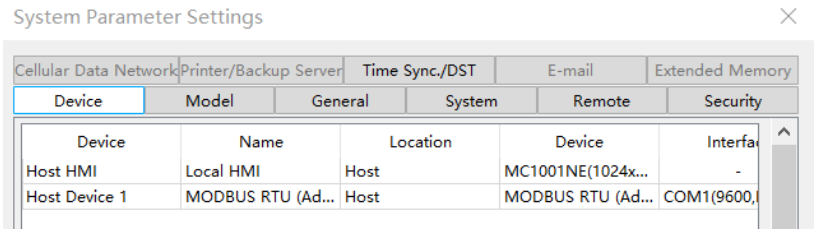
Setting	Description
Name	The name of the device
HMI or Device	In this example a PLC is used, so select [Device].
Location	Select [Local] or [Remote. In this example the PLC is connected to local HMI, so select [Local].
Device type	Select the type of PLC.
I/F	The available interface: [Com], [Ethernet], ● If the interface used is [Com], 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 “Device No Response” message. **Turn around delay** While sending the next command to the device, 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 the device. Default value is “0”.

	Note: If the device 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 device IP address and port number. 
Device default Station No.:	Set the default station number for device address if the device station number is not included in the address. Device station no. can be set in its address. The address format: ABC#Addr. ABC stands for device station number and ranges from 0 to 255. Addr stands for device address. And the “#” sign separates the station number and the address. As shown in the following figure, the data is read from device station number 1, and address 0x-20. 
Default station no. use station no. variable	If checked, the station number variable list will be enable for selection.  When setting Read or Write Address, the station number variable is allowed to be used in the address field. As shown in below picture, where "var2" is one of the 16 station number variables.

	 Syntax of station number variable: varN # address N is an integer from 0 to 15, and address is the Read or Write Address. System provides 16 station number variables var0~var15, and their actual data come from the system registers LW-10000~LW-10015. For example, since the data of "var2 # 123" is read from LW-10002, when the value in LW-10002 is "3", it means that var2 # 123 is equivalent to 3 # 123, that is, the station number is Station 3.
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 below figure,  When HMI sends a command to address 255#200, all the devices will receive this command and will not reply. Only devices 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	The maximum data size to read from the device at one time. Unit: word

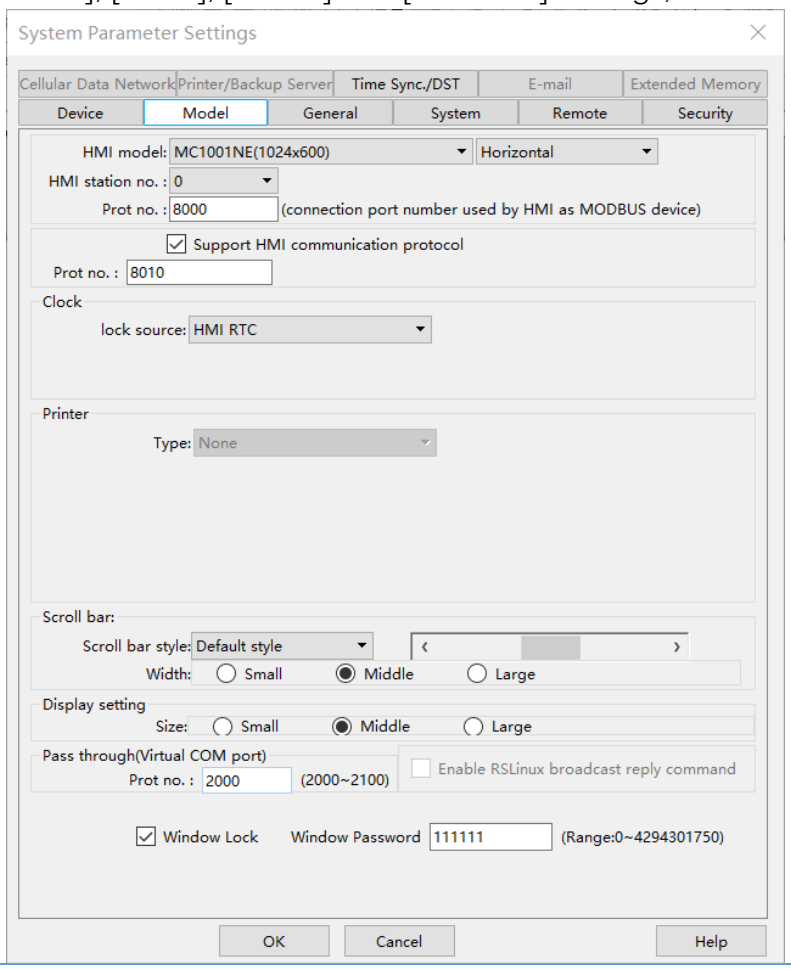
(words)	
Max. Write - command size (words)	The maximum data size to write to the device at one time. Unit: word.

After all settings are completed, a new device named “Local Device 1” is added to the [Device list]

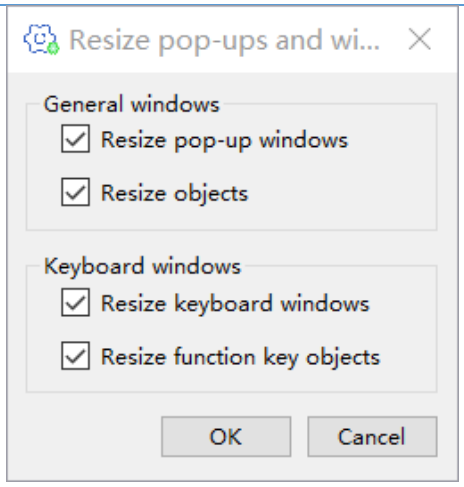


4.2 HMI Model

Configure the [HMI model], [Timer], [Printer] and [Scroll bar] settings, and more.



Settings	Description
HMI Model	Select an HMI model. The [Resize pop-up windows / objects] dialog box will pop up when changing to a model with a different resolution. Select required adjustment and click [OK]. In most cases, select all options.



Landscape/Portrait

Change the orientation of the project.

HMI station no.	Select the station number for HMI. Also can use the default value.
Port Number	Set the port number of current HMI, which is also used for MODBUS server. The default value is “8000”.
Support HMI communication Protocol	Support mutual communication with other series of HMI.
Clock	The interface type used by PLC can be [COM] or [Ethernet]. ● When [Touch Screen Real Time Clock] is selected, the clock of HMI comes from the timer contained in HMI. ● When [External Device] is selected, the clock will come from an external device, and the source address of the clock needs to be set correctly.
Printer	The printer driver can be installed on HMI. And Printer Type will be set as the model number of printer. ● Status Address：shows printer running status.

LW-n	Status
0	Printer driver is not installed yet.
1	Installing printer driver.
2	Printer is ready.
3	Printer is printing
LW-n+1	Error
0	None
1	Printer isn’t found
2	Unknown error.

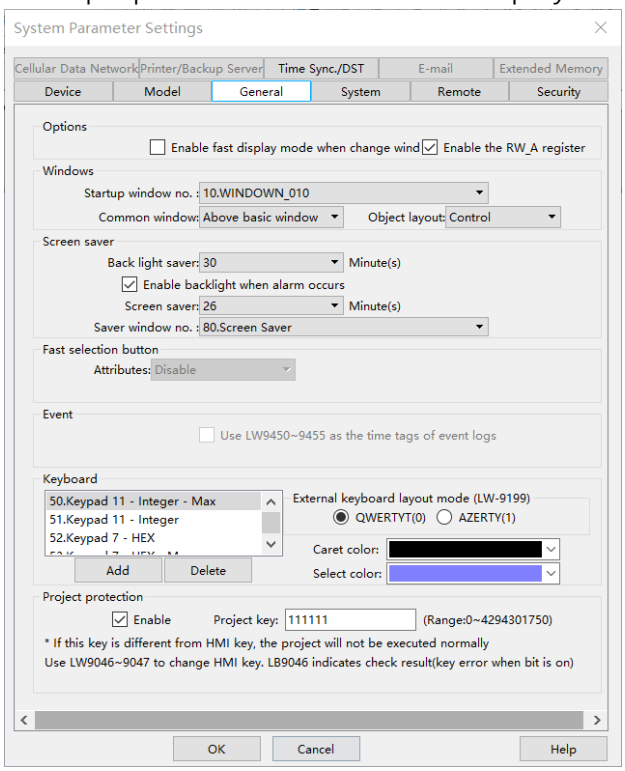
● Control Address：Set & update printer connection parameters.

LW-n	Command
0	None
1	Update printer connection parameters.
LW-n+1	I/O

	0	Ethernet
	1	USB
	LW-n+2	IP address (Total: 4 words)
	LW-n+6	Port (Default value: 9100)
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 (Virtual COM port)	To set the port number for Pass-through communication.	

4.3 General

[General] is used to configure the properties related to screen display.



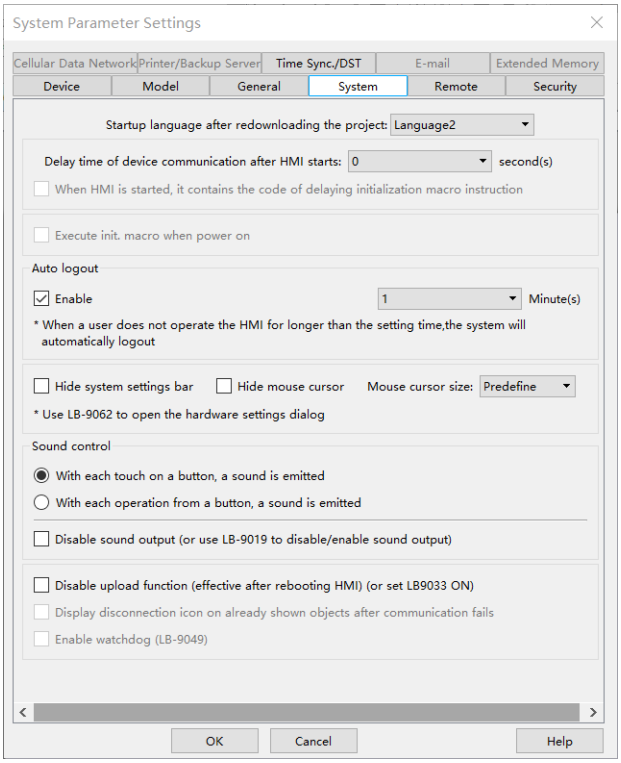
Setting	Description
Options	**Enable fast display mode when change window** With this option selected, certain objects will cache the most recent value and display it after changing window, to achieve fast display. The objects are: Bit Lamp, Word Lamp, Numeric, ASCII, Bar Graph, Meter Display, Animation, and Moving Shape…etc. **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 64K.
Windows	**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.
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 setting 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.** Specify a window to be displayed when HMI goes to screen saver mode.
Fast selection buttons	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 at bottom left side of the screen; if [Right] is chosen, the button will show at the bottom right side of the screen. **Settings** Set the shape and label font of the Fast Selection Button. **Hide button when HMI starts** If check this, the Fast Selection Button will be hidden, calling it out requires system registers LB-9013~LB-9015
Event	**Use LW9450~9455 as time tags of event logs** When this option is selected, Event Log will use the following time tags to show the time that the event is triggered. LW-9450: time tag of event log – second (range: 0~59)

	LW-9451: time tag of event log – minute (range: 0~59) LW-9452: time tag of event log – hour (range: 0~23) LW-9453: time tag of event log – day (range: 1~31) LW-9454: time tag of event log – month (range: 1~12) LW-9455: time tag of event log – year (range: 1970~2037) Please note that the system will not be able to use these time tags when one of them exceeds acceptable range.
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.
Project protection	Projects can be restricted to be executed by a specific HMI.

4.4 System Setting

[System Setting] is used to configure different features of this software.



Some features are duplicated from system registers, such as, [Hide system setting bar (LB-9020)], [Hide mouse cursor (LB-9018)], [Disable buzzer (LB-9019)], and [Disable upload function (LB-9033)]. Users can also set these features via system tag.

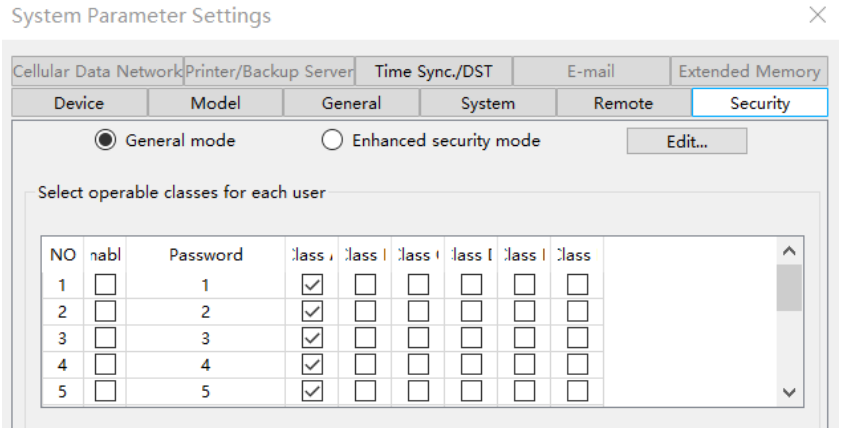
To select a system tag, select [Address] » [System tag] check box when adding a new object and then select the [Address Type]. To browse all the system tags, Select [Project] » [Tag] » [System] from the main menu of this software.

Setting	Description
Startup language after redownloading the project	Set the language to use when HMI starts after the project is re-downloaded.
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 from HMI screen.
Hide mouse cursor	Hide the mouse cursor in HMI screen.
Mouse cursor size	Set mouse cursor size.
Sound Control	**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. With this option selected, the sound is emitted when the object actions. **Disable sound output:** Mute HMI. (Not including the sound played when tapping system setting button on HMI.)
Disable upload function (effective after rebooting HMI) (or set LB9033 ON)	Disable HMI to upload project.
Use a disconnection icon or relative objects when device communication fails	If selected, a disconnection icon is displayed on relevant objects when failing to communicate with the device. This icon will be shown in the lower right corner of the object. This icon will only show for disconnection after successful connection.
Enable watch dog (LB-9049)	Watchdog automatically reboots the system after the HMI stops functioning for a specified period of time.

4.5 Security

[Security] configure the user passwords and security classes. There are two authentication modes: General Mode and Enhanced Security Mode.

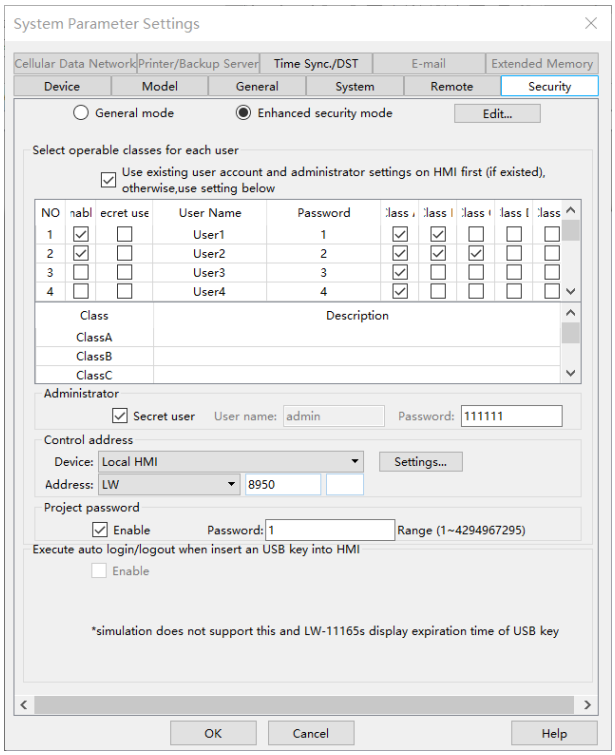
4.5.1 General Mode

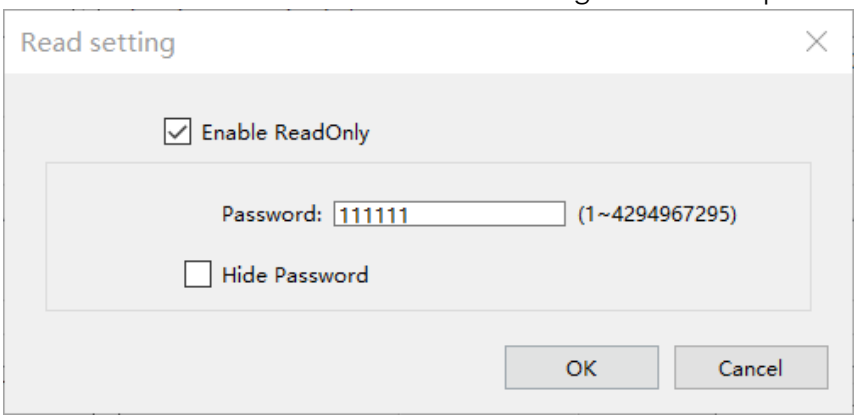


General Mode and Enhanced Security Mode.
Up to 12 sets of user and password are available. Password should be one non-negative integer.
Once the password is entered, the objects that the user can operate are classified.
There are six security classes available: A to F.
If **[None]** is selected for an object, every user can access this object.
For example, when the security class of User No. 1 is set as the preceding figure, User No. 1 could only access objects of classes A, B, C and “none”.

4.5.2 Enhanced Security Mode

At most 12 users can be set here. In addition, [Administrator] user is provided in this mode. An [Administrator] has all privileges and can operate all object classes. User passwords can be English letters or alphanumeric, and each user can have up to 12 classes: [Class A] to [Class L].

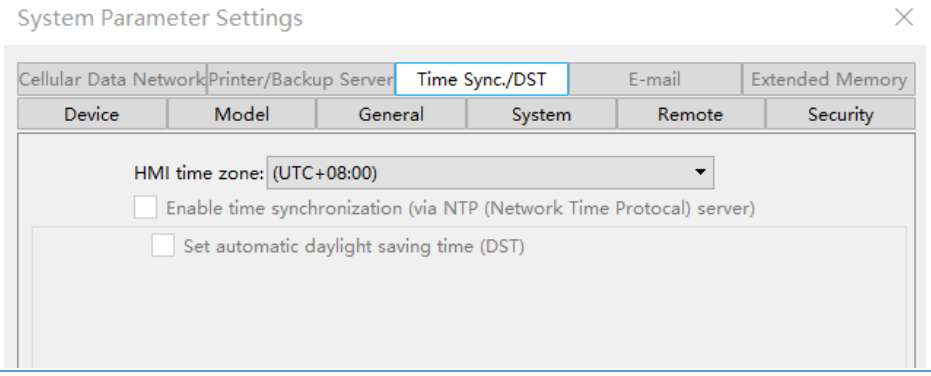


Setting	Description
Editable	Determines whether other users can change or see the password.  **Enable read-only** Under this mode, the settings can be viewed but not changed. **Mask password** Passwords are masked by asterisks (*)
Select operable classes for each user	When [Use existing user accounts on HMI] check box is selected, the user accounts existing on HMI will not be erased after downloading the project file to 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.

Project Password	When this password is enabled, it has to be entered before editing the project file. Select [Enable] and then click [Settings] to set the password.
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.

4.6 Time Sync./DST

Configure HMI time synchronization to NTP server.



Setting	Description
HMI time zone	Set HMI time zone.
Enable time synchronization with the external device when HMI starts	**Execute time synchronization when HMI starts** Automatically synchronizes HMI time with the designated NTP server when HMI starts. **Server response time** Select NTP server time zone. **Network time server** Provides four fields to fill in Network Time Servers for user's device. If the synchronization with server 1 fails, the system will try to synchronize with server 2, and so on. If HMI time cannot synchronize with any of the Network Time Servers, the system register LB-12055 will change to ON status. **Update interval** The frequency of synchronization, the range is from 10 to 86400 seconds.
Automatically set daylight saving Time (DST)	**Start / End** Set the start/end of Daylight Saving Time. The option [Last] may refer to the 5th or 6th week depending on the selected month.

	Daylight bias Set the time zone's offset during DST.

Note

- When DST ends, and the time falls back, the Trend Display object in Real-time mode will stop refreshing for the overlapping hour(s) when it just going back to standard time. However, historical data will not be affected.
- When DST ends, adjusting HMI time (manually or by network) back to the DST period will not be effective to the system. The system register LB-12355 stays OFF.
- When DST ends, and the time falls back, the system will not start DST for the overlapping hour(s). The system register LB-12355 stays OFF.
- Before DST starts, adjusting HMI time (manually or by network) into the DST period will start DST. The system register LB-12355 will be set ON. The HMI time will be the specified time without adding the offset.
- During DST, adjusting HMI time (manually or by network) to a time that is not within DST period will end DST. The system Register LB-12355 will be set OFF. The HMI time will be the specified time without subtracting the offset.
- This feature **does not yet support the DST in Southern Hemisphere.**

4.7 Remote

Parameters in this tab configure remote connections. Certain functions can be carried out using system registers.

Setting	Description
Prohibit remote HMI connecting to this machine	If checked, the remote HMI will be prohibited to connect with the local HMI. Also can use LB-9044 to control the remote connection.
Prohibit password remote-read operation (or set LB9053 ON)	Prohibit Remote HMI to read Local HMI's project password and user password.
Prohibit password remote-write operation (or set LB9054 ON)	Prohibit Remote HMI to write to Local HMI's project password and user password.
Password free	If the check box is selected, the client can connect with HMI via VNC without entering the password. System Register LB-12089 can also be used to control this function.
Password from project	If **[Password from project]** check box is selected, set the password for VNC login.
Monitor mode	If **[Monitor mode]** check box is selected, the HMI connected via VNC can only be monitored but not controlled. System Register LB-12088 can also be used to control this function.
Decompilation is prohibited	If this check box is selected when compiling, the setting in [exhmi password] will be ignored. Furthermore, the .exhmi file cannot be decompiled to .hmi file.
Disable HMI Upload Function	If this check box is selected, the .exhmi file obtained after uploading is 0 byte, and cannot be decompiled. Furthermore, need to reboot HMI for any changed setting to take effect. If set LB-9033 ON, the .exhmi file cannot be uploaded.

4.8 Cellular Data Network (In developing)

4.9 Printer / Backup Server (In developing)

4.10 E-mail (In developing)

4.11 Extended Memory (In developing)

Chapter 5 Windows Operation

5.1 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 this software can be built and edited.

5.2 Window Types

There are 4 types of windows, each with different functions and usages:

- Base Window
- Fast Selection Window
- Common Window
- System Message Window

5.2.1 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

5.2.2 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:

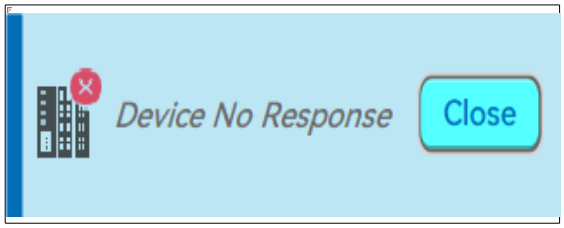
[LB-9013] hide (set ON)/show (set OFF) Fast Selection window
[LB-9014] hide (set ON)/show (set OFF) Fast Selection button
[LB-9015] hide (set ON)/show (set OFF) Fast Selection window/button

5.2.3 Common Window

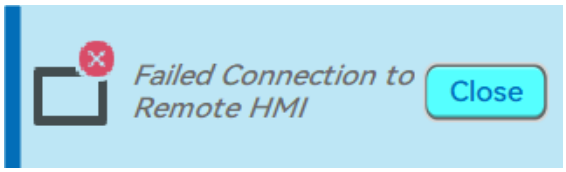
Window no. 4 is the default Common Window. Objects in this window will be displayed in other base windows, not including pop-up windows. Therefore, the common objects in different windows are often placed in common window.
When operating HMI, select [Function Key] » [Change common window] to change the source of common window.

5.2.4 System Message Window

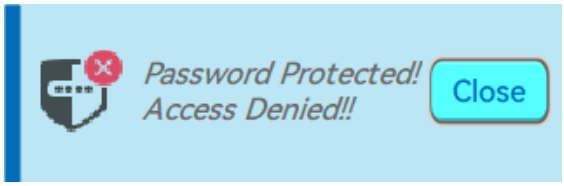
Windows No. 5, 6, 7, 8 are the default System Message Windows:



Window 5: Device Response
When the communication between device 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 6: HMI Connection
When failing to connect HMI with a remote HMI, this message window will pop up automatically.



Window 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 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.

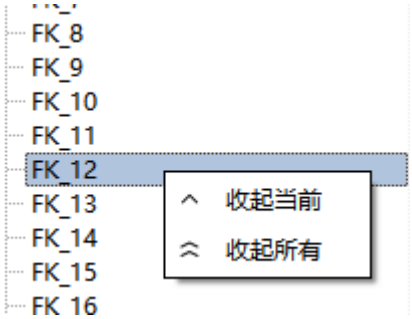
[LB-9035] HMI free space insufficiency alarm (when ON)
[LB-9036] SD card free space insufficiency alarm (when ON)
[LB-9037] USB disk free space insufficiency alarm (when ON)
The text shown in windows no. 5 ~ 8 can be edited for easier reference.

Note

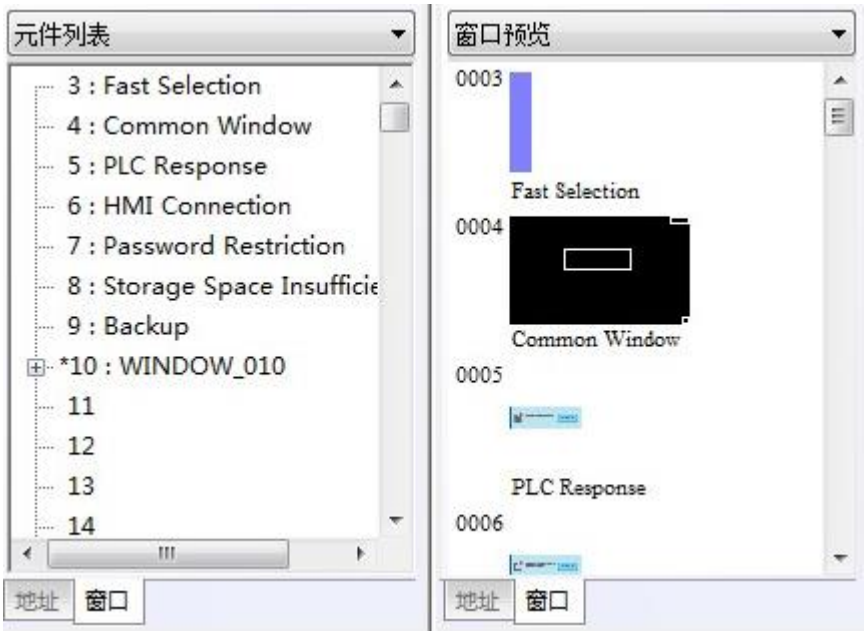
- A screen can display up to 16 pop-up windows simultaneous including System Message Window, Direct Window and Indirect Window.
- The system does not allow opening the same window with two Direct (or Indirect) Windows in one base window.
- Windows no. 3 to 9 are used by the system only, and windows no.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

5.3 Creat, Set and Delete a Window

Check the existing windows in [Windows Tree]
[Object List] displays window numbers and window names. Opened windows are marked With ☐ icon. Press the (+) sign to see the object ID, address and description of the objects used in this window.
Right click in the expanded Object List to call up the control key to control the collapse of the current or all expanded objects list.
,.

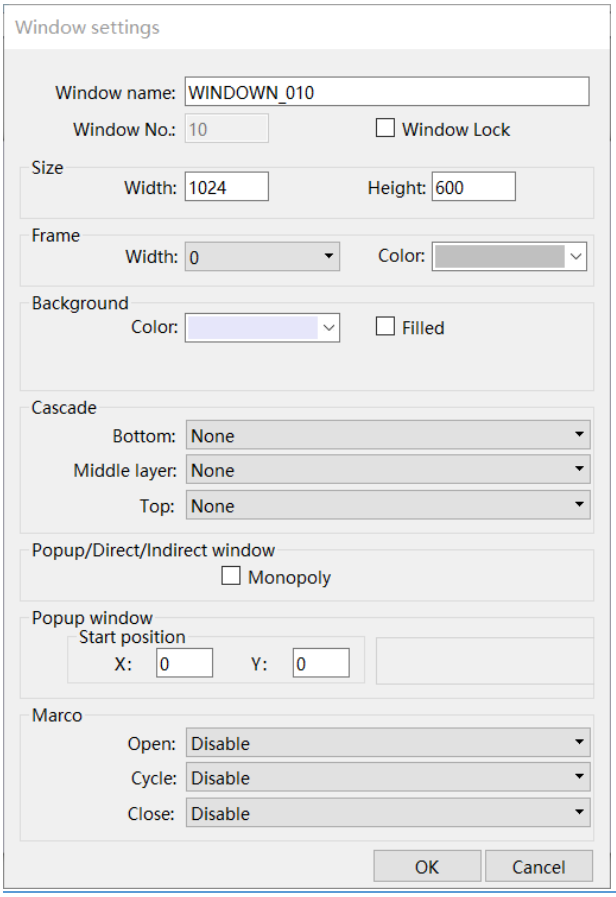


[Window preview] displays the thumbnails of windows.



5.3.1 Creating and Setting a Window

In windows tree right click on a window number then select [New]



Setting	Description
Name	The input name appears on the title bar and also in windows 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.
Macro	Select the macro to be executed when the window opens or closes, or the macro to be executed periodically (each 500 ms). Please build the macro in advance.

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 [View] » [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 windows tree and select [Settings].
- In [View] » [Open Window] select the window then click [Settings].
- In the window, right click when no object is selected, and select [Attribute].

5.3.2 Open, Close and Delete a Window

The ways to open an existing window:

- Double click on the window number in the windows tree.
- In the windows tree, select the window number, right click, and then select [Open].
- In [View] » [Open Window] select the window number and then click [Open].

The ways to close or delete an existing window:

- In the windows tree, select the window number; right click, and then select [Close] or [Delete].
- In [View] » [Open Window] select the window number and then click [Delete].
- To delete a window, you must close it before deleting it.

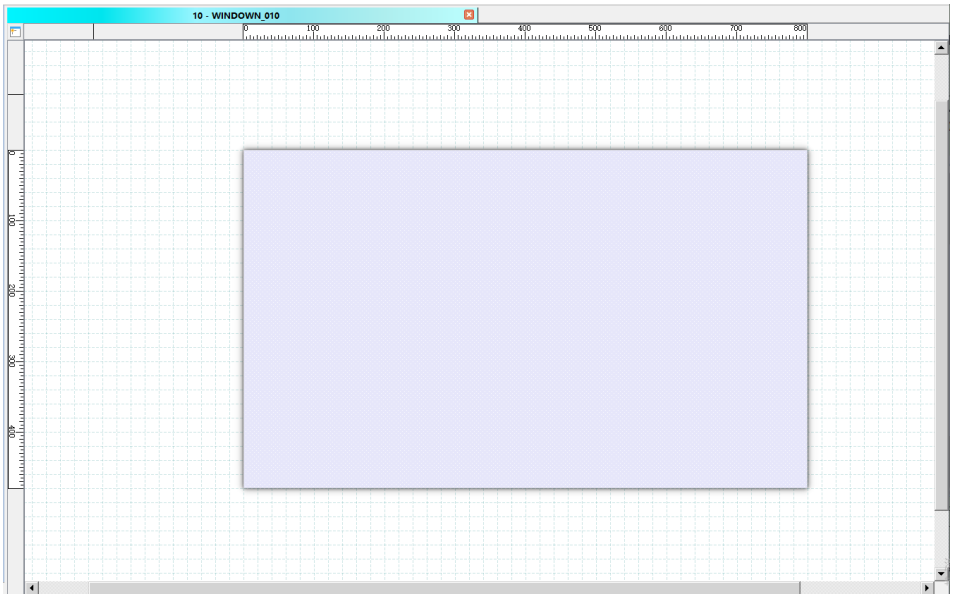
5.4 Editing Tips

This software provides a user-friendly interface with which users can easily edit projects. The following are some editing tips.

,.

5.4.1 Non-display Area, Select Tool, and Hand Tool

As shown in the following screenshot, two areas can be found in the editing window: editing area and non-display area. The non-display area refers to the area outside the editing area (the one in the middle). Objects placed in the non-display area will not be visible on HMI, and communication with devices continues. The invisible objects such as Meter, Time-based Data Transfer, or Text objects used for comments can be placed in this area to avoid overlapping with other objects in the editing area, which adds difficulty to project editing. The objects placed in the non-display area remain effective.



5.4.2 Ruler（In developing）

5.4.3 Quick Copy（In developing）

Chapter 6 Event Log

6.1 Overview

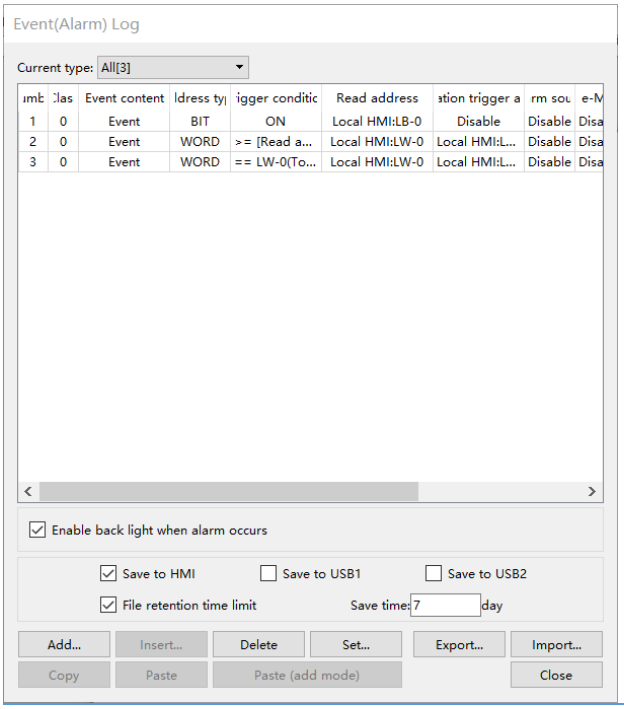
The following are the basic steps to use Event Log:

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

6.2 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. The upper limit for the number of event logs is 10000.



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

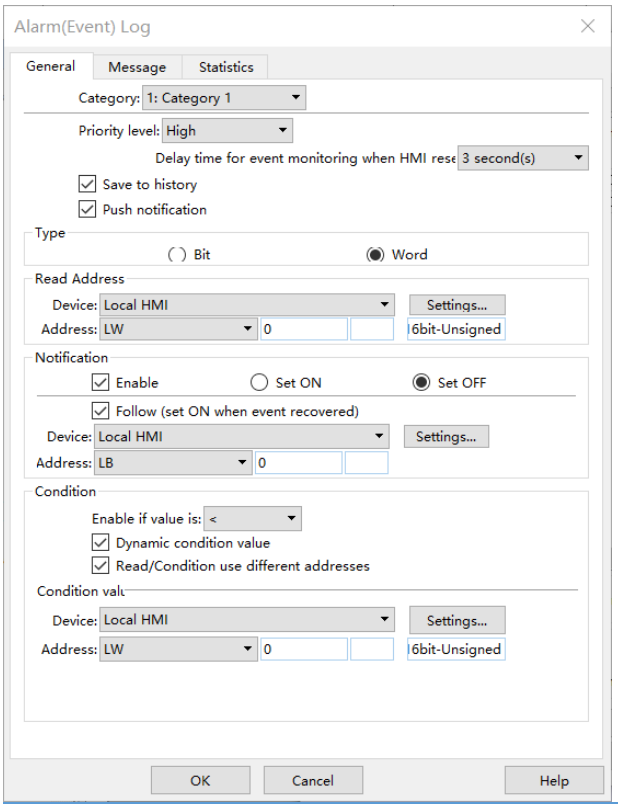
	number of events are in this category.
History files	Saves event log files to the specified location. Once an event occurs, the HMI immediately saves the history file. When executing On-line or Off-line Simulation on PC, the files will be saved in the ParsePro / Project Name / ftp folder under the installation directory. **Preservation limit** This setting determines the maximum number of Event Log files to be preserved in HMI memory. This does not include the file generated today. That is, if [Days of preservation] is set to 2; the two latest files excluding the file generated today will be kept. The files that are not within the range will be deleted automatically for saving the storage space.
Copy	Copy the selected item.
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 when paste.

6.2.1 Quick View of Errors (In developing)

6.3 Creating a New Event Log

6.3.1 General Tab

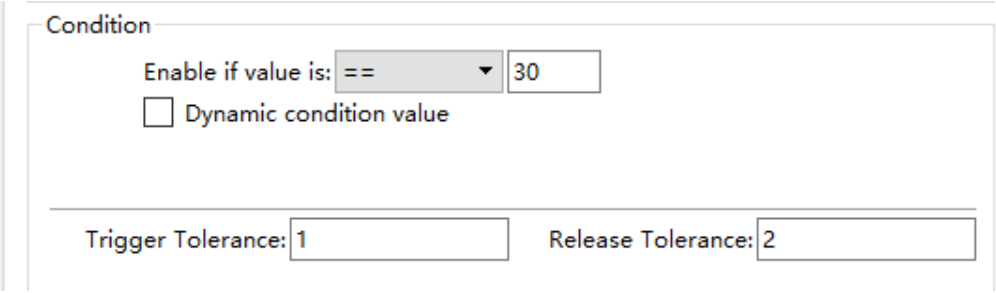
Click [New] in the [Event (Alarm) Log] dialog box, then enter [General] page.



Setting	Description
Category	Select event category, the range is from 0 to 255.
Priority level	Events in Alarm Bar / Alarm Display are ordered by priority level first and then by time. When the number of events is equal to the maximum number (the default value is 1000), the events with low priority level will be deleted from the Event Record and new events will be added.
Save to history	In Event Log main settings, if [Save to HMI memory] check box under [History files] group box is selected, selecting [Save to history] here determines whether each separate event should be saved as historical file.
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. **Auto Reset** The notification bit will reset to its original state once the alarm condition returns to normal. For example, when the alarm is triggered, the state of the notification bit turns ON. When the alarm condition returns to normal, and [Auto Reset] check box is selected, the state of

	the notification bit turns OFF.
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:

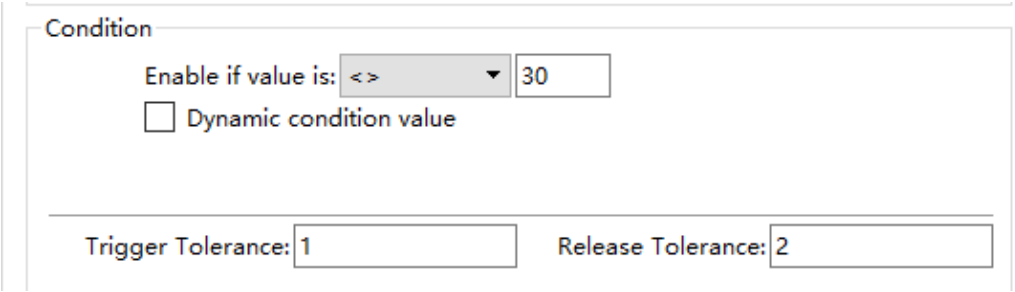
When [Read address] value is greater than or equals to 29 (= 30 - 1) Or less than or equals to 31 (= 30 + 1), the event will be triggered. The trigger condition:

29 ≤ [Read address] value ≤ 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:

[Read address] value < 28 or [Read address] value > 32

Example 2



The setting above indicates:

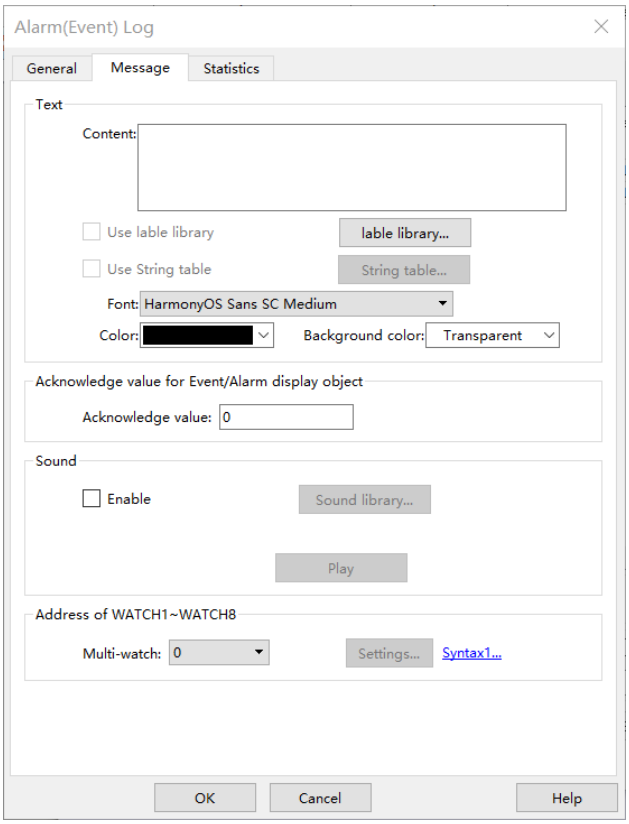
When [Read address] value is less than 29 (= 30 - 1) or greater than 31 (= 30 + 1), the event will be triggered. The trigger condition:

[Read address] value < 29 or [Read address] value > 31

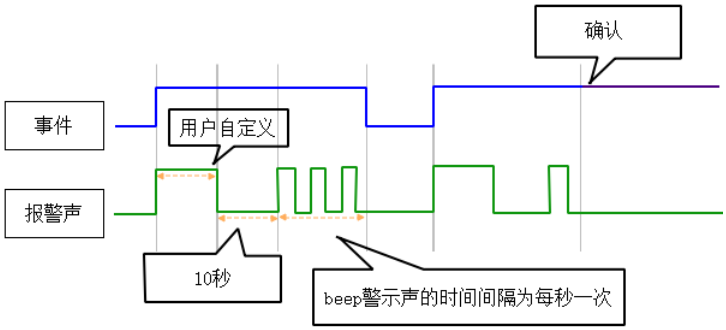
After the event is triggered, when [Read address] value is greater than or equals to 28 (= 30 - 2) or less than or equals to 32 (= 30 + 2) the system will return to normal condition:

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

6.3.2 Message Tab



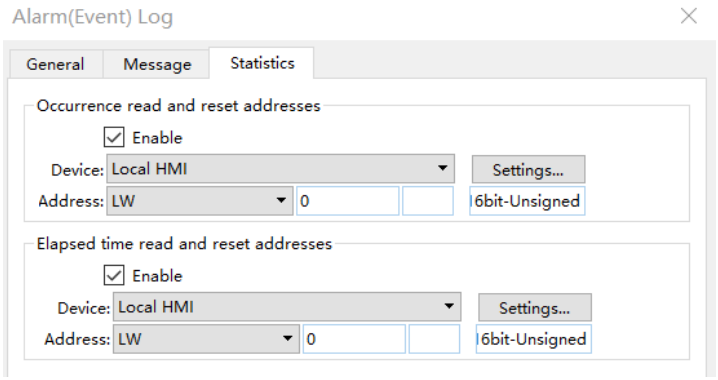
Setting	Description
Content	The text content displayed in [Alarm Bar], [Alarm Display], and [Event Display] objects. Use the formats of [Watch Address] WATCH1 to WATCH 8 to use register data in content or see Example 3 below. The content in Label Tag Library and String Table can be used in the Event Log message.
Font/Color/Background Color	The font / color / background color can be set respectively for each event. The font and color settings determine how [Alarm Bar] shows the text, while The font, color, and background color settings determine how [Alarm Display] and [Event Display] show the text. These settings are not available in the [Event Display] under History mode.
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. 
Watch Address	Users can set how the value is displayed in the designated watch address when an event occurs. Up to 8 watch addresses can be used simultaneously. Click [Syntax] to see how to use the syntax to embed device data in the content of an event log displayed in the watch address.

● Example 3

The display content can contain the data in the LW address when the event is triggered.
The format is: % (WATCH #) d (% ->start, # ->Monitor No. 1~8, d ->Format Type)
Suppose the value in WATCH1: LW-20 is 13 when triggered:
If it is set to "High Temperature=% (WATCH1) d", it will be displayed as "High Temperature=13".

6.3.3 Statistics Tab



Setting	Description
Occurrence read and reset address	If enabled, the number of occurrence of the event after HMI startup will be written to the designated word address. The word address can be read / written.
Elapsed time read and reset address	If enabled, from an event occurs to its recovery, the elapsed time (in seconds) will be written to the designated word address. The word address can be read / written.

Chapter 7 Data Sampling

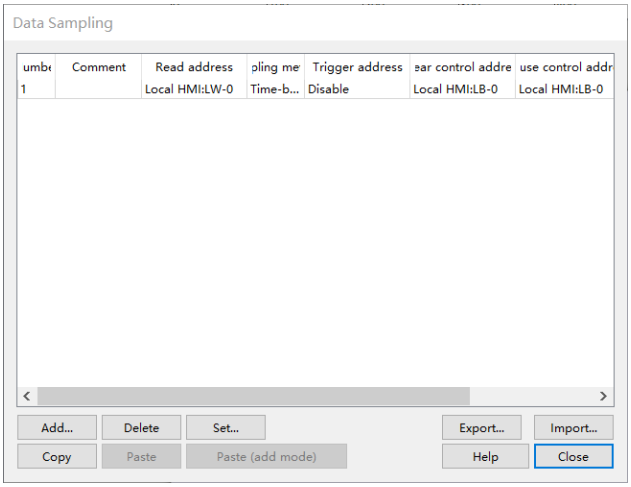
7.1 Overview

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.

7.2 Data Sampling Management

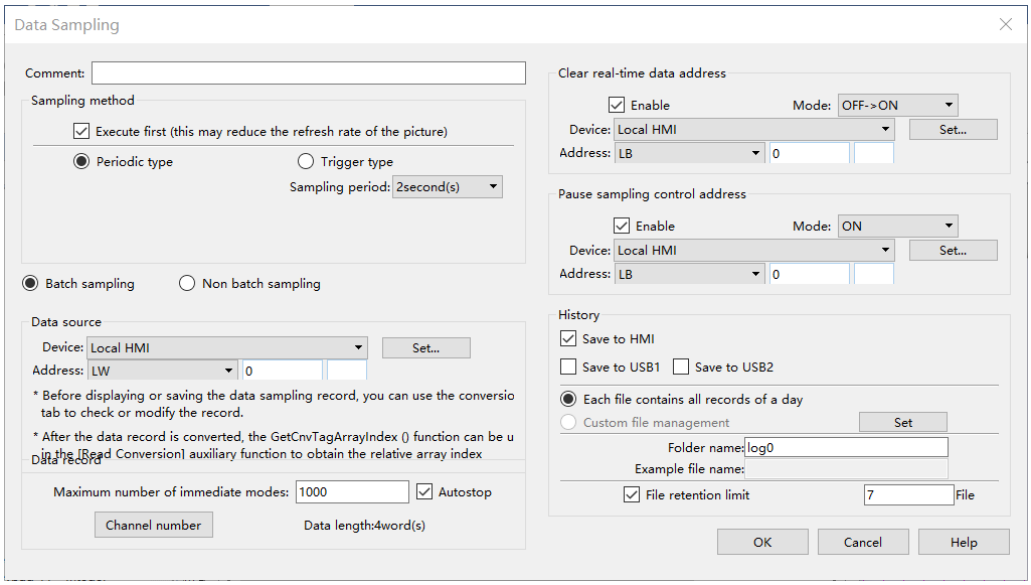
Create a new Data Sampling object first by the following steps:

1. From the menu select [Data/History] and click [Data Sampling].
2. Click [Add] to finish relevant settings

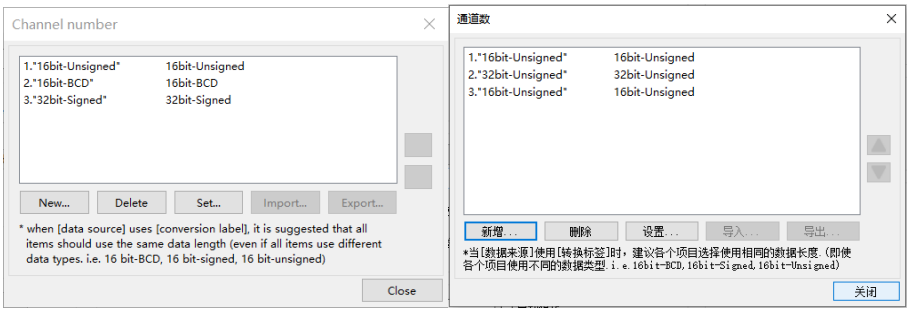


7.3 Creating a new Data Sampling

The following introduces how to set a new Data Sampling.



Setting	Description
Sampling Mode	**High Priority** Data sampling processes with this feature enabled will be prioritized. Please note that too many priorities can slow down update rate of other objects. **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.
Data Source	Specify an address to be the source of Data Sampling. To convert a data record, please use a user-defined conversion tag that allows [Read conversion]. For array addresses, the GetCnvTagArrayIndex function can get the relative array index and then calculate.
Data Record (Real Time)	In Real-time Mode, when [Auto. stop] is not selected, the max. number of data records can be saved on HMI is 86400. When the number of records exceeds 86400, the earliest data will be deleted. **Data Format**

	Data of different formats in consecutive registers can be sampled. For example, setting: LW-0 (16-bit Unsigned), LW-1 (32-bit Unsigned), and LW-3 (16-bit Unsigned). The upper limit for the number of records is 1000.  **Auto. stop** Data sampling will automatically stop when the number of records reaches [Max. data records].
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.
Pause sampling control 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.
History	**Save to HMI** Saves Data Sampling to HMI every 10 seconds. Or, use system register [LB-9034] to force storing data. See details on the restrictions of using LB-9034 in the Note 6 below. **Save to SD card / USB1 / USB2** 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.db,* indicating the date of the file. **Folder name** Specify Data Sampling file name which must be all in ASCII characters. The folder name will be written as: *[Storage Location]*\[ftp] \[datalog] \ *[Folder Name]* \ *[File Name]* **File retention limit** Determines the number of data sampling files to be preserved. Please note that the current .db file is not included in this limit. That is, if this limit is set to 2, apart from the current file, two latest files will be preserved.

Note

1. 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.

2. 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.
3. When using [Customized file handling] and set [File retention 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.
4. When running simulation on PC, all data sampling will be saved to the datalog folder in: this software\ParsePro\[Project Name]\ftp\datalog\[Folder name]. 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.
5. When saving files to USB disk or SD card, the capacity of a FAT32 folder depends on the length of the file names. Fewer files can be saved when the file names are longer.
6. When using LB-9034, the shortest interval between two successful executions is 2 seconds. That is, after LB-9034 is triggered, it can be effective again 2 seconds later.

Chapter 8 Object General Properties

8.1 Overview

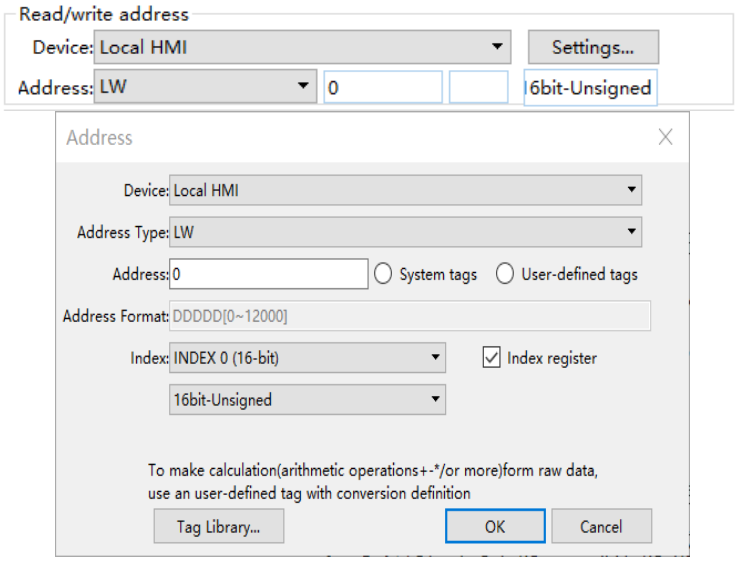
The basic steps to create an object:

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.

This chapter explains the basic settings of an object.

8.2 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 [Device] which comes from [System Parameters Settings] » [Device List].

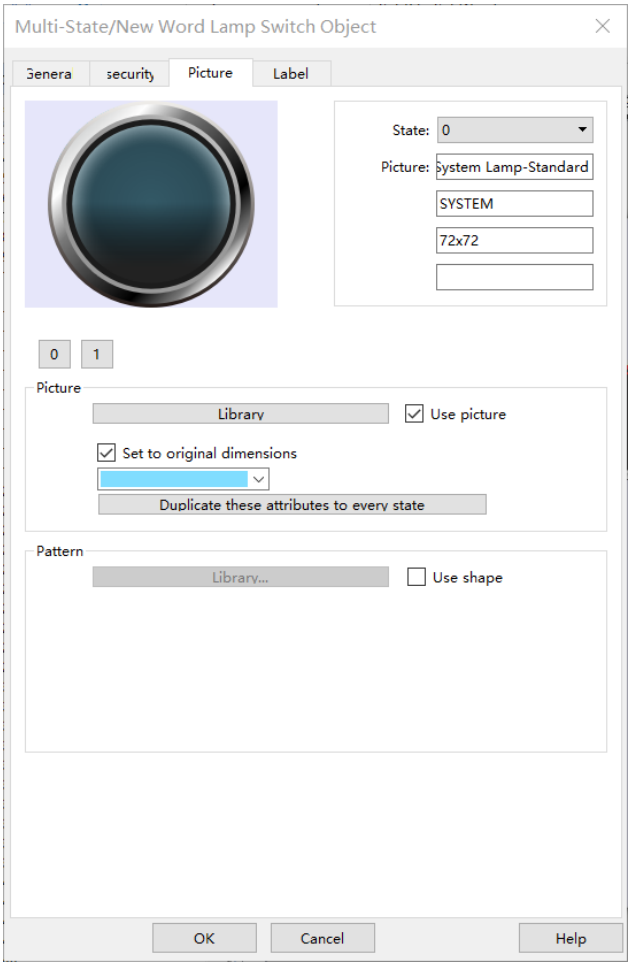


Setting	Description
Device	Select the PLC type.
Address type	Different PLCs have different address types.
Address	Set the read/write address.
System Tag	Address tags include [System Tag] and [User-defined Tag]. This option allows users to use [System Tag]. [System Tag] consists of the preserved addresses 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].

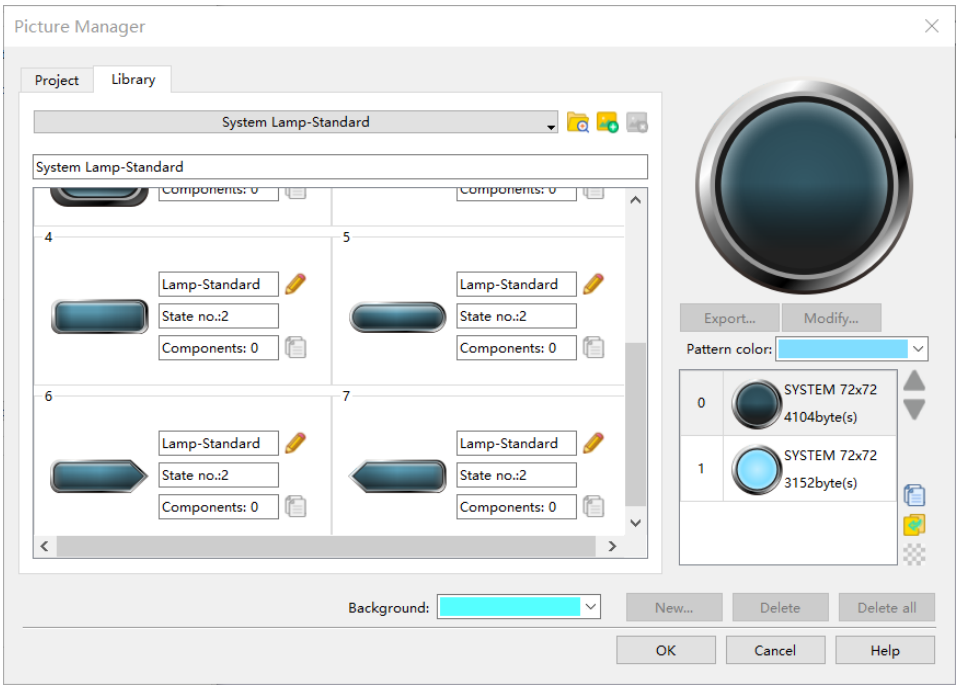
8.3 Using Shape Library & Picture Library

In the settings dialog box of certain objects, “Shape” tab can be found. In this tab, Shape Library and Picture Library can be used to add visual effects on objects. Select Shape tab to use the libraries when creating an object.



8.3.1 Picture

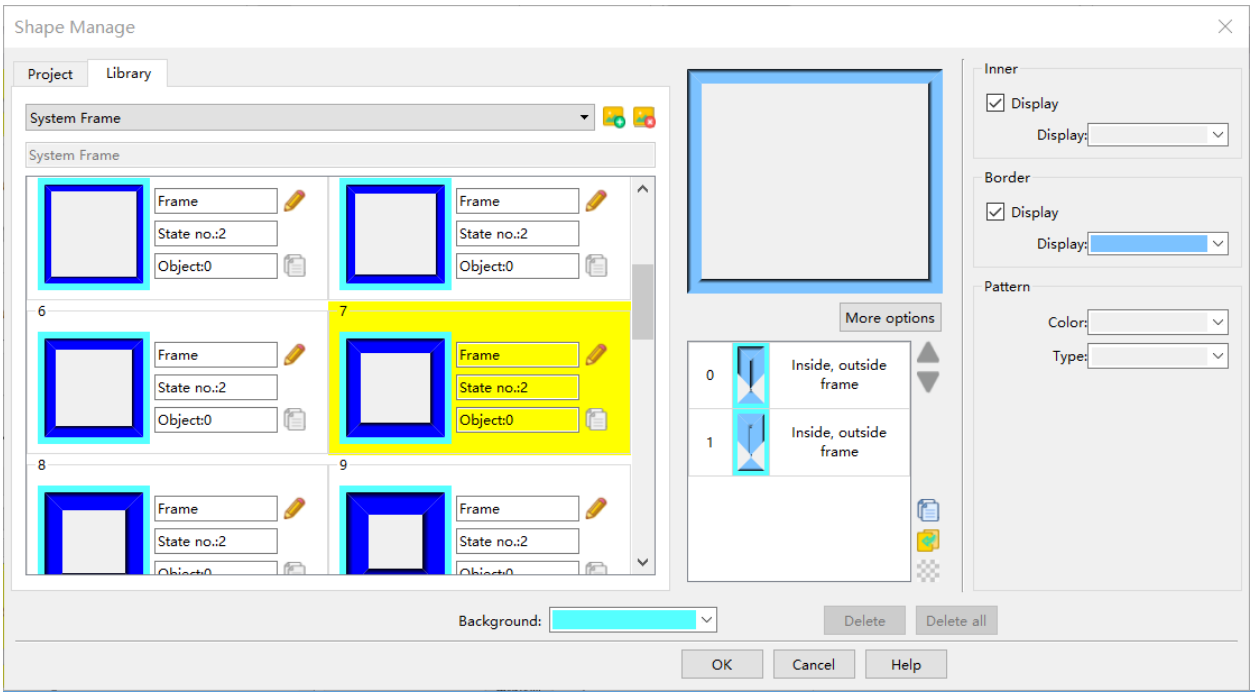
Select [Use picture] and then click [Picture Library] button to open Picture Manager window and select a picture from the window.

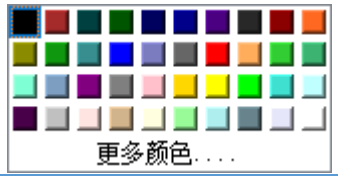
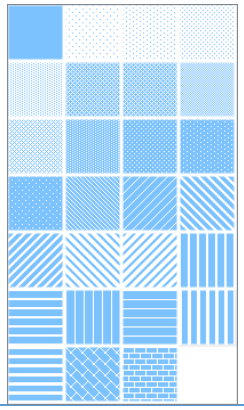


Setting	Description
Picture Library	Click [Picture Library] button in Picture group box to open Picture Manager window.
Set to Original Dimensions	When this checkbox is selected, this software will return the picture to its original size to eliminate the redrawing step.
Duplicate these attributes to every state	This button is available only when a system picture is used. Clicking this button will duplicate the attributes of the current picture to other states.

8.3.2 Shape

Select [Use shape] and then click [Shape Library] button to open Shape Manager window and select a shape from the window.

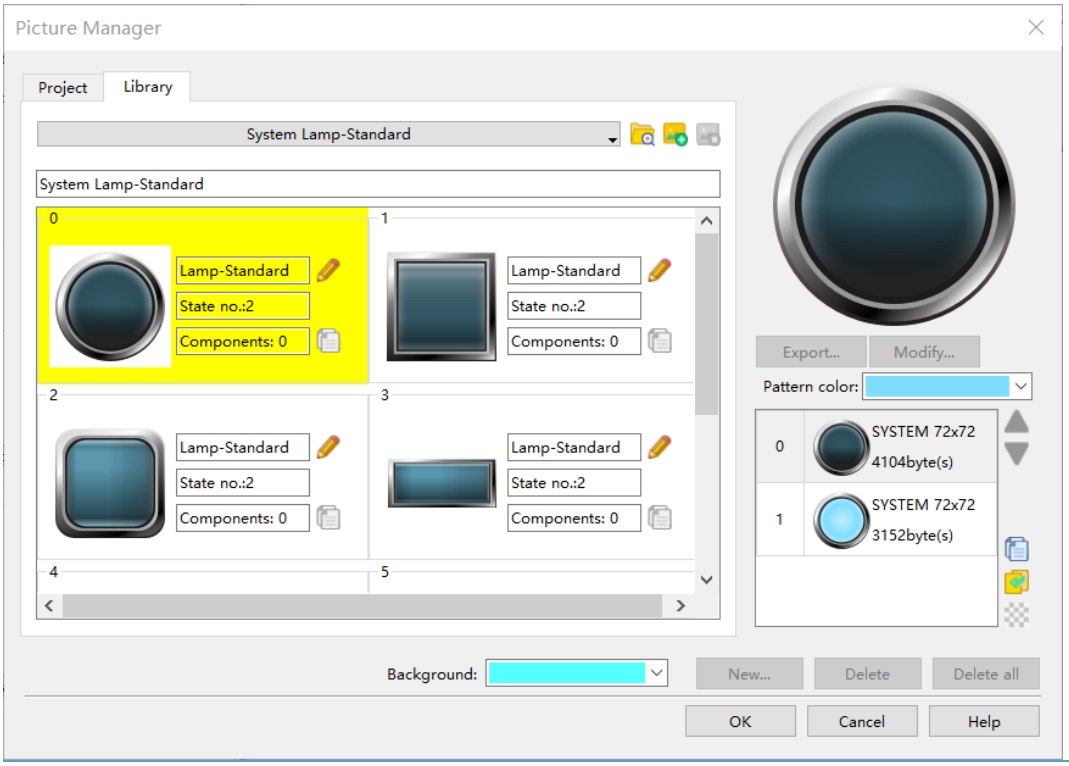


Setting	Description
Inner	Select [Display] to show the inner color of the shape and select a color as the inner color by clicking the drop-down button. 
Border	Select [Display] to show the frame color of the shape and select a color as the frame color by clicking the drop-down button.
Pattern Color	Click the drop-down button to select a color for the interior pattern of the shape.
Pattern Type	Click the drop-down button to select a pattern. 
Set to all states	Duplicate the attributes of the current state to other states.

8.3.3 Shape Manager

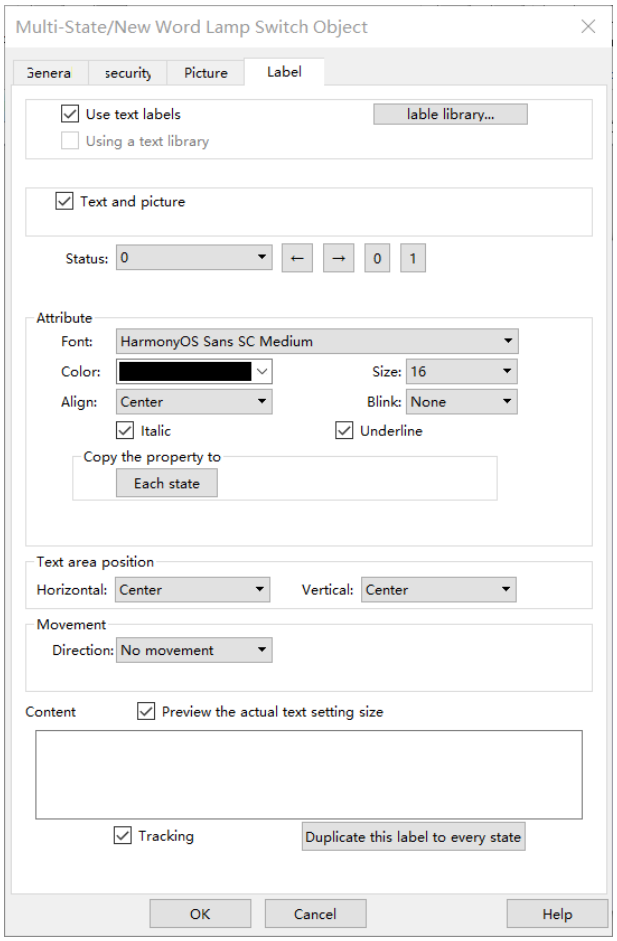
In Picture Manager and Shape Manager windows, the currently selected picture / shape is

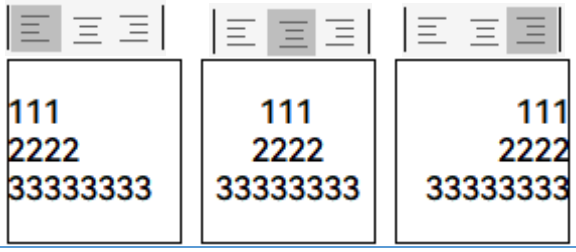
highlighted yellow as shown below. Also include the below messages,

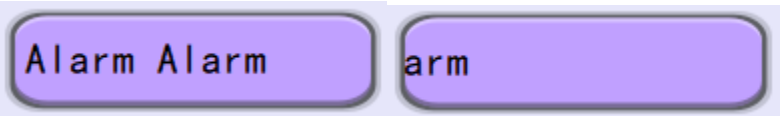


Setting	Description
Lamp-Standard	Name of the shape.
States：2	Number of states of the shape.
Objects：0	This shape is used by 0 object in the project.
0、1	The state 0 and state 1 of this shape.

8.4 Setting Lab Text

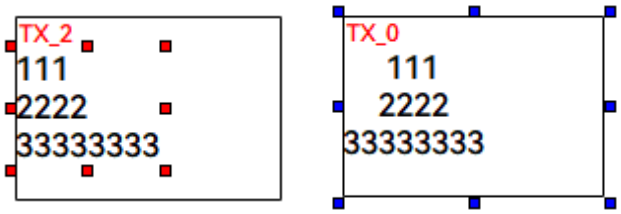


Setting	Description
Use Text Label	Use a label for the object.
Use Label Library	Use a label from the Label Library.
Text and Picture	If this option is checked, the text will be shown in png Format.
Label Library	View all the labels in the Label Library.
Font	Select a font for text from the list.
Color	Select a font color.
Size	Select a font size.
Alignment	Multilines text is aligned within the label outline. Below are the corresponding shortcut keys. 
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 this option is not selected, the next text appears only when the previous text disappears completely.  If this option is selected, the text will be displayed continuously.  **Speed** Adjust the speed of the text movement.
Content	Set the content of the label. If [Use label library] is selected, the content in the Label Library can be used.
Tracking	With this option selected, changing the position of the text in one state will also change the text position in other states.
Duplicate these attributes to every state	Copy the content of the current state to all states.

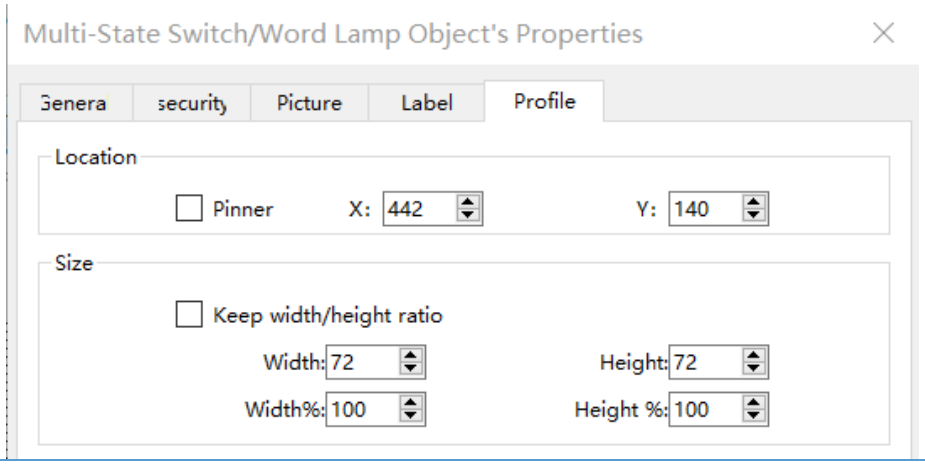
Note

- When Text/Comment object is used with Comment enabled, the text used as comment will not appear on HMI.
- Use shortcut keys  to adjust alignment attributes, such as text labels (red boxes) clicked into objects. When only the object (blue box) is clicked, using the shortcut key will adjust the position of the text in the object, and the alignment attribute will not change.



8.5 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
Location	**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.

Chapter 9 User Password & Object Security

9.1 Overview

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

General Mode

Enhanced Security Mode

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.

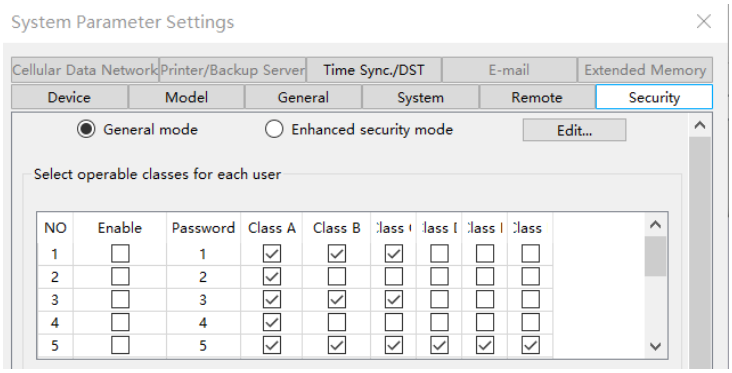
9.2 User Password and Operable Object Classes

The security parameters can be found in [System Parameter Settings] » [Security].

9.2.1 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.

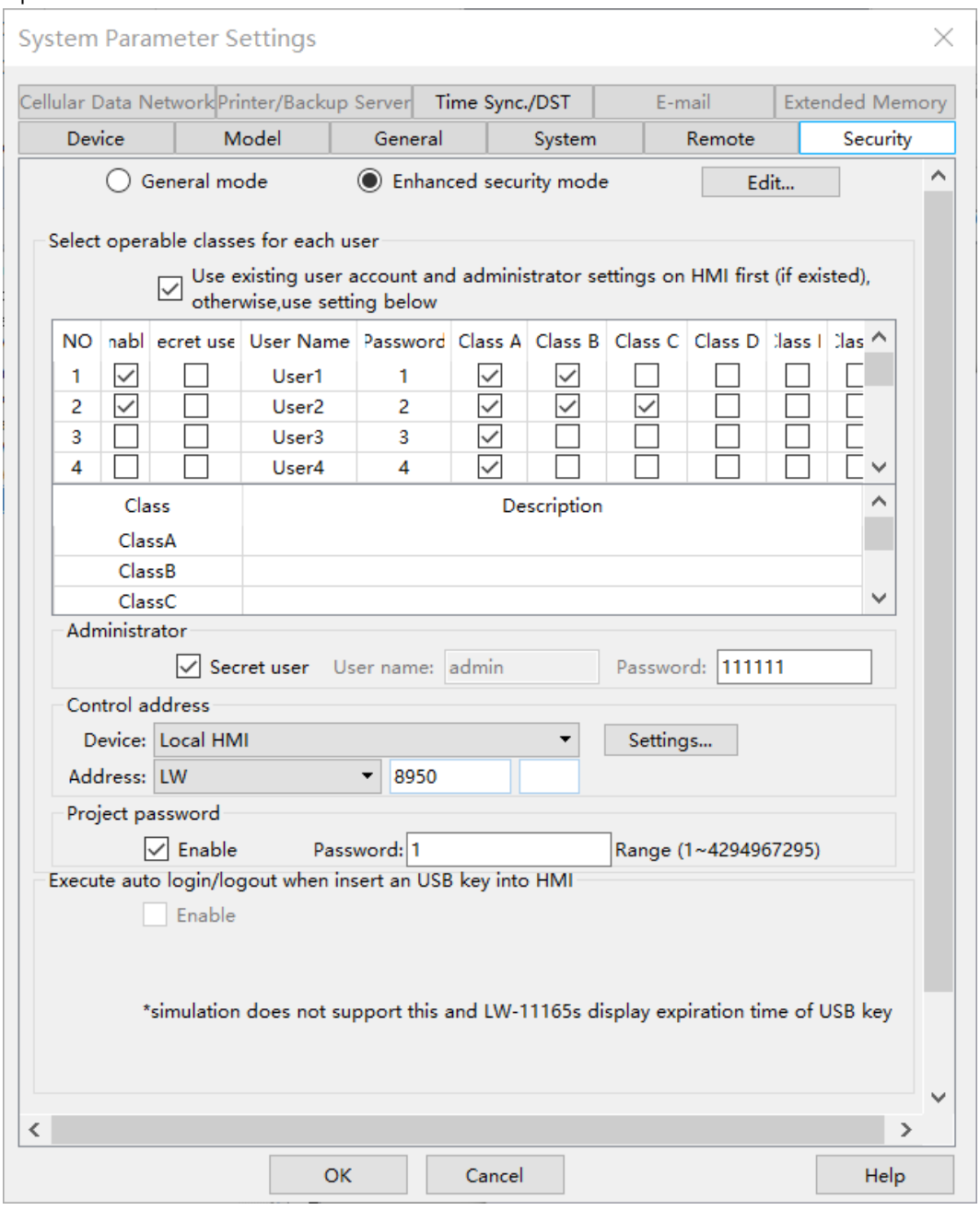
Once the password is entered, the objects that the user can operate are classified. As shown below, “User 2” can only operate objects with class A.



9.2.2 Enhanced Security Mode

Up to 12 users can be set here. In addition, [Administrator] setting is provided. Administrator has

all privileges and can operate all object classes. A username can contain Chinese characters, letters, and numbers, and a password can only contain letters and numbers. Each user can have up to 12 operable classes: A to L.



9.3 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.

9.3.1 Control Address Settings

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

Control 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 (Case-sensitive and only allows Chinese characters, letters and numbers). (,).
LW-n + 12 (8 words)	Password	Account password (Case-sensitive and only allows letters, numbers, or special characters) (,).

After setting the [Control address], the relevant addresses can be found in [Address Tag Library] » [User-defined tags].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]

9.3.2

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] firstly. 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 (minutes)	Set [user name], [password], [user privilege], and [user index] first. [user index] is for specifying a time period (in minutes), within this period the account is valid. If 0 is specified, this account stays valid until the HMI is powered 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.
15	Add a temporary account (days)	Set [user name], [password], [user privilege], and [user index] first. [user index] is for specifying a time period (number of days), within this period the account is valid. If 0 is specified, this account stays valid until the HMI is powered off.
16	Add an expiring account (minutes)	Set [user name], [password], [user privilege], and [user index] first. [user index] is for specifying a time period (in minutes), within this

		period the account is valid. 0 is an invalid value for this setting.
17	Add an expiring account (days)	Set [user name], [password], [user privilege], and [user index] first. [user index] is for specifying a time period (number of days), within this period the account is valid. 0 is an invalid value for this setting.
18	Remaining minutes for user name	Set [user name] first. If succeeded, the remaining time (in minutes) will be displayed in [user index].
19	Remaining minutes for user index	Set [user index] first. If succeeded, the remaining time (in minutes) will be displayed in [user index].
20	Remaining days for user name	Set [user name] first. If succeeded, the remaining time (number of days) will be displayed in [user index].
21	Remaining days for user index	Set [user index] first. If succeeded, the remaining time (number of days) will be displayed in [user index].

Note

- Add a temporary account / expiring account: The difference between temporary accounts and expiring accounts is that temporary accounts are not stored in the system and will be invalid after HMI is turned off. Both temporary accounts and expiring 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.
- The [user privilege] address does not display the privileges assigned to current user account, please use system register LW-9222 to display the privileges.

9.3.3 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 Code	Command execution result
(0x001)	Succeeds
(0x002)	Invalid command
(0x004)	Account exists (when adding a new account)
(0x008)	Account not exists

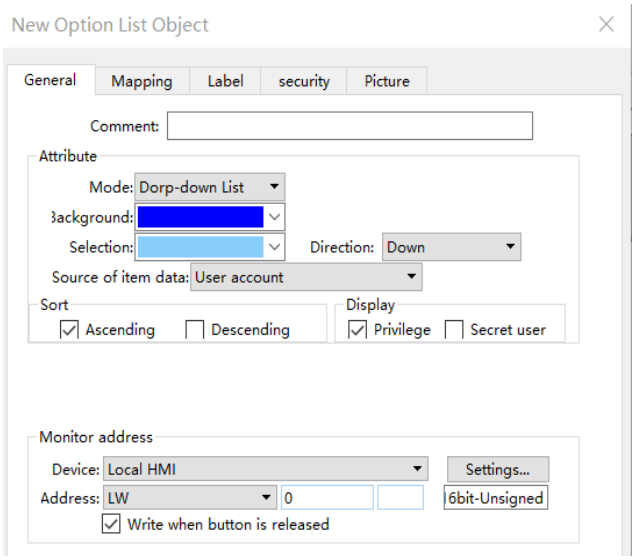
(0x010)	Password error
(0x020)	Deny command
(0x040)	Invalid name
(0x080)	Invalid password character exists
(0x100)	Invalid import data
(0x200)	Out of validity range (when log in by USB Security Key). The [Effective Time] can be set in Administrator Tools.

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.

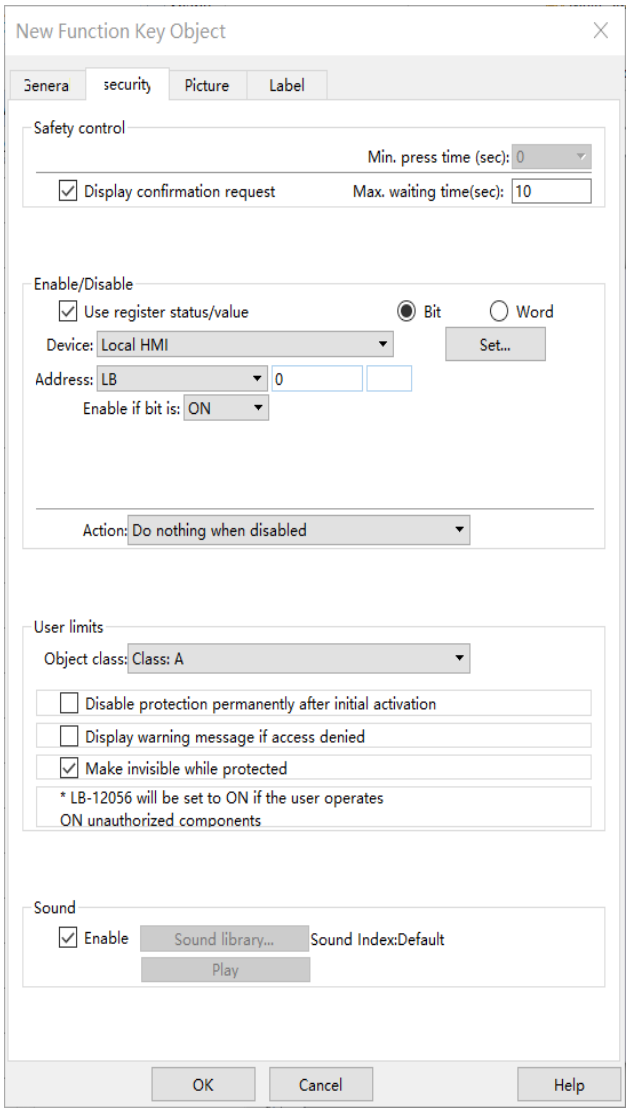
9.4 Enhanced Security Mode Usage

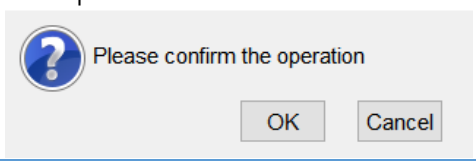
9.4.1 Enhanced Security Mode and Option List Object

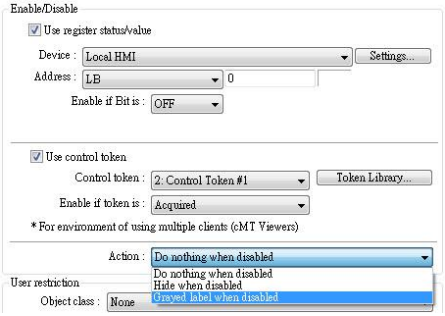
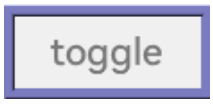
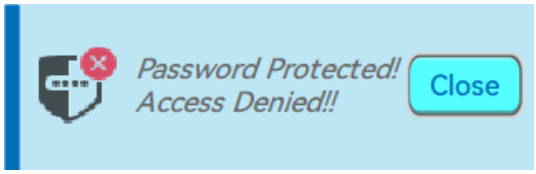
Enhanced Security Mode uses Control Address LW-n + 2 as account index. With Option List Object, account names and privileges can be displayed. Users can select whether or not to display the account privileges and secret users in Option List. Secret users are set to be hidden in [System Parameter Settings] » [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.



9.5 Object Security Settings



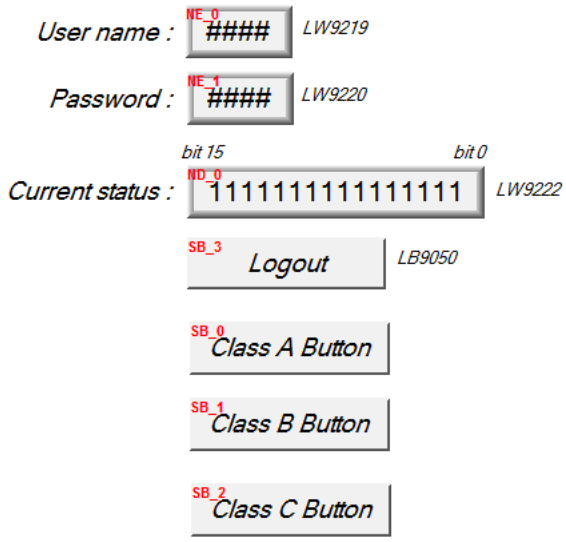
Setting	Description
Security Tab	Min. press time (sec) Press and hold the object for longer than the [Min. press time] set here to activate the object. Confirm before operation After pressing the object, a dialog appears for operation confirmation. If the response to this dialog comes later than the set [Max. waiting time (sec)], this dialog disappears automatically and the operation will be canceled. 

Enable / Disable /	When [Use register status/value] or [Use control token] is selected, whether the object is operable is determined by the status of the designated address. As shown in the following figure, only when LB-0 is in ON state will this object be operable  Hidden when disable When the designated address is disable, the object is hidden. Grayed label when disable When the designated address is disable, the label of the object turns grey.  For the [Enable/Disable] designated character address of [Set Word] and [Numeric] Objects, please refer to the Security Settings in 12.4 and 12.11 of this manual respectively.
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, the object will be hidden.

9.6 Example of Object Security Settings

The following shows an example of setting object security class:

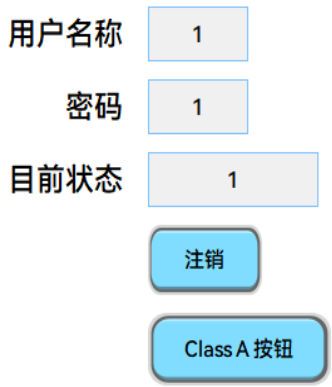
1. Create a project, go to [System Parameter Settings] » [Security] » [General] to enable 3 users:
   User 1 = Operable class: A
   User 2 = Operable class: A, B
   User 3 = Operable class: A, B, C
2. Design Window no. 10 as shown:



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 operable object class 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 “0”, 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.

“ 3” (333), A, B, C ,[LW-9222] bit 0 ~ bit 2 “1”, “A ~ C” .

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

,,“”

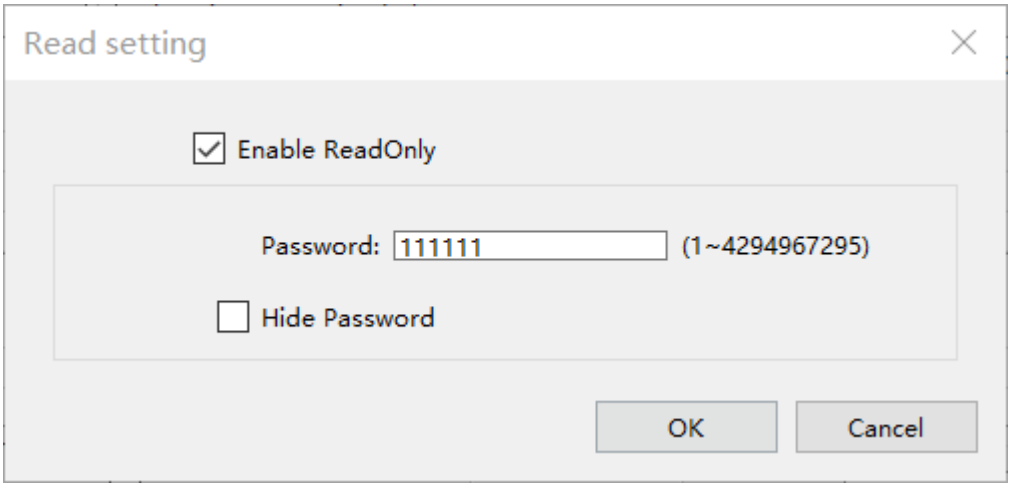
Note

- 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.

9.7 Protecting Password Settings from Unauthorized Editing

Before sending the project to others who may edit the project afterwards, it is recommended to click [Editable] button in Security settings tab to open read-only mode. This mode can protect password settings from unauthorized editing.,, [] .



When [Enable read-only] is selected, a password will be required for changing security settings in the project. When [Mask password] is selected, passwords will be masked by asterisks (*).

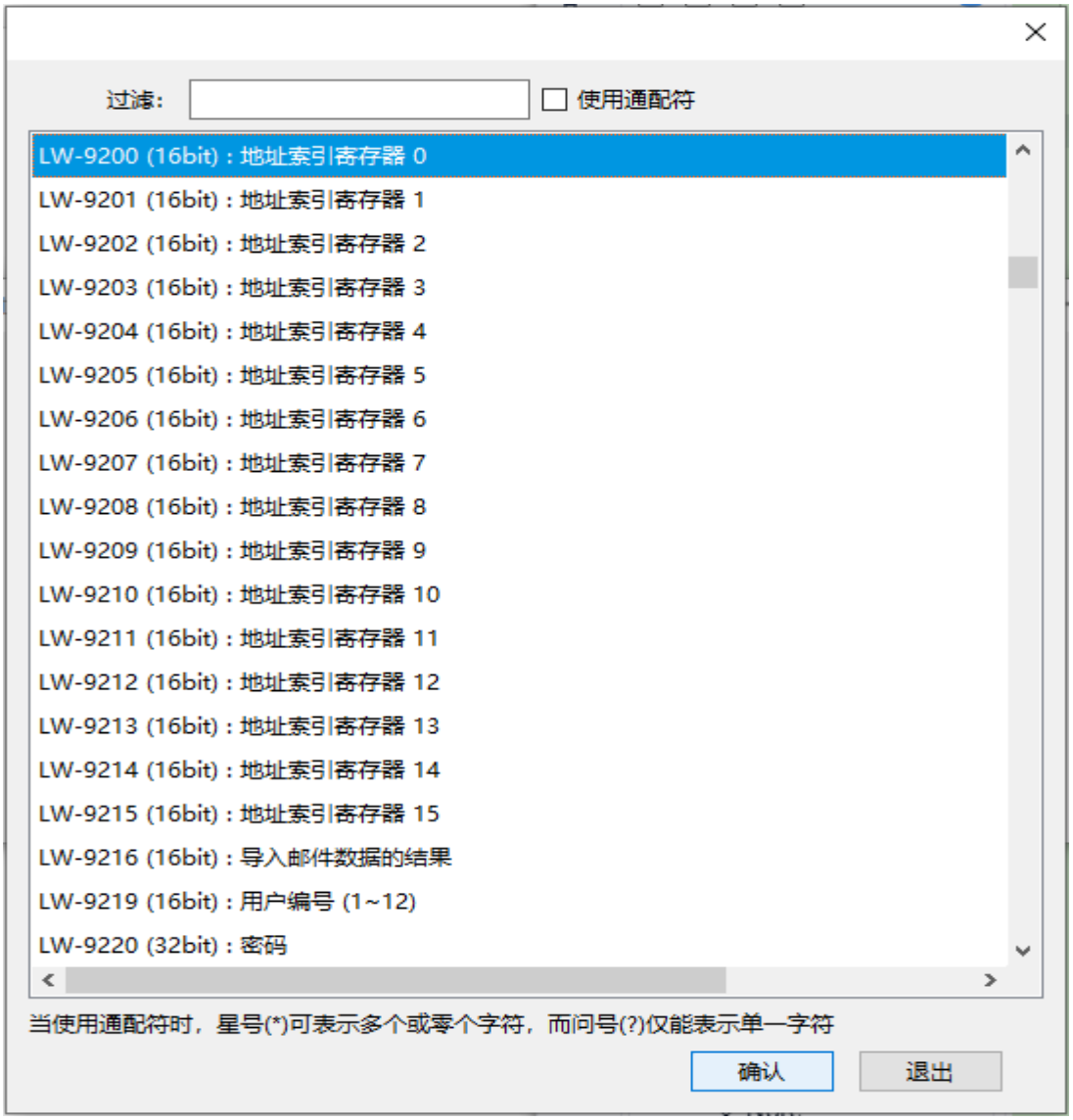
Note

- The protected projects cannot be decrypted since they are encrypted by users, therefore, please remember your password.

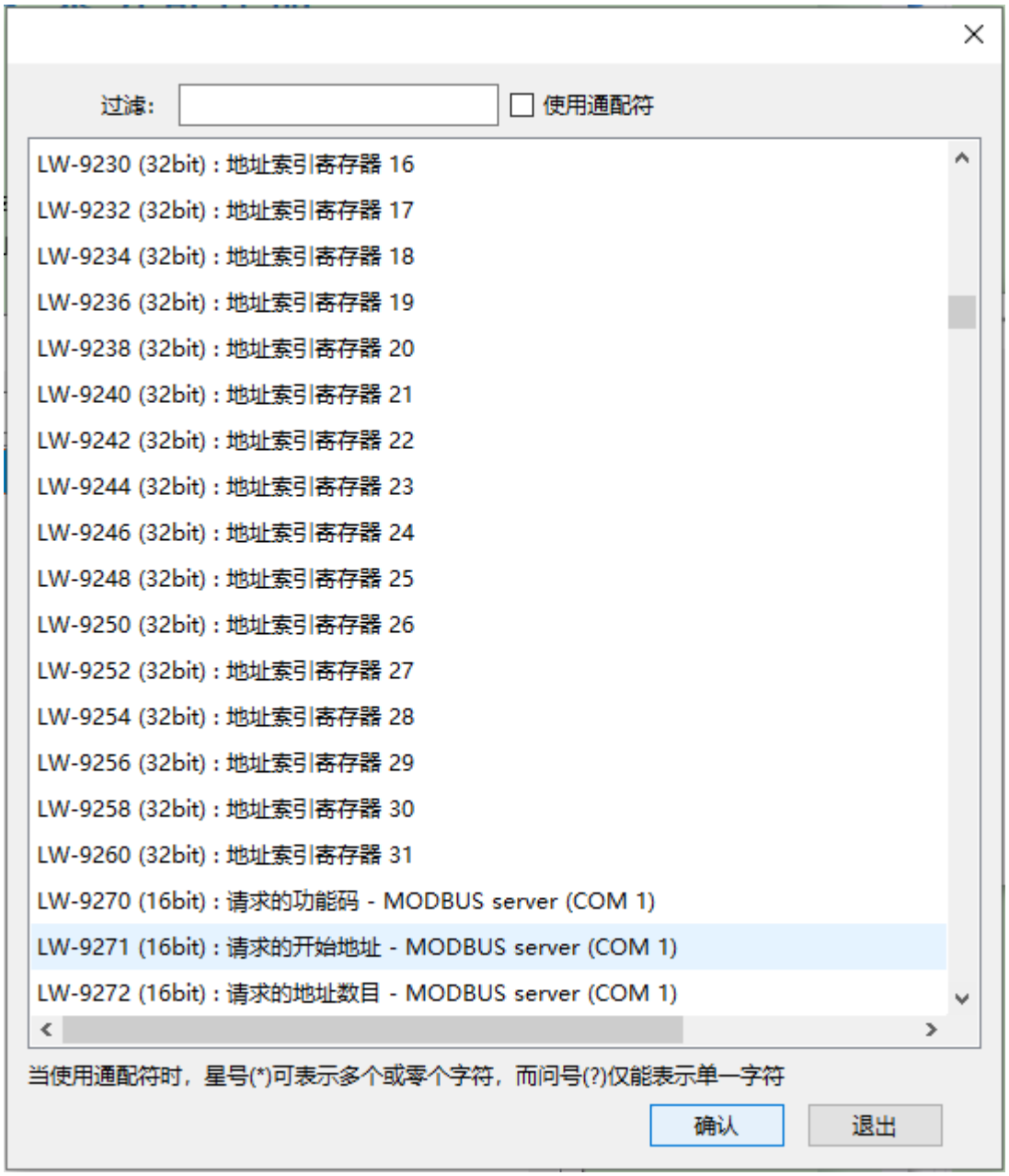
Chpater 10 Index Register

10.1 Overview

This software 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 16pcs of 16-bit and 16pcs of 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 65536 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.

32-bit 16 31 LW-9230 (32-bit) LW-9260 (32-bit), 4294967296 words.

When using [Index register], the address is designated by the following equation:

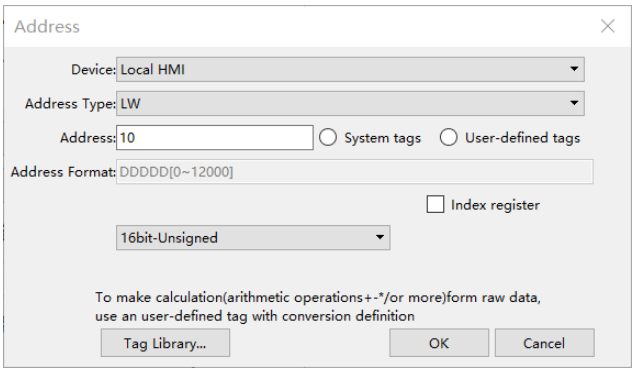
The constant set in [Address] + the value in the chosen Index Register.

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.

10.2 Example of Index Register

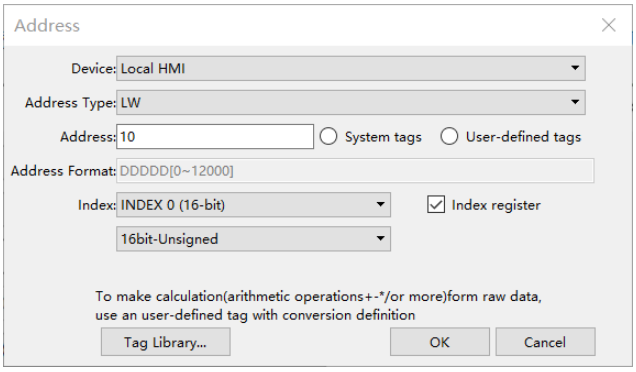
The following explains the way to designate the register while Index Register is used. If **not** selecting **[Index register]** check box and set address to [LW-10]. The system will directly read / write LW-10.



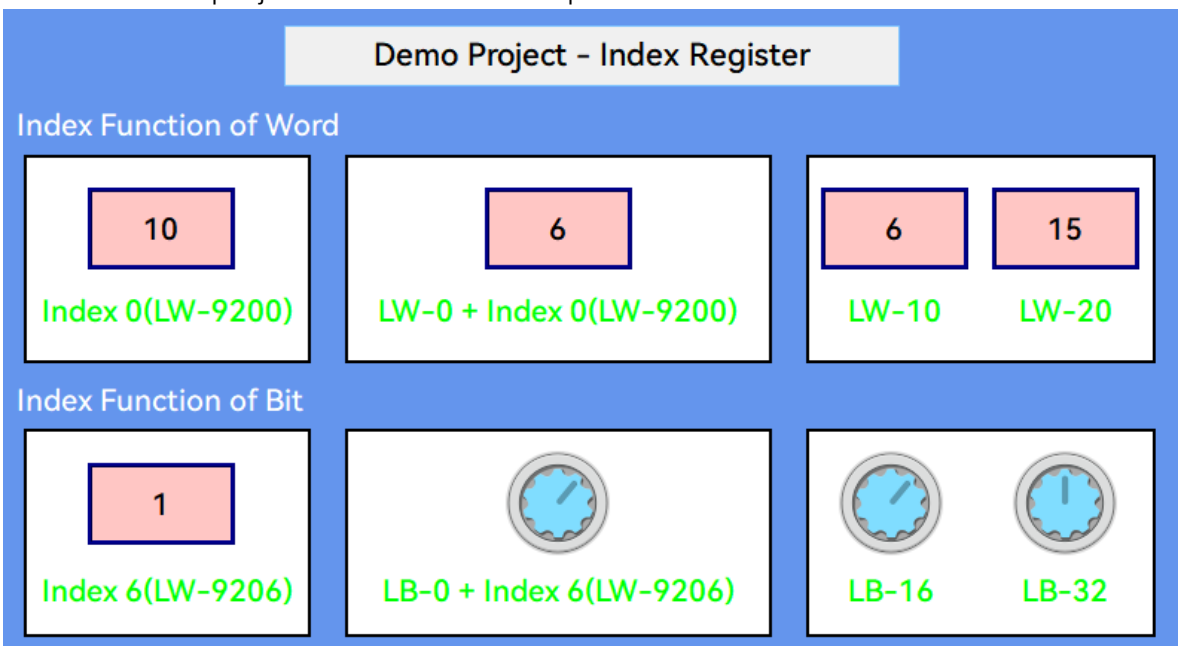
If select **[Index register]** check box and set [Index] to [INDEX 0 (16-bit)], the system will read / write [LW(10 + value in Index Register 0)].

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

[] [] [INDEX 0 (16-bit)], [LW-(10 + 0)].

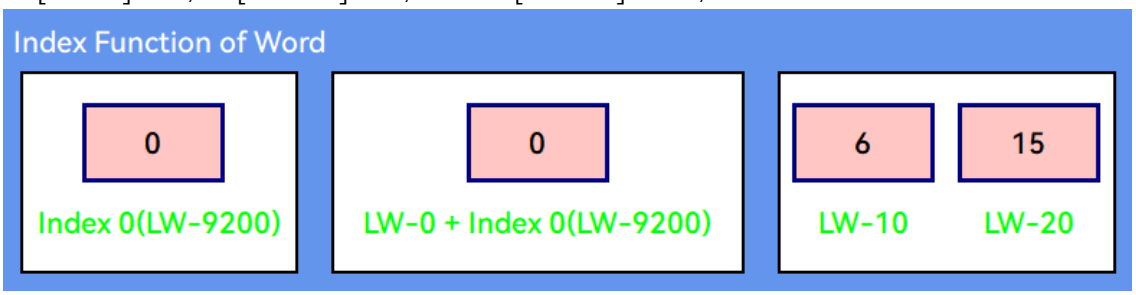


Here’s a demo project shown as an example:

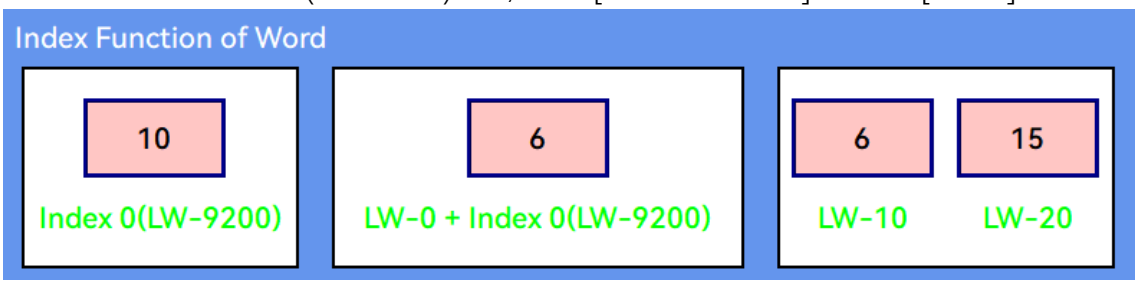


Example 1 1

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 6, and in [LW-20] is 15, the result is:



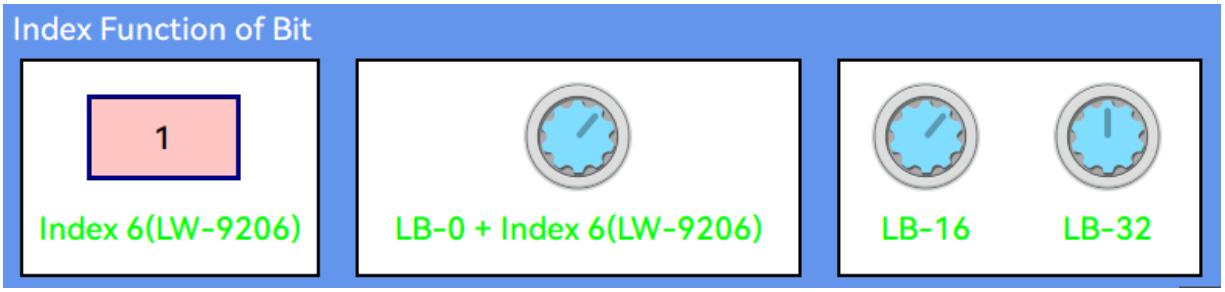
If the value in Index 0 (LW-9200) is 0, then [LW0 + Index 0] = read [LW-0].



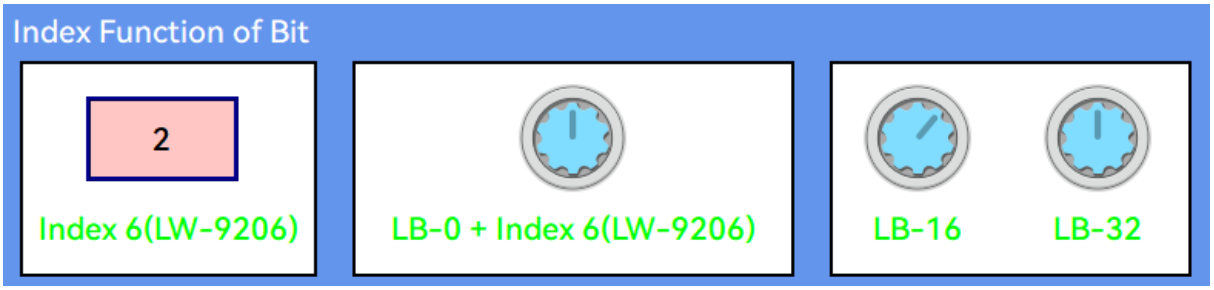
If the value in Index 0 (LW-9200) is 10, then [LW0 + Index 0] = read [LW-10] = 6.

Example 22

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 equals to 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.

Chapter 11 Keyboard Design & Usage

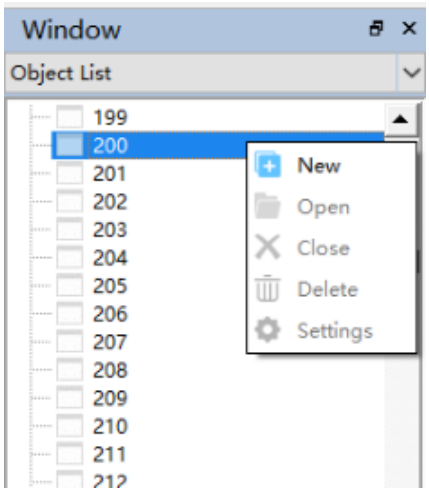
11.1 Overview

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 this software, you can create the keyboard if needed. The types of the keyboards are:

Popup Keyboard (with or without title bar)
Fixed Keyboard
Unicode Keyboard

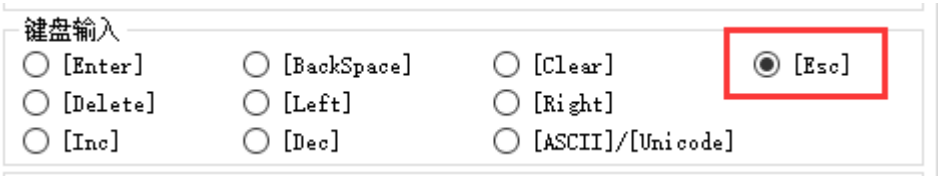
11.2 Steps to Design a Pop-up Keyboard

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.

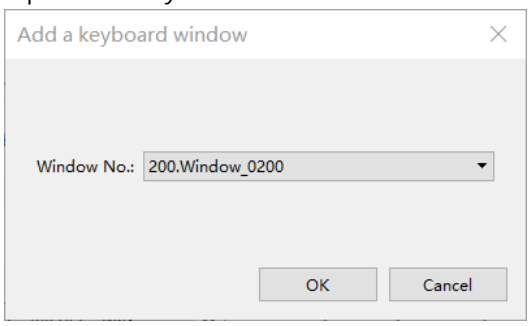


Set another Function Key object as the [Enter] key.

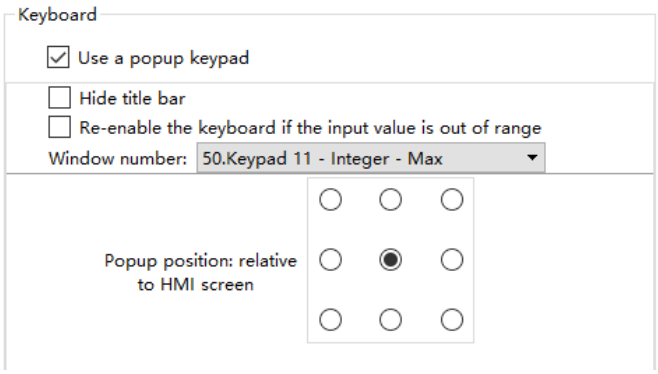
The rest are mostly used to enter numbers, for example for input singal of value 1.

3. Select a suitable picture for each Function Key object..

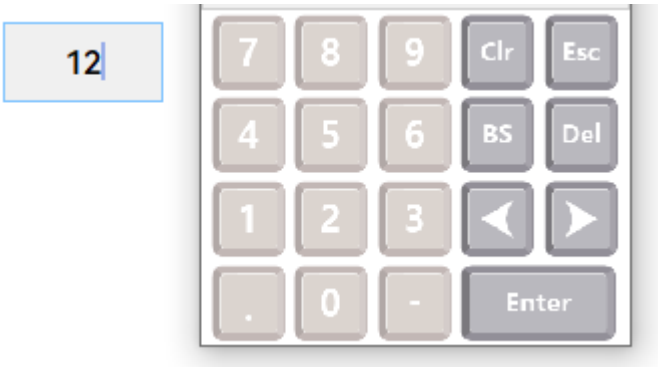
4. Select [System Parameter Settings] » [General] » [Keyboard] » [Add] to add “window no. 200”. Up to 32 keyboards can be added.



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 [200. Keyboard] ,, 200,.



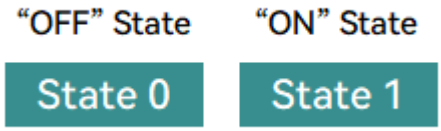
Chapter 12 Objects

This chapter explains how to use different objects. .

12.1 Bit Lamp

12.1.1 Overview

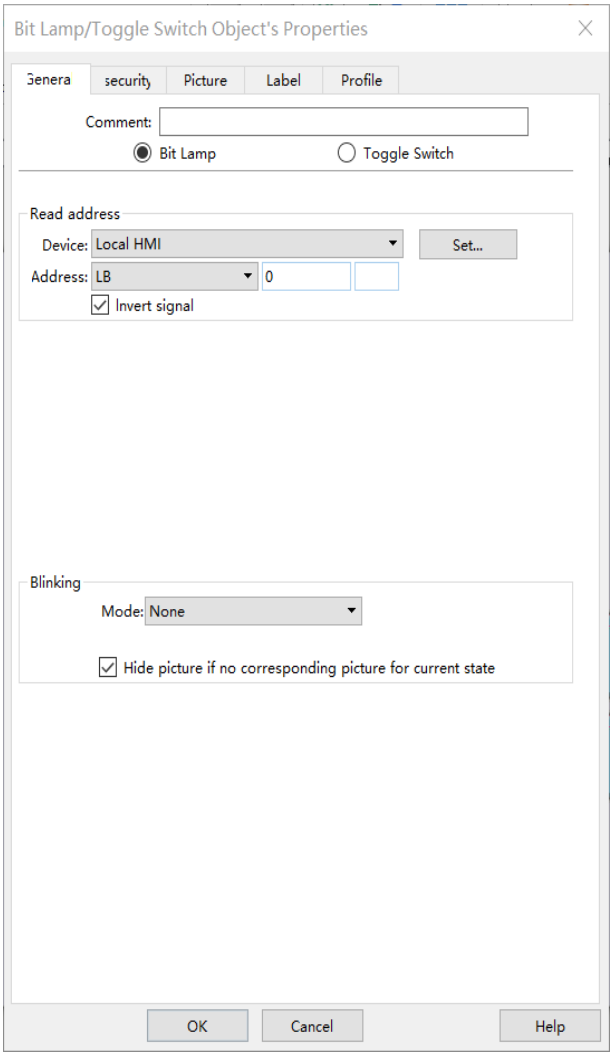
Bit Lamp object displays the state of a designated bit address. If the bit state is OFF, the State 0 shape will be displayed. If the bit state is ON, the State 1 shape will be displayed.



12.1.2 Configuration


Click [Object] » [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.

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.

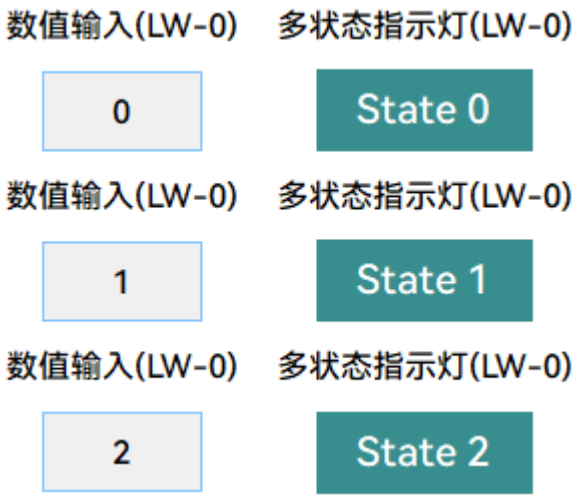
Note

- In [Label] tab, if select [ON=OFF (use state 0)] check box, both state 0 and 1 follow the settings of state 0.

12.2 Word Lamp

12.2.1 Overview

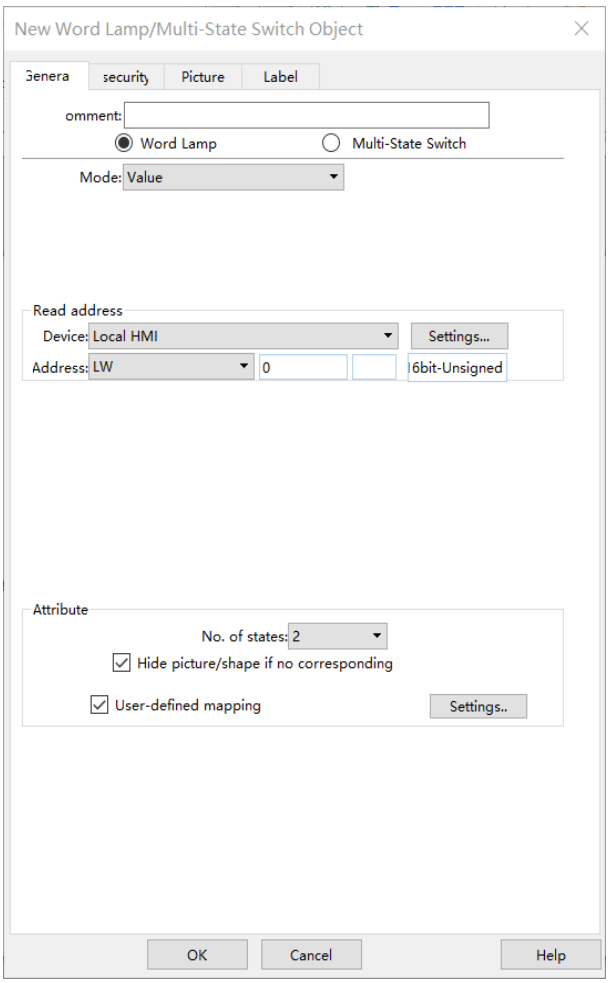
Word Lamp object displays the state according to the value of a designated word register. Up to 256 states are available. When the value of the register is 0, State 0 appearance of the object is displayed, and with the register value being 1 the object displays State 1, and so on.



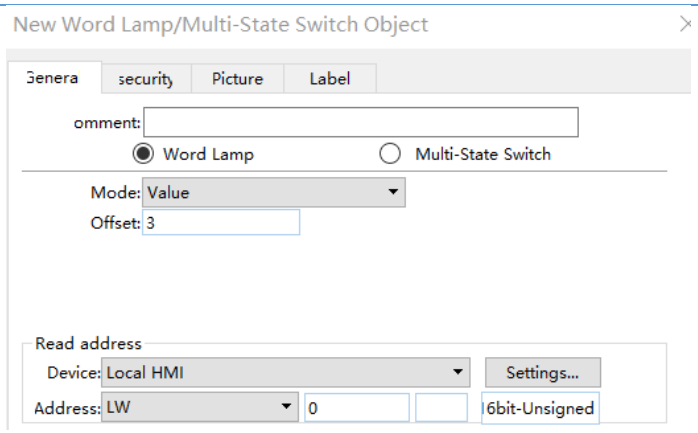
12.2.2 Configuration

Click [Object] » [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.

General Tab



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.
Mode / Offset	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 0 is displayed. (value 3 - offset 3)



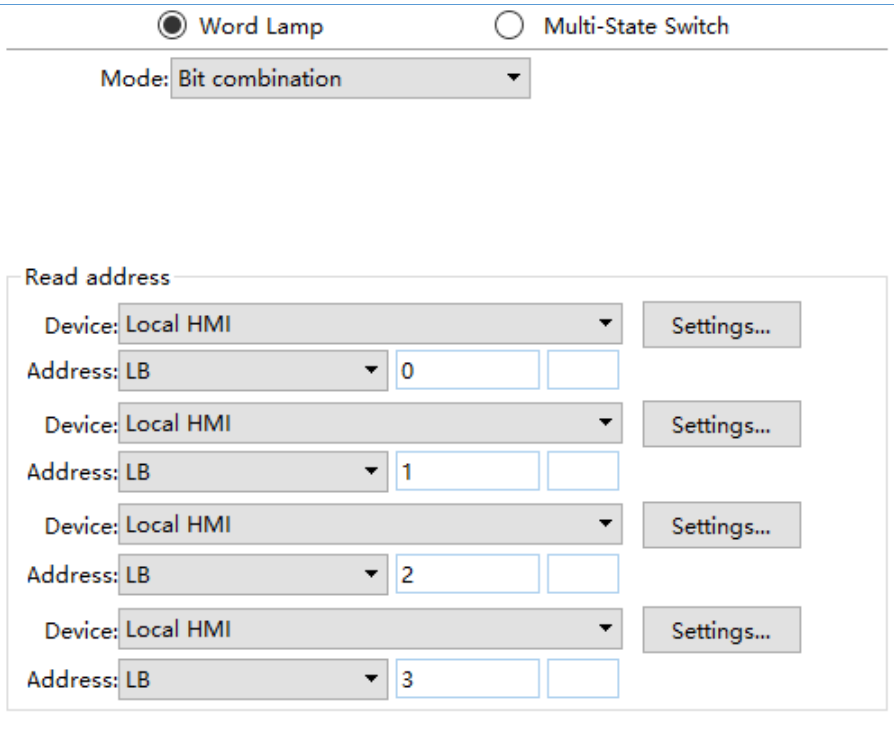
LSB

Convert the value from decimal to binary. The least significant active bit in a binary data word selects the state displayed.

Decimal	Binary	Displayed state
0	0000	State 0 displayed. All the bits are 0.
1	0001	State 1 displayed. The least significant active bit is bit 0.
2	0010	State 2 displayed. The least significant active bit is bit 1.
3	0011	State 1 displayed. The least significant active bit is bit 0.
4	0100	State 3 displayed. The least significant active bit is bit 2.
5	0101	State 1 displayed. The least significant active bit is bit 0.
6	0110	State 2 displayed. The least significant active bit is bit 1.
7	0111	State 1 displayed. The least significant active bit is bit 0.
8	1000	State 4 displayed. The least significant active bit is bit 3.

Bit combination

Lamp state depends on the states of bit combinations, where PLC 1 represents the least significant bit (LSB), PLC 2 represents the next LSB, and so on. Maximum number of bit is 4, for a total of 16 states. Changing [No. of states] in Attribute group box changes the number of read addresses.



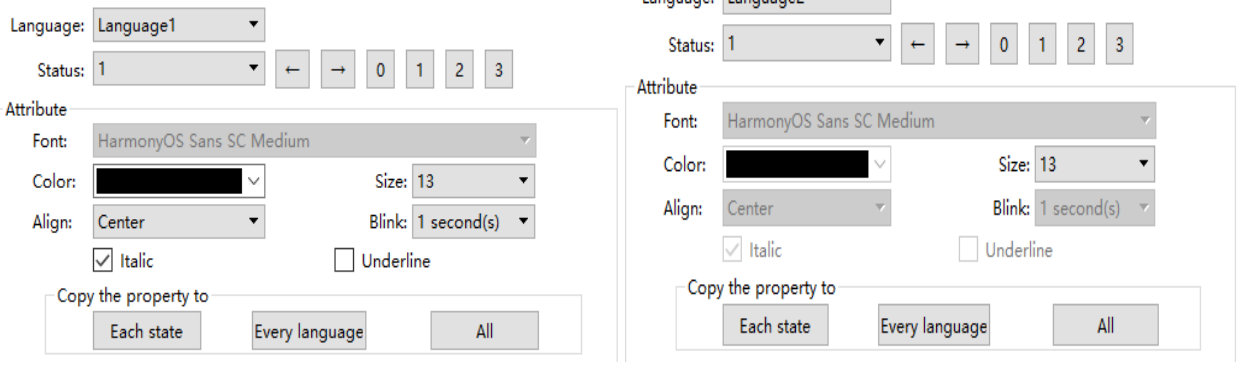
Change state by time

The state displayed changes on a time basis. The frequency can be set.

Read Address	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.
Attribute	**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.

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, 1 , 2~8 , 1 .



12.3 Set Bit

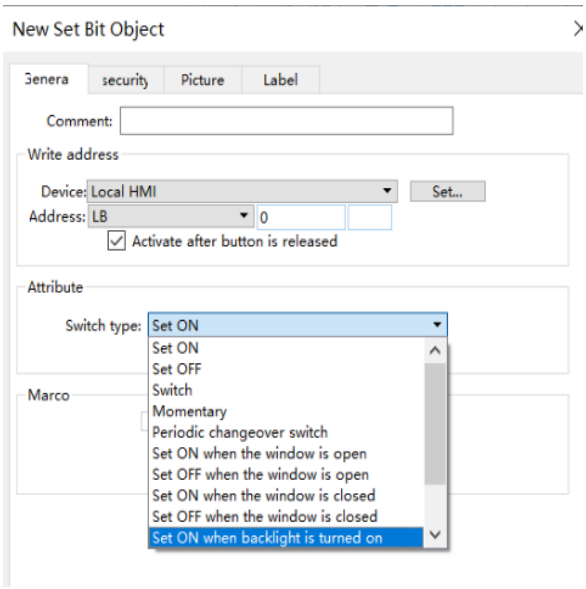
12.3.1 Overview

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.

12.3.2 Configuration

Click [Object] » [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.

General Tab



Setting	Description

<table>
<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>Attribute</td><td>
<table>
<tr><td>Set type</td><td>Description</td></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, i.e. ON->OFF, OFF->ON.</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</td><td>Set the bit ON when the backlight is turned ON.</td></tr>
<tr><td>Set OFF when backlight on</td><td>Set the bit OFF when the backlight is turned ON.</td></tr>
<tr><td>Set ON when backlight off</td><td>Set the bit ON when the backlight is turned OFF.</td></tr>
<tr><td>Set OFF when backlight off</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 “17 Macro References”.</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>

Note

- In [Label] tab, if select [ON=OFF (use state 0)] check box, both state 0 and 1 follow the settings of state 0.

12.4 Set Word

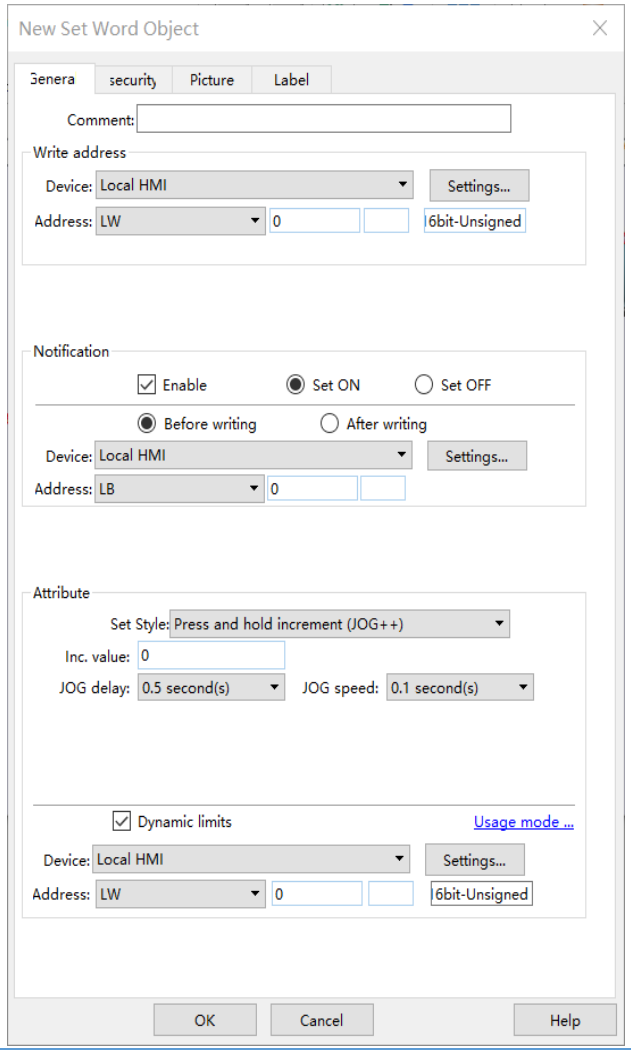
12.4.1 Overview

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.

12.4.2 Configuration

Click [Object] » [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.

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 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:

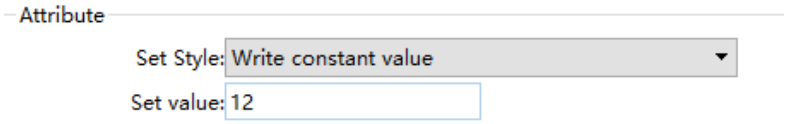
Content	16-bit	32-bit
Dynamic Address	LW-100	LW-100
Bottom limit	LW-100	LW-100
Upper limit	LW-101	LW-102

Example 2 2

The available button actions are:

Write constant value

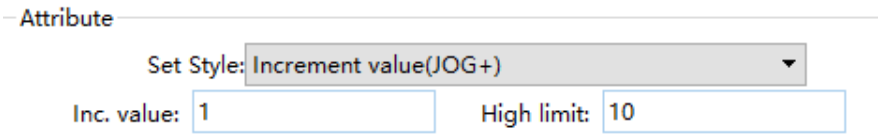
Write a preset value. Each time the button is pressed, the value in [Set value] is written to the designated register. Data format is as set by the [Write address] above; it can be 16-bit BCD, 32-bit BCD, ···32-bit float. In the following figure, when the button is pressed, the number 12 is written to the designated register.



- Increment value (JOG+)

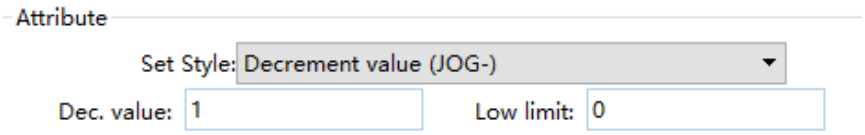
Increase value in register by a set amount in [Inc. value], each time when the button is pressed, up to the [Upper limit]. As shown below, each button press increases the value in the register by 1 until the value is 10.

- (JOG+)



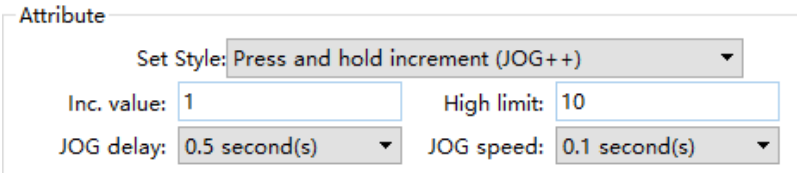
- Decrement Value (JOG-)

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.



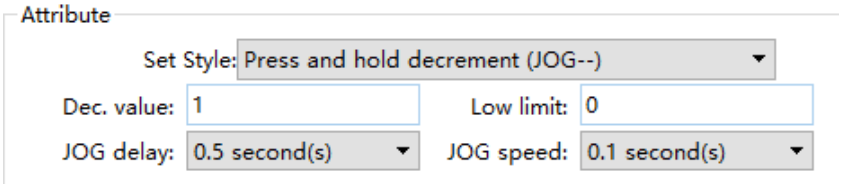
- Press and hold increment (JOG++)

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 0.1 second, till the value is 10.



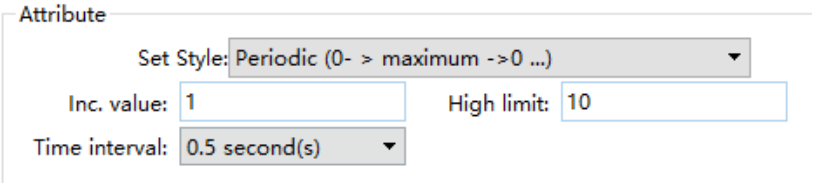
- Press and hold increment (JOG--)

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 0.1 second, till the value is 0.



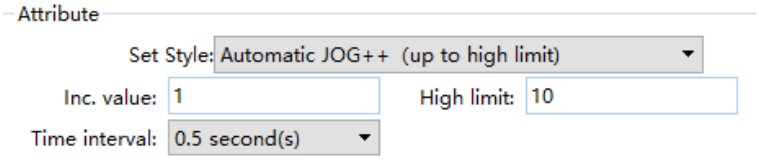
- Periodic JOG++ (0>up>0)

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.



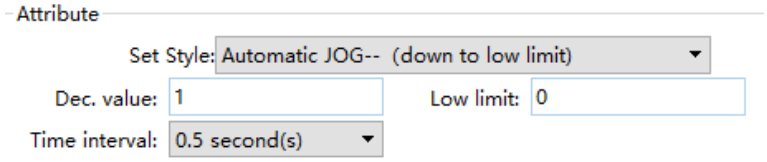
- Automatic JOG++

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.



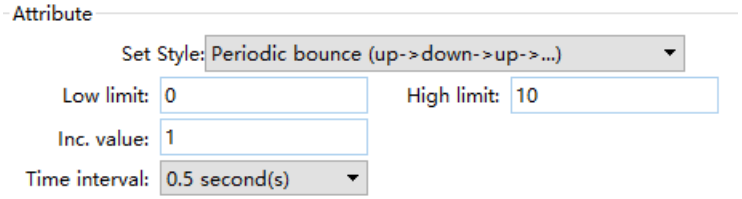
- Automatic JOG--

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.



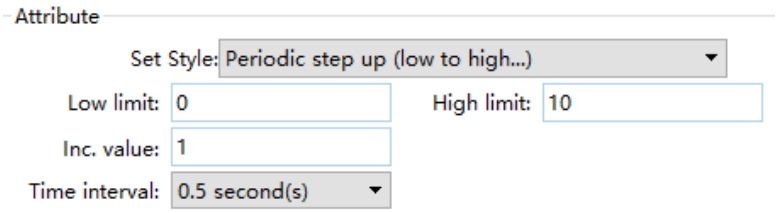
- Periodic bounce

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.



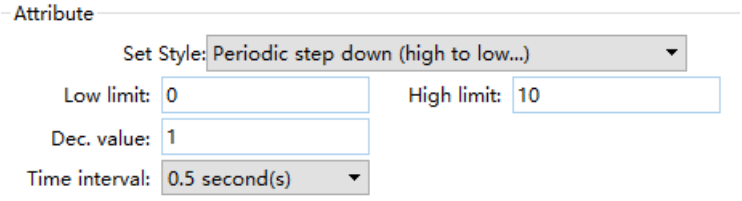
- Periodic step up

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.



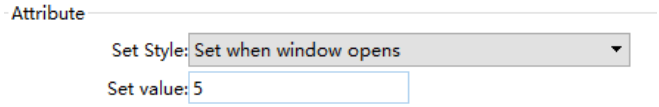
- Periodic step down

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.



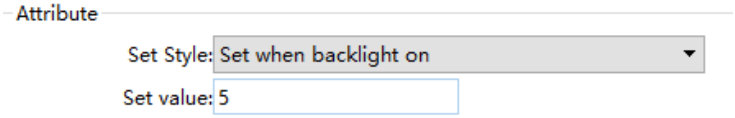
- Set when window opens / Set when window closes

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.



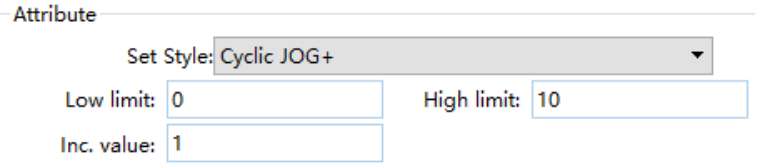
- Set when backlight on / Set when backlight off (Not supported on cMT-SVR)

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.



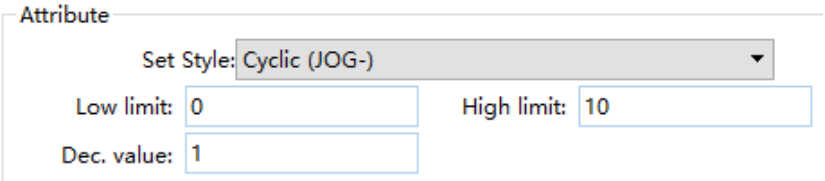
- Cyclic JOG+

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.



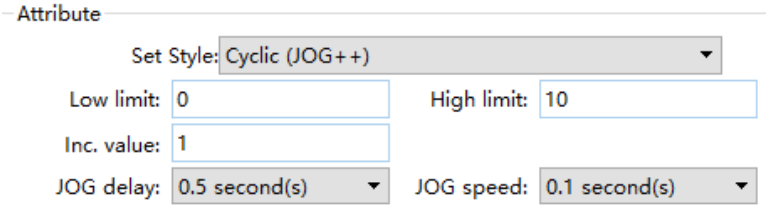
- Cyclic JOG-

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++

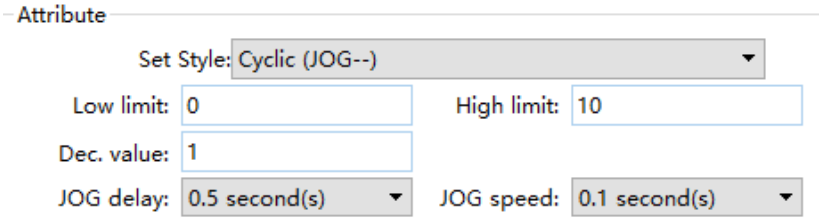
When the button is held longer than a set time in [JOG delay], it increases the value in a register by a set amount in [Inc. value] at a set rate in [JOG speed], to the [Upper limit], then reset to the [Bottom limit]. As shown below, when the button is held longer than 0.5 second, increase the value 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.



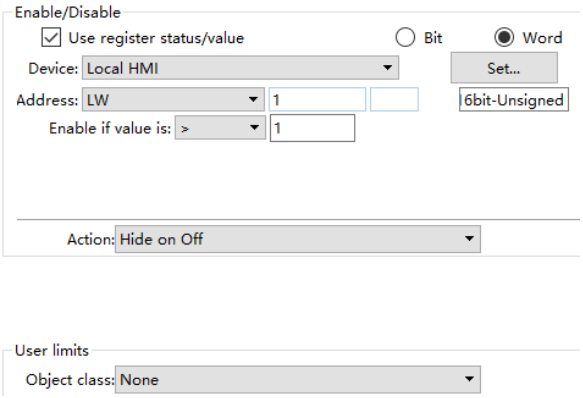
- Cyclic JOG- -

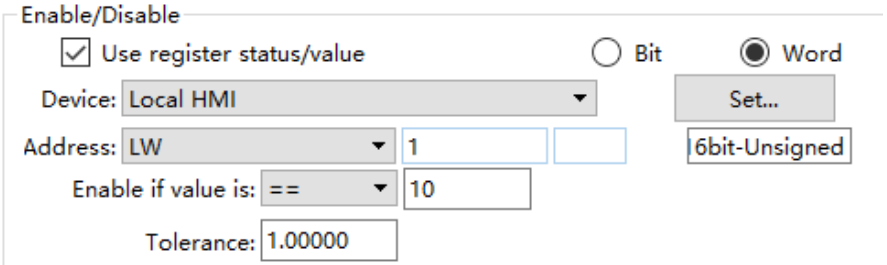
When the button is held longer than a set time in [JOG delay], decrease the value in a register by a set amount in [Dec. value] at a set rate in [JOG speed], to the [Bottom limit], then reset to the [Upper limit]. As shown below, when the button is held longer than 0.5 second, decrease the

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.



Security Tab



Setting	Description
Enable/Disable	If the option [Use register status/value] is enabled and [Word] is selected, whether the object is operable depends on the condition of a word address specified in [Trigger if value is:]. In the settings above, the object is operable only when the value in LW-1 is greater than 1. **Hide when disabled** The object is hidden when the specified condition does not occur in the specified word address. **Grayed label when disabled** The label of the object turns gray when the specified condition does not occur in the specified word address.  **Enable if value is:** This setting is for specifying a trigger condition. The available options are: >, <, ==, <>, >=, and <=. A tolerance value can be set for conditions == and <>. For example: 

	When the value is the specified word address is greater than or equal to 11, or smaller than or equal to 9, the object will be hidden and is not operable.

12.5 Function Key

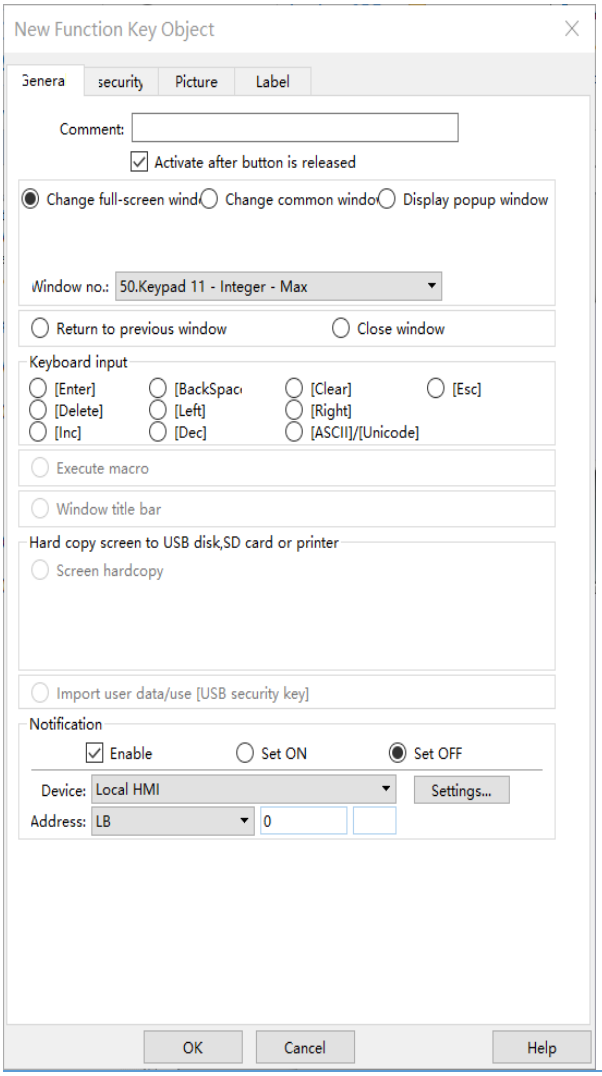
12.5.1 Overview

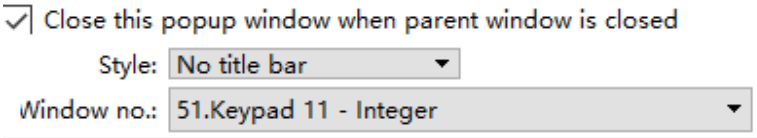
The Function Key object can be used for several tasks, such as switching between windows, keypad design, Macro execution, screen hardcopy, and setting USB security key.

12.5.2 Configuration

Click [Object] » [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.

General Tab



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.
Keyboard Input	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. **Line feed:** Move the cursor down to the next line. **Inc:** Increment by 1. **Dec:** Decrement by 1. **ASCII/UNICODE:** Specify the character to be entered by this key
Execute Macro	Select this check box to execute one of the Macros from the drop down list that has already been configured by users. 
Window Title Bar (Not enabled, grayscale display)	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.
Screen Hardcopy (Not enabled, grayscale display)	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].
Import user data / Use [USB Security Key]	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.
Notification	If this selection is enabled, it will notify a designated bit address to set ON or OFF, each time the button is pressed.

12.6 Toggle Switch

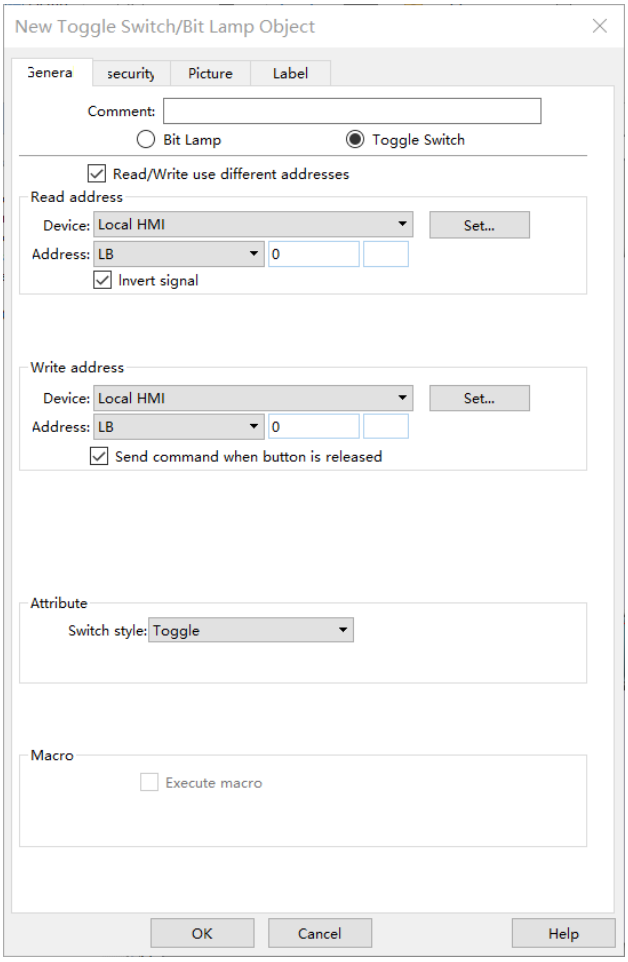
12.6.1 Overview

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.

12.6.2 Configuration

 Click [Object] » [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.

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/Write use different addresses	Different addresses can be used to read data and write data.
Read address	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. When [Read/Write use different addresses] option is not selected, the title of this group box will be “Read/Write address”.
Write address	Click [Setting] to select the [PLC name], [Address], [Device type], [System tag], [Index register] of the bit device that controls the

<table>
<tr><td></td><td>[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><td>Set Style</td><td>Description</td></tr>
<tr><td>Set ON</td><td>Set ON the designated bit of the device after press the object.</td></tr>
<tr><td>Set OFF</td><td>Set OFF the designated bit of the device after press the object.</td></tr>
<tr><td>Toggle</td><td>Alternates the bit state each time pressed. If the state is ON, will be displayed OFF. Vice versa.</td></tr>
<tr><td>Momentary</td><td>Holds the bit ON only while button is pressed while the bit will be OFF once release the pressing.</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.</td></tr>
</table>

12.7 Multe-state Switch

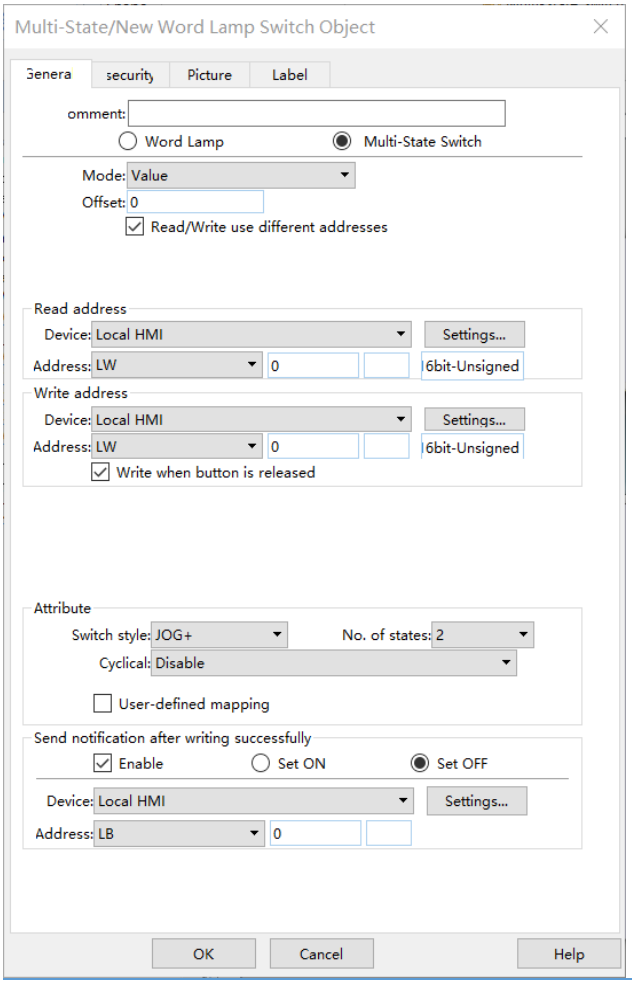
12.7.1 Overview

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.

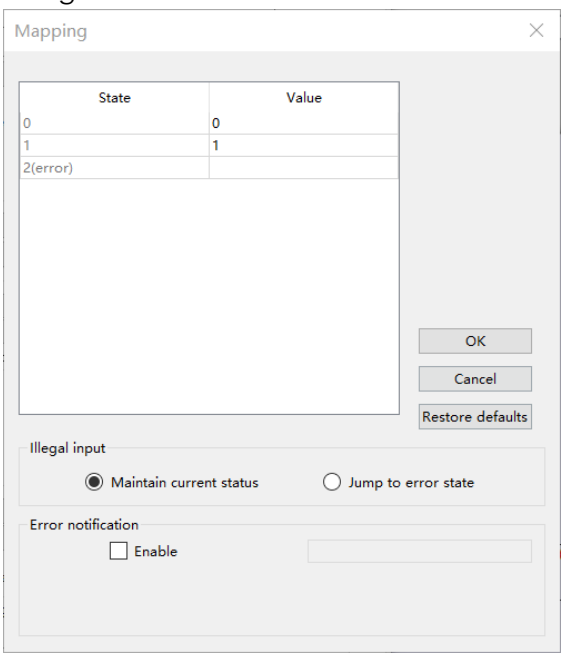
12.7.2 Configuration

 Click [Object] » [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.

General Tab



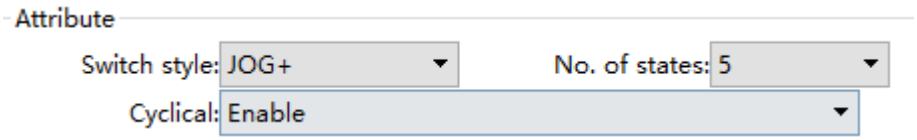
Setting	Description
Comment	User can describe the information of the object. **Word Lamp / Multi-State Switch** Switch between Word Lamp and Multi-State Switch features.
Model/Offset	Different modes can be selected: [Value], [LSB]. For more information, see “12.2 Word Lamp”.
Read/Write use different addresses	Different addresses can be used to read data and write data.
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. When [Read/Write use different addresses] option is not selected, the title of this group box will be “Read/Write address”
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.

Example 1

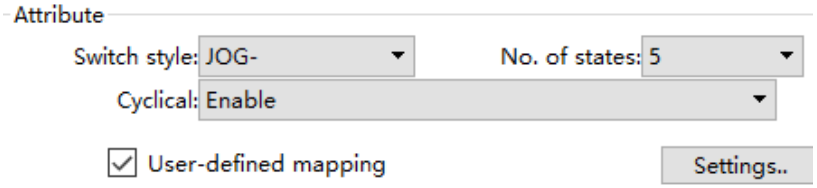
JOG+

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.



12.8 Combo Button Object

12.8.1 Overview

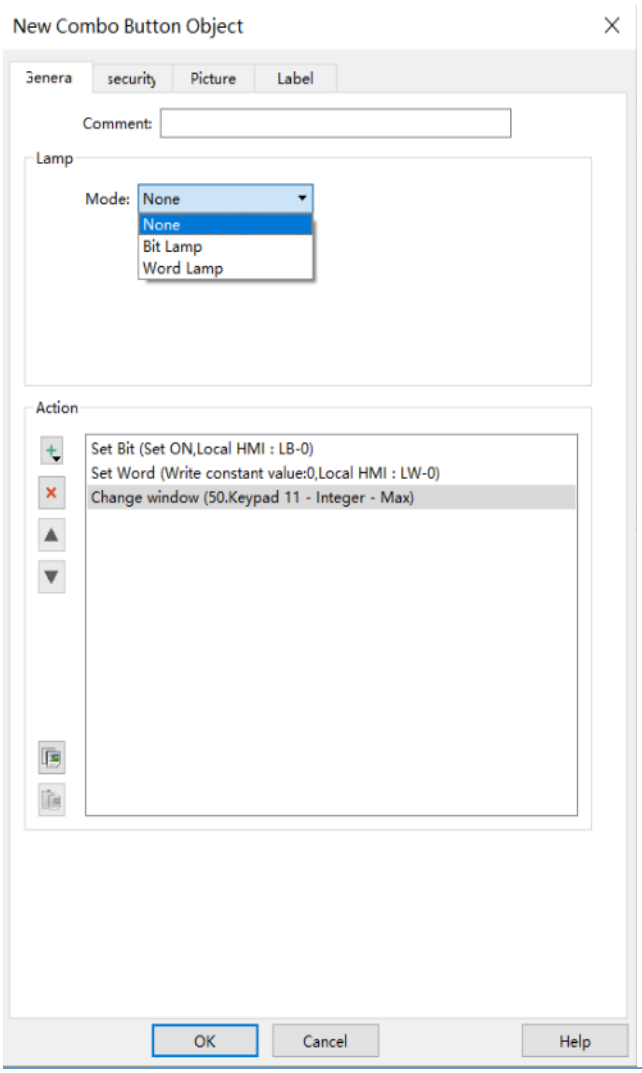
Combo Button can execute multiple commands. The former way was to overlay multiple objects in the same position, and the commands are executed in the order of the layer of the objects. This takes time to test the order when planning the project. Combo Button allows users to easily set multiple commands 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.

12.8.2 Configuration

 Click on the Combo Button icon on the toolbar or select [Objects] » [Combo Button] to create object. Configure object properties, click OK, a new Combo Button object will be created. []

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.

<table>
<tr><td>Add</td><td>Delay
Delays the action for a few milliseconds. A combo button can set one [Delay] action only.
Set Bit
Sets the designated bit ON or OFF.<table><tr><td>Set Style</td><td>Description</td></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></table>Set Word
Sets the value in the designated register.<table><tr><td>Set Style</td><td>Description</td></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></table>Change Window
Switch to the designated window. A Combo Button Object can only one [Change Window] action and the action will be executed at last.</td></tr>
<tr><td>Action</td><td>Configurate the executed action, such as [Delay],[Set Bit].[Set Word] and [Change Window]. A combo button can execute up to 20 actions.
  change the order of actions.
  Copy  Paste  Delete
Copy, Paste or Delete the selected action.</td></tr>
</table>

12.9 Slider

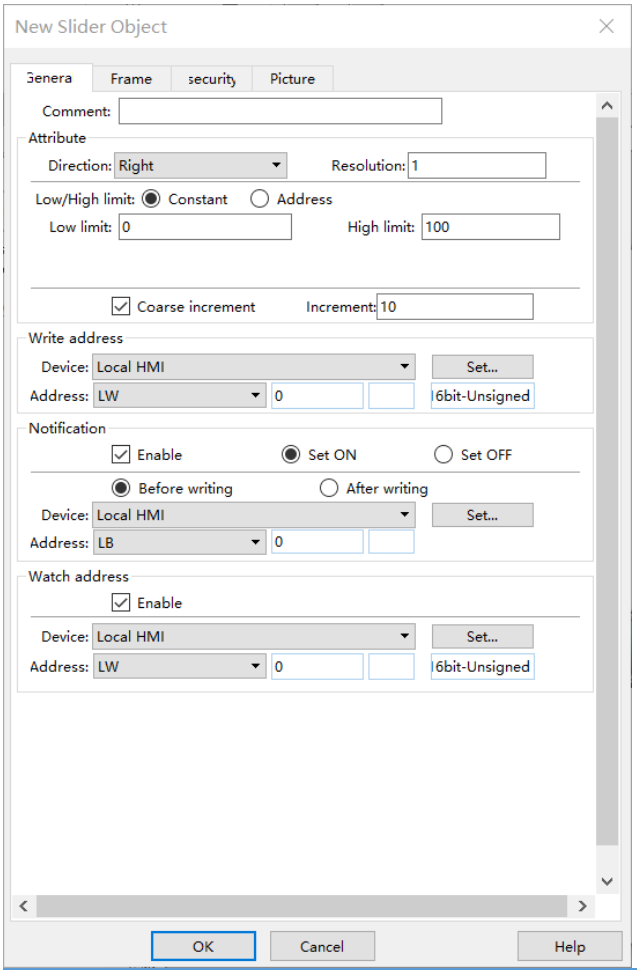
12.9.1 Overview

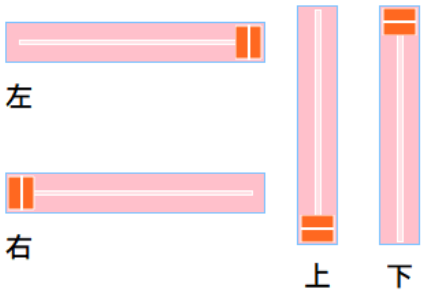
Slider object is used to change the value in a designated word register address by moving the slide on the screen.

12.9.2 Configuration

Click [Object] » [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.

General 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.
Write Address	Click [Setting] to select the [Device], [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 before or after writing. Click **[Setting]** to select the [Device], [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.

Example 1

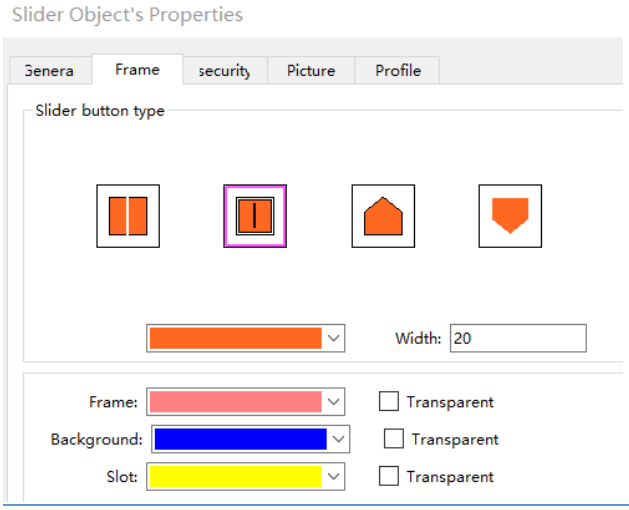
Set the low or high limit by a designated register. When write address is LW-n, where n is an arbitrary number, the rule of setting limits is:

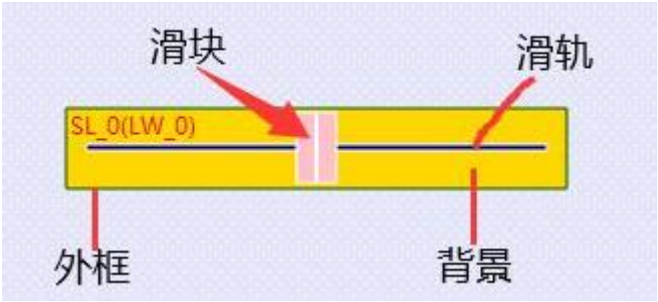
Content	16-bit	32-bit
Address	LW-n	LW-n
Low limit	LW-n	LW-n
High limit	LW-n+1	LW-n+2

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

Outline Tab



Setting	Description
Slider	Four default styles are offered, and the width and color of the Frame, Background, Slot can be set. 

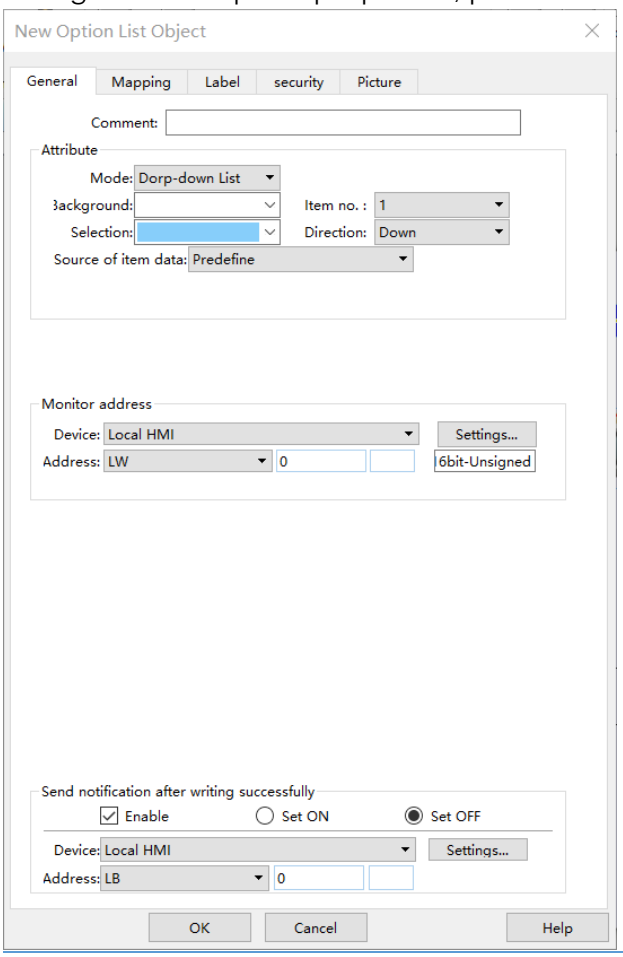
12.10 Option List

12.10.1 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.

12.10.2 Configuration

Click [Object] » [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.



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 12.10.2.1.
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.

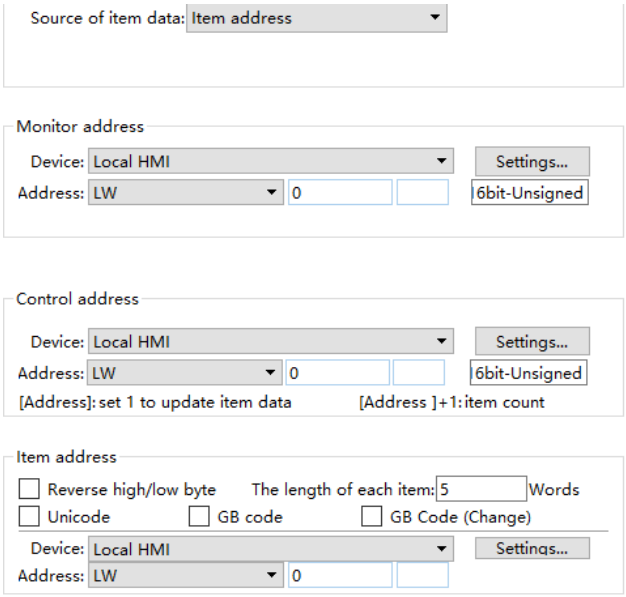
12.10.2.1 Source of Item Data

- **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].

- **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

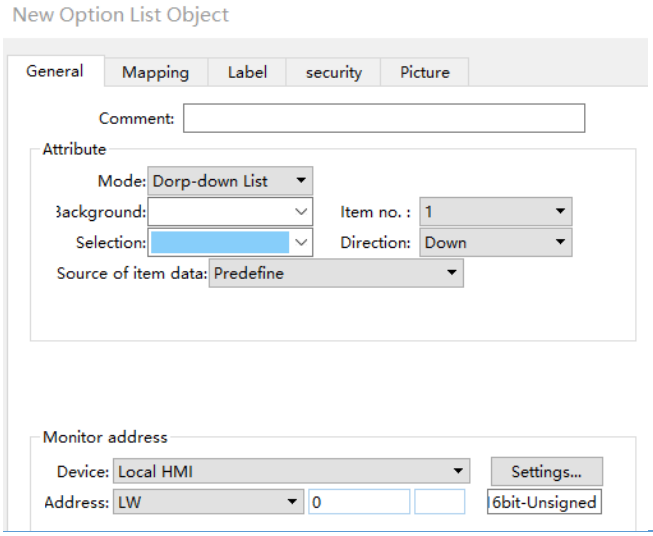
The UNICODE characters used here should be used by Text object, so that this software will compile the needed fonts and download these fonts to HMI, then the UNICODE letters can be correctly displayed.

[The number of items] multiplied by [The Length of each item] must be less than 1024 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.



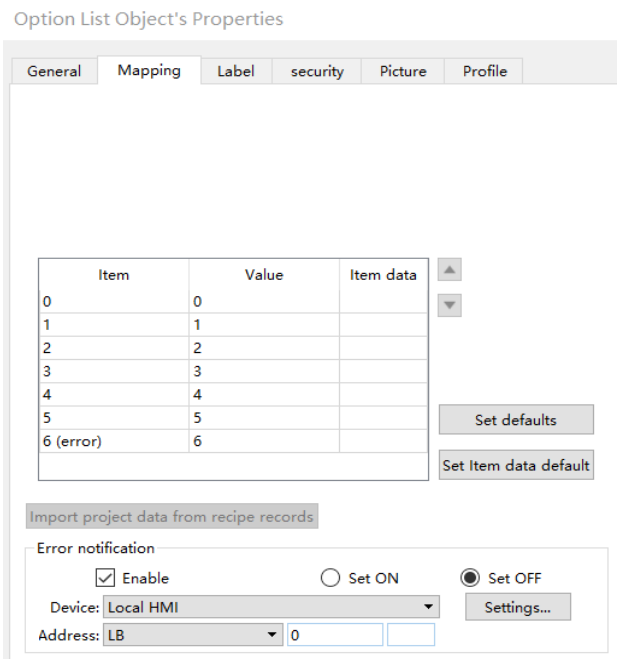
Setting	Description
Sort	Select the sorting method from [Ascending] of [Descending].
Display	If [Privilege] is selected, the privileges for each user will be displayed in option list. If [Secret user] is selected, even though it is defined to be hidden in [System parameter settings] » [Security] » [Enhanced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

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 [Records of Recipe Database] 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 Records of Recipe Database. After importing, modifying Records of Recipe Database 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.

12.11 Numeric

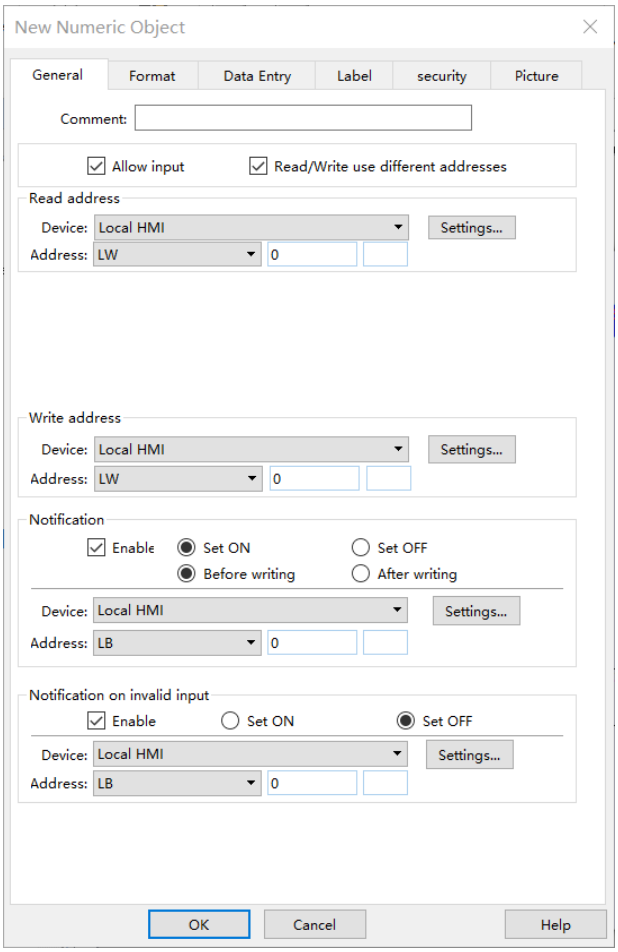
12.11.1 Overview

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

12.11.2 Configuration

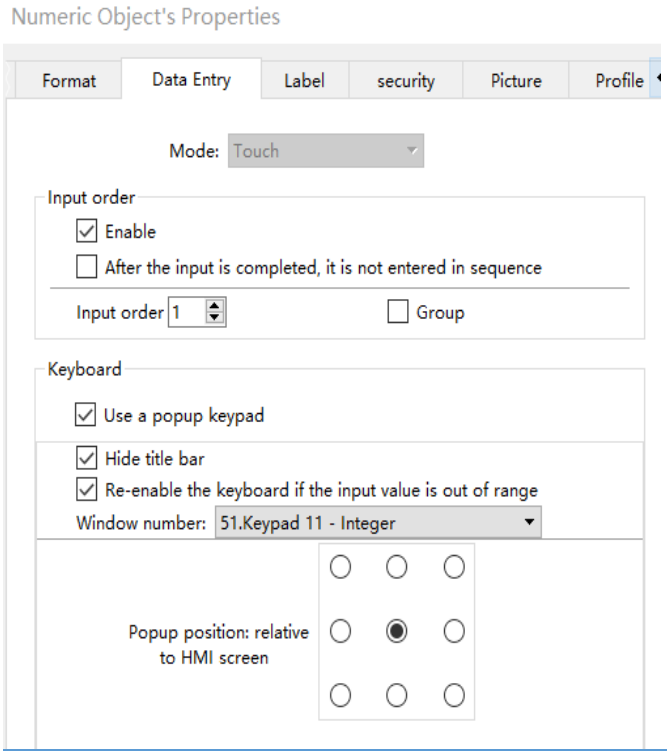

Click [Object] » [Numeric] icon on the toolbar to open a Numeric object property dialog box. Set up the properties, press OK button, and a new Numeric object will be created.

General Tab



Setting	Description
Allow input	If selected, the input features and relevant settings are enabled.
Read/Write use different address	Different addresses can be used to read data and write data.
Read address	Click [Setting] to select the [Device], [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. When [Read/Write use different addresses] option is not selected, the title of this group box will be “Read/Write address”.
Write address	Select the [Device], [Device type], [Address] of the word device that system writes to.
Notification	With notification enabled, the state of the designated bit address can be set on / off, before / after writing.
Notification on invalid input	If an illegal value is entered, automatically set the state of a designated register.

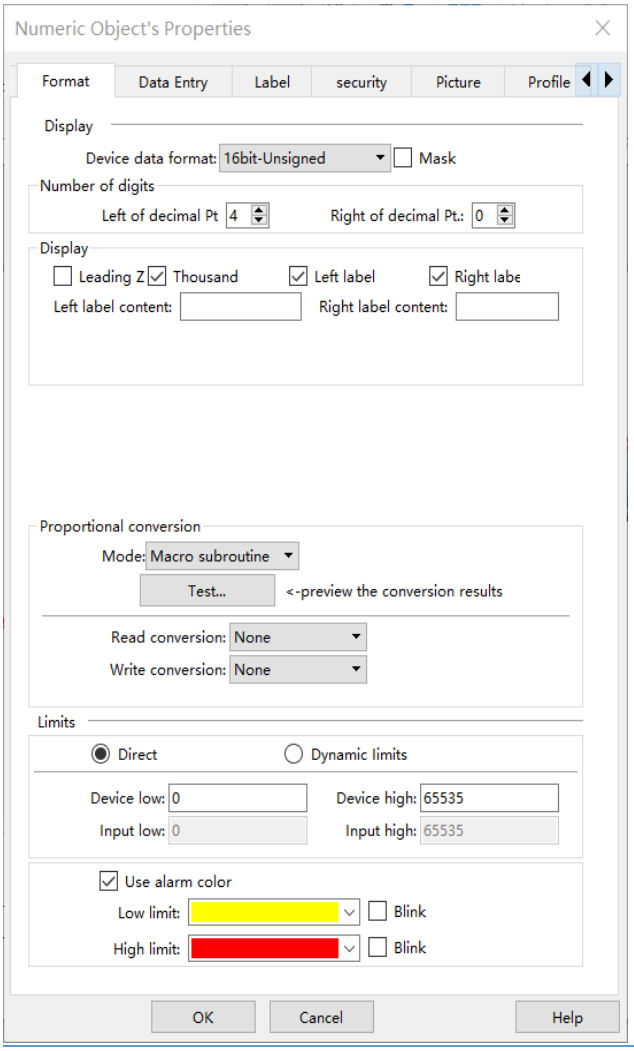
Data Entry Tab



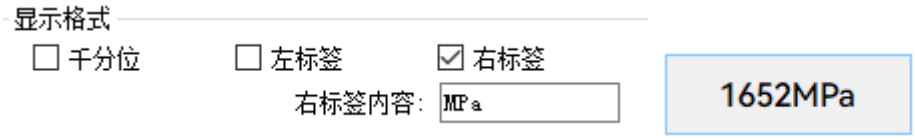
Setting	Description
Mode	**Touch** Used when data entry is initiated by touching the screen object. **Bit control** Used when data entry is enabled by turning ON a designated bit, and entry ends when the bit goes OFF.

Input control	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.
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	**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.

Format Tab



Setting	Description
Device Data Format	Data format Set the data format of a designated word register. Options include: BCD, Binary, Unsigned, Signed, HEX, and Float. 16-bit uses 1 word where 32-bit uses two words. (BCD)、 (Binary)、/ (Signed/Unsigned)、 (Hex)、 (Float) . 16-bit ,, 32-bit ,. **Mask** If selected, any values entered will be hidden by displaying them as asterisks "*".
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.
Display format	**Millimeter** **When the value of the register reaches the thousandth bit, "," will be displayed, as shown in the following figure:**

	1,000 **Left Label** **Add a left label to the value, and you can customize the display content without affecting the value of the register, as shown in the following figure:**  **Right Label** **Add a right label to the value, and you can customize the display content without affecting the value of the register, as shown in the following figure:** 
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** 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 defined in Macro Function Library. To use this feature, see “12.11.2 The rule of using Macro subroutine”.
Limit	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. To use consecutive addresses, 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.

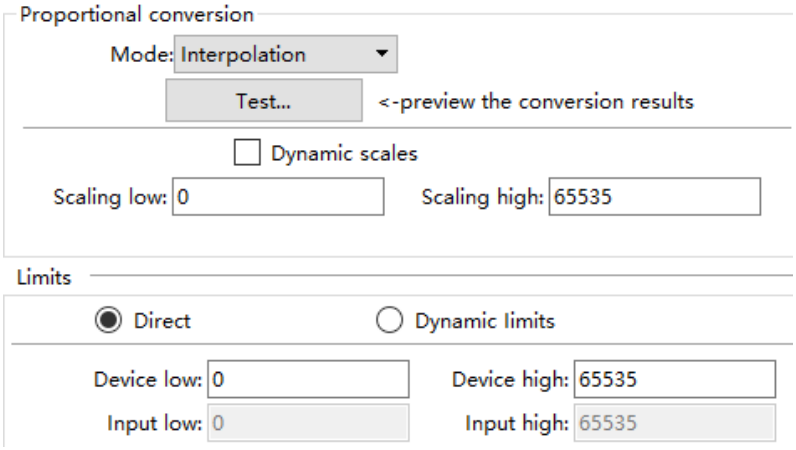
Blink

When the value in the register is outside either limit, the digits flash.

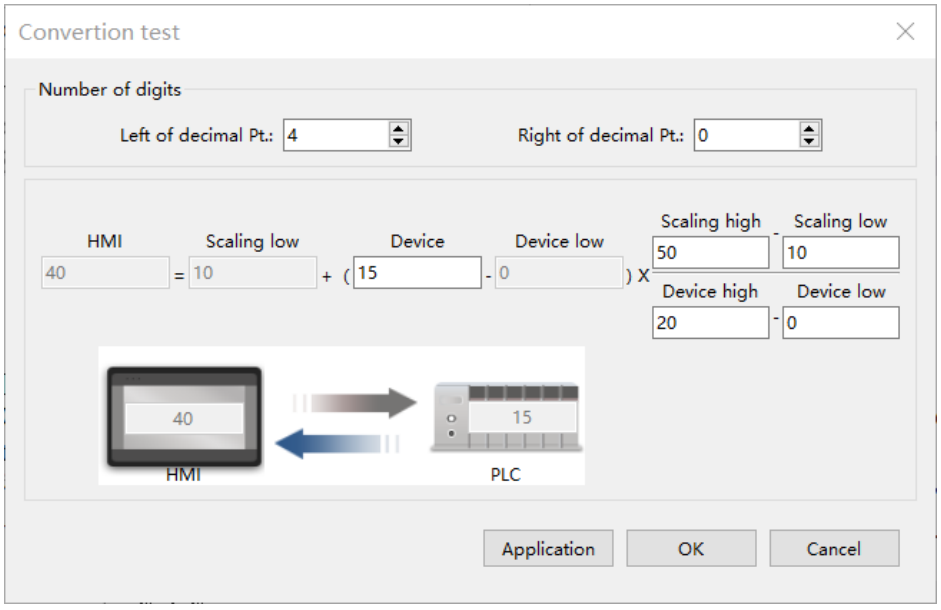
Example 1

If [Interpolation] is selected, the scaling equation is as the following:

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.

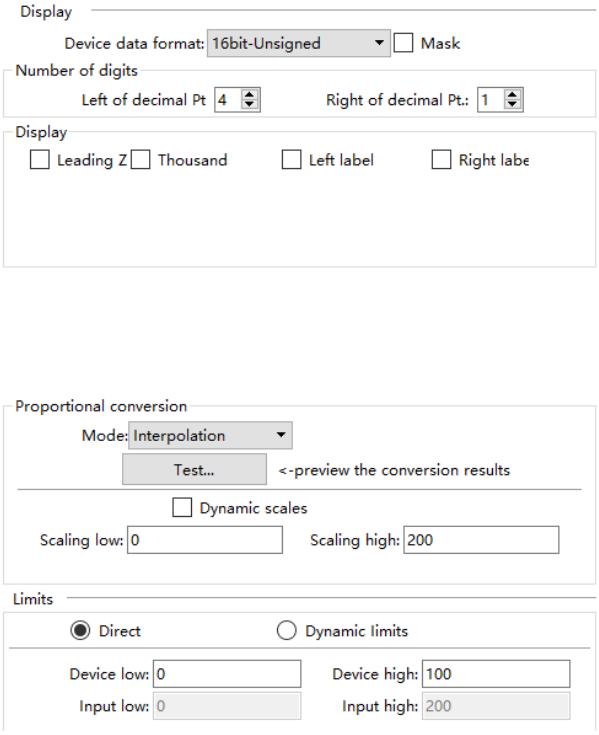


Example 2

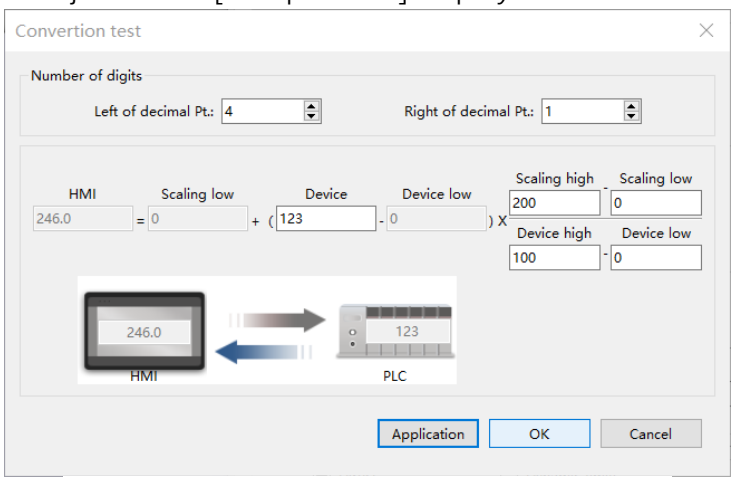
If the numeric format selected is not Float and decimal point is used, the decimal place of the

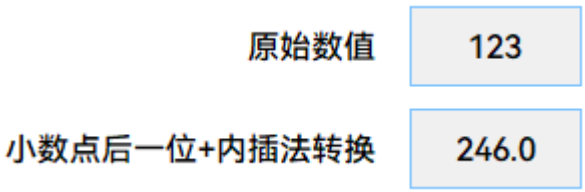
converted result will not be adjusted automatically, please adjust [Engineering high] to correctly place the decimal point of the result gained in [Interpolation] mode. Please see the illustration below.

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.

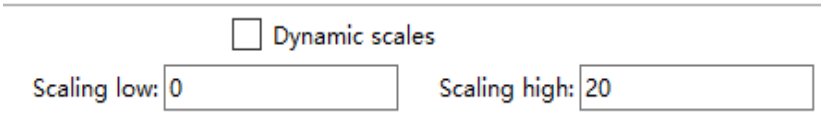


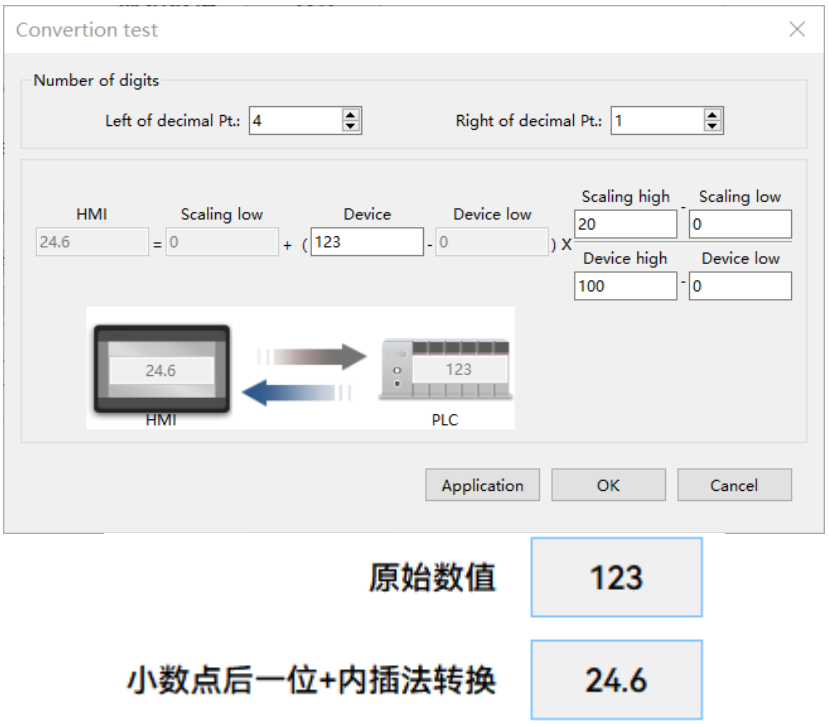
2. Enter value “123”, the object set to [Interpolation] displays “246.0” instead of “24.6”.





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





Example 3

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 4

Set the limits by a designated register. When [Address] is LW-n and [Use consecutive addresses] is selected, 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 5

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].

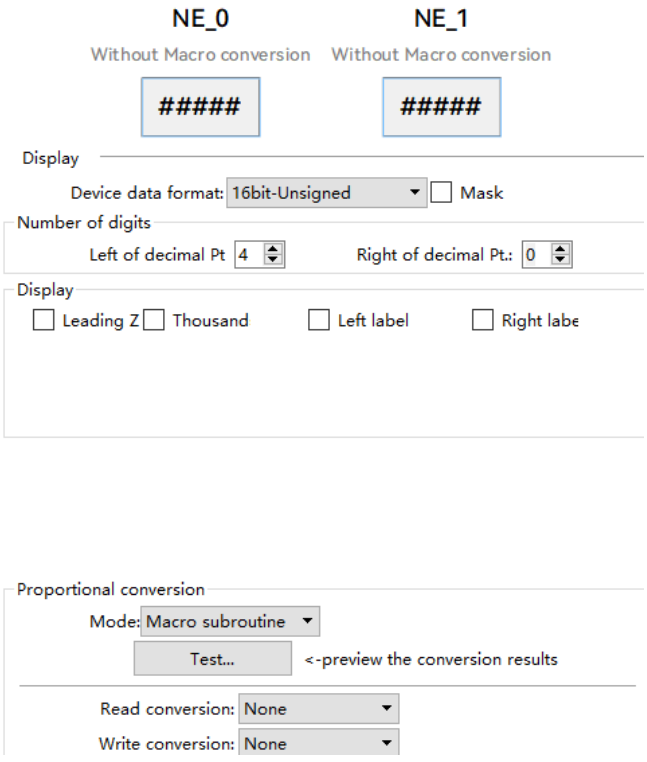
Read Conversion

```
int16 read(uint16 a)
{
   int16 b;
   b = a + 10;
   return b;
}
```

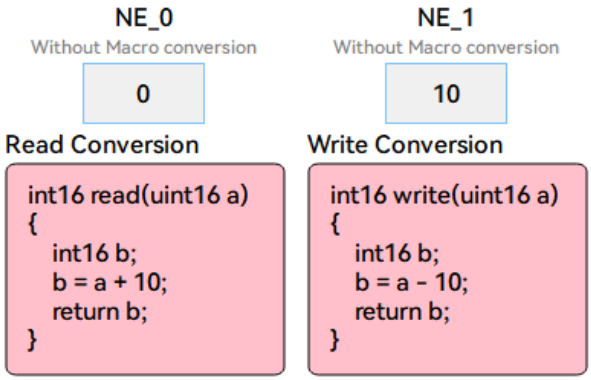
Write Conversion

```
int16 write(uint16 a)
{
   int16 b;
   b = a - 10;
   return b;
}
```

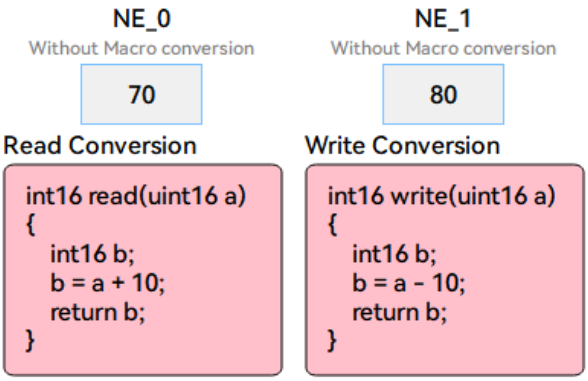
1. Create two Numeric objects: NE_0 and NE_1 and use the same control address. Select [Macro subroutine] for NE_1



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



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



Note

If executing [Read conversion] and [Write conversion] by the 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].

Rules for using macro subfunctions in numerical components

There must be a return value and there can be only one parameter, for example:
sub int test(uint16 a) // (correct)
sub int test(int a) // (wrong, no return value)
sub int test (int a, int b) // (wrong, there are two parameters)
The data format of the numerical component must correspond to the specific parameter type. As shown in the following table:

Macro para. type	Data format of numerical components
int16	16-bit Signed
int	32-bit Signed
uint16	16-bit BCD, 16-bit HEX, 16-bit Binary, 16-bit Unsigned
uint	32-bit BCD, 32-bit HEX, 32-bit Binary, 32-bit Unsigned
float	32-bit Float

If the data format of a numerical component is 16-bit Unsigned, only macro subfunctions with parameter type uint16 can be selected, for example:
sub int test(uint16 a) // (correct)

sub int test(int a) // (incorrect)

Only the address of the local HMI can be accessed. For example:

GetData(var,0, "Local HMI", "LB", 0, 1) // (correct)

GetData(var,0, "MODBUS RTU", "0x", 0, 1) // (wrong)

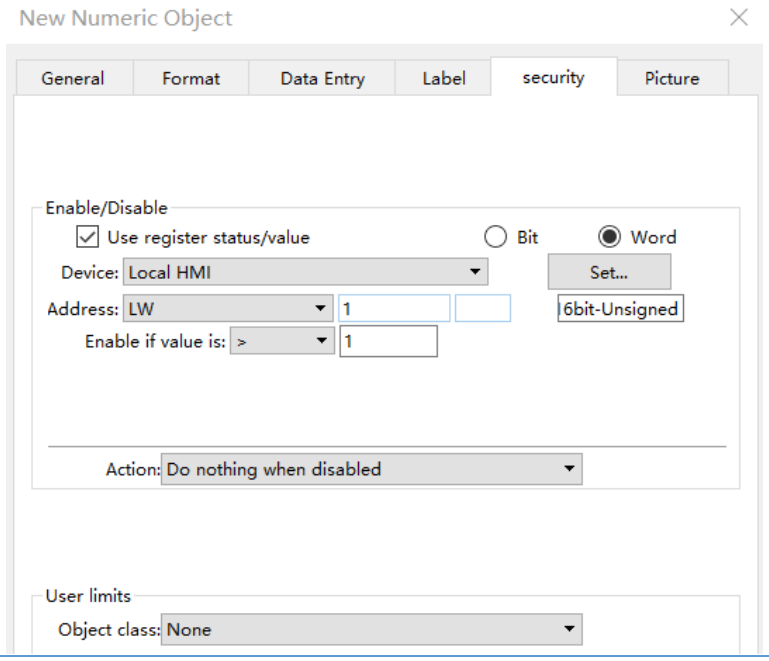
Unable to call the following functions:

ASYNC_TRIG_MACRO, SYNC_TRIG_MACRO, DELAY, FindDataSamplingDate, FindDataSamplingIndex, FindEventLogDate, FindEventLogIndex, INPORT, INPORT2, OUTPORT, PURGE, TRACE

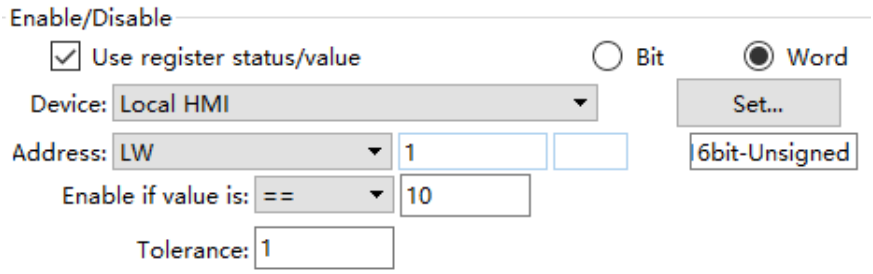
Unable to use the following loop statements

For,while

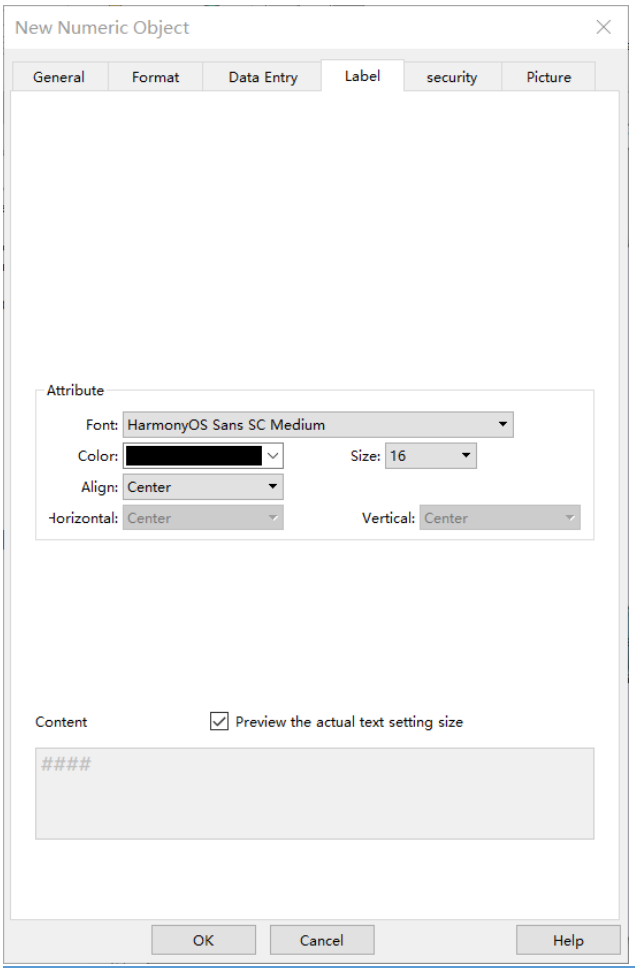
Security Tab



Setting	Description
Enable/Disable	With this option selected ans chose [Word] whether an object is operable depends on whether the specified condition occurs in the designated address. As shown in the figure, this object can only be operated when the value of LW-1 is greater than 1. **Hide when disabled** The object is hidden when the specified condition does not occur in the designated address. **Grayed label when disabled** The value in the object turns gray when the specified condition does not occur in the designated address.  **Use control token** With this option selected, whether an object is operable depends on whether a control token is acquired or unacquired. **Enable if Value is** enable conditions The conditions for specifying the address can be set,

	including>,<,==,<>,>=or<=. Where==and<>can be set as [Tolerance]. For example,  When the value of the specified address is greater than or equal to 11, or less than or equal to 9, the component will be disabled and hidden.

Label 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.
Size	Set the font size.

12.12 ASCII

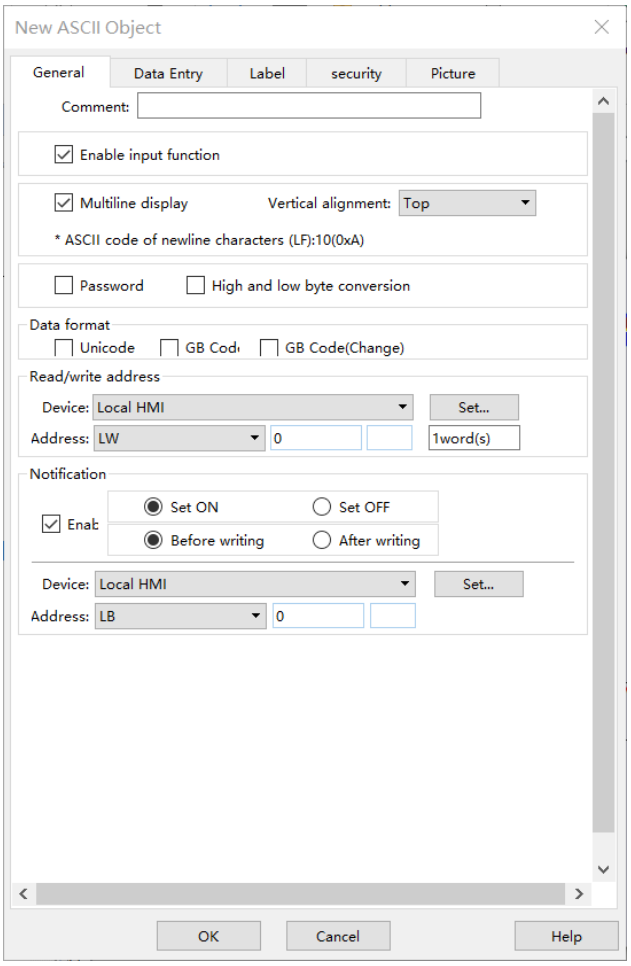
12.12.1 Overview

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

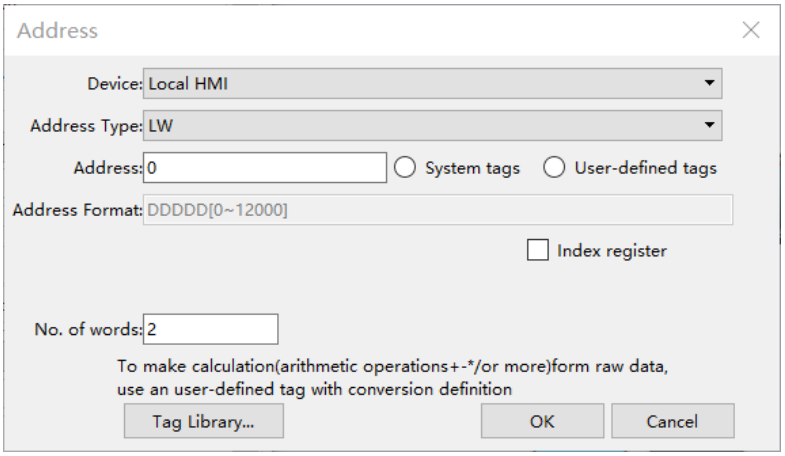
12.12.2 Configuration

Click [Object] » [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.

General Tab



Setting	Description
Enable Input Function	If selected, the input features and relevant settings are enabled.

Multi-line display	If selected, the ASCII object can display multi-lined text. If a line feed character LF (0xA) is used in the string, a newline will be created.
Vertical alignment	When [Multi-line display] is enabled, the method to vertically align multiple lines of text can be selected.
Password	If selected, any values entered will be masked by asterisks (*).
Unicode	Data in UNICODE format can be displayed. Otherwise, the system will display characters in ASCII format. This function can be used with the [ASCII/UNICODE] function key.
GB Code (high & low byte conversion)	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. 
Read address	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. The Unit is word.

Note

- A 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)

12.13 Indirect Window

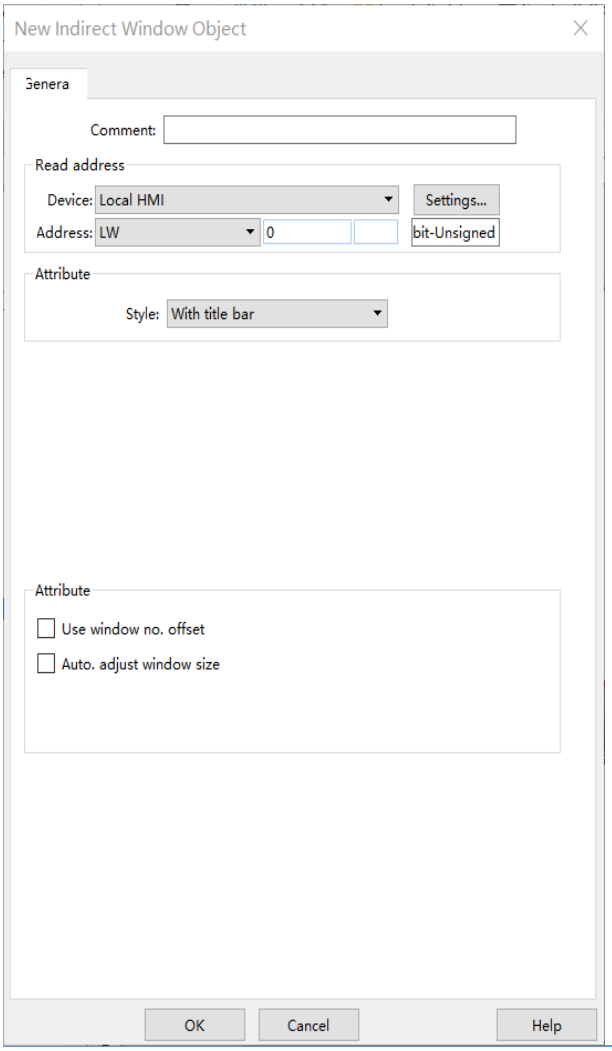
12.13.1 Overview

Indirect Window object opens or closes a 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.

12.13.2 Configuration

 Click [Object] » [Embed Window] » [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.

General Tab



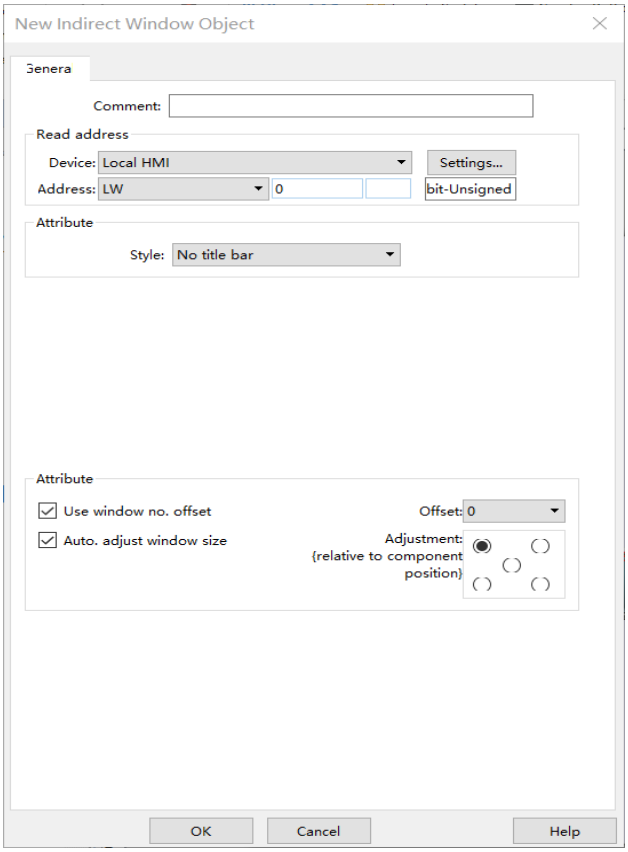
Setting	Description
Read address	Click [Setting] to select the [PLC name], [Address], [Device type], [System tag], [Index register] of the word device that controls the pop-up window. Users can also set address in [General] tab while adding a new object.
Attribute	**Style** Set the display style of the 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.

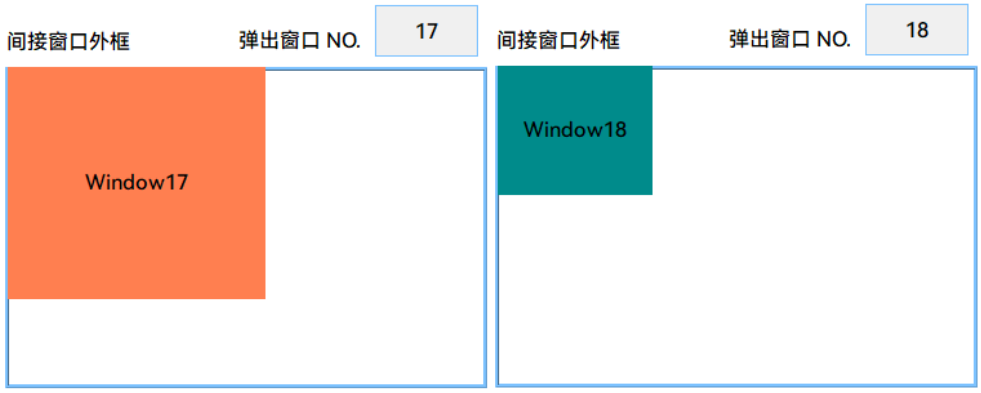
Auto. adjust window size	Automatically resizes the Indirect Window and align the pop-up window to the preset region. **Alignment** Sets a reference point of the pop-up window from one of the five positions on the screen; for example, if the lower-right region is selected, the lower-right corner of the 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

- A screen can display up to 24 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.

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.

12.14 Direct Window

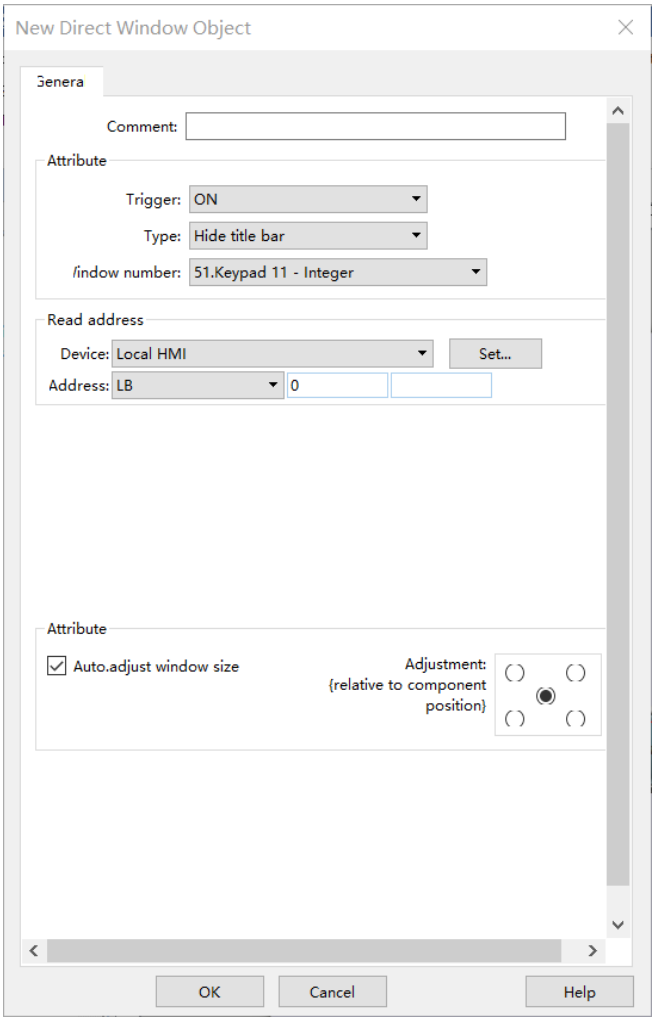
12.14.1 Overview

Direct Window object opens or closes a pop-up window assigned by a designated bit register. When the state of the bit register changes, the pop-up window appears at the predefined location. The display area for the pop-up window is limited by the size of predefined location. Returning the state of the bit register closes the pop-up window. 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.

12.14.2 Configuration

Click [Object] » [Embed Window] » [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.

General Tab



	Description
Read address	Click [Setting] to select the [PLC name], [Address], [Device type], [System tag], [Index register] of the bit 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. **Window no.** Set the pop-up window number
Auto. adjust window size	Automatically resizes the Direct Window and align the pop-up window to the preset region.

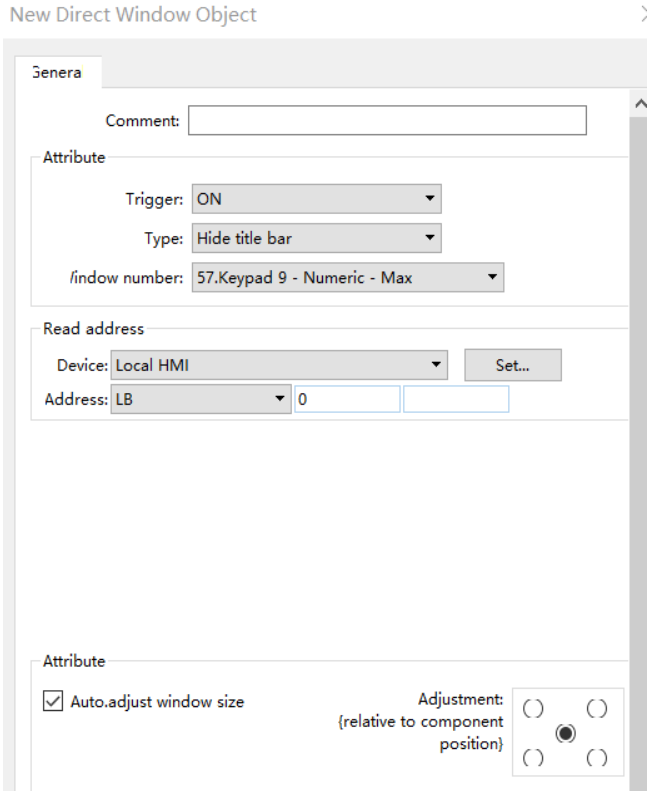
	Alignment Sets a reference point of the pop-up window from one of the five positions on the screen; for example, if the lower-right region is selected, the lower-right corner of the pop-up window is aligned to the lower-right region of the Direct Window. See Example 1.

Example 1

Create window no. 11 which can be controlled by a Toggle Switch with address LB-0.

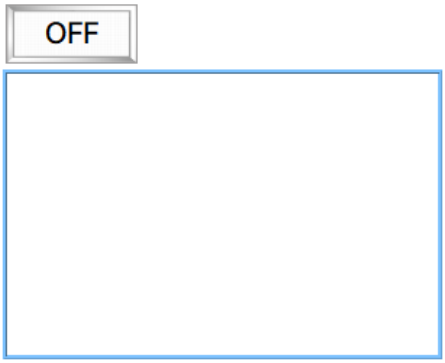
1. Create a Direct Window object and set read address to LB-0.

2. In this example, the reference point for alignment is set to the lower-right region.



3. When LB-0's state is ON, window no. 11 will show.

4. When LB-0's state is OFF, window no. 11 will be hidden.



Note

- A screen can display up to 24 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.

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.

12.15 Moving / Rotating Shape

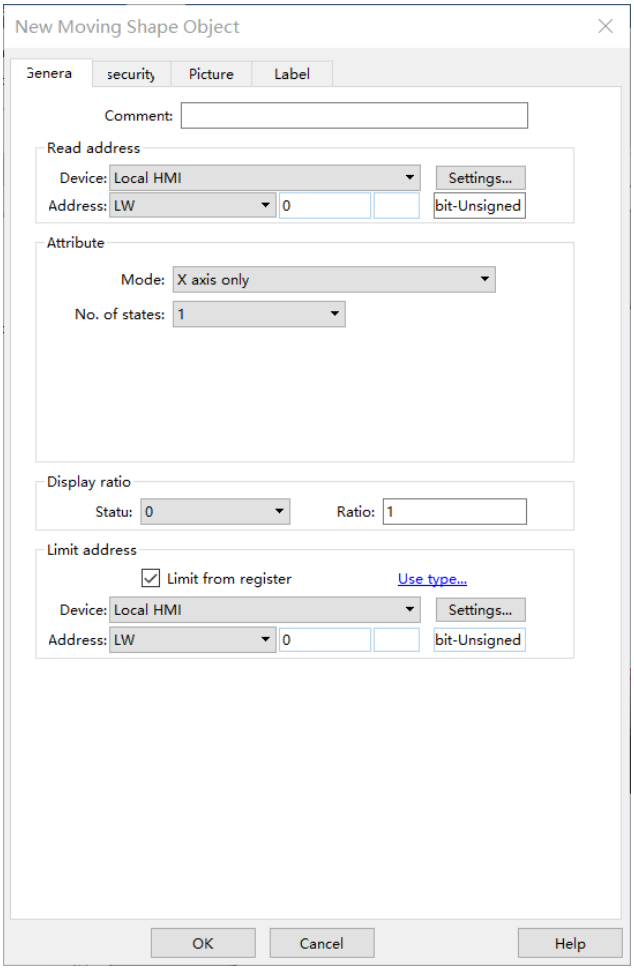
12.15.1 Overview

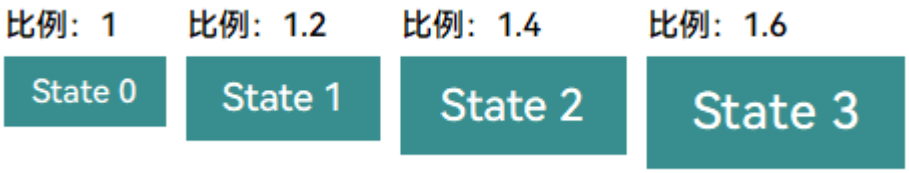
The [Move Graphic] component can define the state and moving distance of the component. The component will change the state and moving distance of the component according to the data in the read address and continuous registers.

12.15.2 Configuration

Click [Object] » [Animation] » [Moving / Rotating Shape] icon on the toolbar to create the object, set up the properties, press OK button, and a new Moving / Rotating Shape object will be created.

General Tab



Setting	Description
Read address	Click [Setting] to configure the [Device], [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 below "Illustration of Modes".
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 / rotating range.

Data format	16-bit	32-bit
Position - [Min. X] address	LW-n	LW-n
Position - [Max. X] address	LW-n+1	LW-n+2
Position - [Min. Y] address	LW-n+2	LW-n+4
Position - [Max. Y] address	LW-n+3	LW-n+6

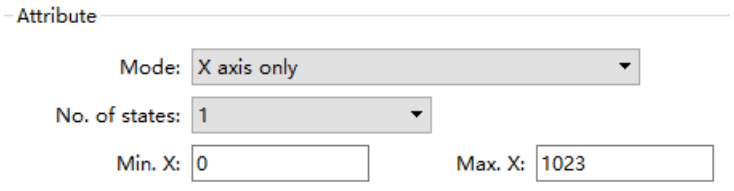
Illustration of Modes

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

(LW-n)

- X axis only

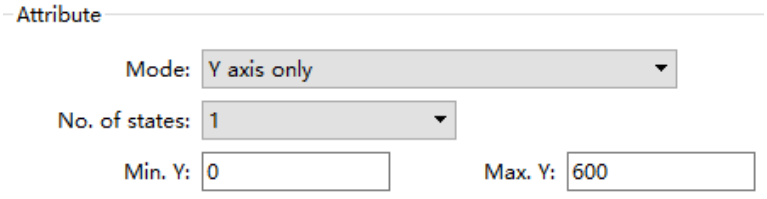
The object is only allowed to move along the X-axis. The moving distance ranges from [Min. X] to [Max. X].



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

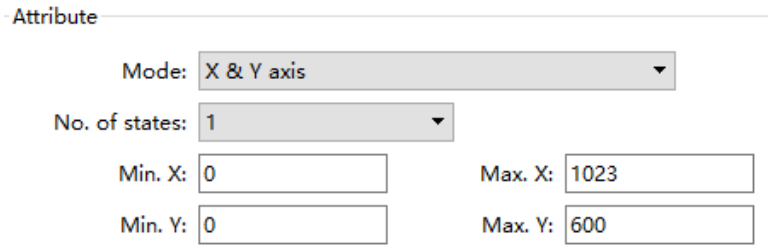
The object is only allowed to move along the Y-axis. The moving distance ranges from [Min. Y] to [Max. Y].



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

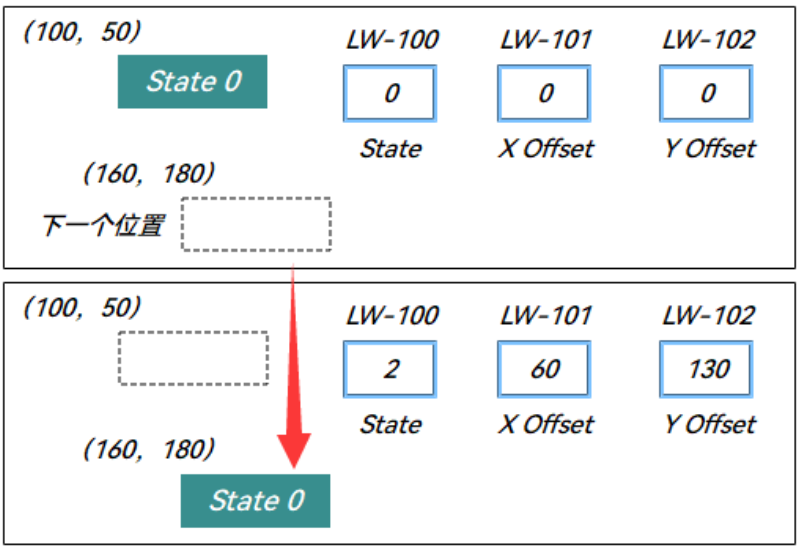
- X & Y axis

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 X	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.
Displacement=(Data-[Input low])× [Scaling high]-[Scaling low]/[Input high]-[Input low]

X axis move distance =

(DATA – [Input low]) x ([Scaling high] – [Scaling low]) / ([Input high] – [input low])

Data format	16-bit	32-bit
Object state	LW-n	LW-n
Moving distance on X-axis X	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.

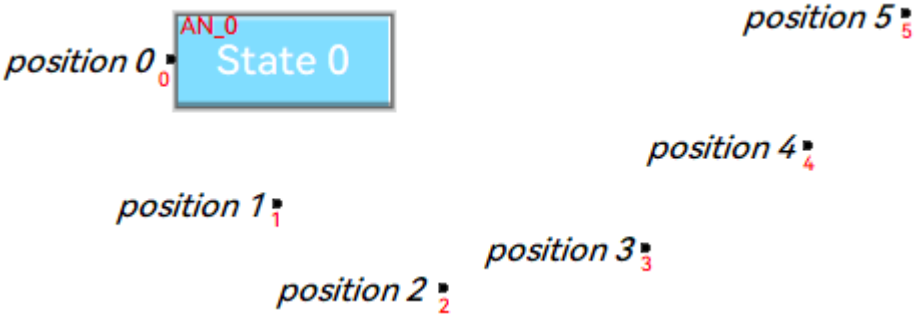
12.16 Animation

12.16.1 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.

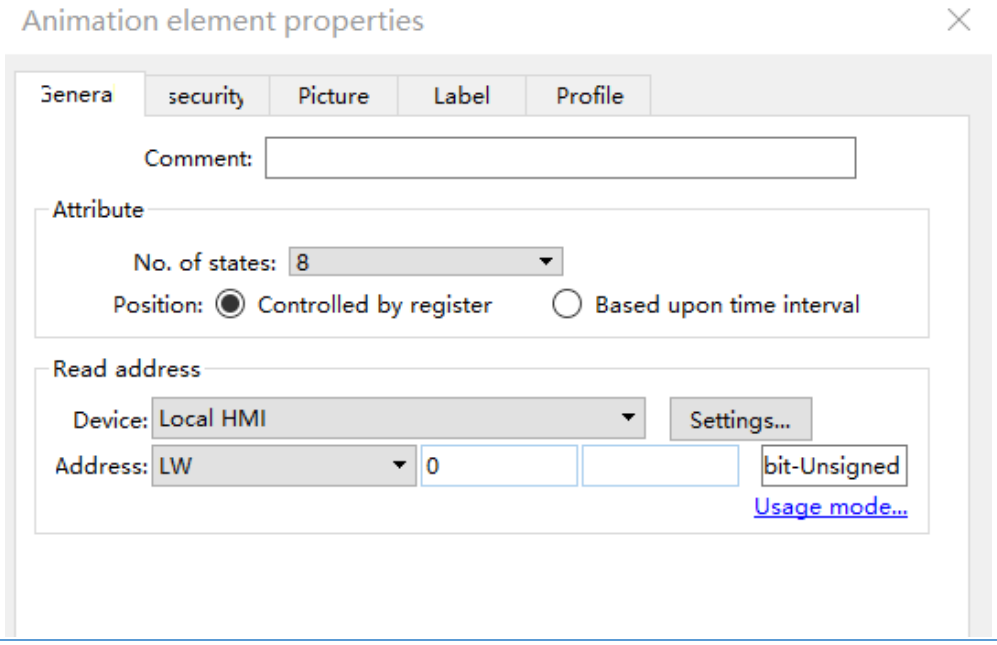
12.16.2 Configuration

Click [Object] » [Animation] » [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.



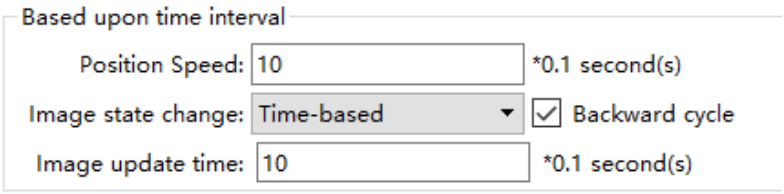
To change the object's attributes, double click on the object to open Animation Object's Properties dialog box.

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···

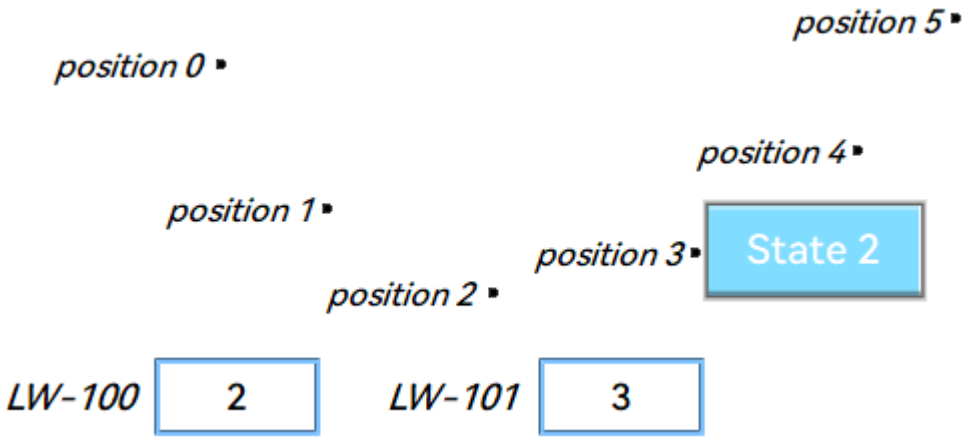


Example 1

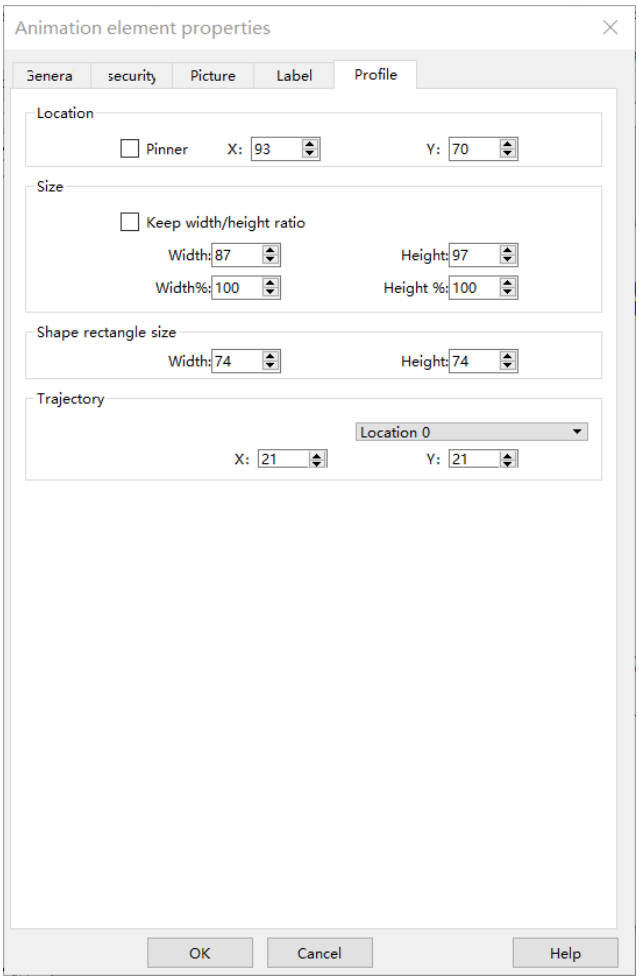
The object's state and position are determined by the registers, and the addresses must be configured correctly, as in the following table:

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.



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.

12.17 Flow Block

12.17.1 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. For cMT / cMT X Series models, drawing non-horizontal and non-vertical lines is possible.

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. For cMT / cMT X Series models, drawing non-horizontal and non-vertical lines is possible (angle limit: 5 degrees).

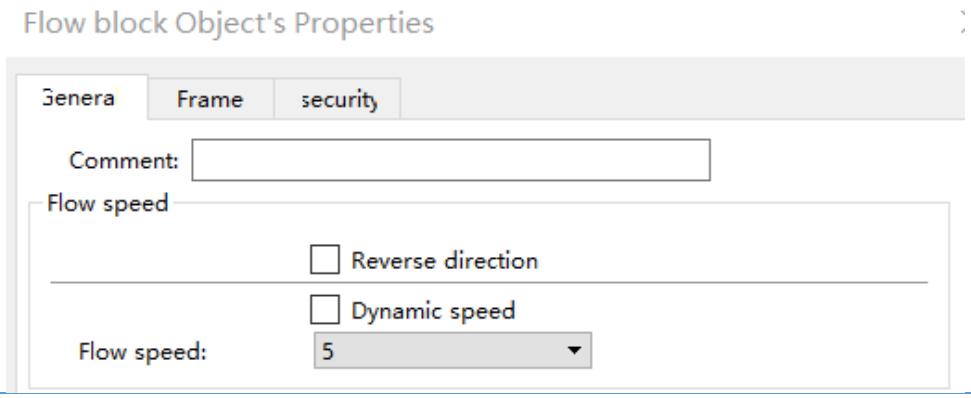
Dynamic speed and direction adjustment (Speed and direction can be controlled by a designated register.

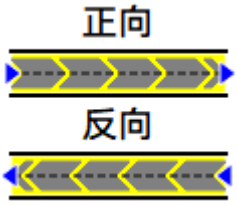
Security mechanism (Interlock), which hides Flow Block when the status of designated bit is invalid.

12.17.2 Configuration

 Click on the Flow Block icon on the toolbar or select [Objects] » [Animation] » [Flow Block] to create object.

General Tab

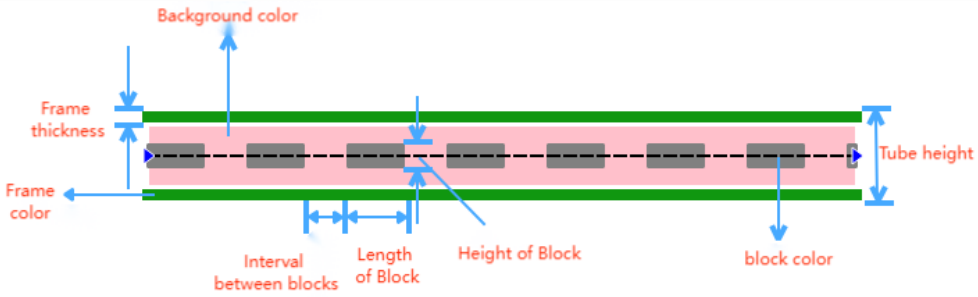


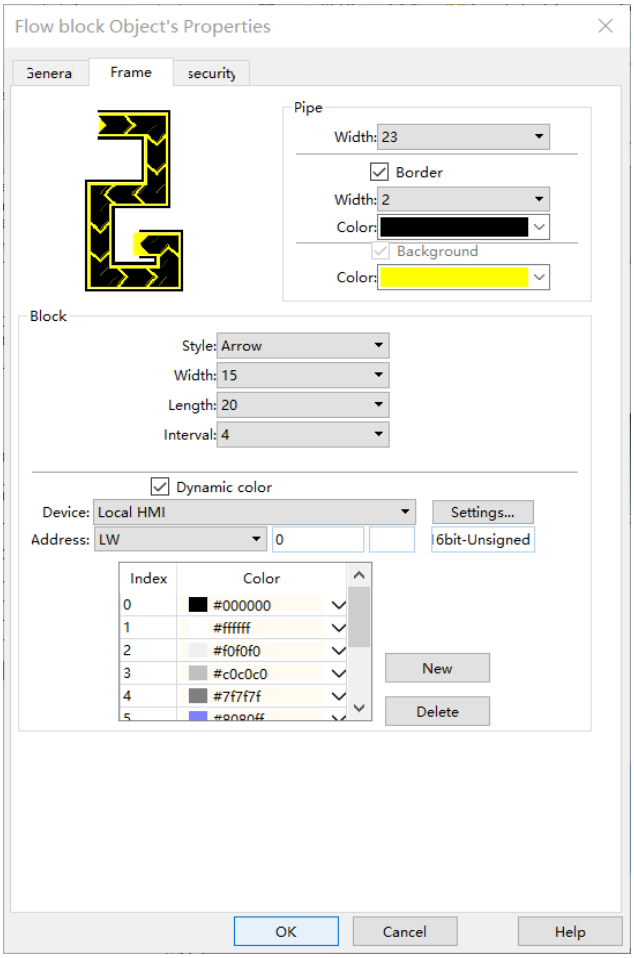
Setting	Description
Reverse direction	The blocks flow in the direction the object is drawn (the blue arrow). If select this check box, the blocks flow in the opposite direction. 
Dynamic speed	**Read address** The direction and speed at which the blocks flow can be controlled by a designated register. 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.

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. Style, width, length, interval and color can be set.
Dynamic color	Allows dynamic color change with 256 customized colors, numbered from 0 to 255 allowed. The color is selected by entering a value in the designated

	register. Entering a value greater than the largest color number changes the Flow Block to the last color in the list.

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.

12.18 Bar Graph

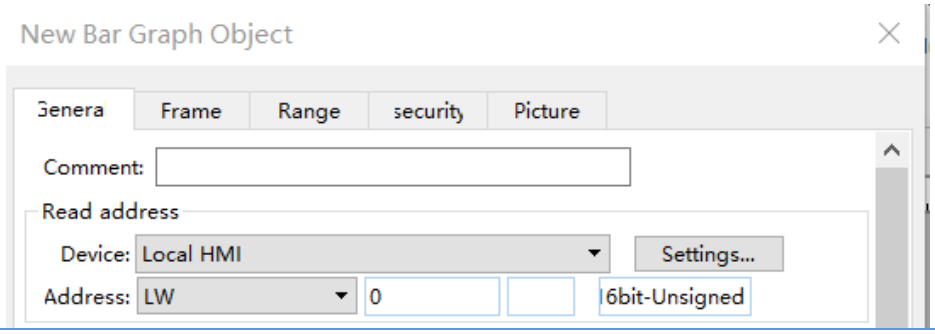
12.18.1 Overview

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

12.18.2

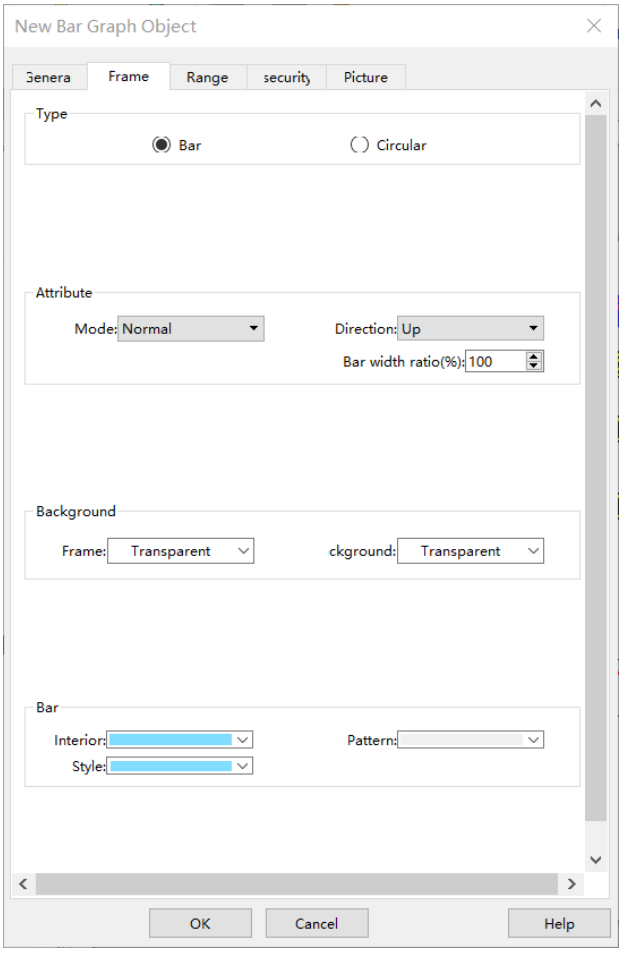
Click [Object] » [Chart] » [Bar Graph] icon on the toolbar to open Bar Graph dialog box. Select properties, click OK button, a new Bar Graph object is created.

General Tab

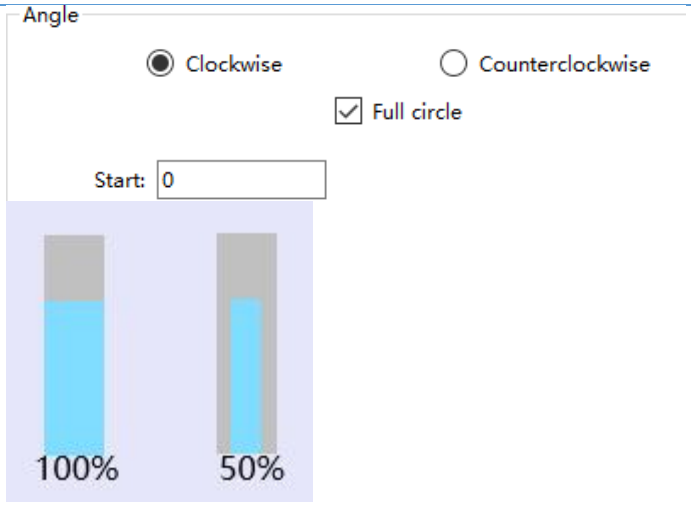


Setting	Description
Read address	Click [Setting] to Select the [Device], [Device type], [Address], [System tag], and [Index register] of the word devices that controls how the bar graph displays.

Outline Tab

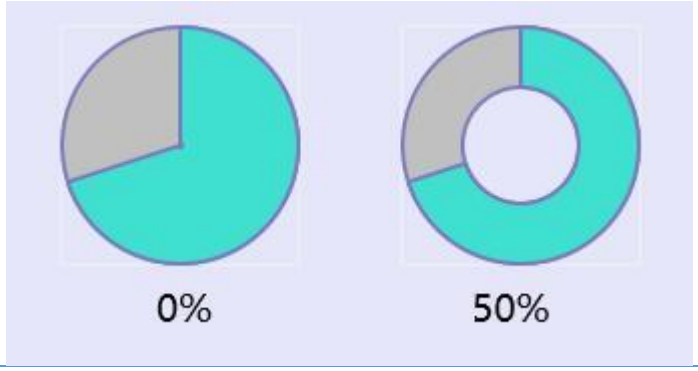


Setting	Description
Type	Choose either [Bar] or [Circular].
Attribute	**Mode** Choose either [Normal] or [Offset]. If [Offset] is selected, an original value [Origin] must be entered for reference. **Direction / Degree** Bar: Determine the bar graph direction. Available options are [Up], [Down], [Right], and [Left]. Circular: Determine the circular bar graph direction. Available options are [Clockwise] and [Counter clockwise]. If [Full circle] is selected, set the start degree. If [Full circle] is not selected, set the start and end degree. **Bar width ratio (%)** The ratio of bar to object width. The figure below shows two ratios, 100% and 50%.



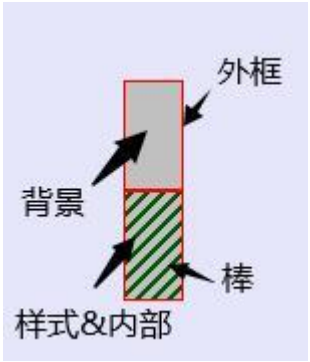
Hole Radius (%)

The ratio of the radius of the hole to the radius of the whole circular bar graph. The figure below shows two ratios, 0% and 50%.

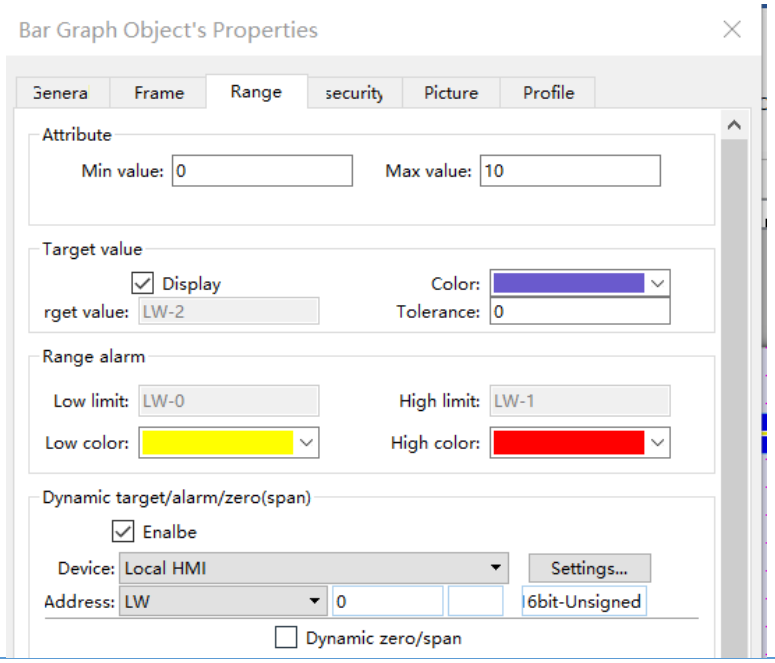


Color/Style

Set the bar's frame and background color, bar style, and bar color. See the picture below.



Range Tab



Setting	Description
Min/Max	The percentage of filling can be calculated by the formula, see Example 1.
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]. Please note that when [Dynamic color] is enabled, the Target indicator and Alarm indicator settings will not be present and their colors are determined by the designated addresses.
Dynamic taget/alarm	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:

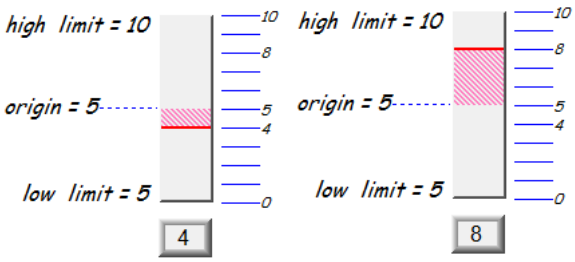
Percentage of filling = (Register value − [Min Value])/([Max value] − [Min value]) × 100%

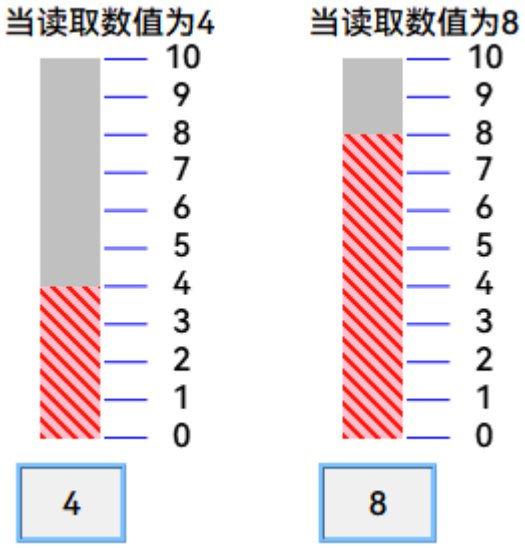
$$\text{显示区域百分比} = \frac{\text{寄存器值} - \text{最小值}}{\text{最大值} - \text{最小值}} \times 100\%$$

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

For example, [Origin] is 5, [Max. Value] is 10, and [Min. Value] is 0. For different value in read address, it will display as below:

If the value at read address is 4 & 8 in respectively:



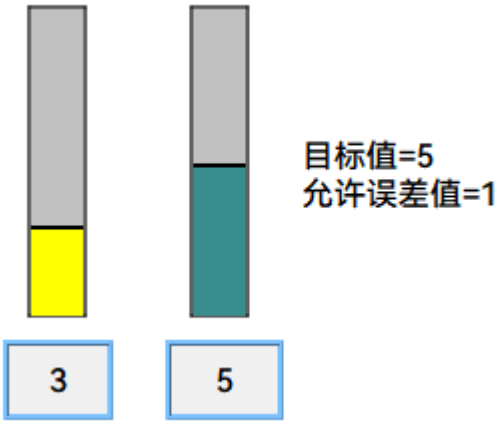


Example 2

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, [Min value], [Max value] 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
Min value	LW-n+3	LW-n+6
Max value	LW-n+4	LW-n+8

12.19 Meter Display

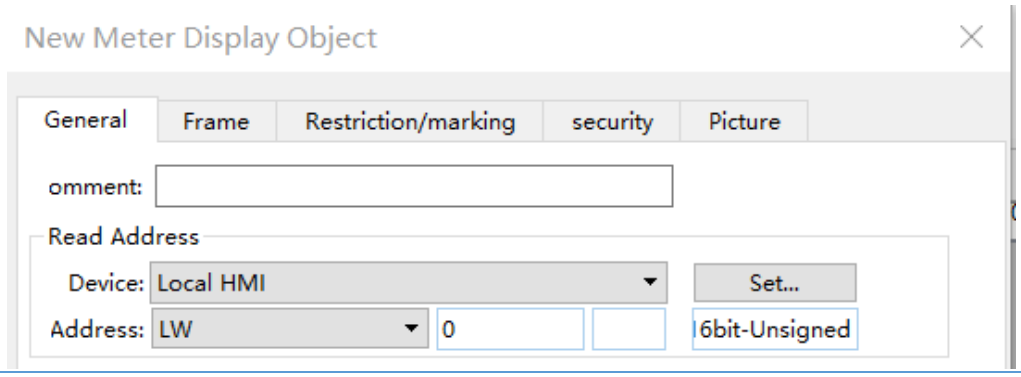
12.19.1 Overview

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

12.19.2 Configuration

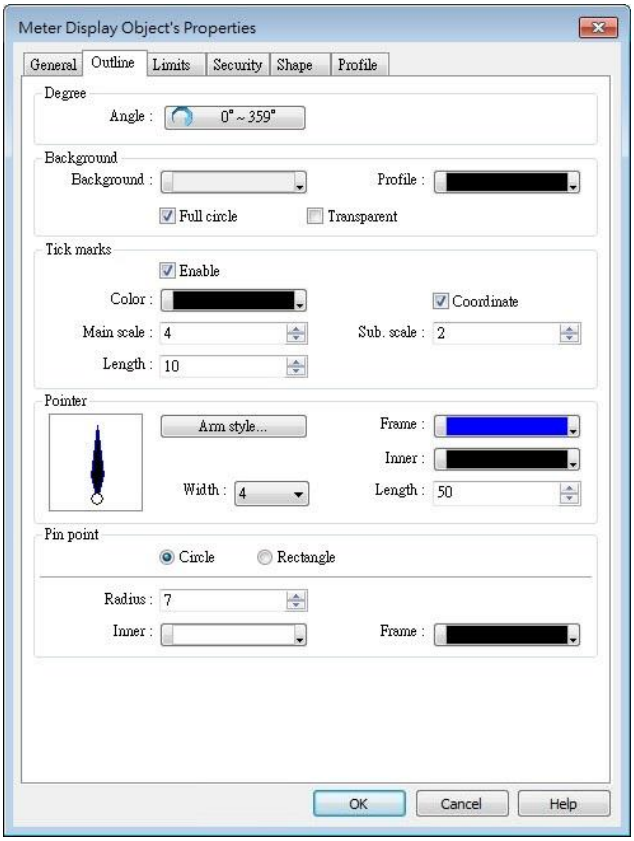
 Click [Object] » [Chart] » [Meter Display] icon on the toolbar to open the Meter Display dialog box. Set the object's attributes and then click OK to create a new Meter Display object.

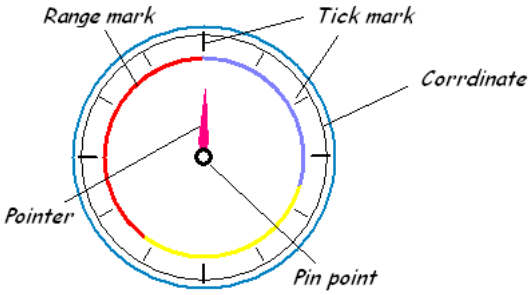
General Tab

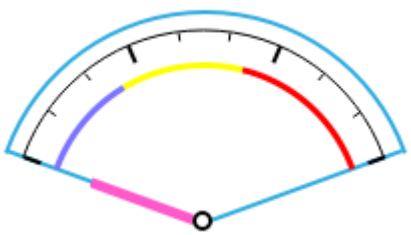
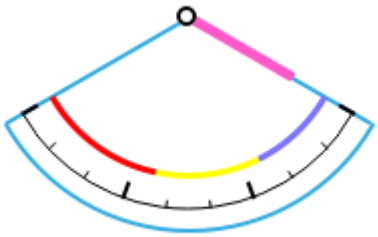


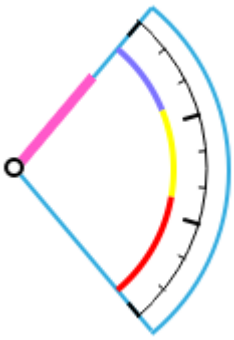
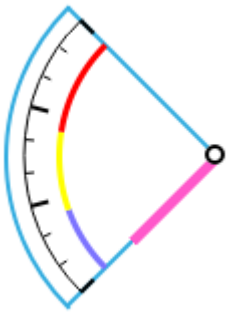
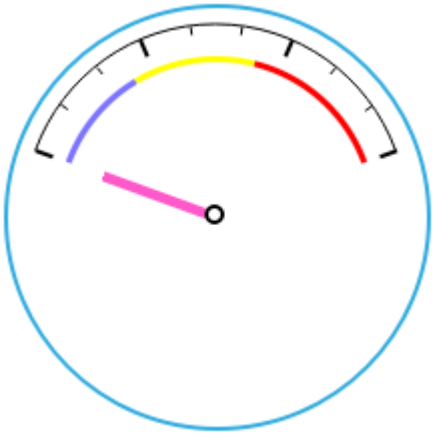
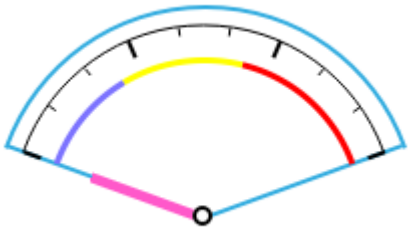
Setting	Description
Read address	Click [Setting] to select the [Device], [Device type], [Address], [System tag], and [Index register] of the word devices that controls the Meter Display object.

Outline Tab



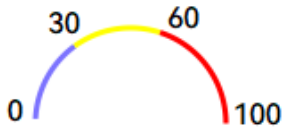


Setting	Description
Degree	Set the pointer to go around the meter clockwise or counterclockwise. Sets the object's start degree and end degree measured clockwise from the 12 o'clock position. The angle range is 0 to 360 degrees. The following shows meters of different settings. ● [Start degree] = 290∘ [End degree] = 70∘  [Start degree] = 120∘ [End degree] = 240∘ 

	40∘ 140∘ ● [Start degree] = 40° [End degree] = 140°  [Start degree] = 225° [End degree] = 315° 
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.  使用全圆  非使用全圆 **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.

Limits Tab /

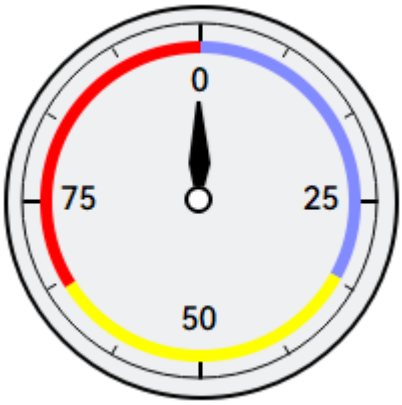
Setting	Description
Value	Sets the object's display range. See Example 1.
	Configures the values of [Low limit], [High limit], their corresponding display colors, and the width.

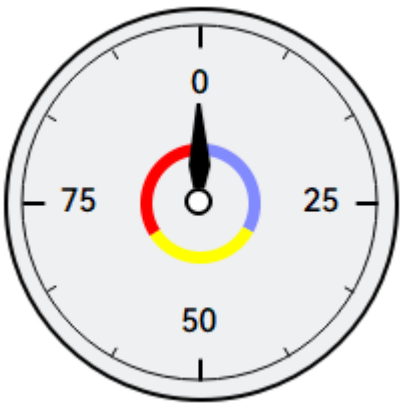


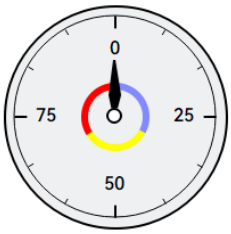
Use user-defined radius

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.
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.

$$\text{角度(度)} = \frac{\text{读取数值 - 最小值}}{\text{最大值 - 最小值}} \times (\text{结束角度 - 起始角度})$$

$$\text{角度(度)} = \frac{30 - 0}{100 - 0} \times (360 - 0) = 180(\text{度})$$

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

12.20 Pie Chart

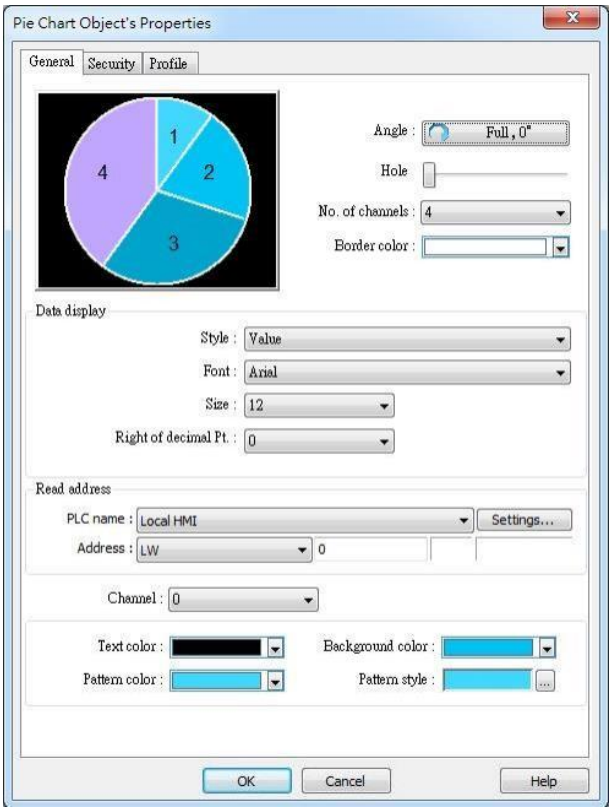
12.20.1 Overview

The Pie Chart object draws a pie chart that is divided into slices to illustrate numerical proportion, according to the value of the designated read address.

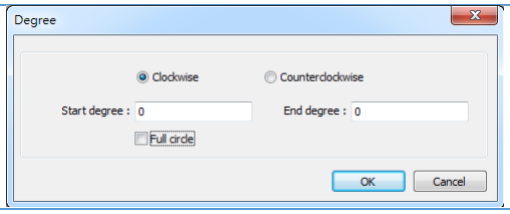
12.20.2 Configuration

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

General Tab



Setting	Description
Angle	Set the [Start degree] of the chart. Choose the Chart to be [Clockwise] or [Counter clockwise]. If [Full circle] isn't selected, then [End degree] must be set.

	
Hole	Set the size of the hollow circle in the center of Pie Chart. 
No. of channels	Set the number of channels to be displayed in the chart. The range is from 2 to 16.
Border color	Set the color of the border.
Data display	The [Style] can be set as [None], [Value] and [Percentage]. The [Font] and [Size] are for the text on the chart. For Value Style, the [Right of decimal Pt.] can set the value on the chart to be displayed with the decimal point. The [Right of decimal Pt.] option is only available for [Value] style.
Read Address	The address is for channel 1. The following consecutive addresses are for the rest of the channels. For example, if the Read Address is LW-0, then the Read Address for channel 2 is LW-1; channel 3 is LW-2…and so on.
Channel	Set the [Text color], [Background color], [Pattern color], and [Pattern style] of the selected channel. The [Background color] is for the [Pattern style] that has background. If the [Pattern style] doesn't have a background, then the [Background color] doesn't need to be set.

12.21 Dynamic Scale

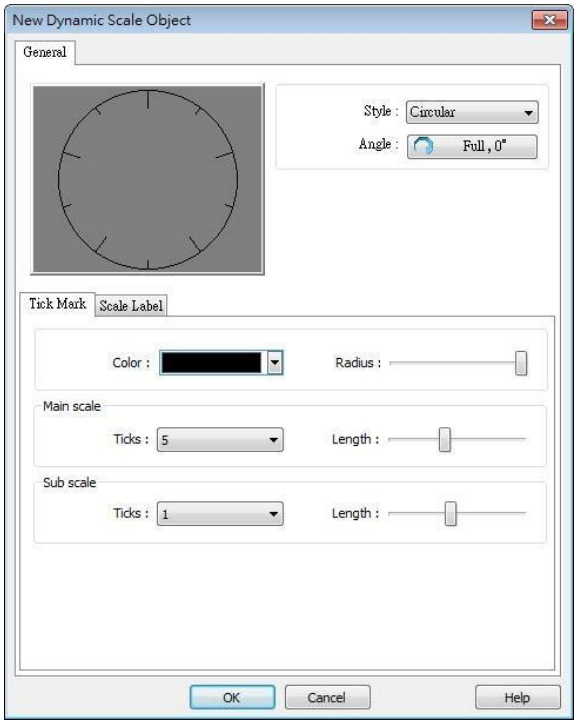
12.21.1 Overview

Dynamic Scale offers customizable tick marks and scale labels and can be used together with objects such as Trend Display, Bar Graph…, etc.

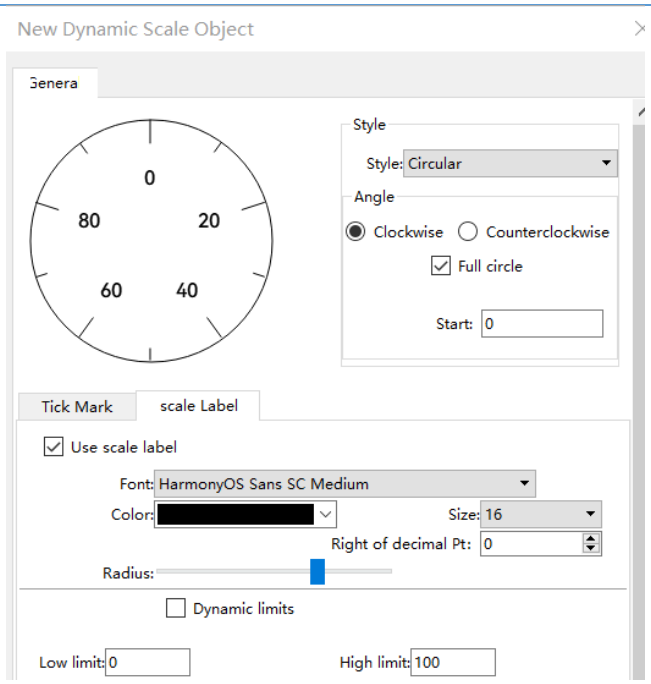
12.21.2 Configuration

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

General Tab



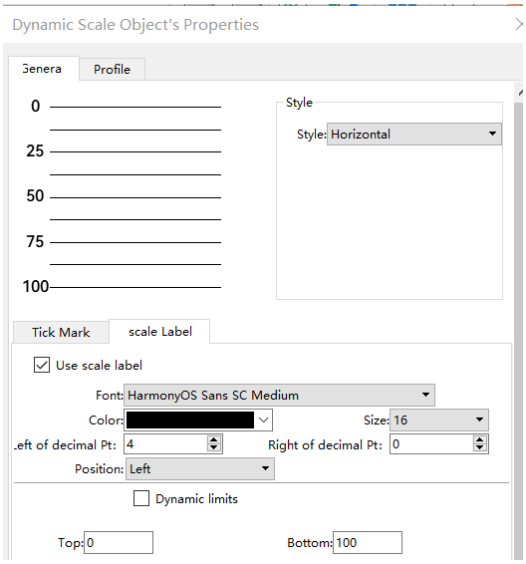
Setting	Description
Style	Select the style from [Circular], [Horizontal], or [Vertical]. If select [Circular], set the [Direction] and [Degree]. For **Direction,** Select from [Clockwise] or [Counterclockwise]. For **Degree,** If [Full circle] is selected, set the start degree. If [Full circle] is not selected, set the start and end degree. ☐ Full circle Start: 0 Sweep range: 360
Tick Mark	Select the color of the tick mark, and set the number of ticks for the main and sub scale (major and minor tick mark). For [Circular] style, the length of the major and minor tick mark, and the radius of the circular tick mark can be set.
Scale Label	Displays major tick labels. **Circular**



Set the font, font color, font size, and decimal point of the scale label.

Set the radius start from the center of the object to the position to place the scale label.

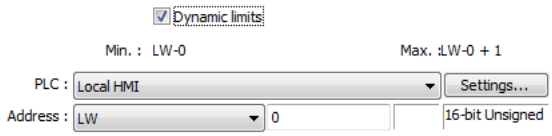
Virtical / Horizontal



Set the font, font color, font size, and decimal point of the scale label.

Set the position to display the scale label.

The Max. and Min. limits of the scale label can be specified. If [Dynamic limits] is selected, the Max. and Min. limits can be set by the designated word addresses.



12.22 Dynamic Drawing

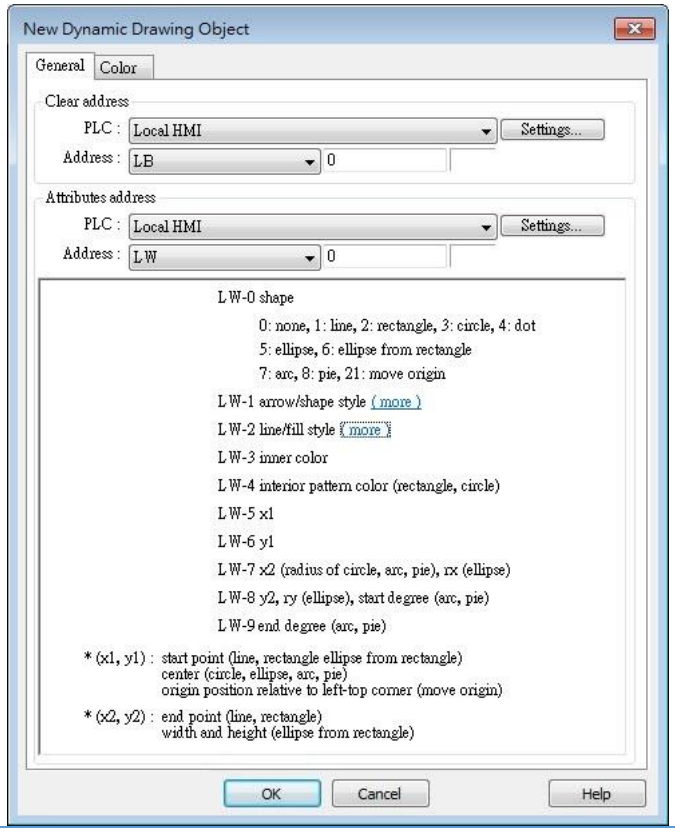
12.22.1 Overview

Dynamic Drawing object enables drawing a shape in a specified region on HMI screen at run time. The shape can be a line, a rectangle, a circle, or a dot. By setting the Attributes Addresses, the style and the color of the shape can be customized.

12.22.2 Configuration


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

General Tab



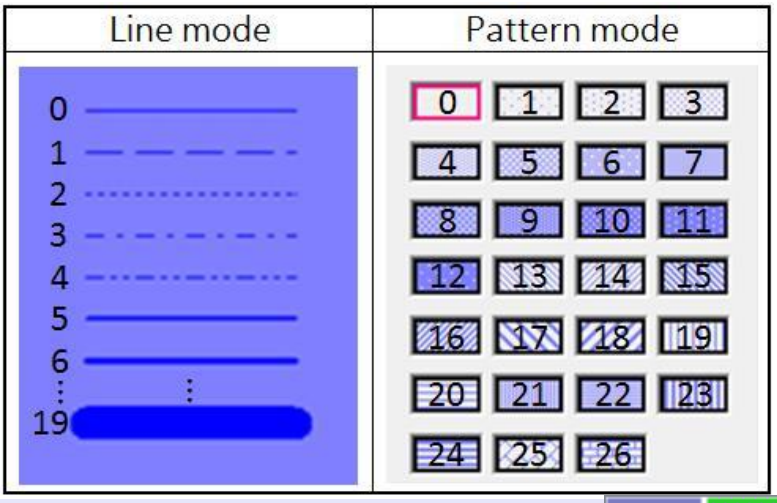
Setting	Description
Clear Address	Clears the drawing.
Attributes Address	Changes the attributes in the drawing.

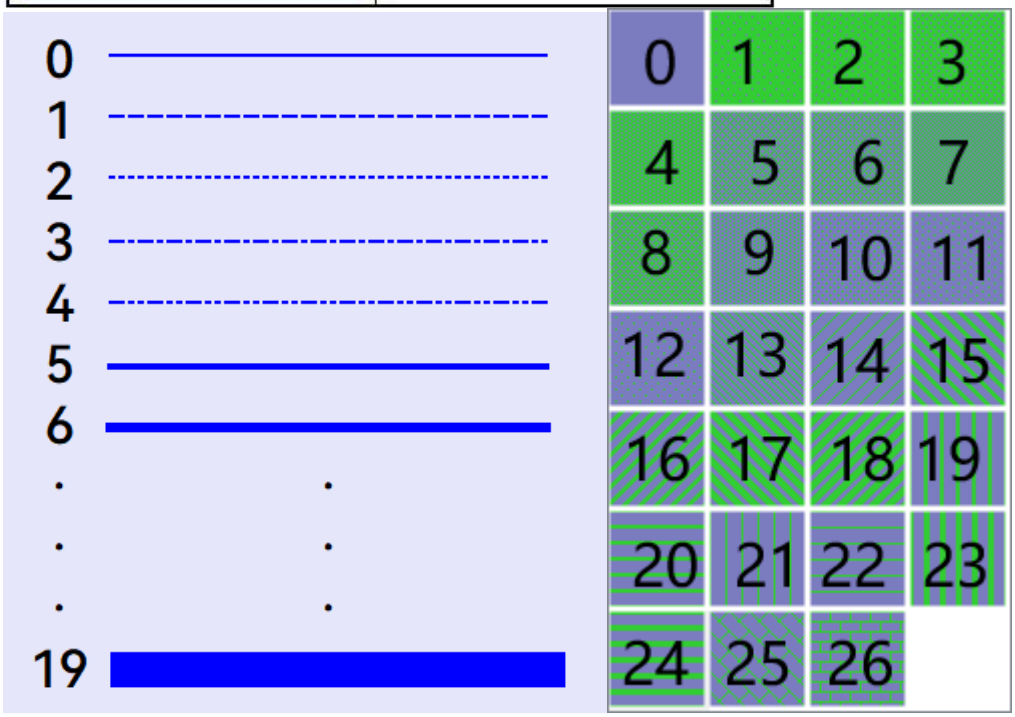
Entering different values in different Attributes Addresses brings different effects, as shown in the following table.

Attributes Address	Attributes Address+0	Attributes Address+1		Attributes Address+2	Attributes Address+3	Attributes Address+4
Default	0	Ones	Tens		Customizable	Customizable
Line	1	0: Non-arrow	0: Small	0: Solid line	Line color	
		1: Single-ended arrow (Hollow)	1: Large	1: Dashed line		
		2: Double-ended arrow (Hollow)		2: Dotted line		
		3: Single-ended arrow (Solid)		3: Dash Dot line		
		4: Double-ended arrow (Solid)		4: Dash Dot Dot line		
				5 and up: Solid line with thicknesses greater than 2		
Rectangle	2	0: Hollow		Drawn in Line mode	Rectangle color	Interior pattern color
		1: Solid		Drawn in Pattern mode		
Circle	3	0: Hollow		Drawn in Line mode	Circle color	Interior pattern color
		1: Solid		Drawn in Pattern mode		
Dot	4				Dot color	
Ellipse	5	0: Hollow		Drawn in Line mode	Ellipse color	Interior pattern color
		1: Solid		Drawn in Pattern mode		
Ellipse from Rectangle	6	0: Hollow		Drawn in Line mode	Ellipse color	Interior pattern color
		1: Solid		Drawn in Pattern mode		
Arc	7			Drawn in Line mode	Arc color	
Pie	8	0: Hollow		Drawn in Line mode	Pie color	Interior pattern color
		1: Solid		Drawn in Pattern mode		
Move Origin	21					

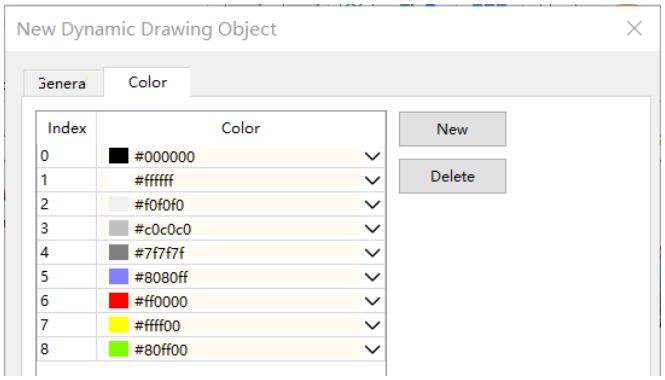
Attributes Address	Attributes Address+0	Attributes Address+5	Attributes Address+6	Attributes Address+7	Attributes Address+8	Attributes Address+9
Default	0					
Line	1	Start point X	Start point Y	End point X	End point Y	
Rectangle	2	Left-top point X	Left-top point Y	Right-bottom point X	Right-bottom point Y	
Circle	3	Center point X	Center point Y	Radius		
Dot	4	Dot X	Dot Y			
Ellipse	5	Center point X	Center point Y	Radius on the X axis	Radius on the Y axis	
Ellipse from Rectangle	6	Left-top point X	Left-top point Y	Width	Height	
Arc	7	Center point X	Center point Y	Radius	Start degree	End degree
Pie	8	Center point X	Center point Y	Radius	Start degree	End degree
Move Origin	21	New origin X	New origin Y			

The values in [Attributes Address+2] represent different Line or Pattern styles, as shown in the following table.





Color Tab



Setting	Description

New	Adds a color to be used in the drawing.
Delete	Deletes the selected color.

Example 1

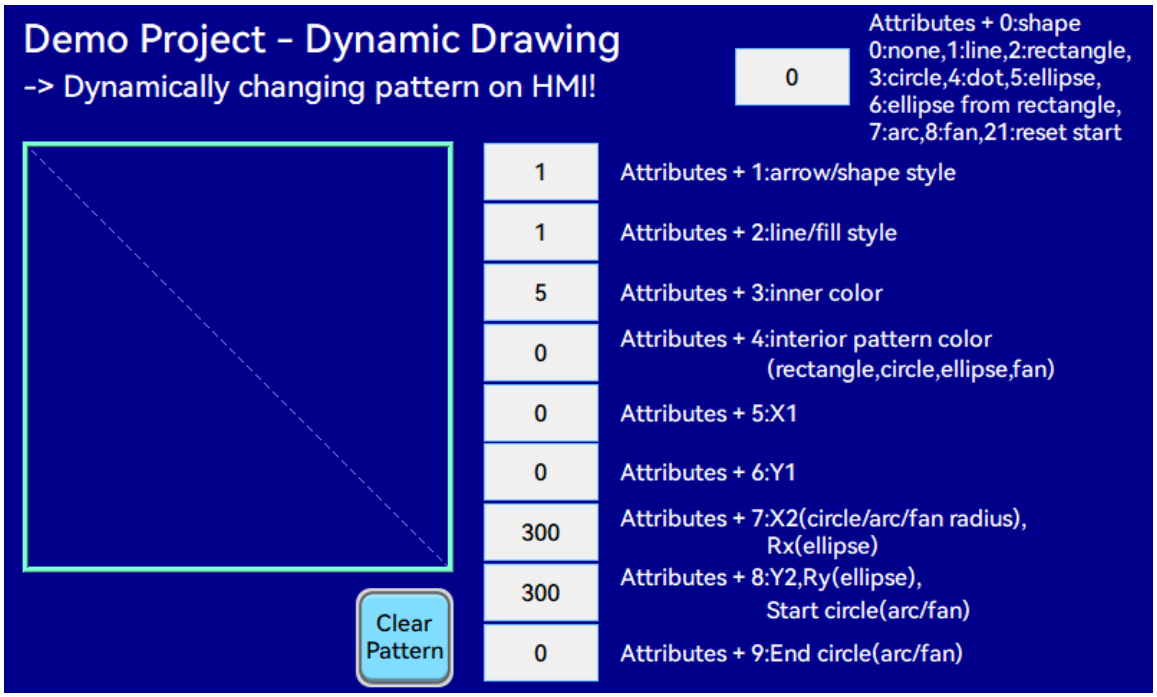
The following demo project demonstrates how to dynamically draw an arrow or a circle on HMI screen. To build the project, follow these steps:

1. Create a Dynamic Drawing object, set Clear Address to LB-0, and Attributes Address to LW-0.

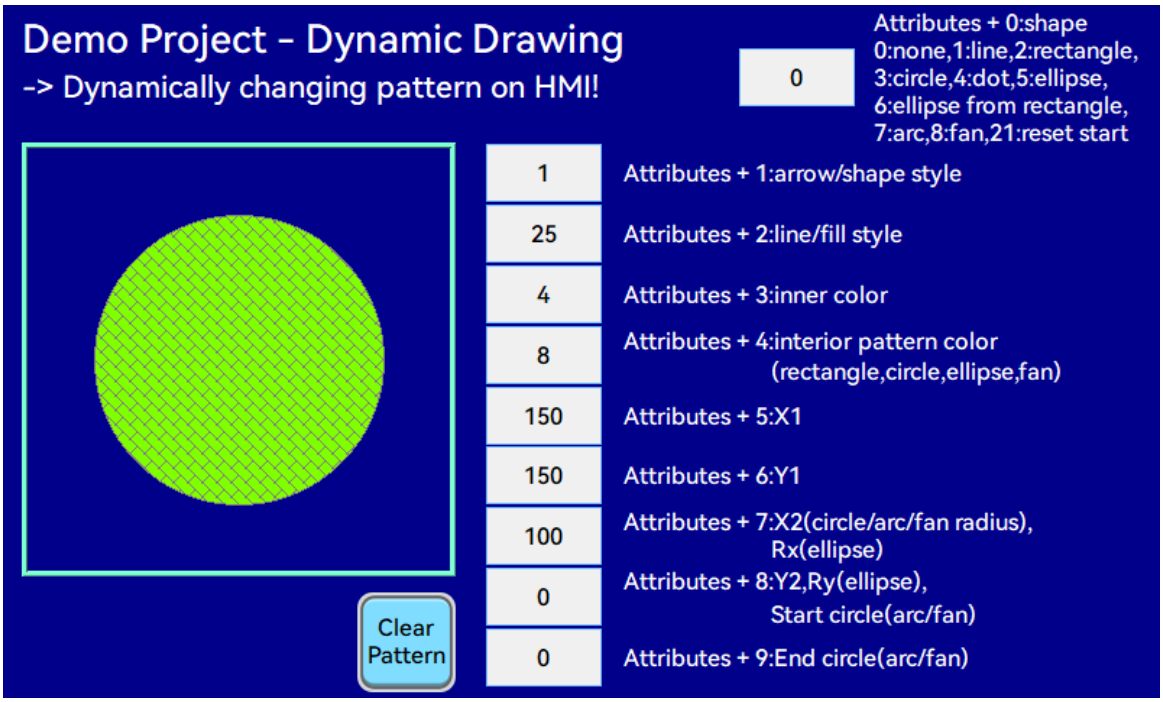
2. Create a Toggle Switch object, set address to LB-0, and select Toggle as switch style, for clearing the drawing.

3. Create 10 Numeric objects, set addresses to LW-0~LW-9, for specifying the attributes in the drawing.

4. Run simulation or download the project to HMI to see the result. By entering 1 in LW-0, a line is drawn, and entering different values in LW-1~9 can change the style, the color, and the position of the line.



1. Press Clear Pattern button, and enter 3 in LW-0, a circle is drawn, and entering different values in LW-1~9 can change the style, the color, and the position of the circle.



Note

Before using Attributes Address, please define [Attributes Address + 1] ~ [Attributes Address + 9]. The system will reset the Attributes Address, after it is used.

If the drawing is not cleared, the new drawing will overlap the previous one, and the maximum acceptable number of drawings in a Dynamic Drawing object is 1000.

The maximum number of line styles is 19, which means the maximum thickness of a solid line is 16. The style numbers that exceed 19 will be displayed as 19.

Find the color number in Color tab.

The range of the start and end degree for Arc and Pie is 0 to 360 degrees.

Origin position is relative to left-top corner (0,0). Giving “move origin” command will make (x1, y1) the new origin, and x1, y1 will keep on accumulating until being reset to (0,0) by clearing the drawing.

12.23 Data group display (not yet enabled)

12.24 XY Plot

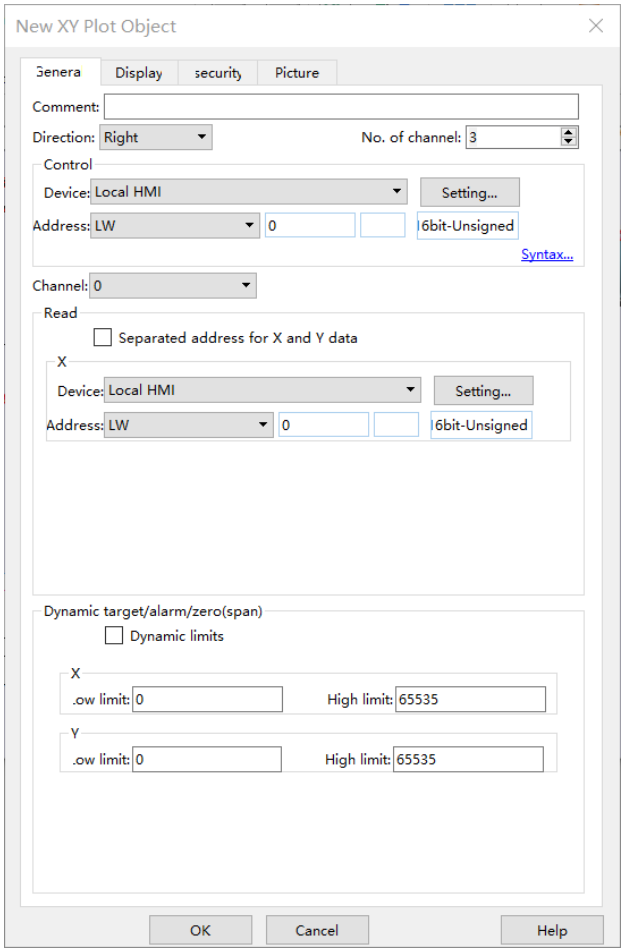
12.24.1 Overview

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 16 channels can be displayed simultaneously. This object facilitates data observation and analysis. Additionally, negative numbers can be displayed as well.

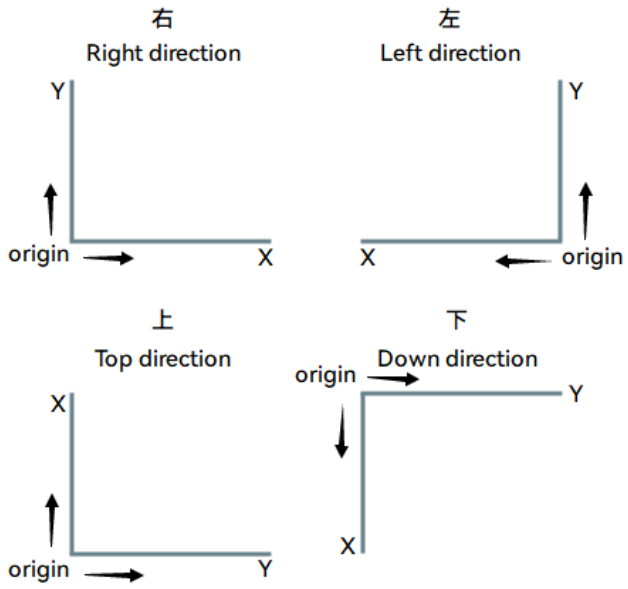
12.24.2 Configuration

 Click [Object] » [Chart] » [XY Plot] icon on the toolbar to open a [XY Plot] object property dialog box.

General Tab



Setting	Description
Directionv	There are four selections, right, left, up or down, as shown as below,



No. of channels	Set the number of channels for observation.
Control address	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 l controls the number of data points plotted. After operation, the [Control address] will be reset to 0.

Control address	Value	Result
LW-n	1	Plots point on XY curve. (The plotted points are kept.)
	2	Clears all XY curves.
	3	Clears then plots new XY curve.
LW-n+1	Any number	Controls the number of data points plotted.

No. of data address

Controls the number of data points. Each channel can plot up to 1023 points.

Channel	Select a channel to configure.
Read Address	**PLC Name** Select a PLC which will be the source of [X data] and [Y data] and designate a read address.
Dynamic limits	When **not** selected (See Example 1) 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 2) A zoom effect can be created by changing the Low / High Limits.

Example 1

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.

Scale (%) = (Read Address Value − Low Limit)/(High Limit − Low Lmit)

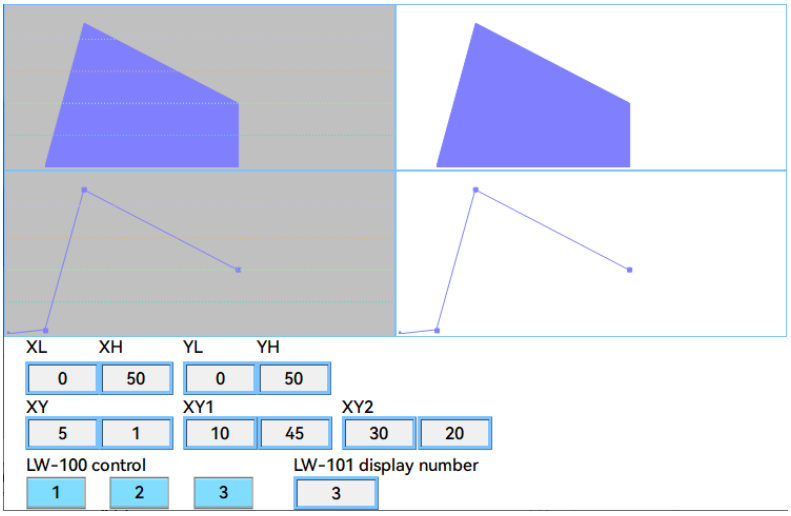
$$\text{刻度百分比(\%)} = \frac{\text{寄存器值 - 下限}}{\text{上限 - 下限}}$$

Example 2

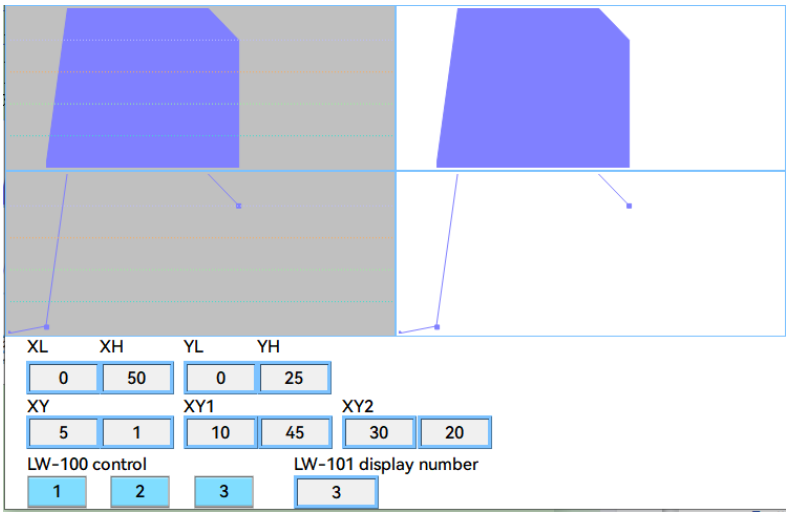
If **[Dynamic limits]** is selected, assume the address is LW-n, the corresponding limits are retrieved from the addresses as shown in the following table.

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-m	LW-m
Y axis high limit	LW-m+1	LW-m+2

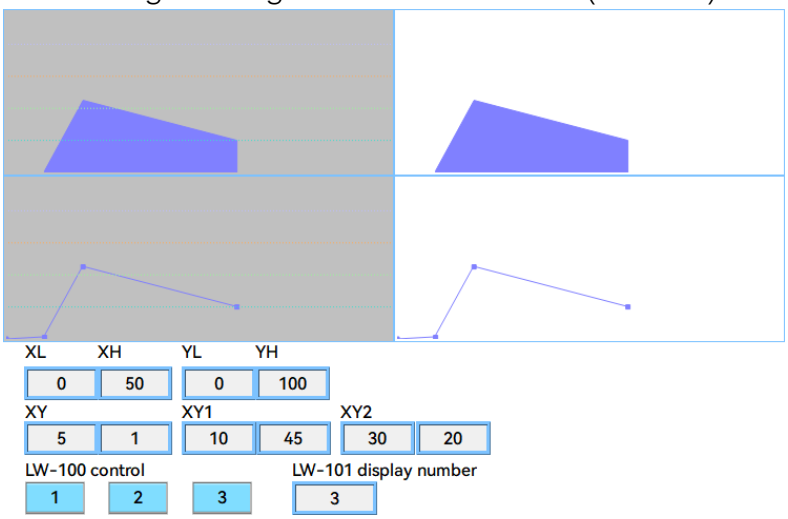
If **[Dynamic limits]** is selected, a zoom effect can be created by changing the setting of Low/ High Limits. In the following example, XL=X low limit, XH=X high limit, YL=Y low limit, YH=Y high limit, and XY, XY1, XY2 are three XY data. When changing the high limits of X and Y axis, the result is shown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



Original

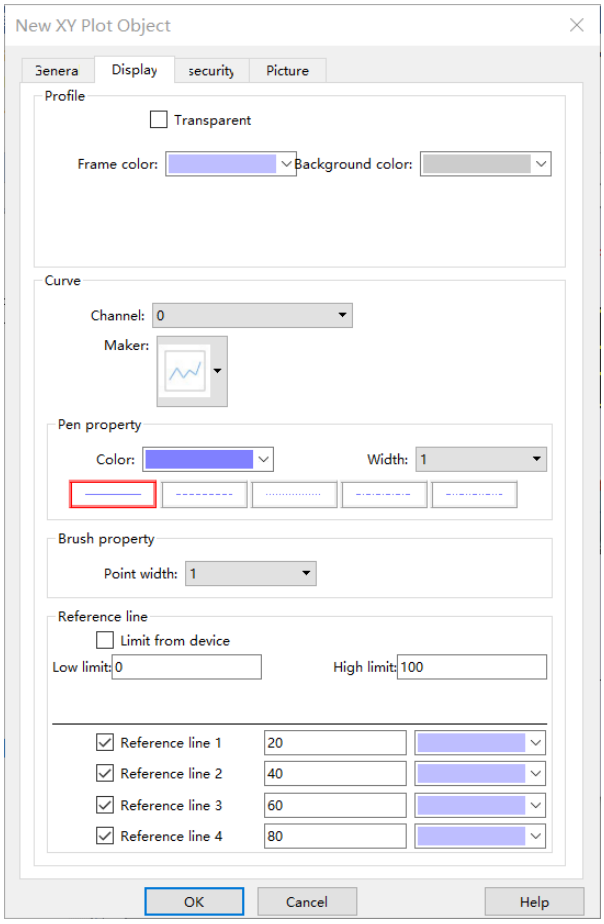


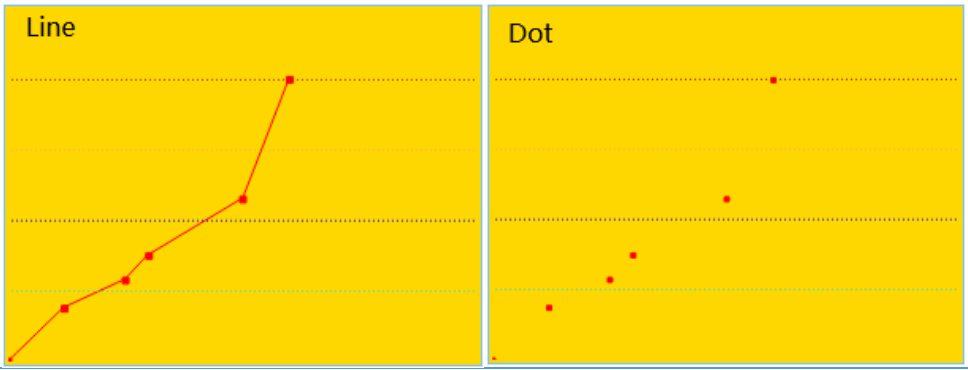
Change the high limit of Y axis to 25. (zoom in)

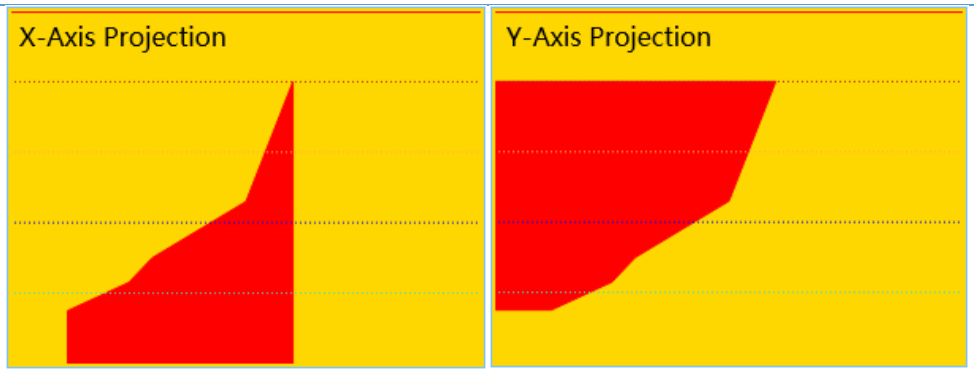


Change the high limit of Y axis to 100 (zoom out)

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	Select the style of XY Plot. there are four different types of XY plot as shown below: 

	 See Example 1 as below.
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.

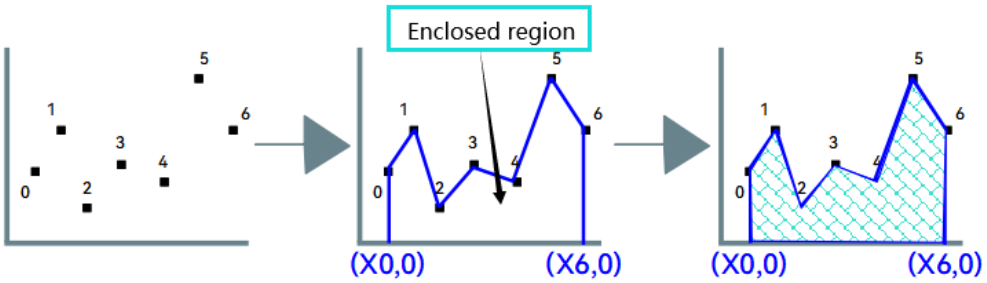
Example 1

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 – (X_0, 0) and (X_6, 0).

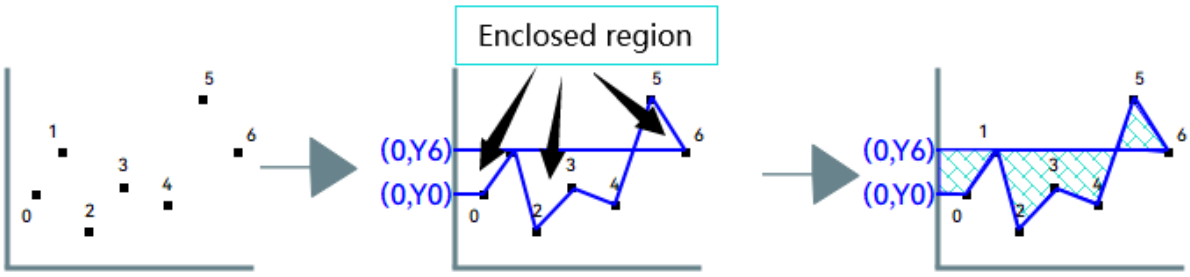
2. Link all the points in the order of (X_0, 0), P0⋯ P6, (X_6, 0) and returns to (X_0, 0) at last.

3. Fill out all enclosed areas.



Similarly for Y-axis projection:

[Y] :



Note

Without clearing the drawn points, XY Plot can be drawn a number of times by entering value 1 in the control address repeatedly. The way to calculate the maximum number of times is: 32 divided by the number of channels.

1 channel → 32 times

2 channels → 16 times

16 channels → 2 times

12.25 Trend Display

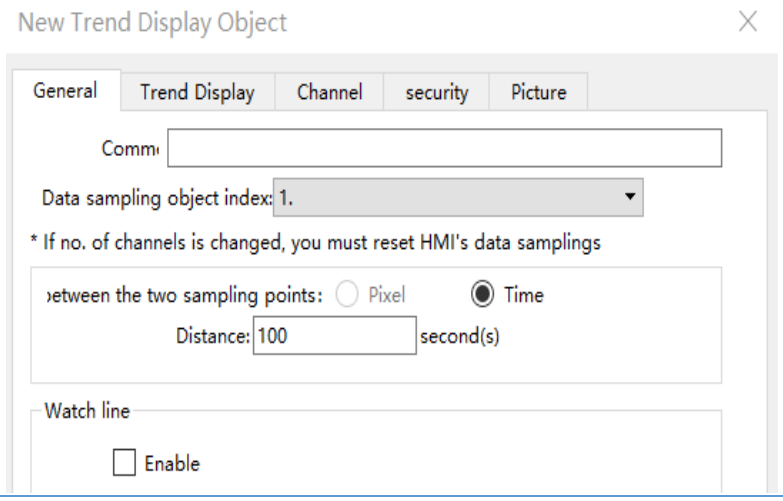
12.25.1 Overview

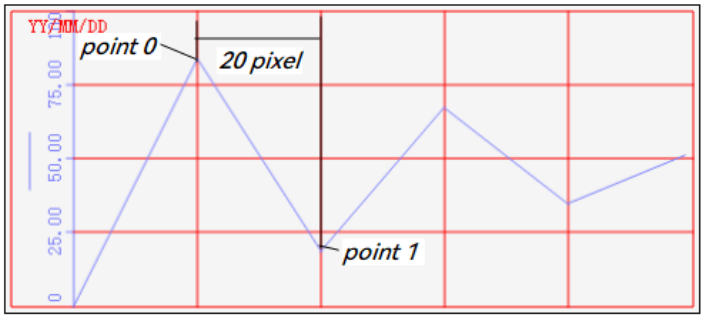
Trend display objects draw curves of the data recorded by Data Sampling object.

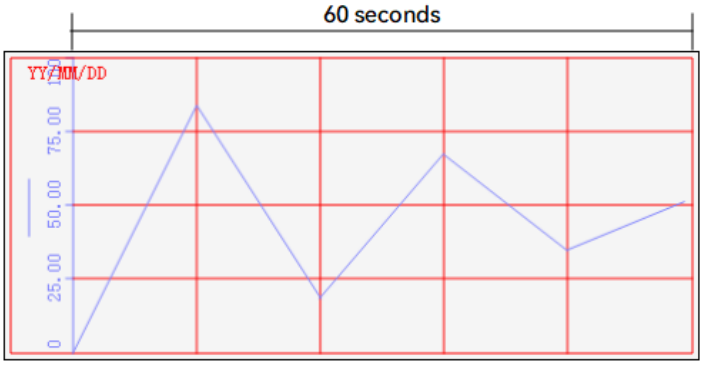
12.25.2 Configuration

Click [Data/History] » [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.

General Tab

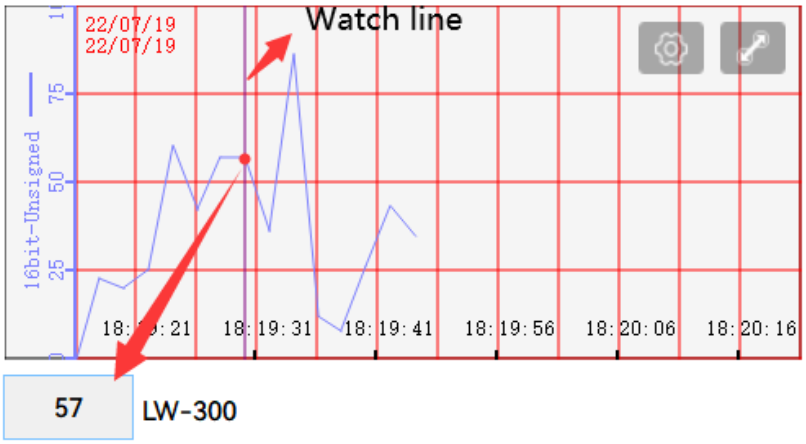


Setting	Description
Data Sampling Object index	Select a Data Sampling object as the source data.
Pixel	[Distance] is used to set the distance between two sampling points, as shown in the following figure. 
Time	[Distance] is used to set the X-axis in unit of time, as shown in the following figure



Watch line

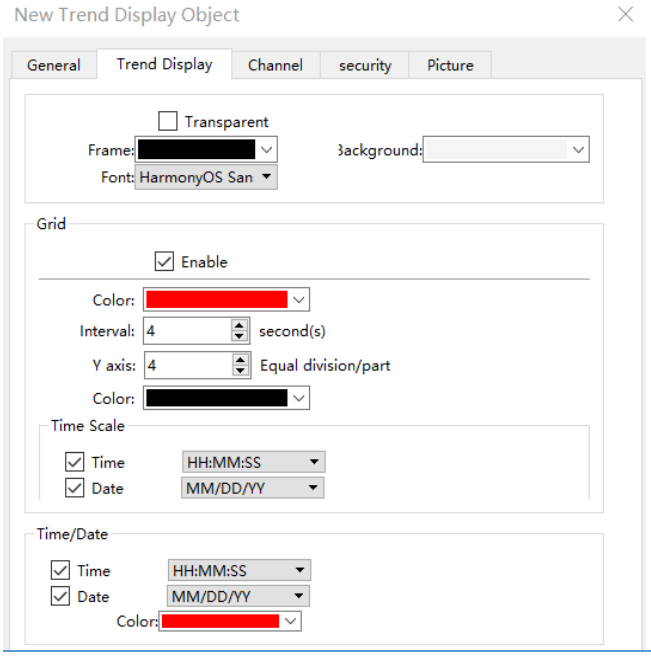
Use the [Watch line] function to display a “watch line” when user touches the Trend Display object. It will also export the sampling data at the position of watch line to the designated word device and use Numeric objects to display the results, as shown in the following figure.

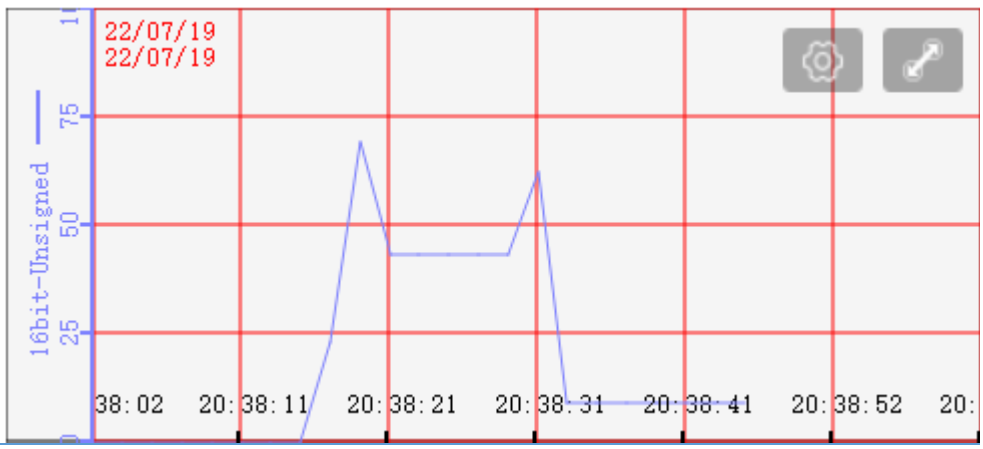


[Watch line] can also export sampling data with multiple channels. The system will consecutively write each channel to the specified address and the following addresses, in the same order as in [Data Sampling] object. The address assigned to [Watch line] is the start address, and sampling data for each 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:

Register	Channel	Data format
LW-300	0	16-bit Unsigned(1 word)
LW-301	1	32-bit Unsigned(1 word)
LW-303	2	32-bit Float(1 word)
LW-305	3	16-bit Signed(1 word)

Trend Tab



Setting	Description
Transparent / Frame / Background/Font	Select the color of frame and background. Also can set the font of text.
Grid	Set the number of dividing lines and the line color. The number of divisions depends on the setting in General tab » [Distance between data samples] / [X axis time range]. **X-axis interval** The number of vertical grid lines. Select [Distance between data samples] in General tab: Select how many sampling points will be included between two vertical grid lines. Select [X axis time range] in General tab: Select the time range between two vertical grid lines. **Y-axis division** The number of horizontal grid lines.
Time scale	Select [Time] / [Date] check box to display the time scale along the x axis. 

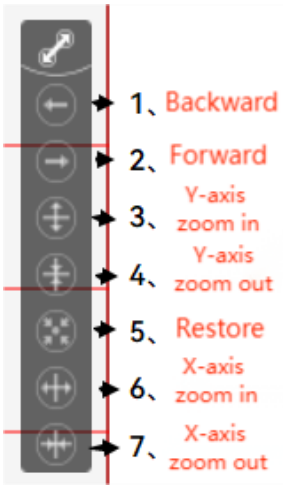
	Time mode Select time mode as HH:MM OR HH:MM:SS HH:MM HH:MM:SS. **Date** Select date mode of Time Scale.
Time / Date	The time of the latest sampling data will be marked on the top left corner of the object. This group box is used to set the time / date display format and font color.

Note

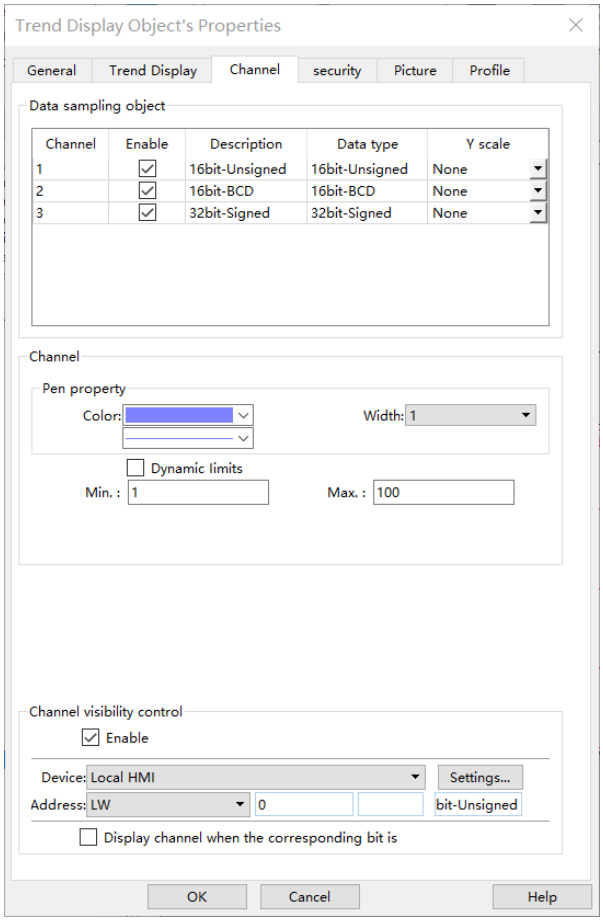
In addition to supporting the query of data at the specified time (see Y-axis Tab for details), the Trend Display can also adjust the X/Y axis (forward/backward, zoom in and out) on the touch screen. Please refer to the following steps.

1. Click the botton  on the Top-right of [Trend Display] Object.

2. Select the function shown as following. If the icon  shows, the Trend Display is in History Mode. On the contrary, it is in Real Time Mode.



Channel Tab



Setting	Description
Y scale	Set Y-axis to be Main Axis or Aux. Axis.
Channel	Configure each sampling line's format and color. Can display up to 64 channels simultaneously. **Dynamic limits** ● Not selected: [Min.] and [Max.] 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:

Data Format	16-bit	32-bit
Low Limit	LW-n	LW-n
High Limit	LW-n+1	LW-n+2

For example, if LW-100 is used here, the low limit and the high limit will be read from:

Data Format	16-bit	32-bit
Low Limit	LW-100	LW-100
High Limit	LW-101	LW-102

A typical usage of this is to zoom in and zoom out of Trend Display. See Example 1.

Channel visibility control

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:

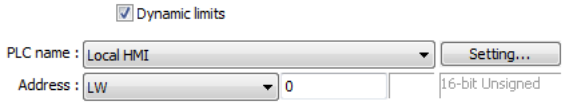
Channel	Control Bit	State	Displayed
1	LW_bit-000	OFF	YES
2	LW_bit-001	ON	NO
3	LW_bit-002	ON	NO
4	LW_bit-003	OFF	YES
5	LW_bit-004	OFF	YES

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.

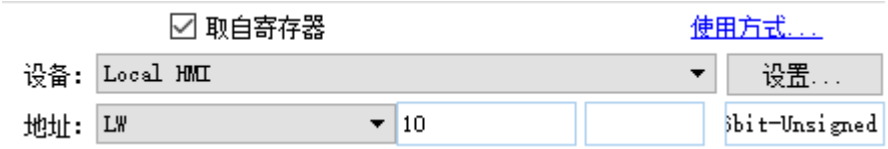
Example 1

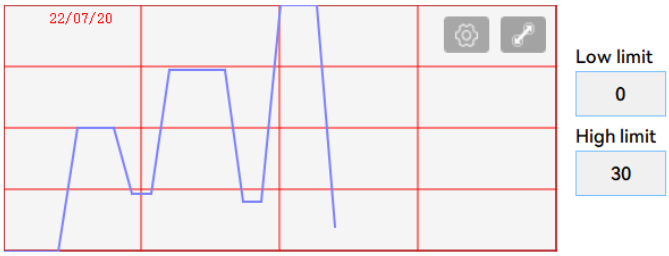
The example explains how to zoom in or zoom out Trend Display.

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-10 and create two Numeric objects for entering the low / high limit. The address that controls the low limit is LW-10; the address that controls the high limit is LW-11. 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.





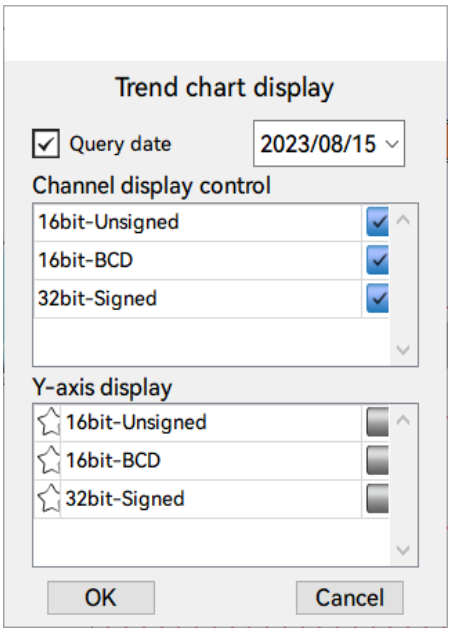
To zoom out the Trend Display, enter a value greater than 30, for example, enter 100 in [High limit], then the Trend Display will be shown as the following figure.

To zoom in the Trend Display, enter a value less than 30, such as 20, in [High limit] as shown in the following figure.

Y Scale Tab

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. Also, Y scale can be adjusted as shown in below steps.

1. Tap the button  on the upper right corner of Trend Display object.
2. Tap [Trend Display Setting] » [Y Scale], select the channel which show the Y Scale.



12.26 Disk curve graph (not yet enabled)

12.27 History Data Display

12.27.1 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. The following is an example of a history data display object.

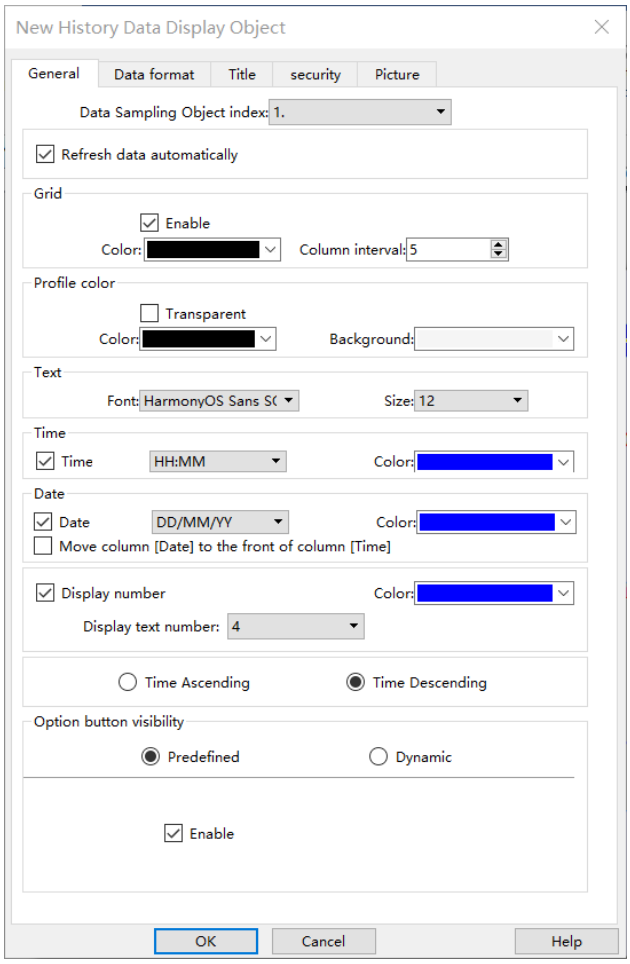
For details of Time Login, please refer to Chapter 7 Data Sampling.

编号	时间	日期	ch.1	ch.2	ch.3
59	16:31:05	20/07/22	9	60	0
58	16:31:03	20/07/22	5	60	8
57	16:31:01	20/07/22	1	60	16
56	16:30:59	20/07/22	8	60	24
55	16:30:57	20/07/22	4	60	32
54	16:30:55	20/07/22	0	60	40
53	16:30:53	20/07/22	7	60	32
52	16:30:51	20/07/22	3	60	24
51	16:30:49	20/07/22	10	60	16
50	16:30:47	20/07/22	6	60	8
49	16:30:45	20/07/22	2	60	0
48	16:30:43	20/07/22	9	60	8
47	16:30:41	20/07/22	5	60	16
46	16:30:39	20/07/22	1	60	24

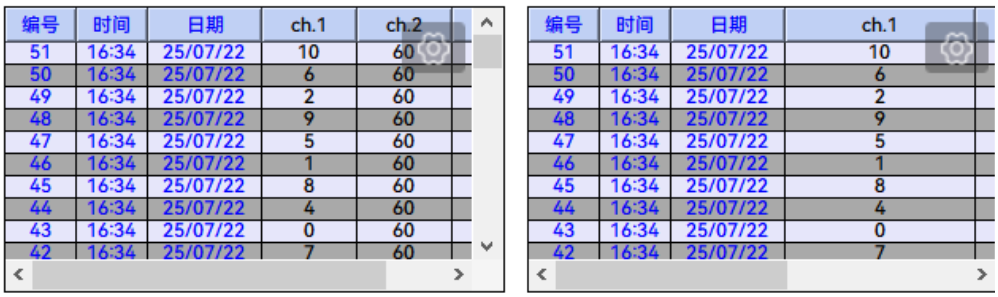
12.27.2 Configuration

 Click [Data/History] » [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.

General Tab



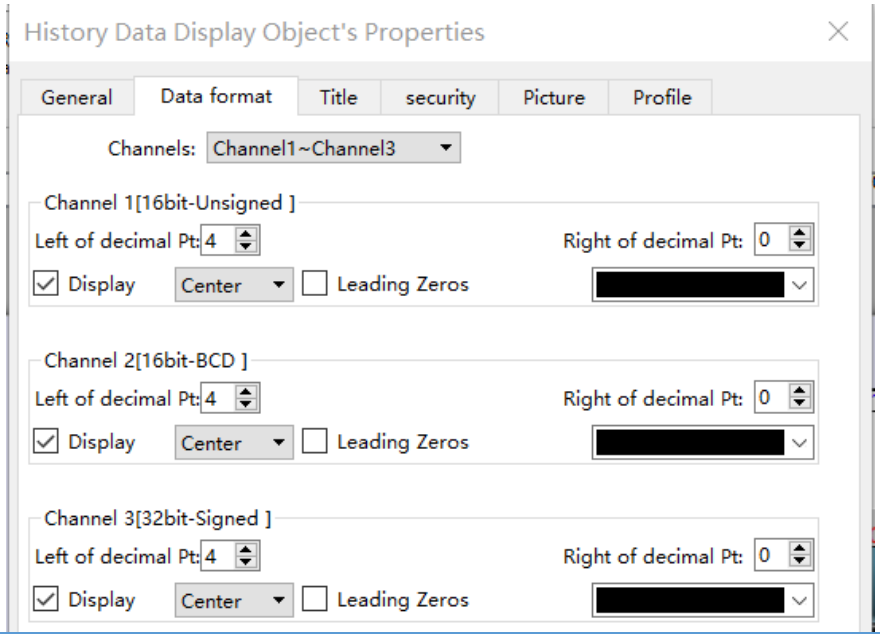
Setting	Description
Data Sampling object index	Select a Data Sampling object as the source data.
Refresh Data Automatically	The system will refresh data every 10 seconds. When this option is not selected, data can be freshed only by changing windows.
Grid	Shows grids between rows and columns. **Color** 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.
Text	Change the font and font size.
Time / Date / Display Number / Sorting style	Select to show the Sampling Data's Date & Time or not and change their display color and mode. Move column [Date] to the front of column [Time] If this option is checked, these two columns will change their display order. **Show the No.** **If this option is selected, the No. Column will be displayed.** **Time ascending** Put earliest data at the top and the latest data at the bottom. **Time descending** Put the latest data at the top and the earliest data at the bottom.

Note

- When you need to query historical data, click the icon  in the upper right corner of the History Data Display Object to specify the date and display the data.

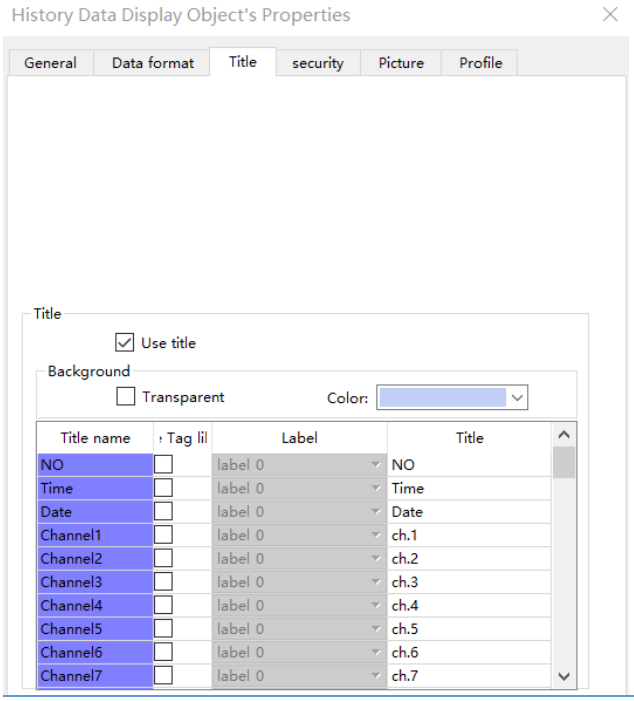
Data Format Tab



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 1 to channel 4) in the Data Sampling object, and only Ch.1 and Ch.4 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: 64 . [] 、 、 .: [] 4 (1~ 4), 16-bit Unsigned. 1 4,.  The display modes are available when displaying [String] format in History Data Display object:  ● Use [UNICODE] mode to display. ● Reverse high byte and low byte data and then display.

Title Tab



Setting	Description

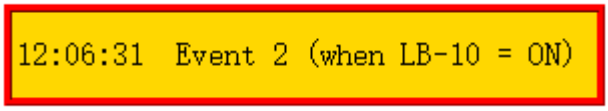
Use headers	Enable or disable title, which is marked as shown below: 
Background	**Transparent** When selected, hide the background for title area. **Color** Set the background color of title.
Setting	Defines the text to be shown on the title. The text can be edited in Label Tag Library. Build the Label Tag Library first, and in History Data Display settings dialog select the checkbox in the Label Library column, and then select the label tag.

12.28 Alarm Bar and Alarm Display

12.28.1 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 “6 Event Log”.



Alarm Bar - Displays alarm messages in one scrolling line.

Alarm Display – Displays alarm messages in multiple lines.

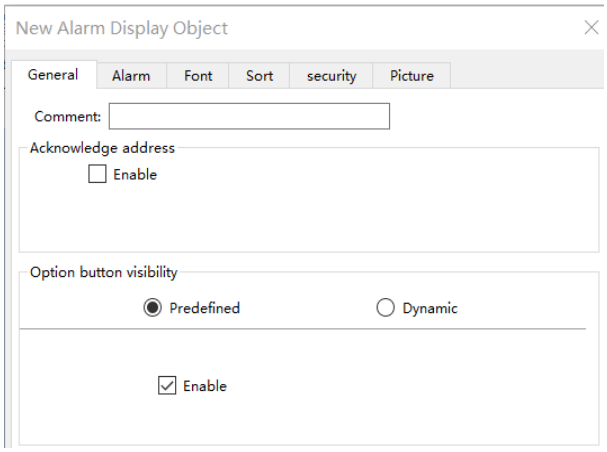
[] ,

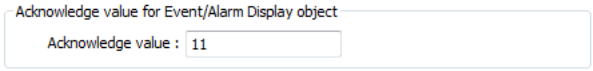
12.28.2 Configuration

Click [Data/History] » [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.

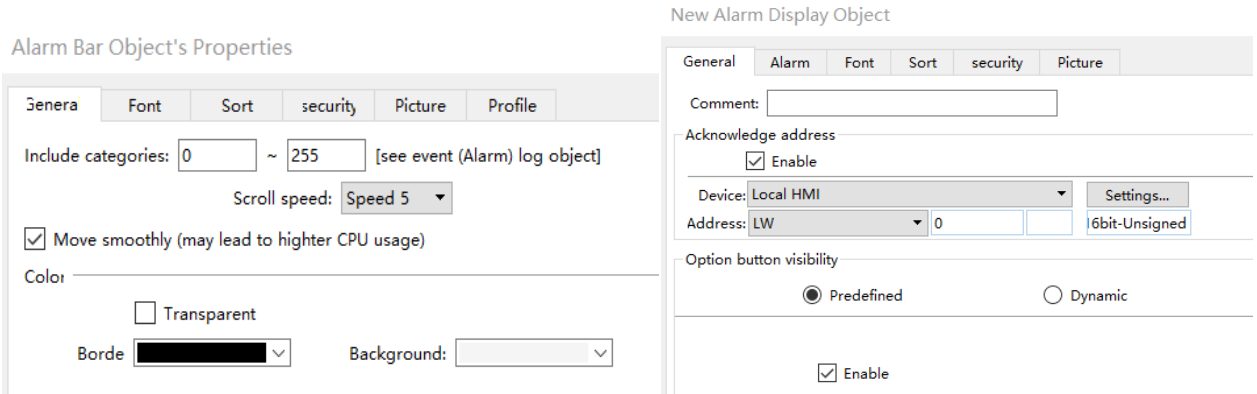
General Tab

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 “6 Event Log”. 

Alarm Tab

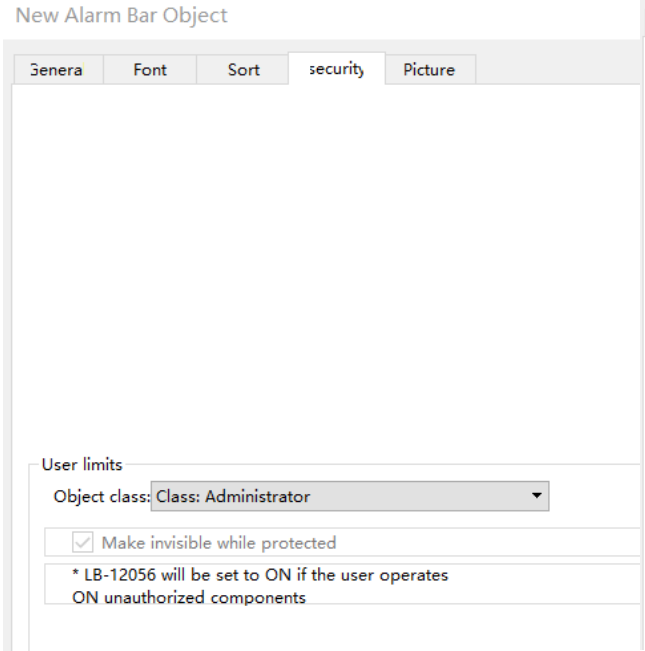


Setting	Description
Include categories	Events in the selected category will be displayed. The categories are set in Event (Alarm) Log object. For example, if the category is set to “2 to 4” here,

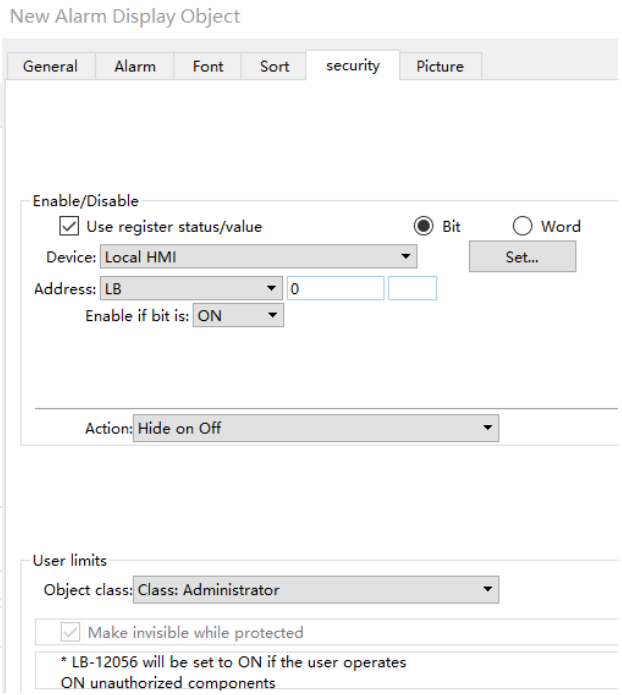
	only events in categories 2, 3, 4 will be displayed. For more information, see “6 Event Log”.
Scroll Speed	Only for Alar Bar. Select one of the speed settings at which the messages scroll.
Move smoothly	When this option is selected, the message will move along the alarm bar more smoothly. Please note that enabling this may lead to high CPU loading.
Sort	**Time ascending** Latest event is placed last in the list (the bottom). **Time descending** Latest event 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

Security Tab

Alarm Bar



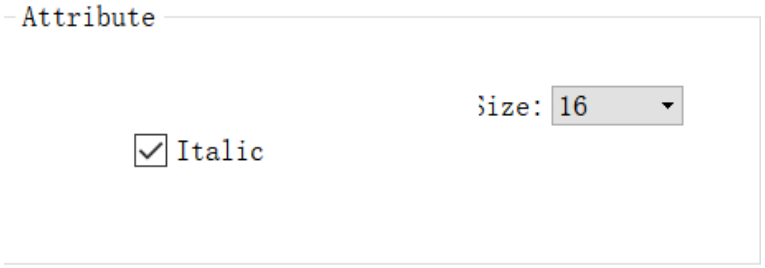
Alarm Display



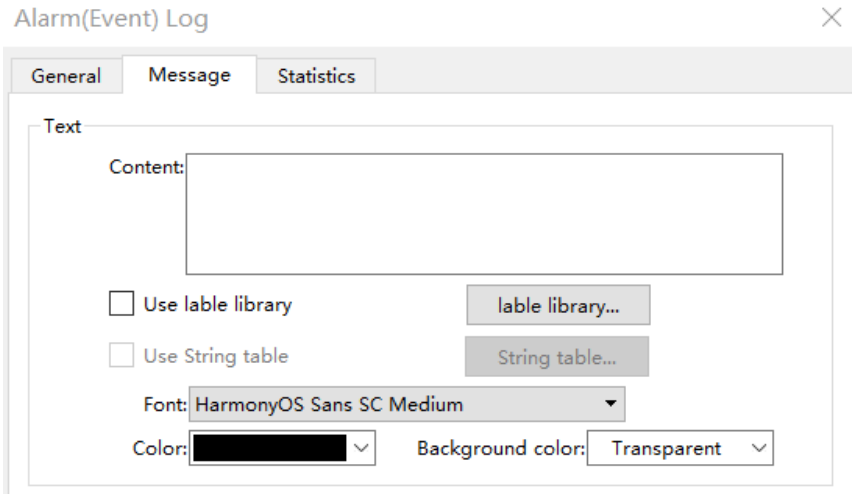
Setting	Description
Enable/Disable	When [Use Register status/value] check box is selected, whether the object is operable is determined by the state of a designated Bit address. As shown in the above settings, if LB-0 is ON, the object is operable. **Hide when disabled** When the designated Bit is OFF, the object will be hidden. **Do nothing when disabled** the unauthorized user can see the Alarm Display object, but cannot trigger the object or make any change.
User Limits	**Object class** “None” means any user can operate this object. Only account “admin” can operate “Administrator” object class. When the user's privilege does not match the object class, or the user is unauthorized to operate the object, the object will be hidden.

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:



12.29 Event Display

12.29.1 Overview

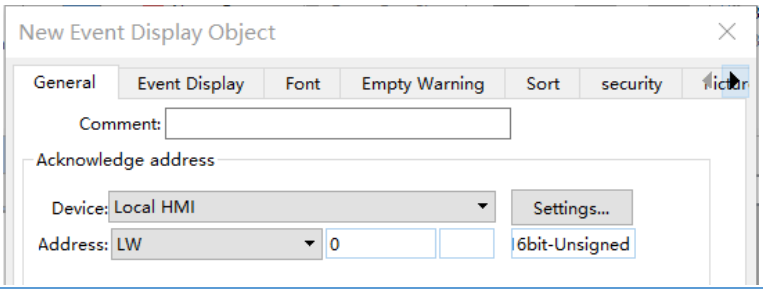
Event Display object is used to display event messages which are defined in Event (Alarm) Log and have met a trigger condition. The triggered events are displayed in the chronological order. Event Display object displays: the date and time the event occurs, the time the event is acknowledged, the time the event returns to normal, the event message, the occurrence count, and the elapsed time. Multi-lined messages can also be displayed.

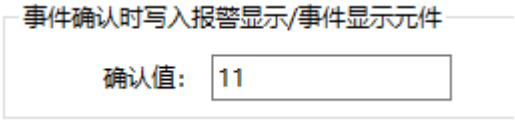
序号	事件发生日期	事件发生时间	确认时间	事件信息
1	07/26/22	15:22:33	15:22:36	Event 2 (when LB-10 = ON)
2	07/26/22	15:22:35	15:23:02	Event 3 (when LB-11 = ON)
3	07/26/22	15:22:42	15:22:48	LW20为 0% ， 大于0;LW20为 0 ， 大于0
4	07/26/22	15:22:45		Event 1 (when LW-1 >= 10)

12.29.2 Configuration

 Click [Data/History] »[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.

General Tab

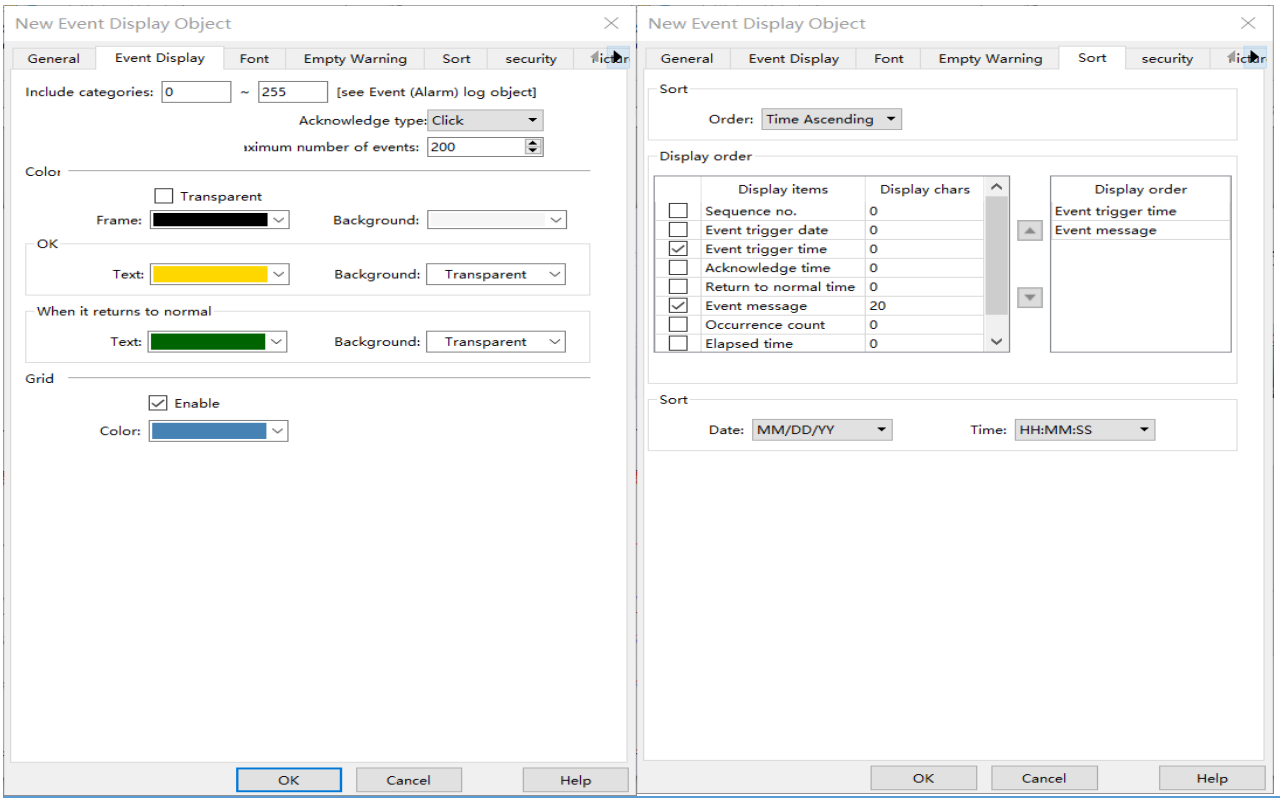


Setting	Description
	When in Real-time mode and an event is acknowledged by touching an active display line, the [Acknowledge value] specified in Event (Alarm) Log object, Message tab, is output to the [Acknowledge address] of Event Display object. For more information, see “6 Event Log” 

all the events that occurred are displayed and updated in real-time.

Press the Option Button  in the upper-right corner of the object and set the start and end date. If the dates are not set specifically, all the events are displayed.

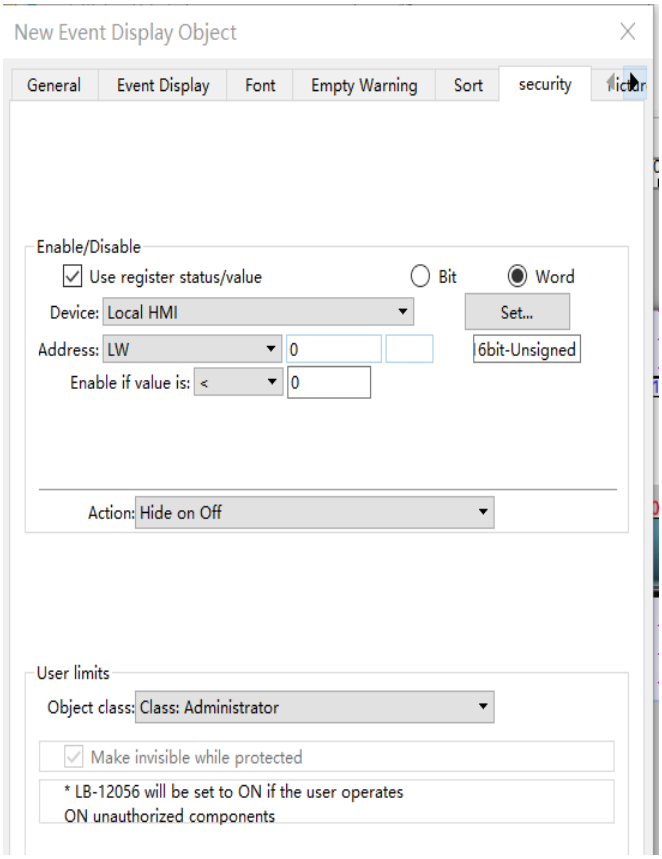
Event Disply Tab



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 “6 Event Log”.

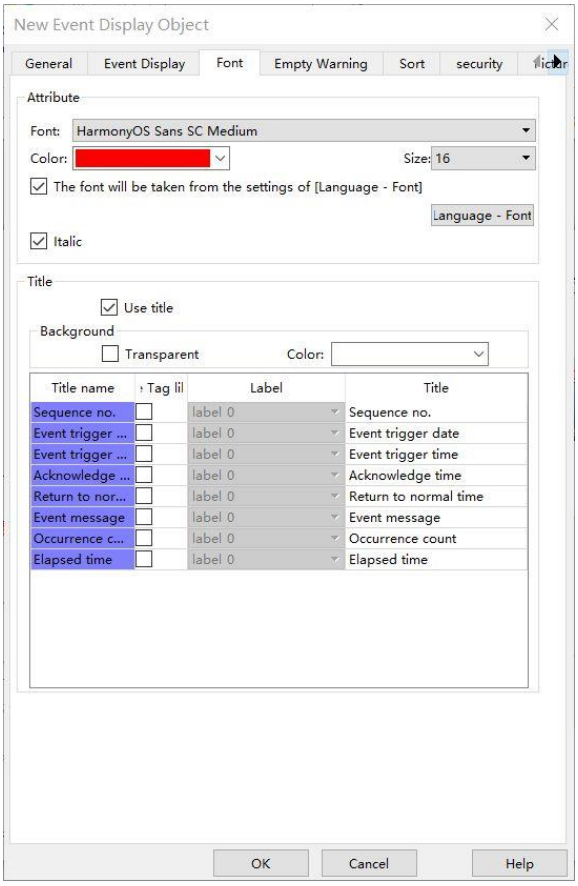
Acknowledge style	Select [Click] or [Double Click] to acknowledge each single event. When an event occurs the user can tap the event line once or twice to acknowledge the new event. When acknowledged, the text color of the event will change to the selected color, and the acknowledge value associated 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.
Sort	**Time ascending** Latest event is placed last in the list (the bottom). **Time descending** Latest event 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 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

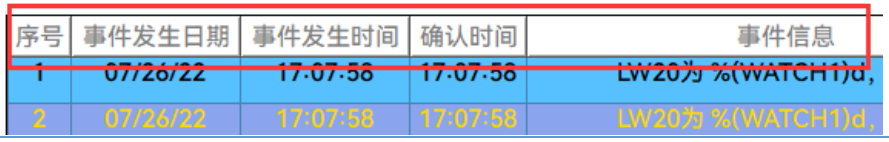
Security Tab



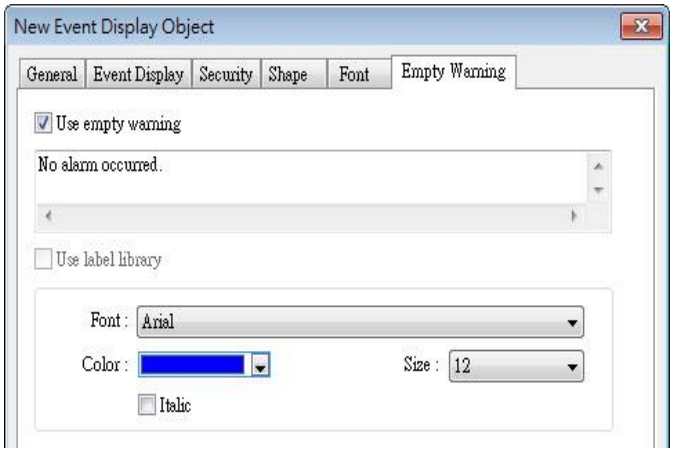
Setting	Description
Enable/Disable	When [Use Register status/value] check box is selected, whether the object is operable is determined by the state of a designated Bit address. As shown in the above settings, if LB-0 is ON, the object is operable. **Hide when disabled** When the designated Bit is OFF, the object will be hidden. **Do nothing when disabled** the unauthorized user can see the Alarm Display object, but cannot trigger the object or make any change.
User Limits	**Object class** “None” means any user can operate this object. Only account “admin” can operate “Administrator” object class. When the user's privilege does not match the object class, or the user is unauthorized to operate the object, the object will be hidden.

Font Tab



Setting	Description
Attribute	Set the font, font size, color of Title or select [Italic].
Use Title	This option is checked to select to use Title. 
Background	**Transparent** When selected, hide the background for title area. **Color** Set the background color of title.
Settings	Defines the text to be shown on the title. The text can be edited in Label Tag Library. Build the Label Tag Library first, and in History Data Display settings dialog select the checkbox in the Label Library column, and then select the label tag.

Empty Warning



When [Use empty warning] is enabled, the text displayed when no event has occurred can be specified.

12.30 Data Transfer (Per-page)()

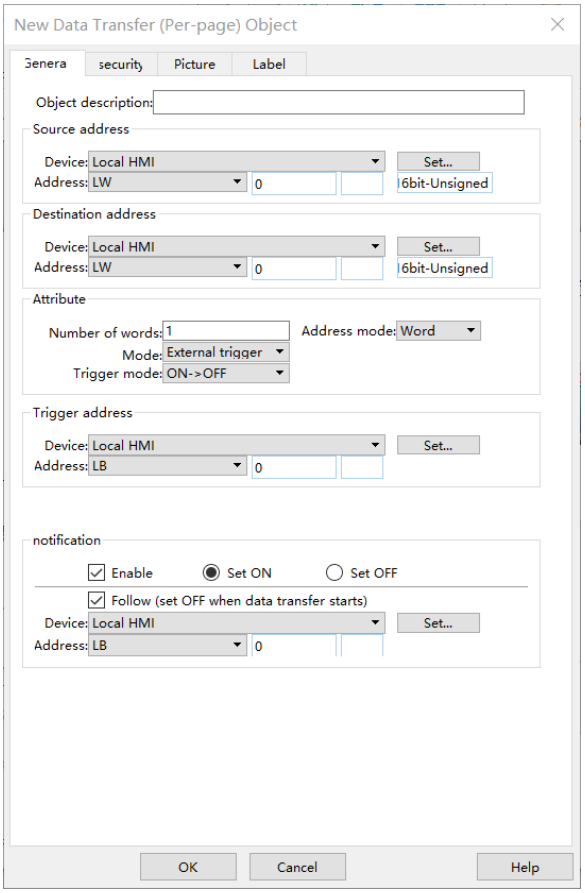
12.30.1 Overview

Data Transfer (Per-page) object can transfer values from the source address to the destination address when the window in which this object is placed opens. The data transfer operation can be activated by manually pressing the object or by detecting the change of state of the designated bit.

12.30.2 Configuration


Click [Object] » [Data Transfer] » [Data Transfer (Per-page)] icon on the toolbar to open the property dialog box. Set up the properties, press OK button, and a new Data Transfer (Per-page) object will be created.

General 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].
Attribute	**No. of word** Enter the number of words to be transferred. Unit: 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.
Notification	When enabled, the system will set the designated address ON or OFF when it's ready for data transfer. **Follow**

	The notification bit will reset to its original state once the system starts data transfer

Note

- When Data Transfer (Per-page) object in Bit-Trigger mode is placed in the common window, it activates data transfer operation once the specified condition is met, regardless of the current base window. But if it is placed in another windows, it activates data transfer operation only when the designated [address] is placed in same window.

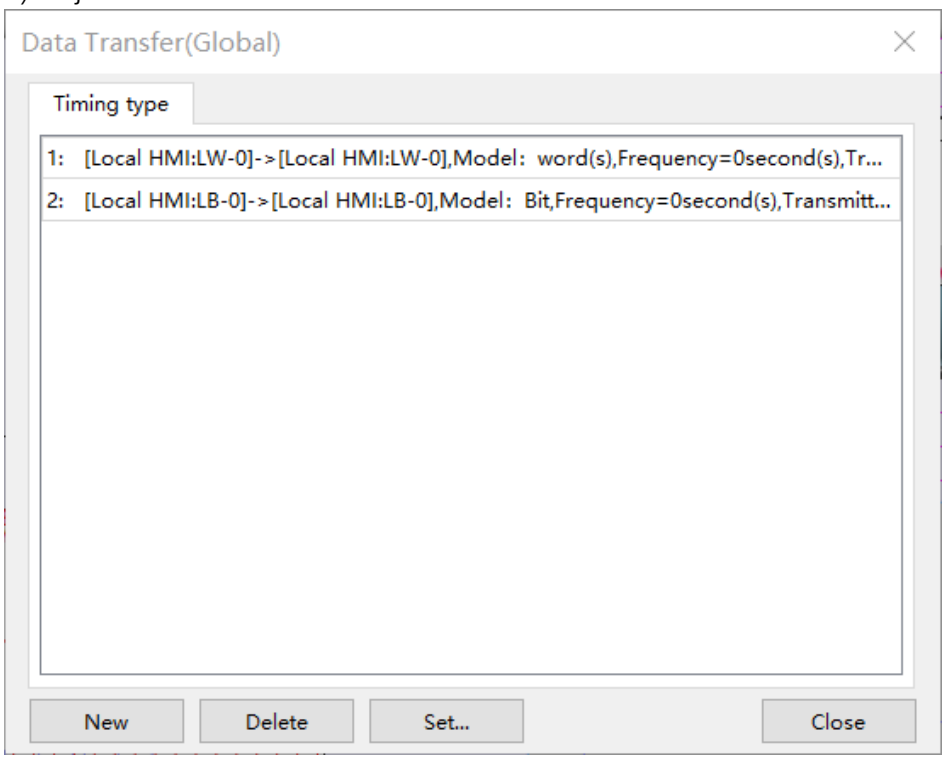
12.31 Data Transfer (Global)

12.31.1 Overview

Data Transfer (Global) object in Time-based mode can transfer data from the source address to the destination address based on a time schedule.

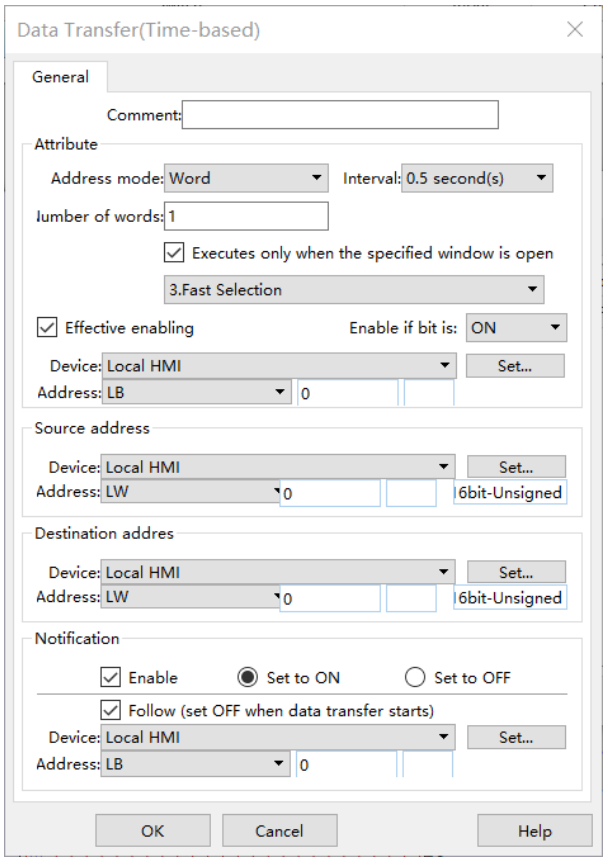
12.31.2

Click [Object] » [Data Transfer] » [Data Transfer (Global)] icon on the toolbar and open the Time-based tab. Click [New] and set up the properties, press OK button, and a new Data Transfer (Global) object will be created.

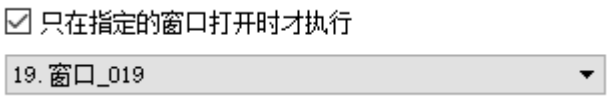
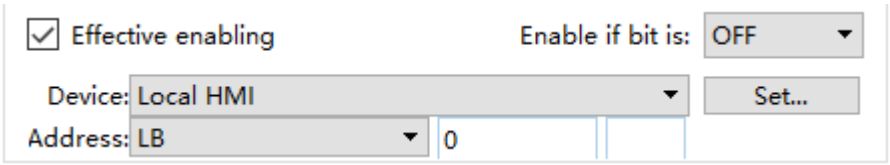


General Tab

Click the [New] button in the Data Transfer management dialog box.



Setting	Description
Attribute	**Address mode** Select the data type to be transferred from [Bit] or [Word]. **No. of bit /No. of word** When [Bit] is selected in [Address mode], set the number of bits transferred each time when data transfer is triggered. When [Word] is selected in [Address mode], 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 shorter time interval or a greater amount of data to transfer may decrease overall system performance; therefore, a longer time interval or a smaller amount of data to transfer is recommended. When a short interval is inevitable, please at least set an interval longer than the time needed for data transfer operation; for example, when the data transfer operation takes 2 seconds, the interval must be longer than 2 seconds.

Activate only when designated window opened	Data Transfer object transfers data only when the designated window is opened. 
Source address	Data Transfer object reads the data from [Source Address].
Destination address	Data Transfer object writes the data to [Destination Address].
Notification	When enabled, the system will set the designated address ON or OFF when it's ready for data transfer **Follow** The notification bit will reset to its original state once the system starts transferring data.
Effective enabling	When enabled, we can set [Address] and Trigger Mode.  If the designated state is same as the set Enable bit, the Data Transfer operation is activated. The Enable bit can be ON or OFF.

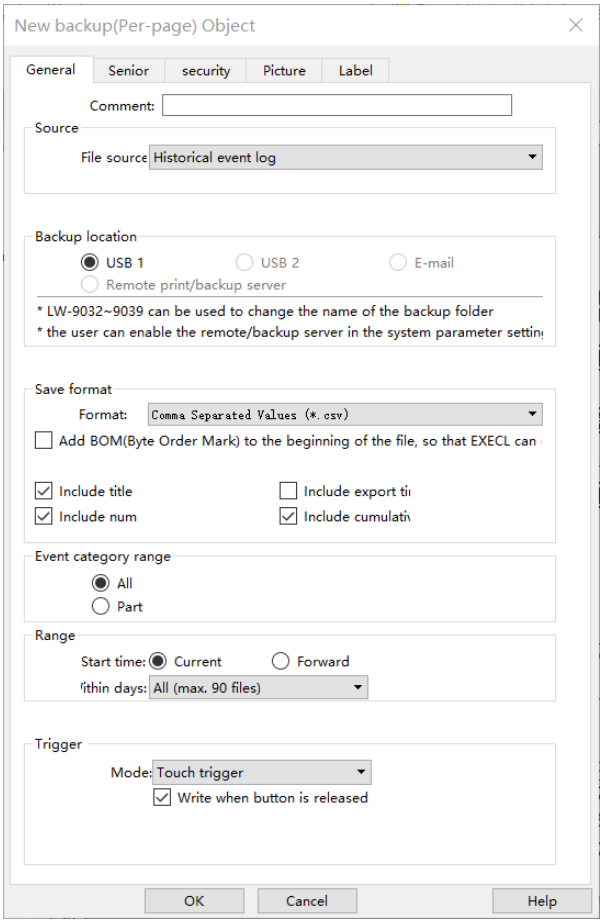
12.32 Backup

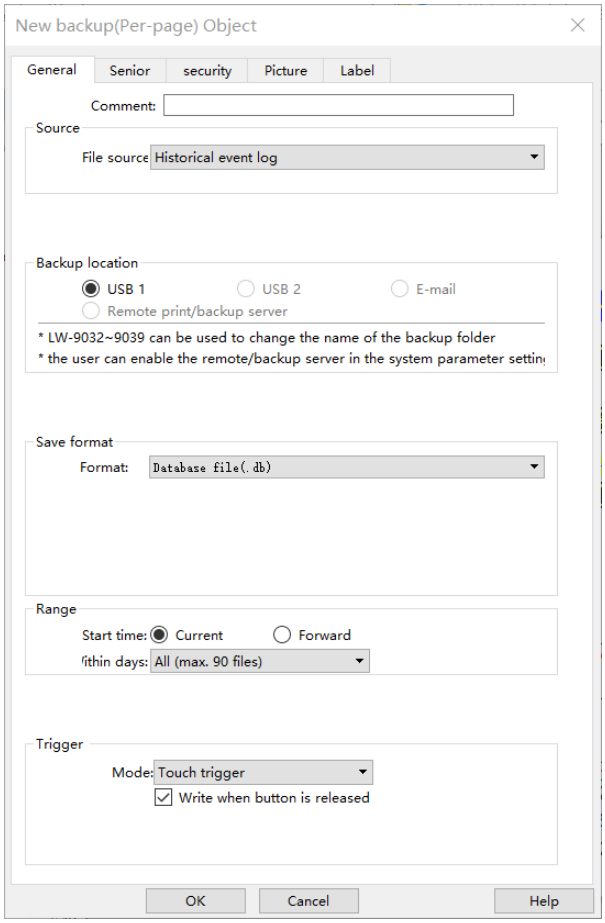
12.32.1 Overview

Backup (Trigger-based) object and Backup (Global) 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.
Backup (Trigger-based) object is triggered by pressing the object on the screen while Backup (Global) object runs in the background regardless of screens being viewed.

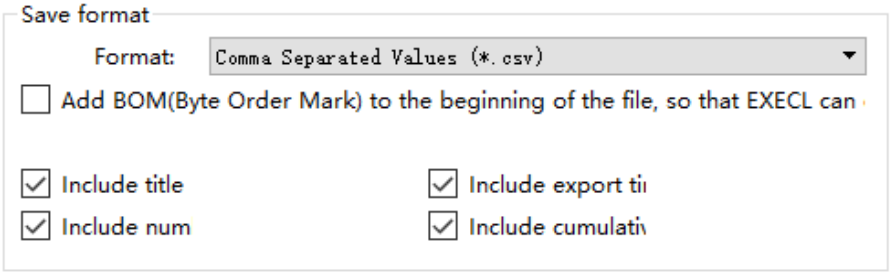
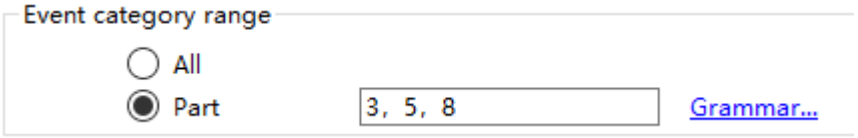
12.32.2

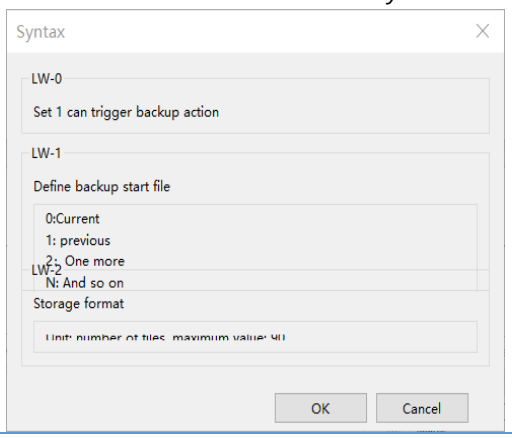
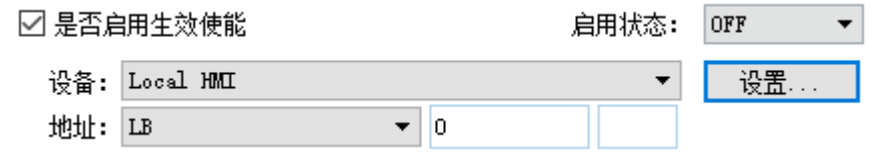
On the toolbar click [Object] » [File Operation] to find [Backup (Trigger-based)] and [Backup (Global) objects. Open [Backup (Trigger-based)] object property dialog box, set up the properties, press OK button, and a new Backup (Trigger-based) object will be created.
Open [Backup (Global)] object managing dialog box, click [New] to open the object property dialog box, set up the properties, press OK button, and a new Backup (Global) object will be created.





Setting	Description
Source	**[RW], [RW_A], [Recipe database], [Historical event log], [Historical data sampling], [Operation log], [SQL Query]** Select one from the above for the source. When backing up [Historical data sampling], use [Data Sampling object index] to select the one to back up. Options other than RW, RW_A will be available only when they are used in the project file.
Backup position	Select the destination where the source files will be saved to. **SD card / USB disk** The external device connected to HMI. **Remote printer/backup server**

	This option is available when [System Parameters] » [Printer/Backup Server] » [Use Remote Printer/Backup Server] is selected. Please note that [Operation log] backup can only be saved to Remote printer/backup server. To save into a SD card or USB drive, use the control address of Operation Log object. **E-mail** To send backup using e-mail, go to [System Parameters] » [e-Mail] tab to configure first, and then open Backup object's e-Mail tab to configure the recipient address, subject, and message.
Save format	• Database file (.db) • Comma Separated Values (.csv) **Add BOM (Byte Order Mark) to file header for EXCEL can interpreting non-ASCII strings correctly.** To back up history logs 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 containing non-ASCII strings can directly be opened in EXCEL. The title, export time, occurrence count, and elapsed time can be included or omitted in the backed-up .csv file. 
Event category range	This groupbox will be available only when backing up a historical event log in CSV format. Two options can be found in this group box: All and Partial. Selecting partial and entering "3, 5, 8" in the field will backup events in categories 3, 5, and 8. Selecting partial and entering "3-8" will backup events in categories 3 to 8. 
Range	**Start time** Current: start on today: , Forward: start by yesterday: . **Within days** Select a number of days. A file covers all logs of one day. For example, if [Start] is set to [Current], and [Within days] is set to [2 file(s)], today & yesterday files in memory will be backed up. If [Start] is set to [Forward, and [Within days] is set to [2 file(s)], the files obtained yesterday and the day before yesterday will be backed up. Select [All] to save all files, and the maximum is 90 days.

Trigger	**Mode** There are three ways to activate Backup function. **Touch trigger** Touch the Backup object to activate backup operation. **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. **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). 
Effective enabling	When enabled, we can set [Address] and Trigger Mode.  If the designated state is same as the set Enable bit, the Data Transfer operation is activated. The Enable bit can be ON or OFF.

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.

When saving files to USB disk or SD card, the capacity of a FAT32 folder depends on the length of the file names. Fewer files can be saved when the file names are longer.

Advanced Tab (in developing)

e-Mail Tab (in Developing)

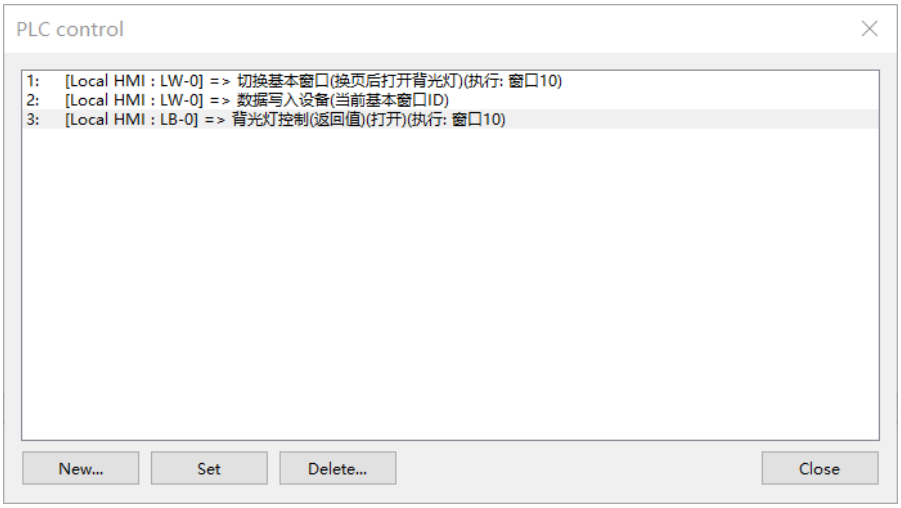
12.33 PLC Control

12.33.1 Overview

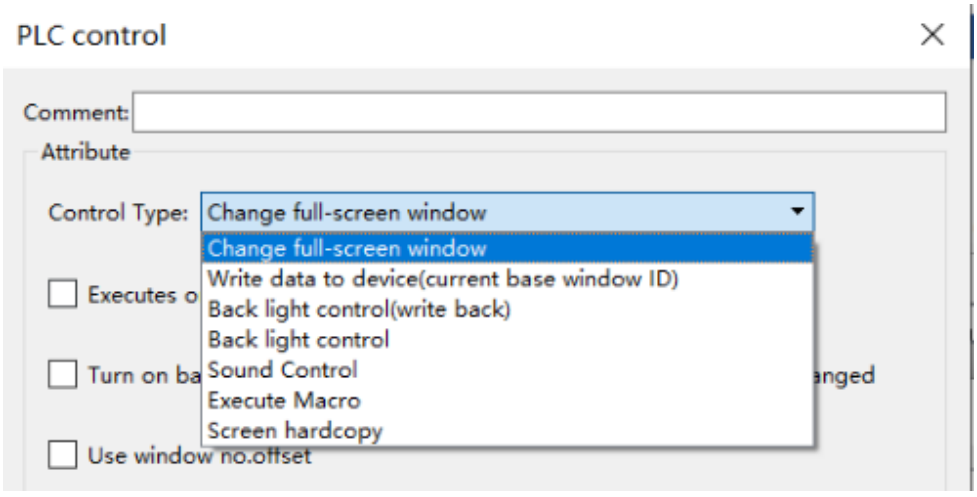
PLC Control object can execute commands when it is triggered.

12.33.2 Configuration

 Click [Object] » [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.

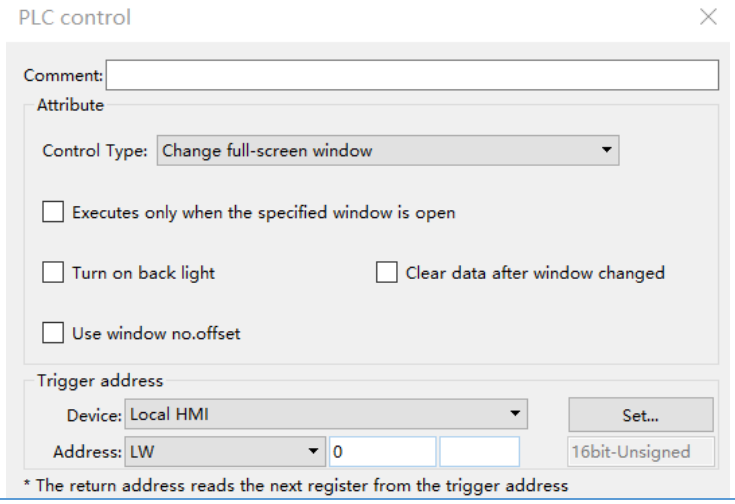


Click [New] and the following dialog box appears. See “Type of Control”.



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 is changed.
Clear data after window changed	Reset the value at trigger address to zero when the window object is changed. If [Use window no. offset] is selected, this option will only show when a negative offset is used.
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 -1024 to 1024.

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: [Trigger address + 1 (16bit)] or [Trigger address + 2 (32bit)].
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 + 1
32-bit BCD	Address	Address+2 + 2
16-bit Unsigned	Address	Address+1 + 1
16-bit Signed	Address	Address+1 + 1
32-bit Unsigned	Address	Address+2 + 2
32-bit Signed	Address	Address+2 + 2

- Write data to PLC (Current base window) PLC (ID)

Each time the base window is changed, the new window number will be written into the [Trigger address]. If [Use window no. offset] is selected, the window number of the base window plus the window number offset will be written into the [Trigger address].

- PLC Control

Transfer word data blocks from PLC to HMI, and vise-versa, and the transfer direction is controlled by the value in the [Trigger address]. Detail as below table.

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	Purposes	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)**

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**

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. If select [SD card] or [USB disk] as [Printer], a “hardcopy” folder will be generated in the selected external device for saving the printed screen in JPG format. The name of the JPG files starts from yyyymmdd_0000.

To print the screen using a printer, go to System Parameter Settings » Model tab and set the printer.

To print the screen using a remote printer, go to System Parameter Settings » Printer/Backup Server tab and configure the parameters.

There are three options to specify the source window for hardcopy:

Current base window

Print the base window currently opened.

Window no. from register

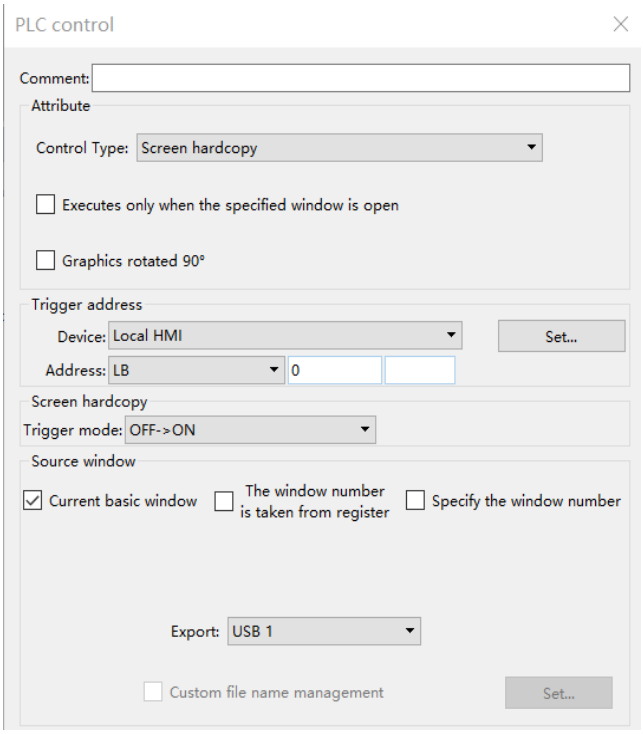
Print the window designated by the value in a designated word address.

Designate window no.

Directly select a window to be printed.

Customized File Handling

This feature can be used to customize naming of the folders and the JPG files.



Setting	Description
Folder name	The folder name can be an alphanumeric name, and certain half-width symbols are allowed: !@#$%^&()_+{}`-=;',. The folder name can also be specified by a naming syntax. **Dynamic format** The folder names can be set by a designated word address, or by a naming syntax indicating the current system time. The syntax can be specified by selecting time buttons or entering the syntax in Format field. The length limit is from 1 to 25. **Note:** Up to 10 layers of folders can be created. The exceeding layers will be ignored.
File name	The way to specify a file name is similar to specifying a folder name. **Note:** If the file name already exists, the system will add "_0001" to the file name as a serial number. The numbers of the later files add up until "_9999". The files after "_9999" will be ignored. For example, if the three file names exist: "A.jpg", "A_0001.jpg","A_0003.jpg". When trigger screen hardcopy with the same file name, the coming files will be named in this order: "A_0002.jpg", "A_0004.jpg", "A_0005.jpg", and so on.

Note

- Printed window will save the image in the iPad's image folder.

A background printing procedure is performed when the printed window is not the current base window.

If the hard-copied window is a background window, its [Direct Window] and [Indirect Window] objects will not be printed.

When using a dynamic format name, the system will use a "_" sign as a substitute for invalid symbols.

When using a dynamic format name, if screen hardcopy is triggered without specifying the format first, the system will save the file in the default directory, which is: hardcopy\yymmdd_0000.JPG

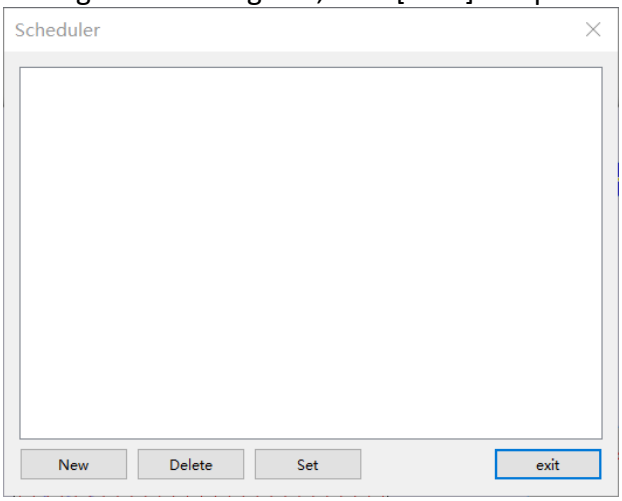
12.34 Scheduler

12.34.1 Overview

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

12.34.2 Configuration

 Click [Object] » [Time-related] » [Scheduler] icon on the toolbar to open the Scheduler management dialog box, click [New] to open the Scheduler property setting dialog box.



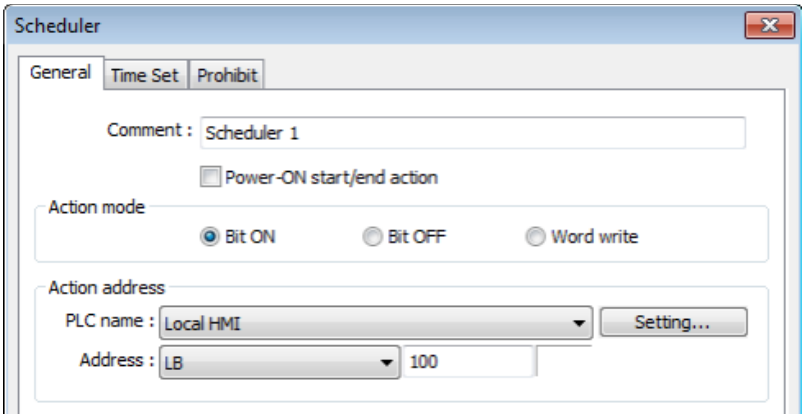
The following two demonstrations explain the usage of Scheduler.

Example 1

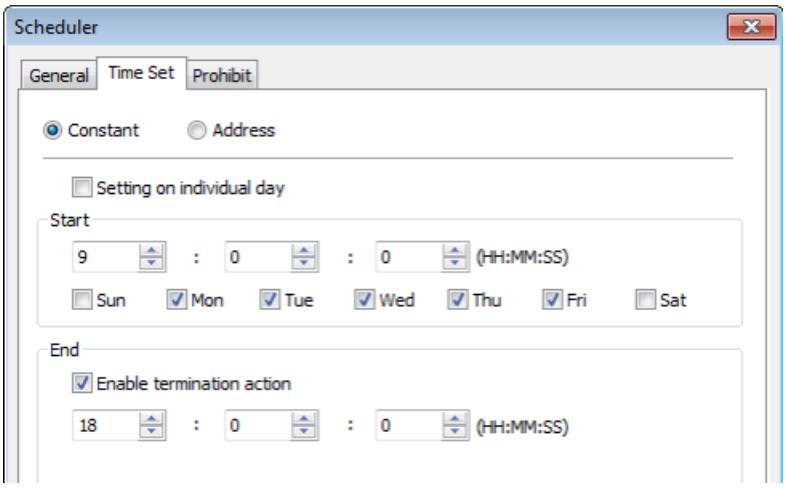
A motor is scheduled to power - ON at 9:00 and power – OFF at 18:00, Monday to Friday.
We are using LB-100 to control the motor state. LB-100 will be set ON at 9:00 and OFF at 18:00.

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.

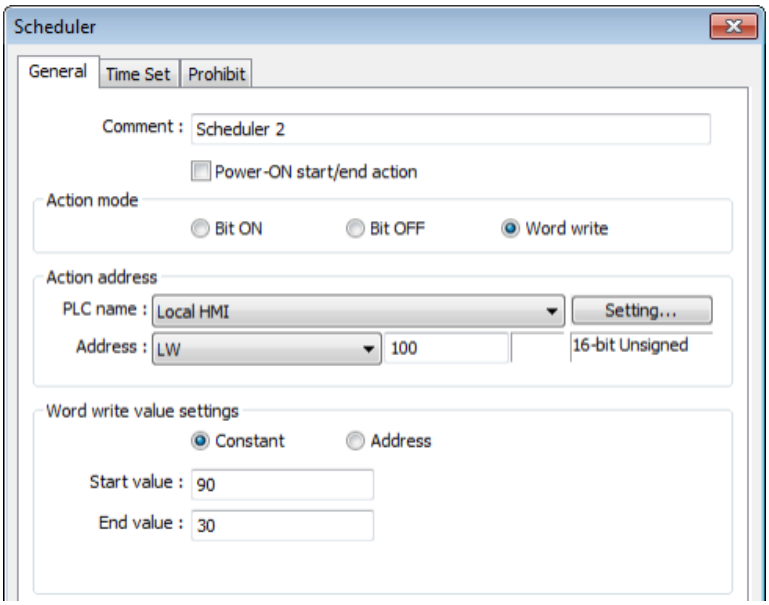
Example 2

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].

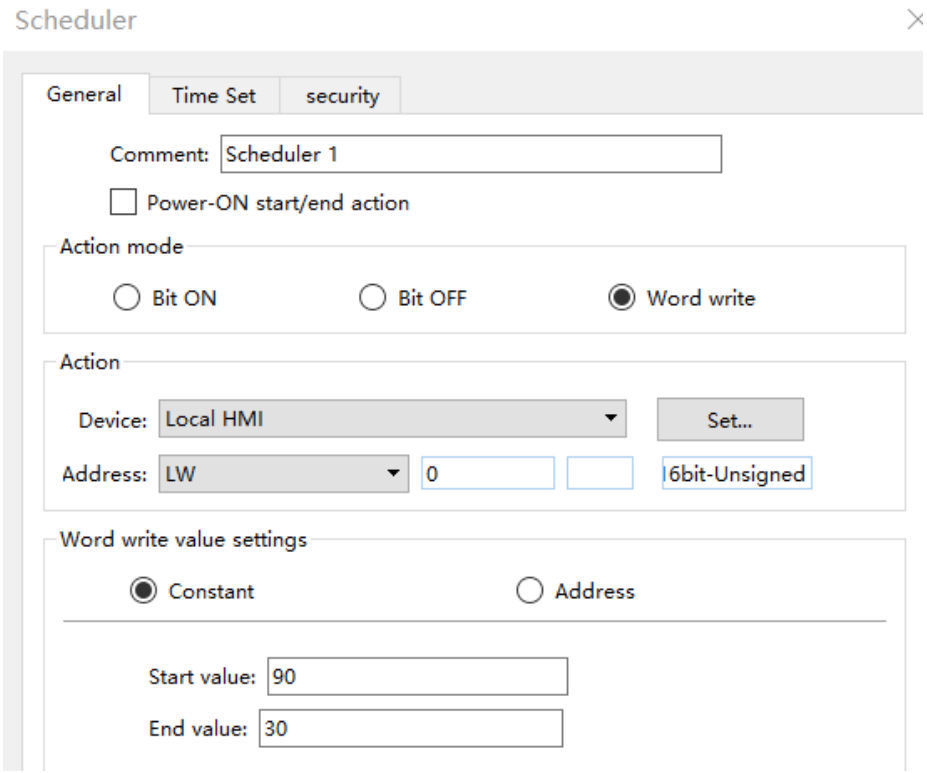
2. In [General] tab, select [Word write] in [Action mode] and set [Action address] to LW-100.

3. Select [Constant] for [Word write value settings] and enter 90 in [Start value] & 30 in [End 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. Click [OK], a new schedule object will be created on the [Scheduler] list.

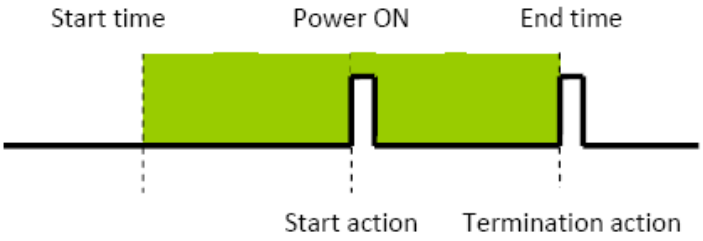
General Tab



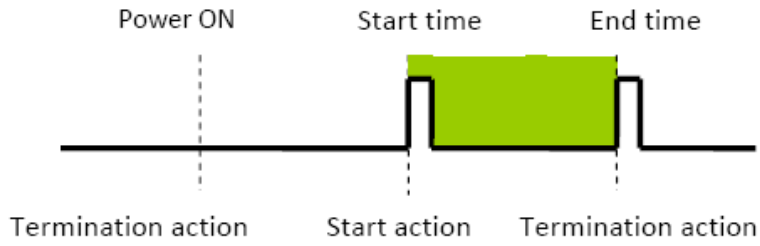
Setting	Description
Power ON start/end action	Execute the defined action when the HMI is powered ON. ● Enabled

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.

Inside scheduled range



Outside scheduled range



- 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.

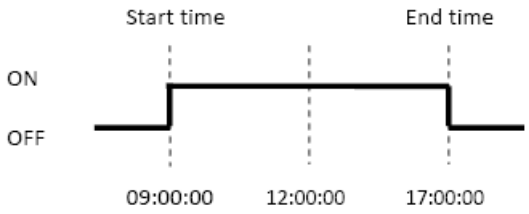
Action mode

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.

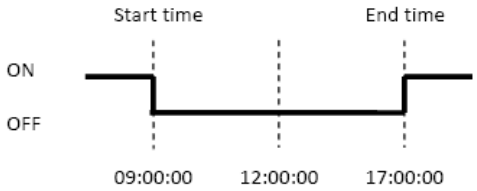
Example: Start time : 09:00:00 End time : 17:00:00



Bit OFF

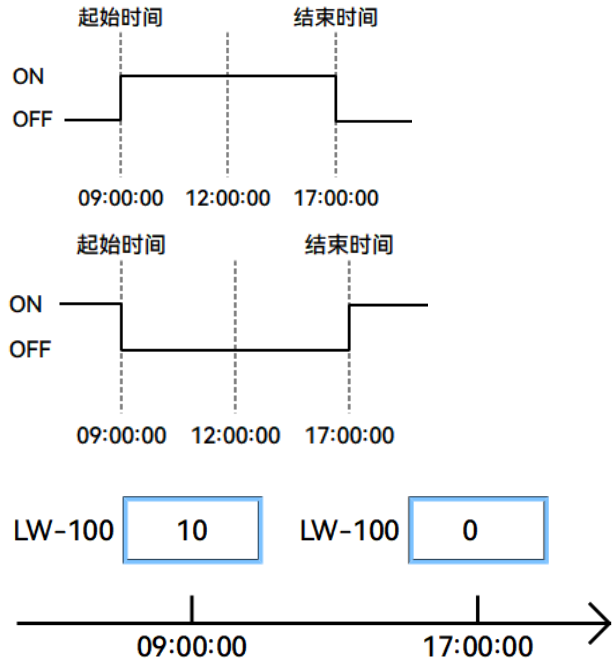
At the start time, set the designated bit OFF. At the end time, set the designated bit ON.

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).

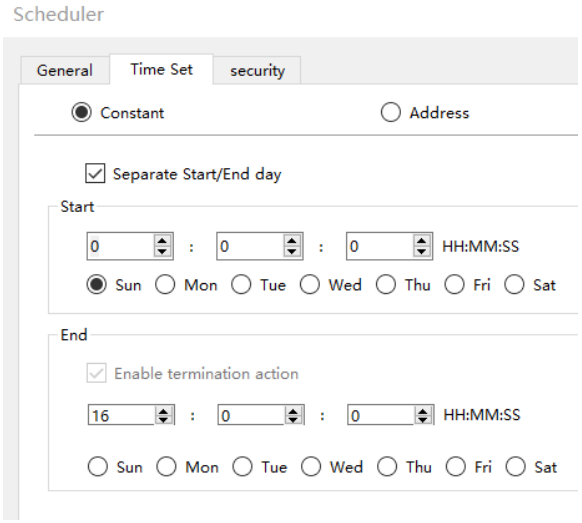


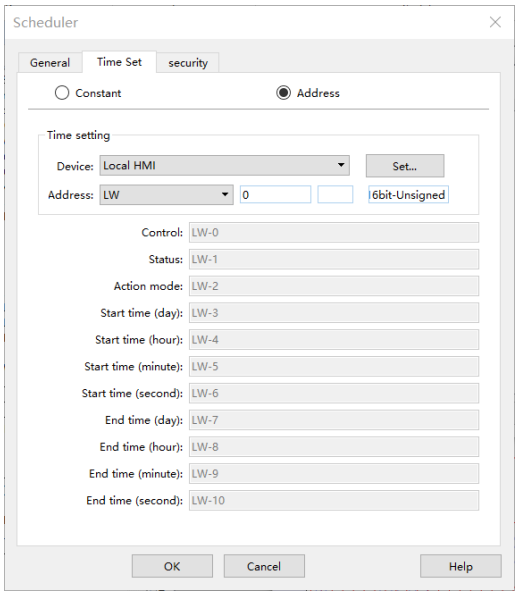
Note

- Only is an [End time] is set in the [Time set] tab will the [Write end value] box appear.

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.

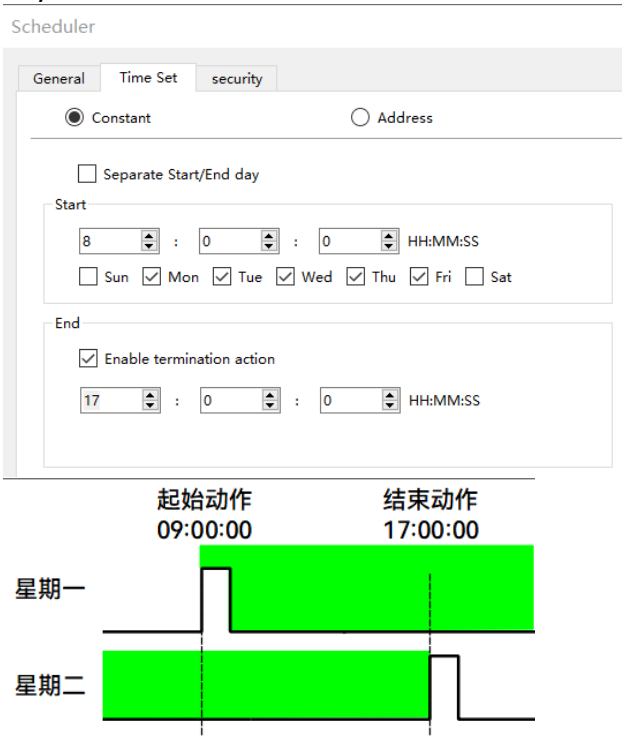




- Constant

Setting on individual day

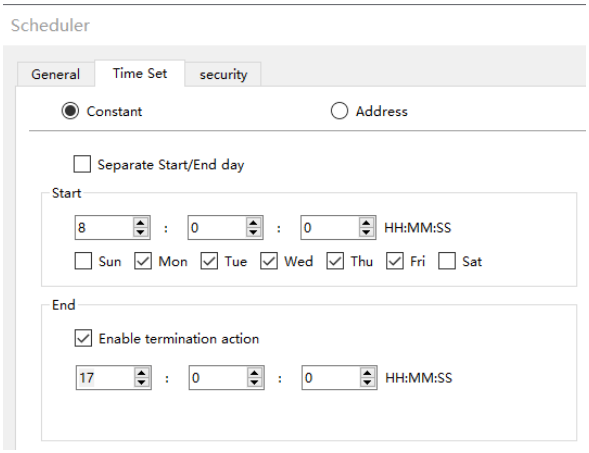
If [Setting on individual day] is selected, the same start time and end time can be assigned to different days of the week.

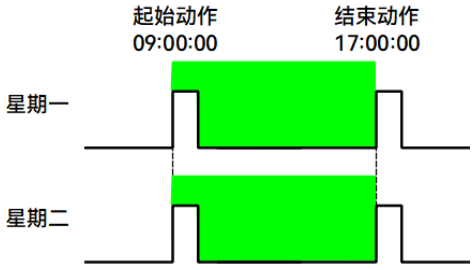


Note

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

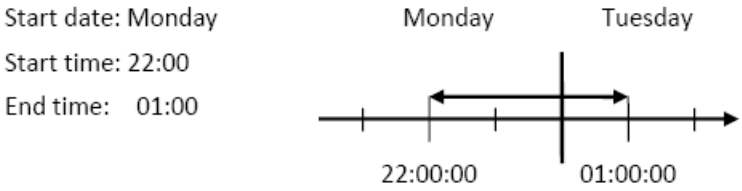
If [Setting on individual day] is not selected, start time and end time entered must start and end within 24-hours.





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.



- 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)

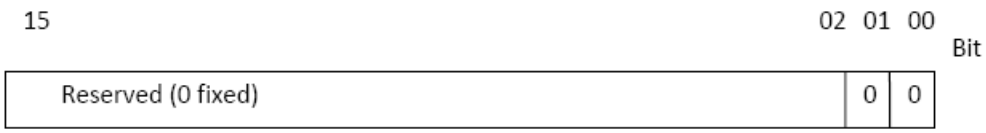
When [Control] bit is ON, the HMI will read and update [Action mode], [Start time], and [End time] values. For example, [Control address] LW-0, then updated bit is LW-Bit-00 (Bit 0: no action; 1: read times/action mode.)

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.



Bit 00: Status bit: Read operation completed. (0: reading or reading not started. 1: reading completed.)

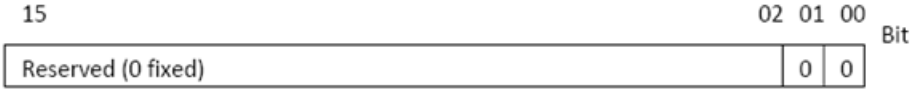
Bit 01: Error bit: Start or end time format incorrect. (0: corrected 1: error)

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.



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.

For example, if the control address is LW-3, then

LW-Bit-300 Sunday (0: not used, 1: used);
LW-Bit-301 Monday (0: not used, 1: used);
LW-Bit-302 Tuesday (0: not used, 1: used);
LW-Bit-303 Wednesday (0: not used, 1: used);
LW-Bit-304 Thursday (0: not used, 1: used);
LW-Bit-305 Friday (0: not used, 1: used);
LW-Bit-306 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)

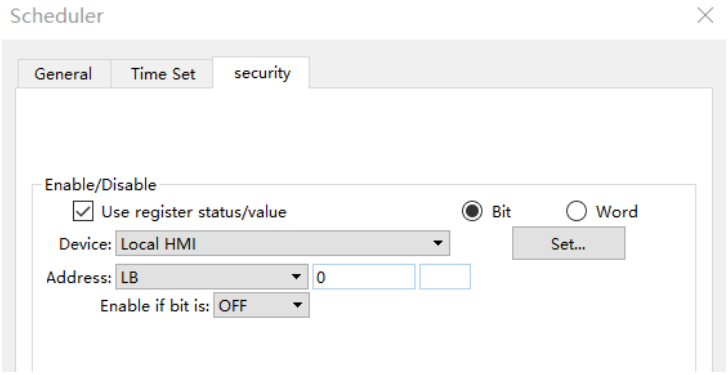
Hour: 0 – 23 Minute: 0 – 59 Second: 0 - 59

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.

Security 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

- The maximum number of Scheduler objects in a project is 64.
- 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.

12.35 Timer

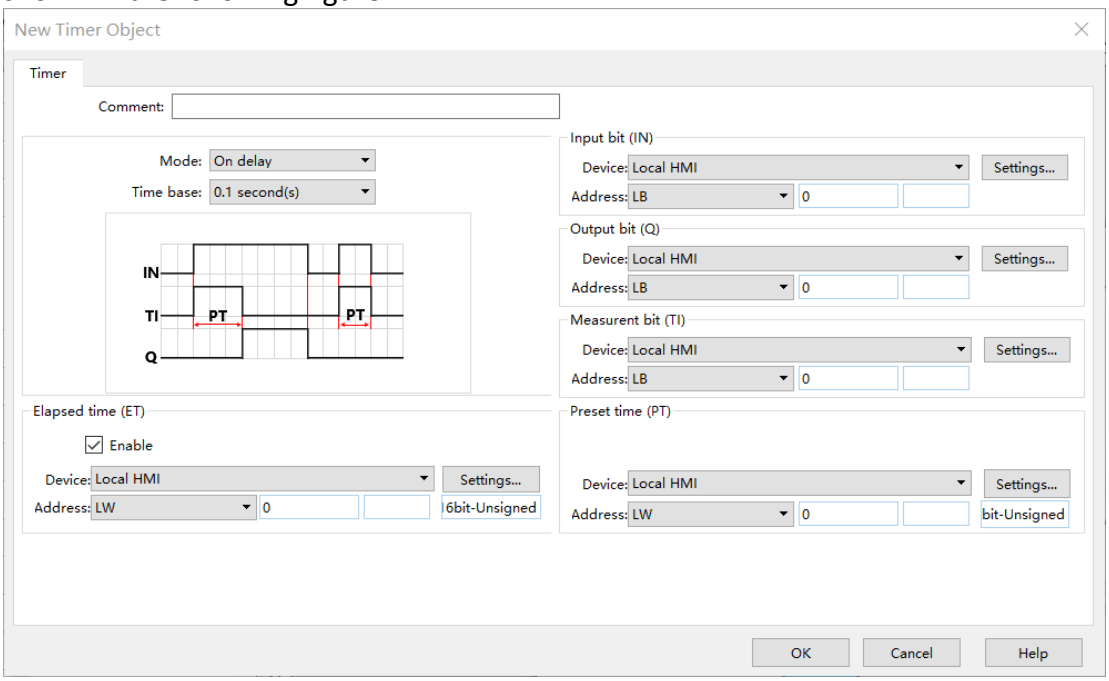
12.35.1 Overview

Timer object can be used to count the time in a specified mode. 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.

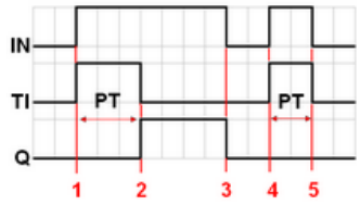
12.35.2 Configuration


Click [Object] » [Time-related] » [Timer] icon on the toolbar to open the property dialog box as shown in the following figure.

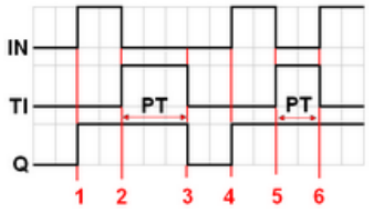


Mode description

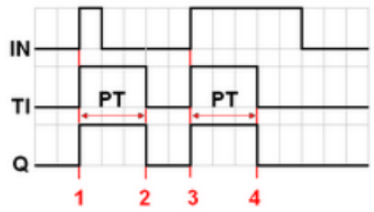
- *On delay*

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.
Description	
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.	

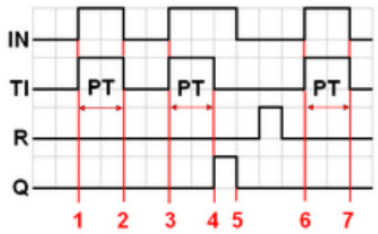
- *Off delay*

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 time. **Elapsed time (ET):** Displays the elapsed time.
Description	
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,.	

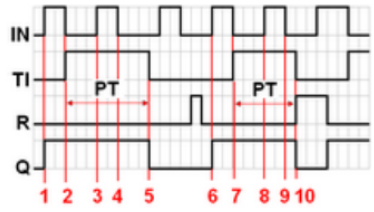
Pulse

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 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.
Description	
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.	

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.	

12.36 Media Player (in developing)

12.37 Vedio in (in developing)

12.38 Picture viewer (in developing)

12.39 PDF Reader (in developing)

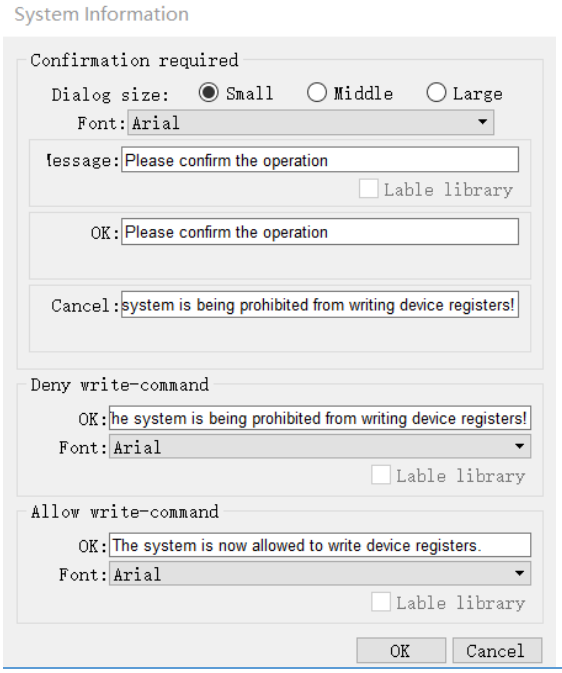
12.40 System Information

12.40.1 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.

12.40.2 Configuration

Click [Object] » [System Information] icon on the toolbar to open the setting dialog box.



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.

12.41 Recipe Database

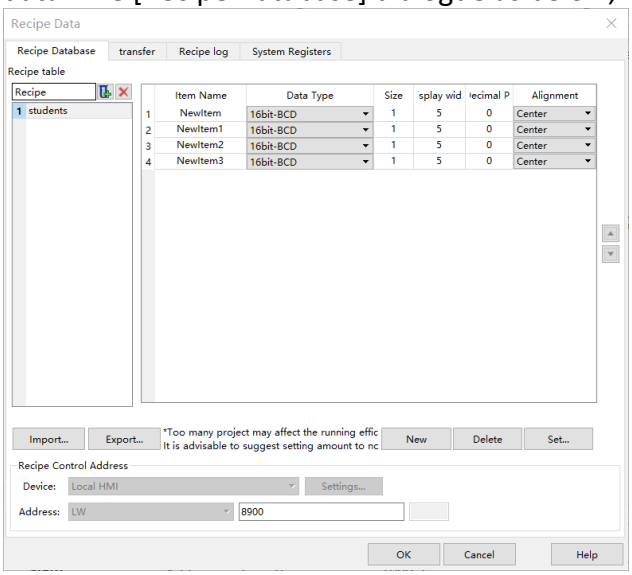
12.41.1 Recipe Database

Overview

[Recipe Database] to configure recipe database settings including recipe name, item name, data type…etc.

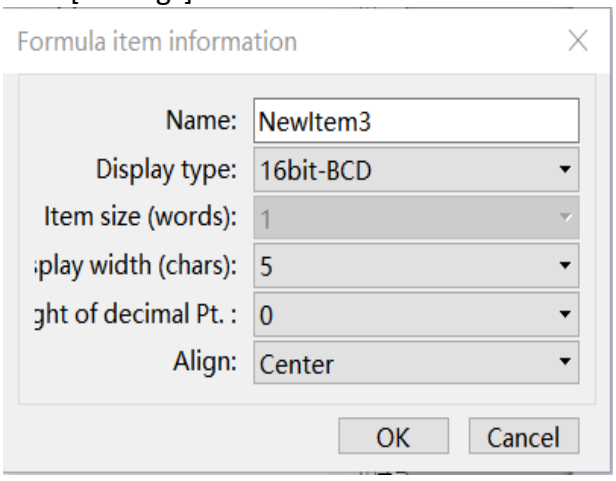
Configuration

Click [Data/History] » [Recipe Database] to define recipes, set transfer method and edit recipe data. The [Recipe Database] dialogue as below,



Setting	Description
Recipes List	Add or delete a recipe, the maximum is 100 recipes.
New	Add a new item, the maximum is 1000 items.
Settings	Configure recipe settings, see the following description.
Delete	Delete the selected items.
Import	Import recipe definition.
Export	Export recipe definition. The exported file will not contain recipe records.

Click [Settings]:



Setting	Description
Name	Enter recipe item name.
Display type	Setting item data type.
Item size (words)	Setting the size of the item.

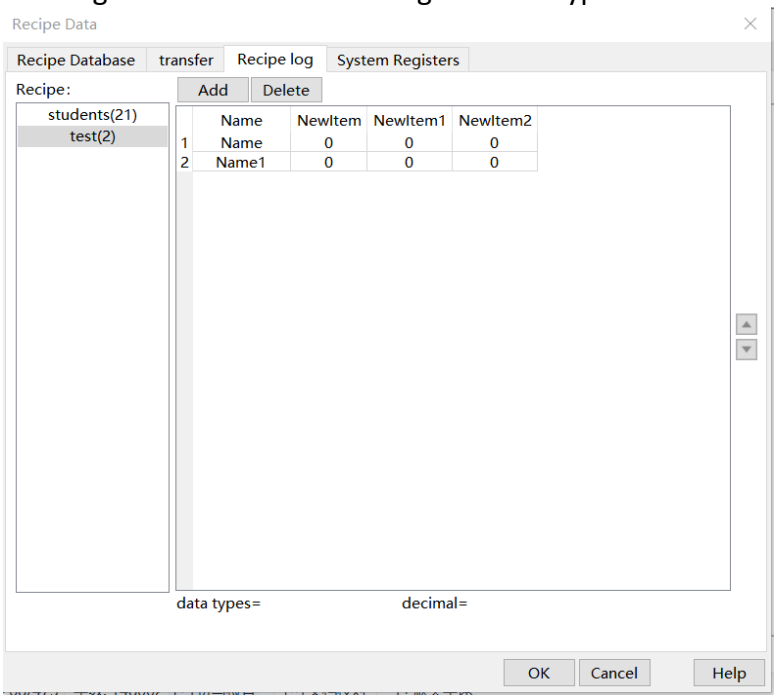
Display width (Chars)	Setting the number of characters of the item to be displayed.
Right of decimal pt.	Setting the decimal place when displaying data.
Align	Setting the alignment when displaying data. [Align left], [Align center], and [Align right] can be selected.

Note

- The maximum number of Recipe Database acceptable in a project file is 100 Recipe Databases. A Recipe Database can contain 1000 items.
- The maximum data length acceptable in a Recipe Database is 1024 words. Exceeding the limit can lead to compilation failure.

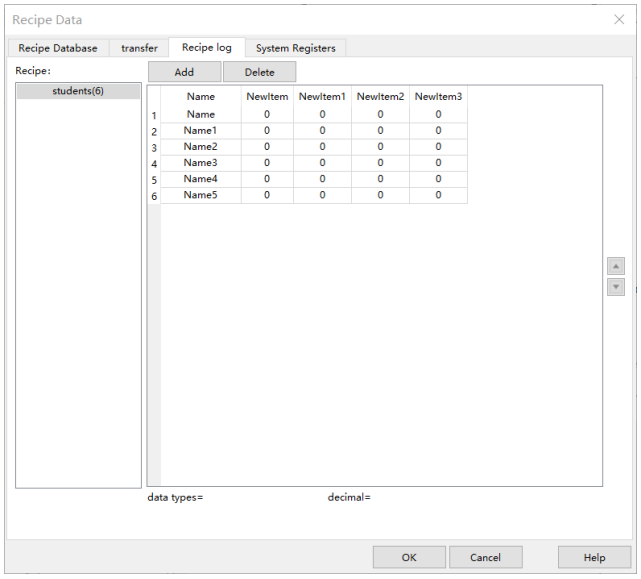
Recipe Log

After setting the Recipe Database, open [Recipe log], then Recipe created in the Definition tab can be found. Items shown on the right hand side come from item names added in Definition tab. Data of each item name can be configured in this tab following the data type defined in the Definition tab.



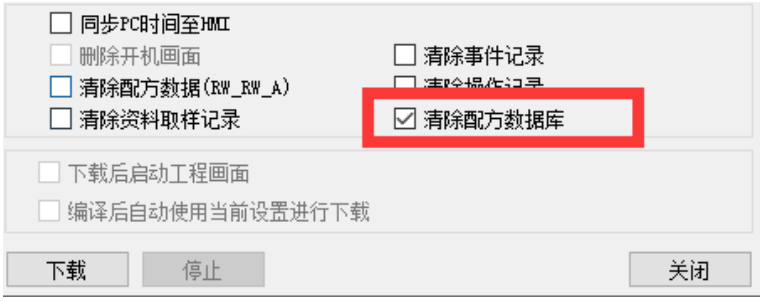
Setting	Description
Recipes	The recipes created in Definition tab. The number enclosed in brackets shows the total number of records in the corresponding recipe.
Add	Adds records into the recipe according to the defined data type.
Delete	Deletes the edited content.
Up/Down arrows	Moves the selected record upward / downward.

To define recipes according to the specified data type, click [Add] button above the record list to insert a new record and start editing each item. When click on the item, the item's data type will be shown under the record list. This helps users to fill in each item with appropriate value. Click [OK] to confirm and save the records.



Note

- Each recipe can hold a maximum of 10000 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.



12.41.2 Recipe Viewer

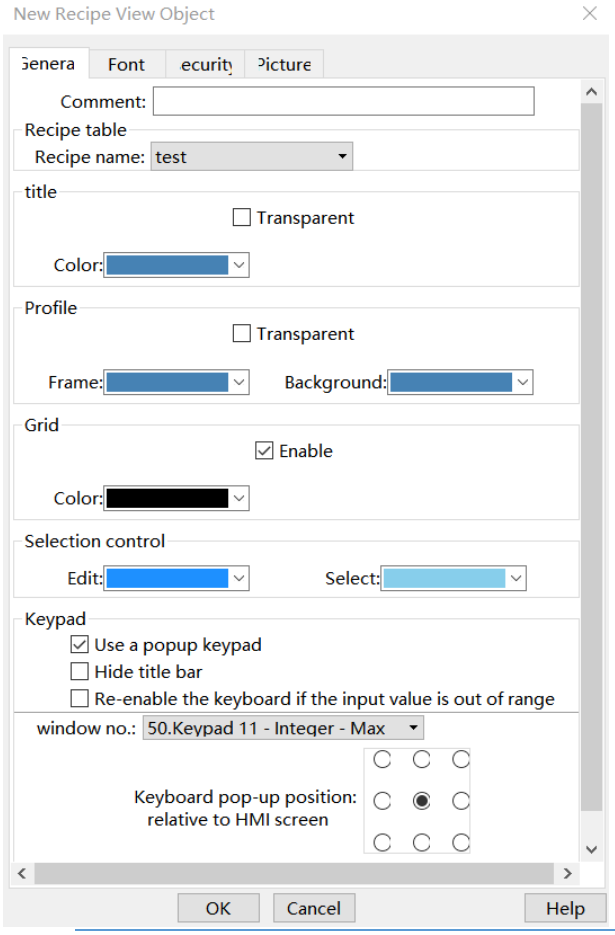
Overview

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.

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.

General Tab



Setting	Description
Recipe Table	Choose the recipe name or look for other recipes from the drop-down list.
Title	The item name assigned in [Data/History] » [Recipe Database]. **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	The dividing lines between columns and rows. **Transparent** If selected, the Grid row has no shading; the color selection is not available.
Selection control	Edit It selects the displayed color when click the specified cells. Select It selects the displayed color when click the specified rows.
Keypad	When double click the specified cell of Database, the keypad dialogue will pop-up and we can revise the data of recipe directly by it.

Note

- There are 4 system registers that can be used to view/update/add/delete recipe database, for example, the first control address is LW-8900, or LW-8900~LW-8903 (refer to Example 1 of 12.41.3 Recipe Transfer for detail), then the second control address should be LW-8904, and so on.

  Selection (LW-n)
  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。
  Count (LW-n+1)
  Show the number of records in current recipe.
  Command (LW-n+2)
  Enter certain value will send command to the selected record.
  Enter “1”: Add a new recipe record.
  Enter “2”: Update the selected recipe record.
  Enter “3”: Delete the selected recipe record.
  Enter “4”: Delete all recipe records.
  Enter “5”: Write the selected recipe record to PLC.
  Enter “6”: Update the recipe record selected from PLC.
  Enter “21”: Add a new recipe record before the selected record.
  Enter “22”: Add a new recipe record after the selected record.
  Result (LW-n+3)
  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 (10000 records), no new records can be added.
  Display “16”: Another command is being executed.

12.41.3 Recipe Transfer

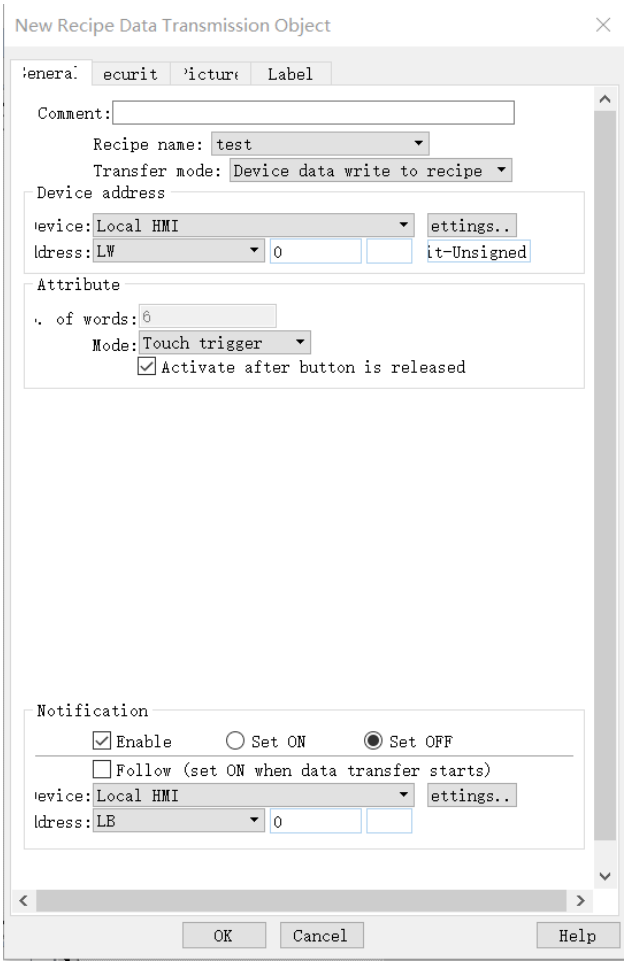
Overview

[Recipe Transfer] can be used to read recipe data to a specified address or write data from a specified address to a recipe.

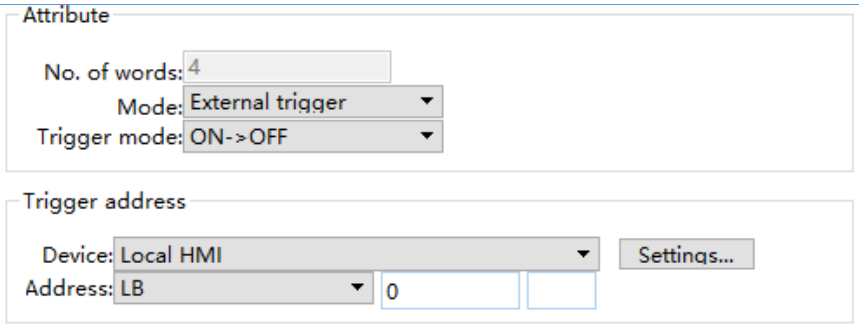
Configuration


click [Data/History] » [Recipe Transfer] button on the toolbar, the [Recipe Data Transmission Object] dialog window will pop-up. After setting various attributes correctly, click "OK" key to add a new [Recipe Transfer] object.

General Tab



Setting	Description
Recipe Name	Select the recipe to transfer, click the drop-down list to select the recipe you want to transfer.
Transfer mode	Device data write to recipe. Write the data from the specified address to recipe. Recipe date write to device
Device address	Specified address.
Attributes	**Touch trigger** Touch the object to activate Recipe 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.



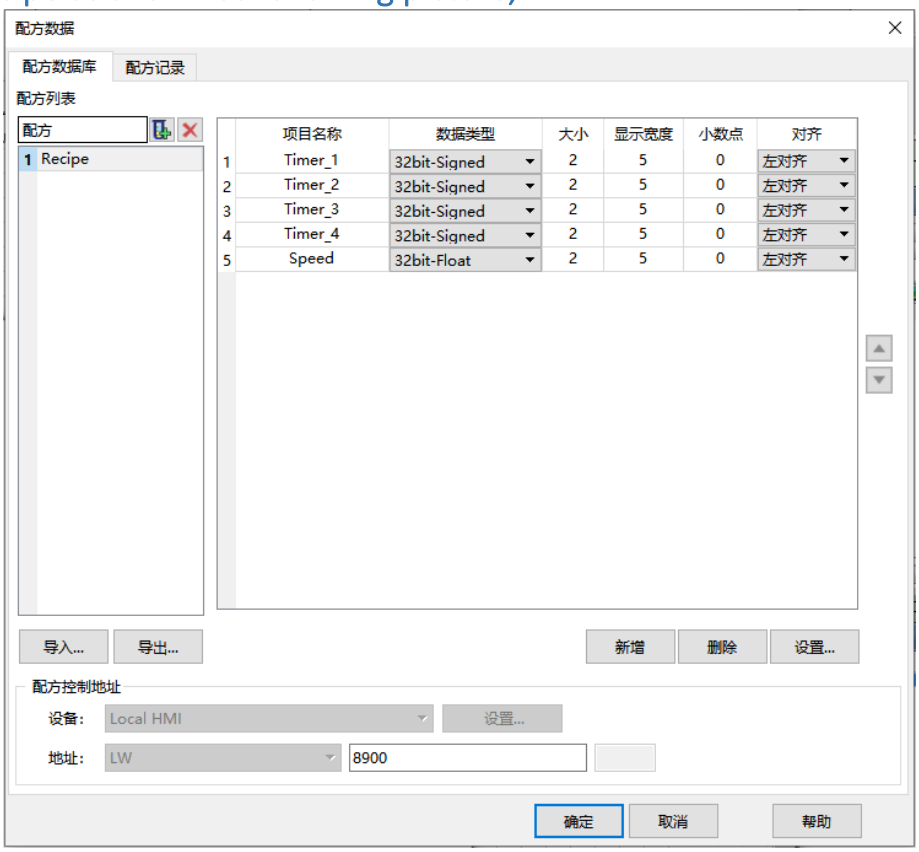
Notification	When enabled, the system will set the designated address ON or OFF when it's ready for data transfer. **Follow** The notification bit will reset to its original state once the system starts data transfer

Example 1

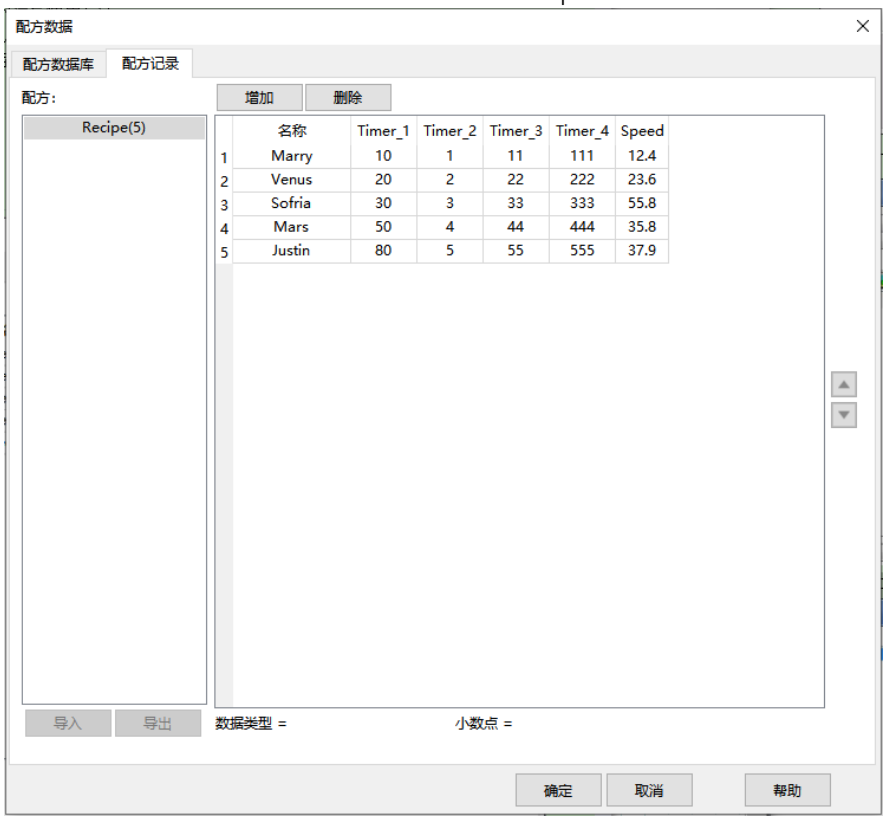
The following example demonstrates an easy way to use [Recipe View], [Recipe Transfer] and a recipe database.

In the example, a recipe database will be created, using "Recipe View" to view and select recipe data, and then using "Recipe Transfer" to transfer recipe data. When you click any record in the Recipe View, the [Selection] and corresponding register values will change. Finally, you can use [Command] to edit and modify the recipe database.

1. Creat a Recipe as shown as following picture,



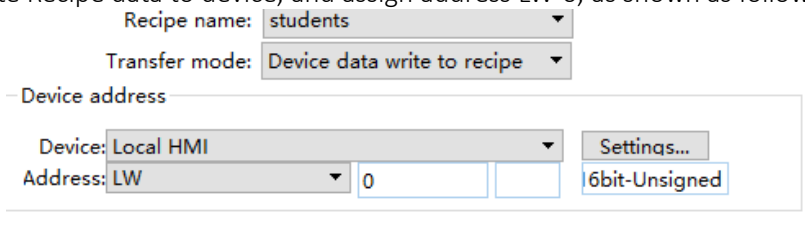
2. In [Recipe log] tab creat a number of records as shown as below picture



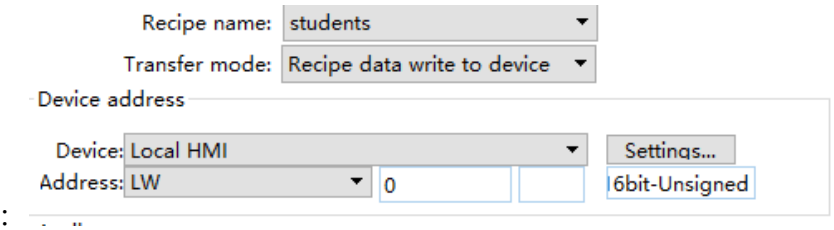
3. Create a Recipe View object and use the recipe database created in the preceding steps.
4. Create 4 Numeric objects using registers “Selection”, “Count”, “Command”, and “Result”.

5. Create two [Recipe Transfer] objects, set one ‘s Transfer mode as Write data from device to Recipe and another Transfer mode as Write Recipe data to device, and assign address LW-0, as shown as following pictures.

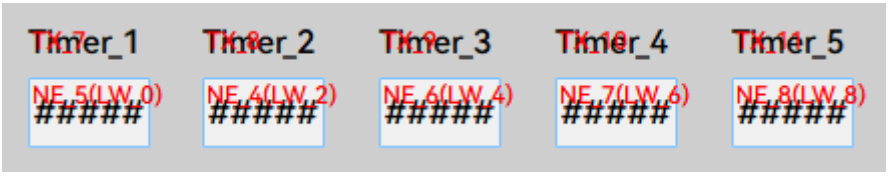
Set:



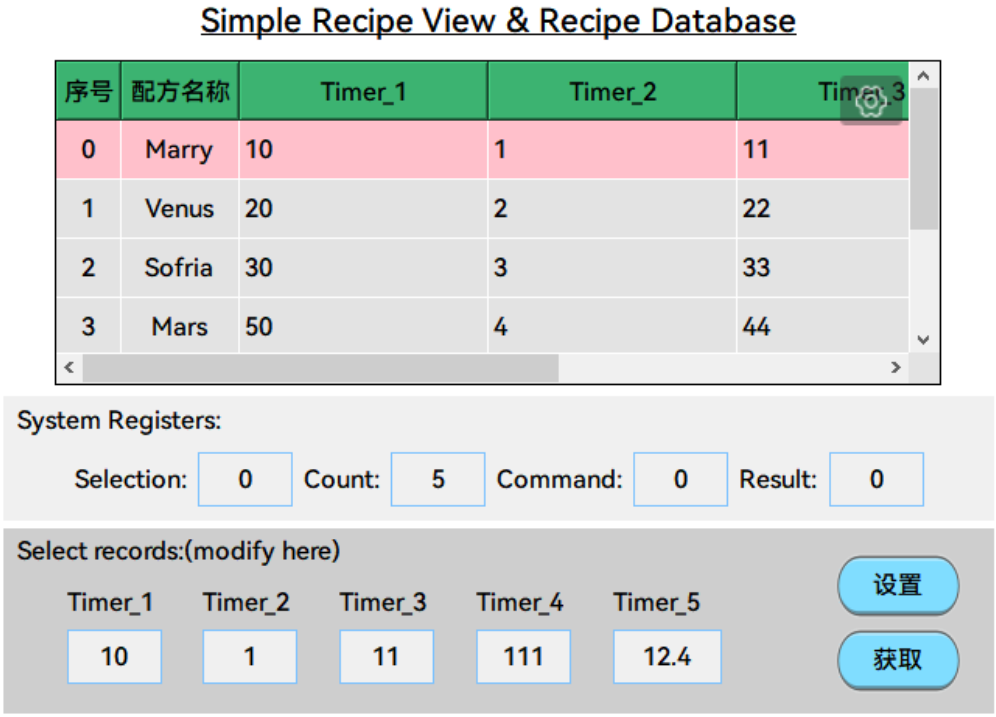
Receipt:



6. Create LW-0~LW-8, ans use 4 Numeric objects to store the corresponding data of Recipes, as below picture.



7. The project is then completed.
8. 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.



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”.

12.42 Operation Log

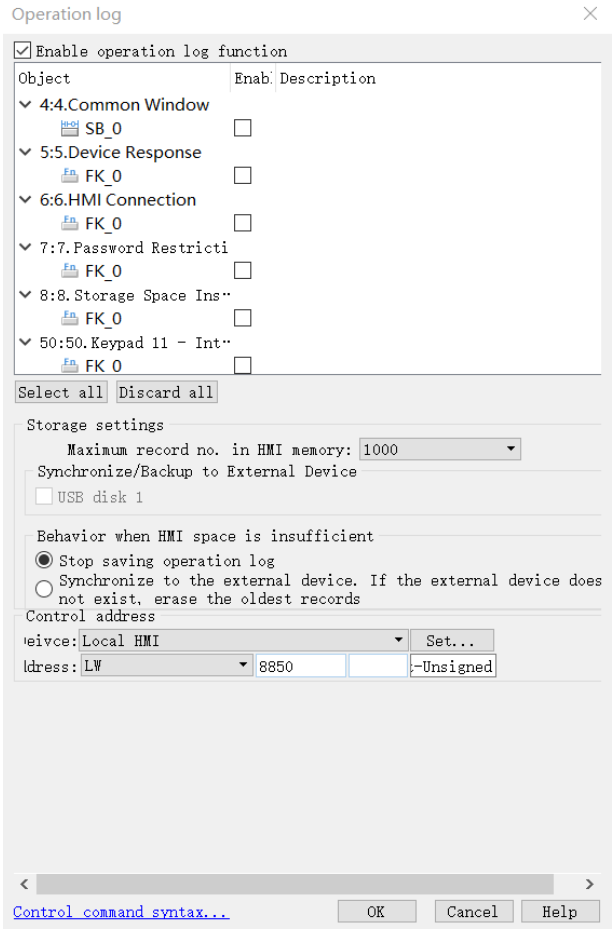
12.42.1 Operation Log Setting

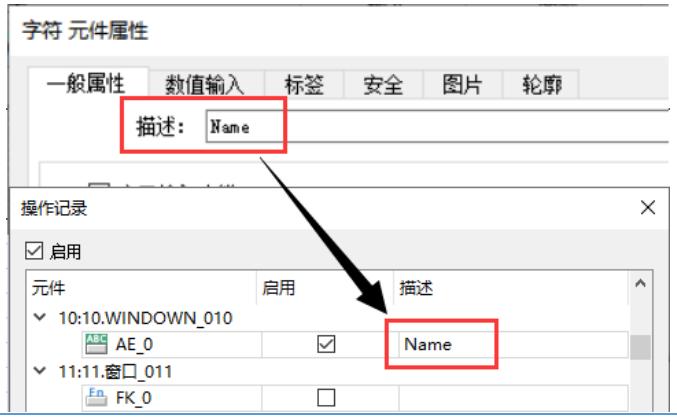
12.42.1.1 Overview

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. Operation log, if configured properly, may also be used to satisfy the requirement for audit trail and electronic signature.

12.42.1.2 Configuration

Select the objects to be recorded. Click [Data/History] » [Operation Log Setting], and then select [Enable operation log function] check box.



Setting	Description
Object	With Operation Log function enabled, user login/logout activities and the objects that can be recorded are listed in the setting dialog box sorted by window numbers.
Enable	The selected objects are recorded by Operation Log.
Description	The description of the object. The selected label will be shown in this column, and if Label Library is not used, the comment in the object's settings will be shown as below. 

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 command 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. (6): Enable Operation Log. (7): Disable Operation Log. (8): Use history data stored in USB disk after changing HMI. (9): Use history data stored in SD card after changing HMI. 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 installation directory: ParsePro\[Project name]\ftp\operationlog\operationlog.db
- Triggering Macro with a Set Bit object generates two records, the triggering of bit and the triggering of Macro.

12.42.2 Operation Log View

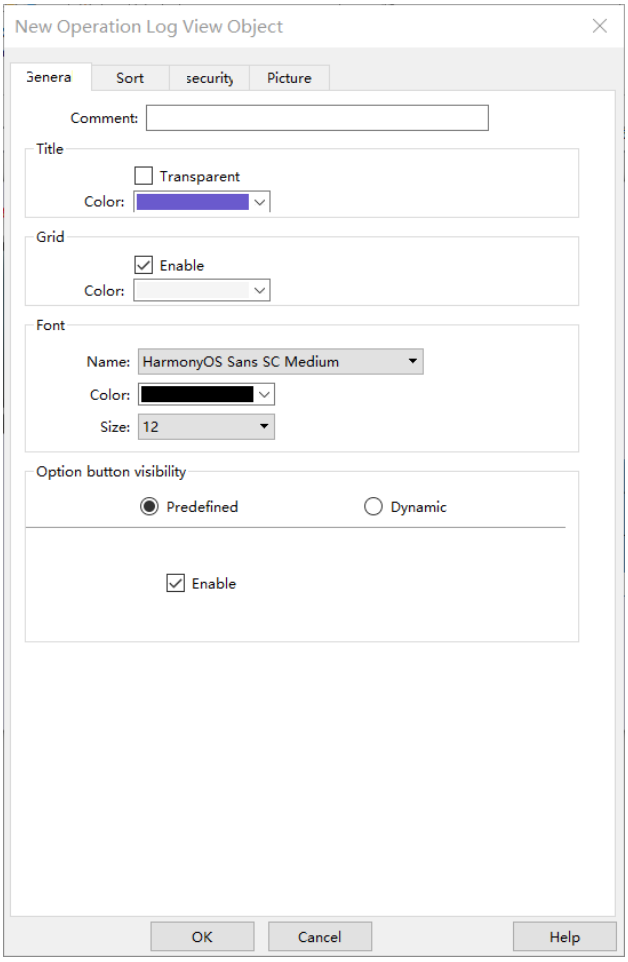
Overview

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

Configuration

Before using Operation Log View, please follow the steps described in the preceding part to finish Operation Log Settings. Click [Data/History], and then click [Operation Log View].

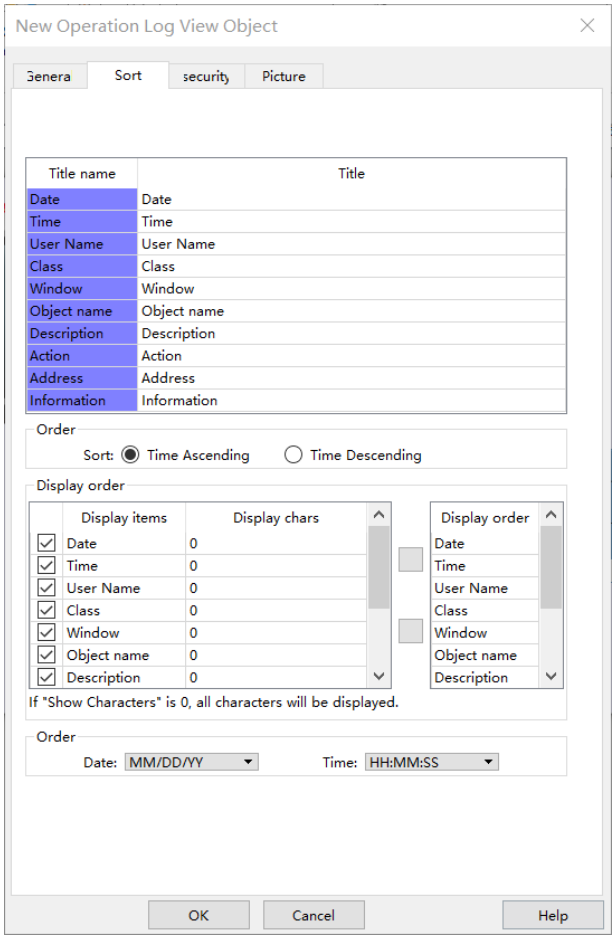
General Tab



Setting	Description
Title	If Transparent option is 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	The dividing lines between columns and rows. **Enable** If not selected, the Grid row has no shading; the color selection is not available.
Selection Control	It selects the displayed color when click the specified rows.
Font	To set the font, size & color of text of [Operation Log View]

Sort Tab

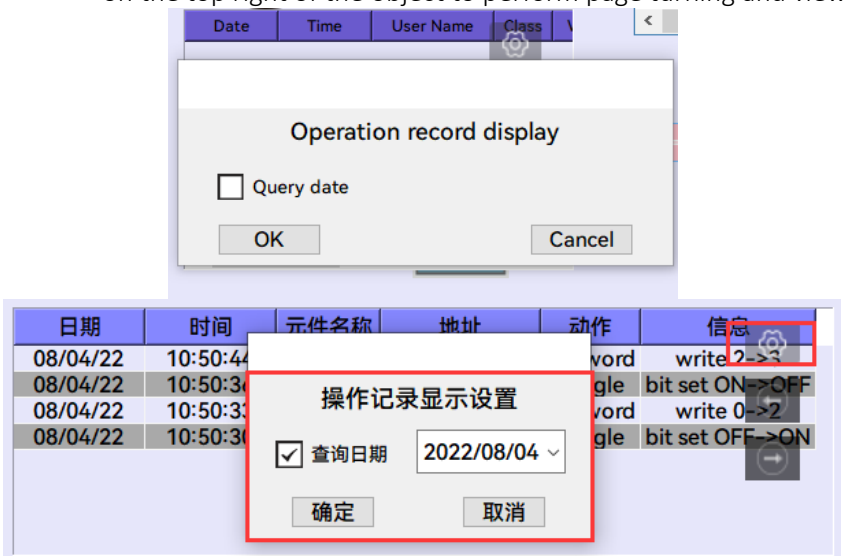


Setting	Description
Title	Sets the title name of the columns displayed in Operation Log View object.
Sort	Sorts the records in time ascending or descending order.
Display order	Sets the order of the displayed item. If [Display chars.] is 0, all characters are displayed.
Date / Time	Sets the format of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

Note

- Click the icon  at the top right of the object to set the query date. If none is checked, all operation records will be displayed.
- [Operation Log view] has been optimized for pagination, with 20 pieces of data displayed on each page. Users

can click  on the top right of the object to perform page turning and viewing operation logs.



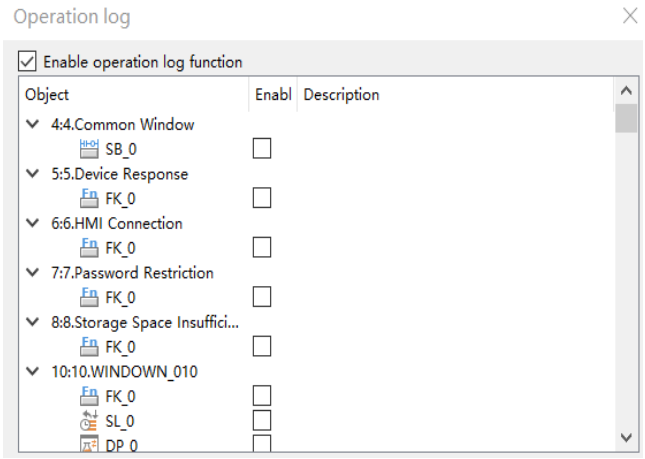
12.42.3 Operation Log Printing (in developing)

Example 1

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

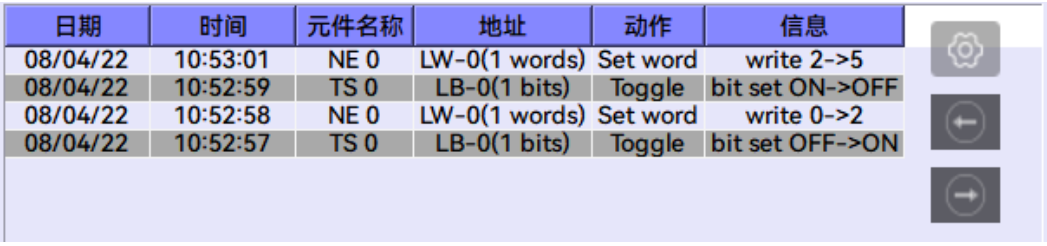
1. Create a Toggle Switch object and a Numeric object on window number 10.

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.



日期	时间	元件名称	地址	动作	信息
08/04/22	10:53:01	NE 0	LW-0(1 words)	Set word	write 2->5
08/04/22	10:52:59	TS 0	LB-0(1 bits)	Toggle	bit set ON->OFF
08/04/22	10:52:58	NE 0	LW-0(1 words)	Set word	write 0->2
08/04/22	10:52:57	TS 0	LB-0(1 bits)	Toggle	bit set OFF->ON

12.43 File Browser (in developing)

12.44 Import/Export (in developing)

12.45 2D Barcode Display

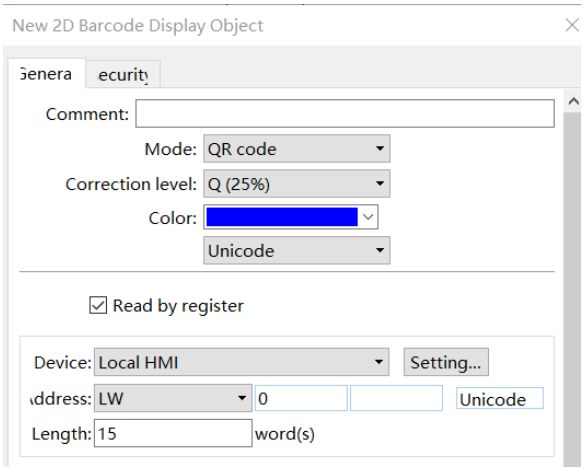
12.45.1 Overview

The 2D Barcode Display object transfers the information from the read address into QR Code or Aztec Code.

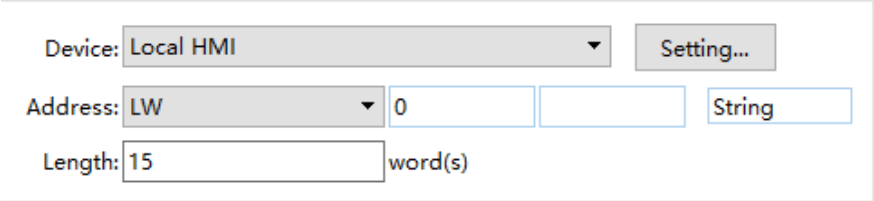
12.45.2

Click the 2D Barcode Display icon on the toolbar to open the property dialog box. Set up the properties, press OK button, and a new 2D Barcode Display object will be created.

General Tab



Setting	Description
Mode	Supports QR code and Aztec code.

Correction level	2D barcodes have error correction capability to restore data if the barcode is dirty or damaged. **QR code** Four correction levels are available: L, M, Q, and H. The data restoration rate is listed below. (The data restoration rate for total codewords. Codeword is a unit that constructs the data area.) **Correction Level** L 7% M 15% Q 25% H 30% **Aztec code** Aztec code supports error correction levels from 5% to 95%. Specifying a higher correction value results in a larger printed symbol and increases accuracy.
Color	Set the 2D barcode color.
Unicode	By default, the 2D barcode is generated via ASCII encoding. If this check box is selected, the 2D barcode is generated via UNICODE encoding. For characters that are not ASCII defined English alphabets or numbers, for example, Chinese or Korean characters, please select the Unicode check box.
Read by register	Whether the data come from Register 是否取自寄存器 If the option isn't checked, then can set the STRING Value directly. If it is checked, then specify the read address of Register. 

12.46 Barcode Scanner (in developing)

12.47 String Table (in developing)

12.48 Database (in developing)

12.49 Table

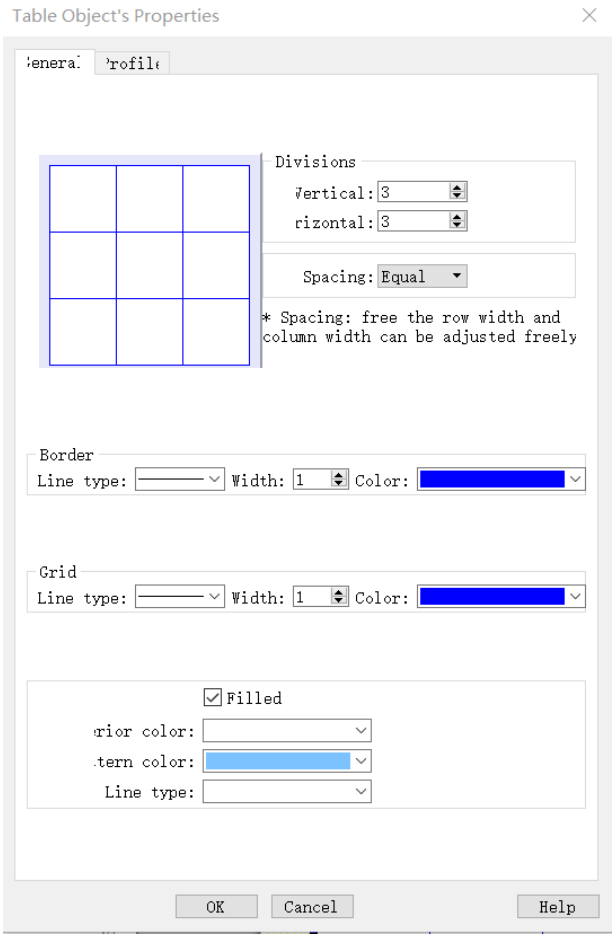
12.49.1 Overview

Table object allows users to draw a table in the project window and customize the border, grid, and pattern of the table.

12.49.2 Configuration

 Click the Table icon on the toolbar to create a Table object. Or, click [Object] » [Table] in the menu.

General Tab



Setting	Description
Preview Window	Displays the settings result.
Vertical	Sets the number of columns in the table. Range: 1~255
Horizontal	Sets the number of rows in the table. Range: 1~255
Spacing	The available options are [Equal] and [Free]. When [Free] is selected, the user can manually adjust the column width or row height in the editing window.
Border	Sets the type, width, and color of the border. [Line width] setting is available only when Solid line type is selected. The range of width is 0~8. The line will become invisible when the width is set to 0.
Grid	Sets the type, width, and color of the grid. [Line width] setting is available only when Solid line type is selected. The range of width is 0~8. The line will become invisible when the width is set to 0.
Fill	Sets the pattern style and color.

12.50 VNC Viewer (in developing)

12.51 Contacts Editor (in developing)

Chapter 13 Shape Library and Picture Library

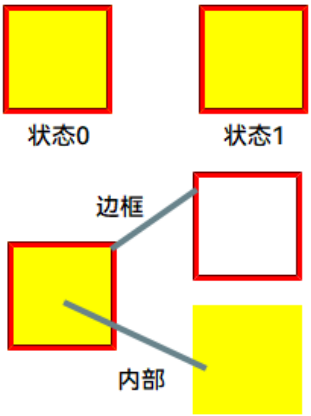
13.1 Overview

SmartManager 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 .exhmi project file. Pictures in [Library] mode will be stored in SmartManager 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 “8 Object General Properties”.

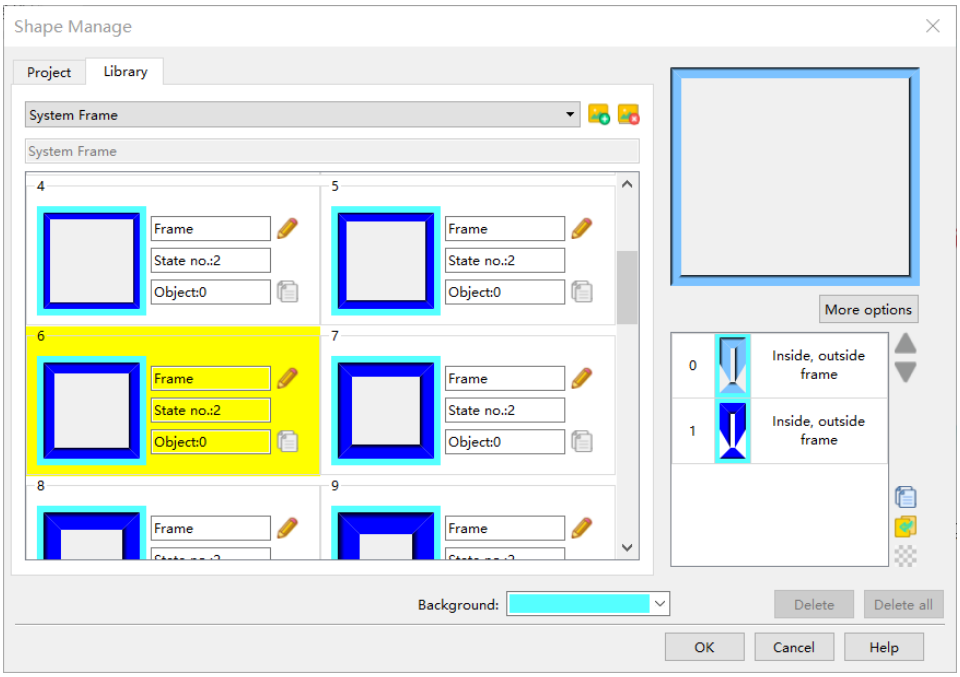
13.2 Build Shape Library

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.



13.2.1 Shape manager

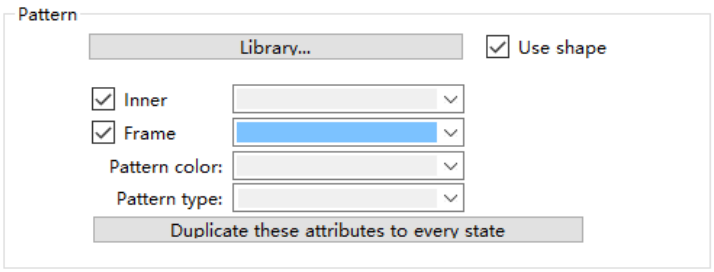
An object can use frame, inner or both. Click [Project] » [Shape], and the [Shape manager] dialog box appears.



Setting	Description
Project	The Shape edited here will be saved in .exhmi Up to 1000 Shapes can be added.
Library	The Shape edited here will be saved to the library directory on PC and will not be saved to .exhmi 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.
Delete library	Exclude currently selected library.
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.
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.
Delete all	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

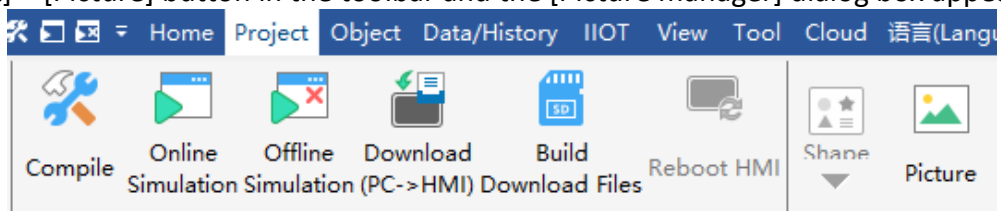
■ The color of [Inner] and [Frame] can be selected in Shape Library. The selection of [Pattern Style] is only available in System Frame / System Button Library.

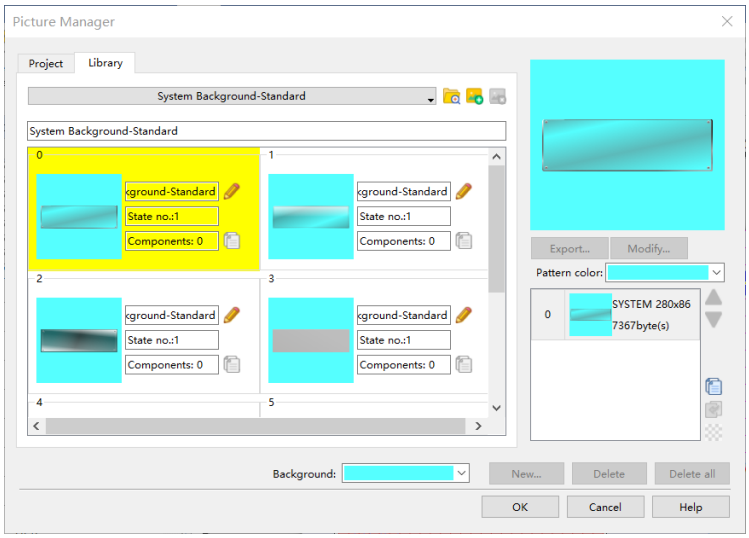


13.3 Building Picture Library

13.3.1 Picture manager

Click [Project] » [Picture] button in the toolbar and the [Picture manager] dialog box appears.





Setting	Description
Project	The Picture edited here will be saved in .exhmi Up to 1000 Pictures can be added.
Library	The Picture edited here will be saved to the library directory on PC and will not be saved to .exhmi project file.
New library	Include existing .plb picture 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.
Delete library	Exclude currently selected library.
Browse libraries	Search for path and view the pictures in the path.

Background	Select and preview 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.
Delete all	Delete all the Pictures listed here.
OK	Confirm to save the edited Picture.
Cancel	Cancel the editing event.
Help	Open help files.

Chapter 14 Label Tag Library and Multi-Language

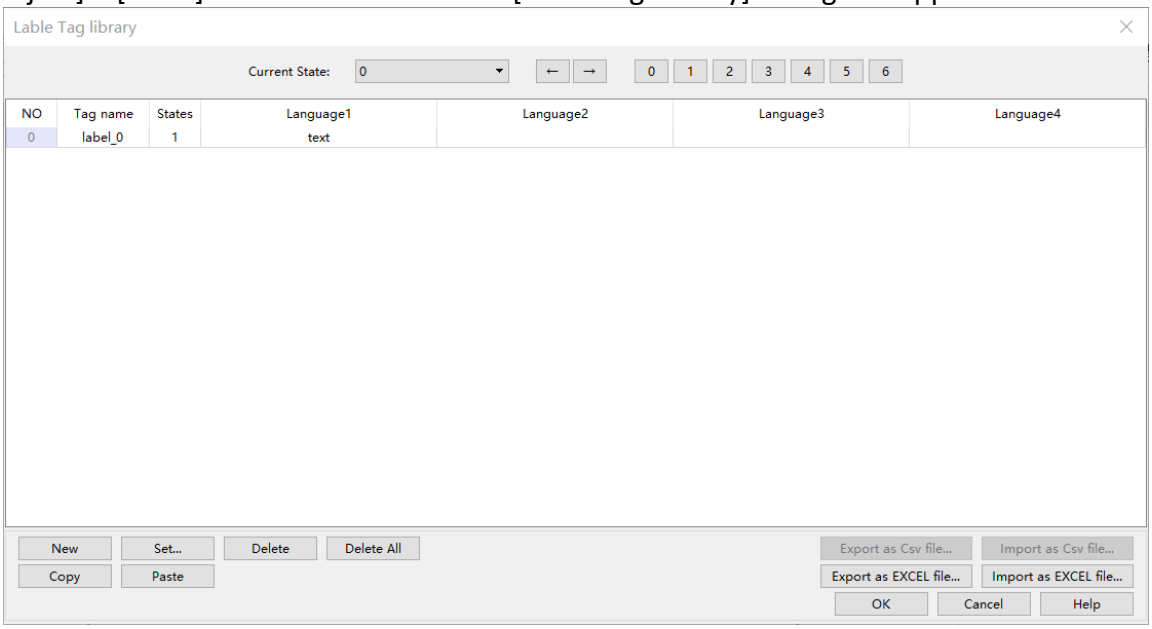
This chapter explains how to build and use Label Tag Library.

14.1 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. SmartManager Pro supports up to 24 different languages simultaneously. This chapter will explain how to create and use the Label Tag Library.

14.2 Label Tag Library Manager

Click [Project] » [Label] on the toolbar and the [Label Tag Library] dialog box appears.



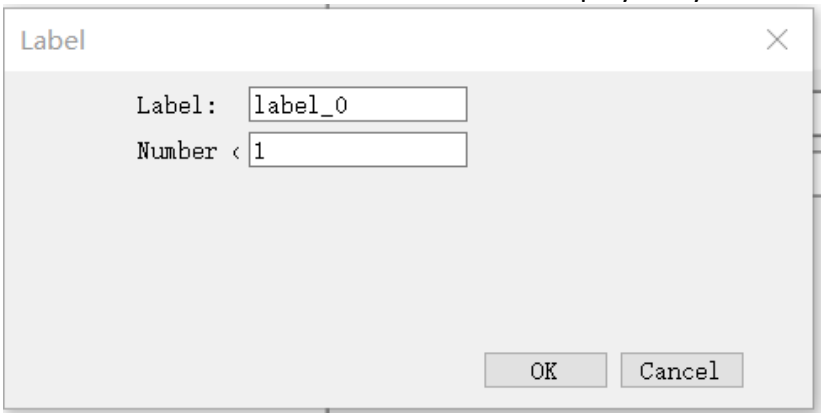
Setting	Description
Current State	Indicates the current state. Each Label has a maximum of 256 states (state no. 0 ~ 255). The state no. is determined by [Language no.]. If less than 3 languages are used, the maximum state no. is 256. If more than 4 languages are used, divide 768 by the language number 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.
Export as Csv file	Saves all Labels in .csv format.

Import as Csv file	Loads the existing .csv file to the Label Library
Export as EXCEL file	Saves all Labels in .xlsx format.
Import as EXCEL file	Loads the existing .xlsx file to the Label Library

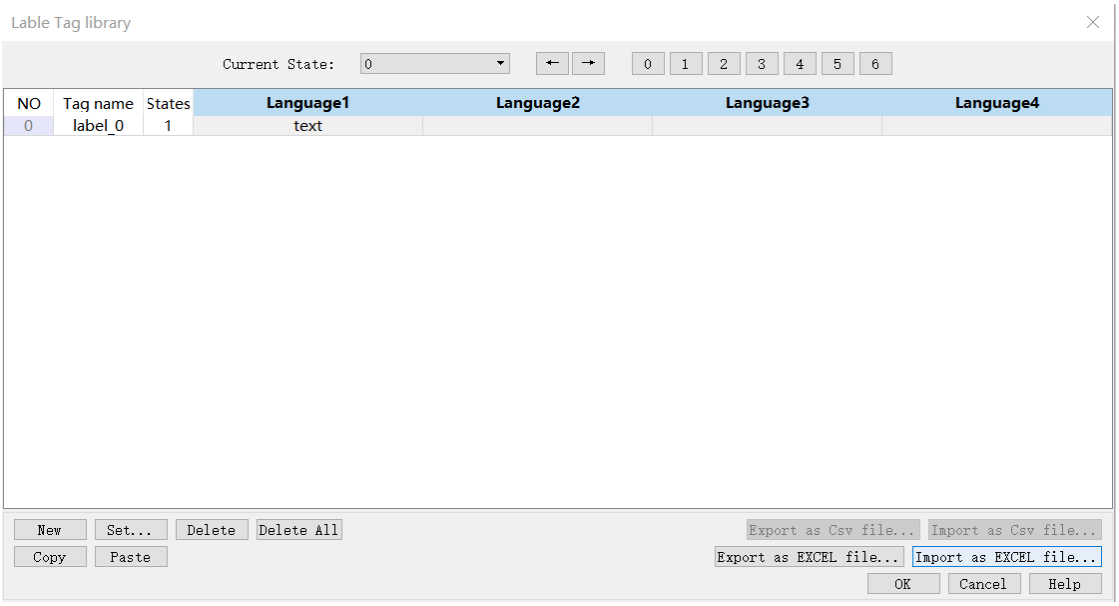
14.3 Steps to create Label Tag Library

Please follow the steps to create a Label Tag Library.

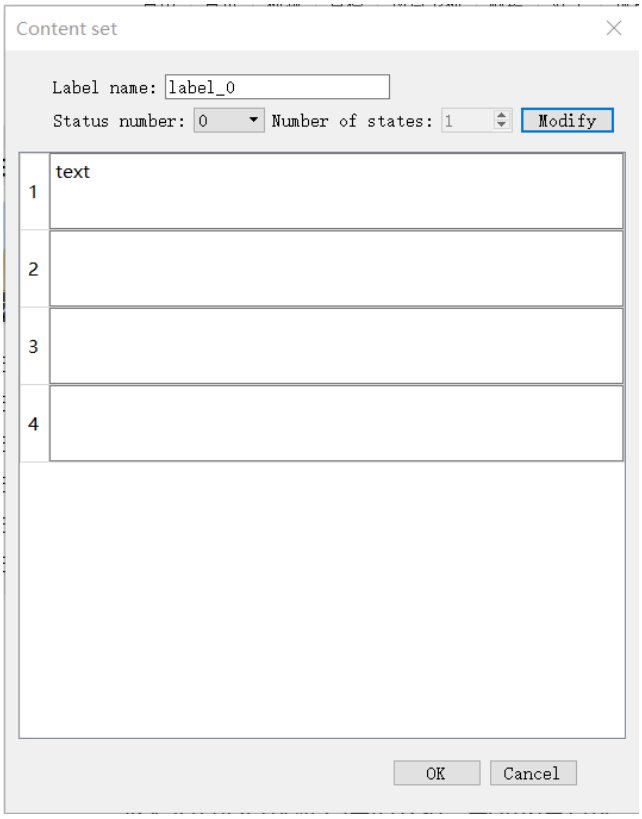
1. From the Library menu, click [Label]. The Label Tag Library dialog box appears. Click [New] to specify the name of the Label and the number of states to be displayed by this Label.



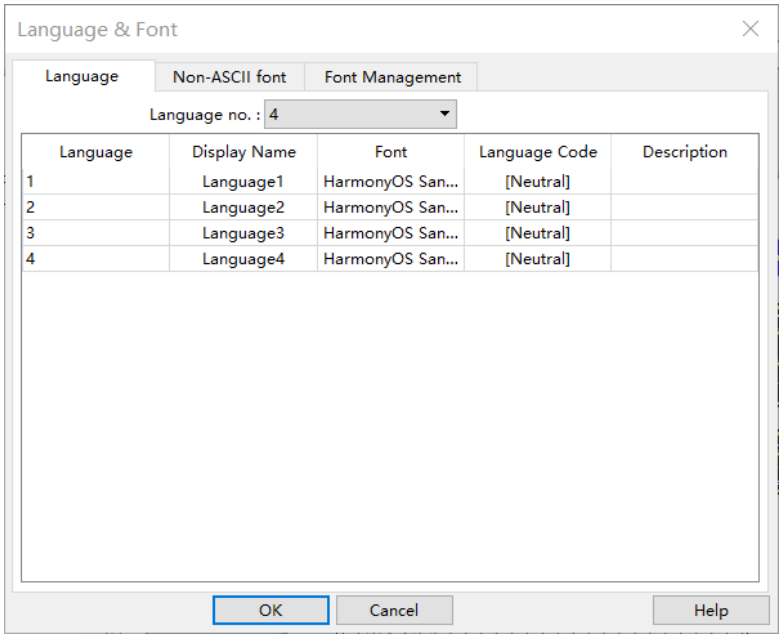
2. Click [OK] and a new label is added to the Label Tag Library. Select the label and click [Settings] to edit its content.



3. Edit the corresponding language content.



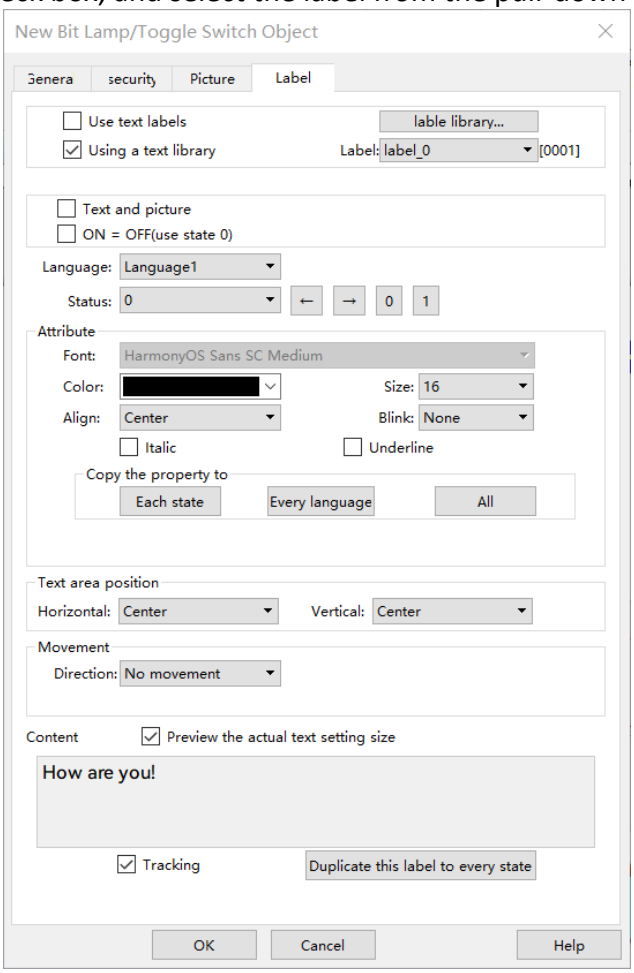
4. Click "Project"»"Language&Fonts"»"Language", you can set the language fonts contained in existing labels. Different fonts can be selected for different languages, and also can attach the comments for each font. The "Language Code" is used to set the language for event push when the "Push Notification" function is enabled in the Event Log on the APP.



14.4 Using Label Tag Library

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].



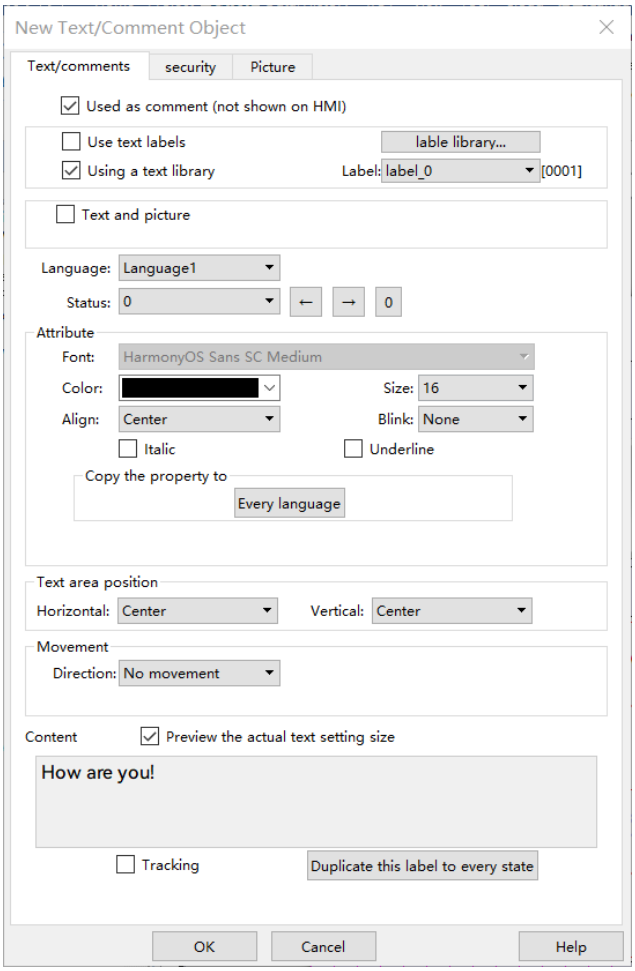
When a tag is selected, the content of the selected tag is shown in the [Content] field in its corresponding font style. Please note that from Language 2 to Language 24 can only be set the Font [Size], the others such as [Color], [Align], [Blink], etc. will follow the settings of Language 1.

14.5 Using of Multi-Language

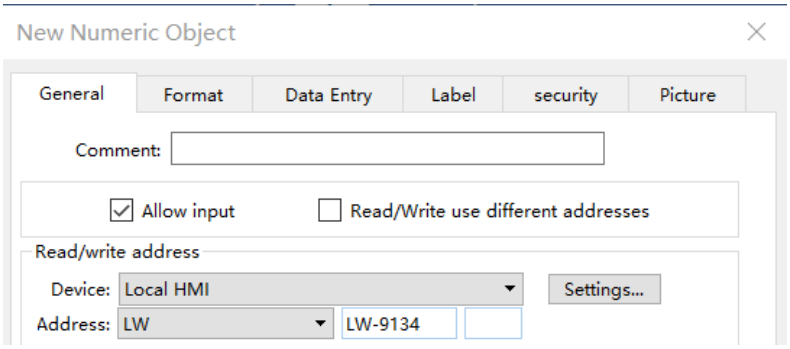
When displaying the texts in multiple languages, the system register "[LW-9134]: language mode" should be used too.
The value of [LW-9134: language mode] is ranged from 0 to 23. Different values correspond to different languages.
If not all languages are selected to compile and download, [LW-9134] will work differently.
For example, user defines 5 different languages in the Label Tag 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.

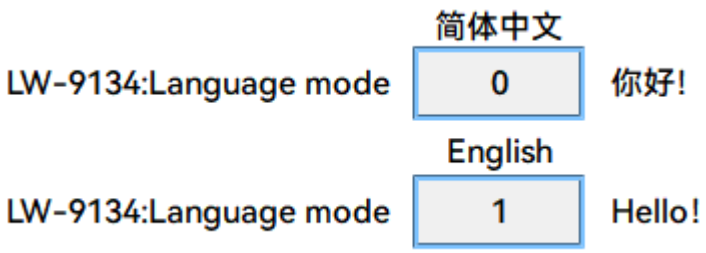
1. Create a Text/Comment object and select [Use label library] checkbox.



2. Create a Numeric Input Object and set the Read/Write Address as the system register [LW-9134].



3. The simulation is shown as followed: If the value of [LW-9134] is changed, the content of the Text object will be changed.



Chapter 15 Building & Using of Address Tag Library

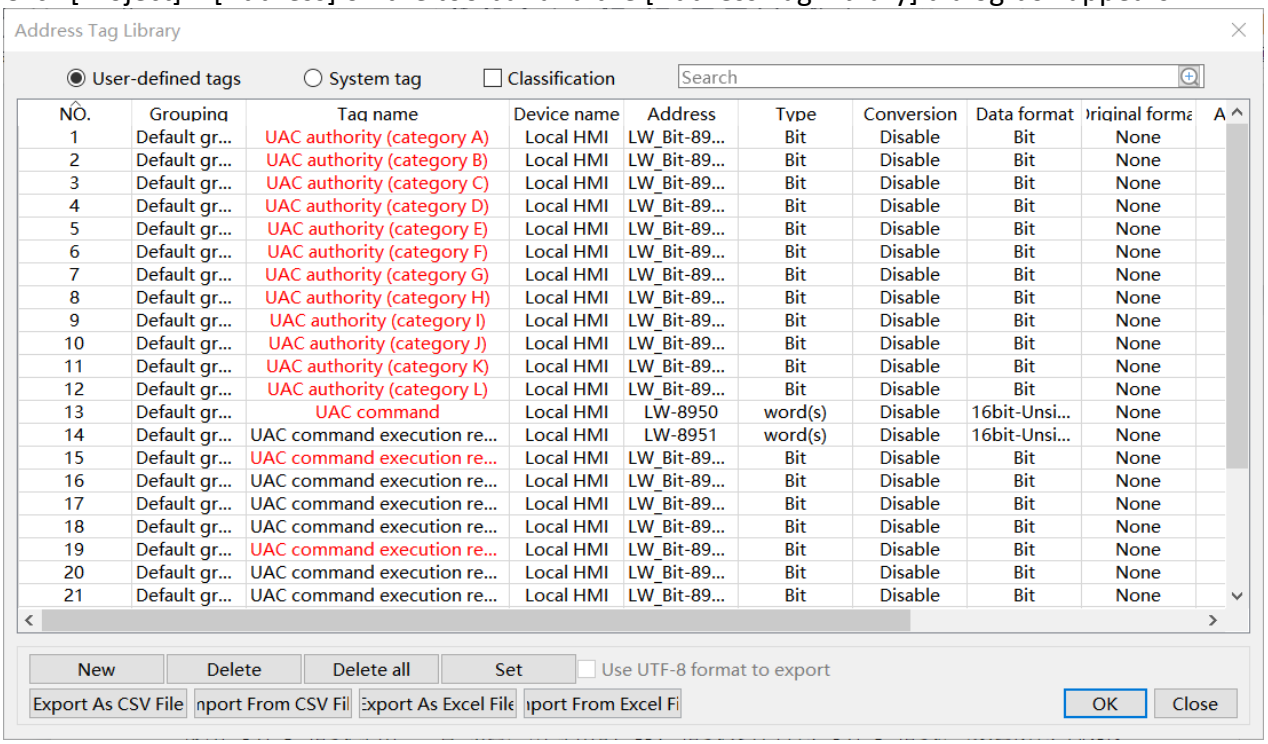
This chapter explains how to build and use Address Tag Library.

15.1 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.

15.2 Building Address Tag Labrary

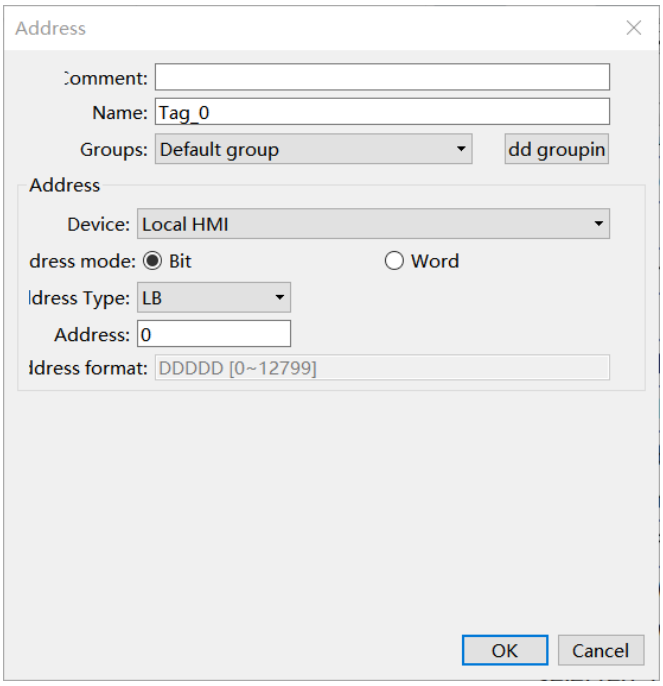
Click [Project] » [Address] on the toolbar and the [Address Tag Library] dialog box appears.

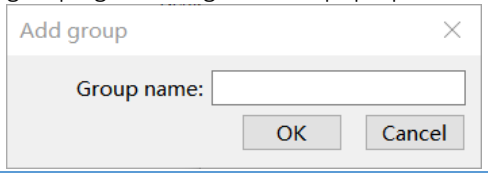


Setting	Description
User-defined tags	Displays user-defined address tags.
System tag	Displays system registers. The registers listed cannot be deleted or changed.
New	Adds a new address tag. Please see the steps next page
Set	Sets the selected address tag.
Export as Csv file	Saves all current address tags as .csv file.
Import as Csv file	Loads the existing .csv file of address tag to the current project.
Export as EXCEL file	Saves all current address tags as .xls file.

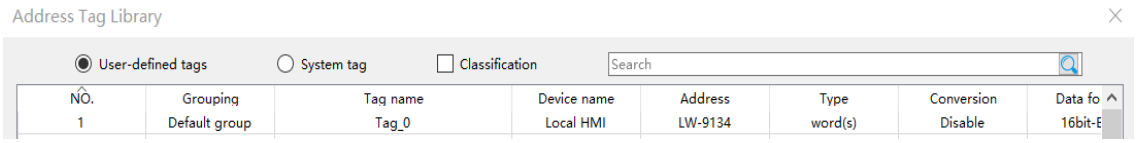
Import as EXCEL file	Loads the existing .xls file of address tag to the current project.
Use UTF-8 format to export	If selected, the .csv file will be exported in UTF-8 format. If not selected, in ANSI format.

1. Click [New] and set the relevant properties.



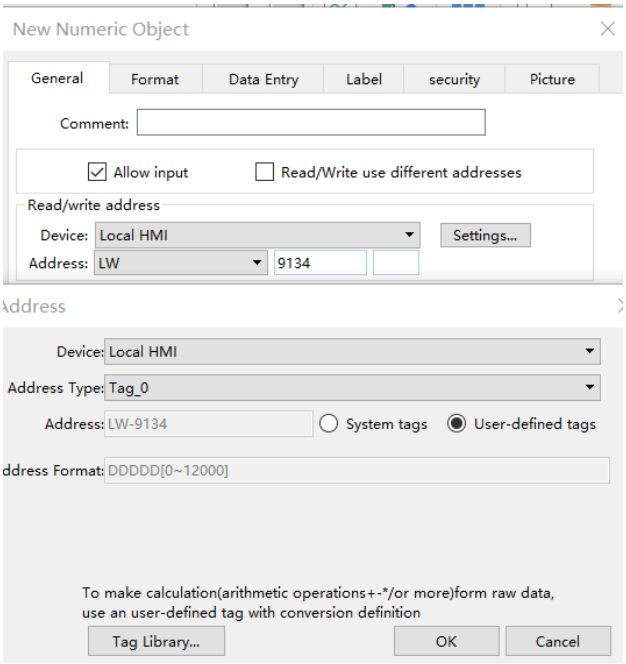
Setting	Description
Comment	The information about the address tag.
Name	The name of the address tag.
Groups	To set groups for user to classify & store Address Tags. The right button is "Add grouping". Clicking on it will pop-up the "Add group" dialogue as shown as below, 
Device	As defined in [System Parameter Settings] » [Device list].
Address mode	The tag address type; select [Bit] or [Word].
Address Type	The available address types depend on [Device] and [Address mode].
Address	Address of the tag.
Adress format	If select [Word] in [Address mode], the data format can be specified.
Conversion/Calculation (Use macro subroutine)	When enabled, the data format that the address tag will be converted into can be specified. Macro subroutines can be selected to do read/write conversion
Read / Write conversion	In Macro Subroutine mode, select the macro subroutine to do read/write conversion. The macro subroutine can only be selected when the data format is identical to the one in the macro subroutine.

2. Click [OK], a newly added tag can be found in the [User-defined tags] library.

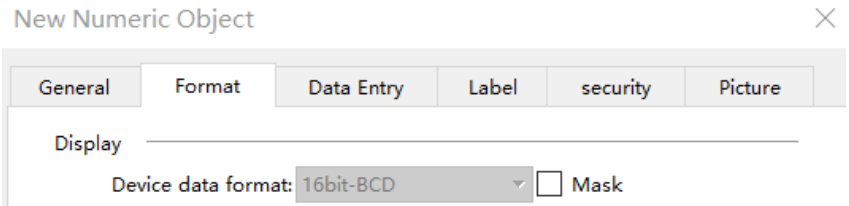


15.3 Using Address Tag Library

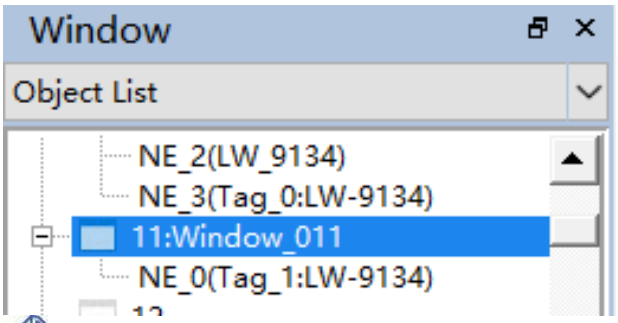
1. Create a tag in Address Tag Library.
2. Create an object, select [General] » [Device].
3. Click [Settings] to finish the settings.
4. Select [User-defined tag] check box.



5. In [Address type] select the defined tag.
6. If the data type is selected when creating the address tag, the system automatically restricts the data format to the one selected.

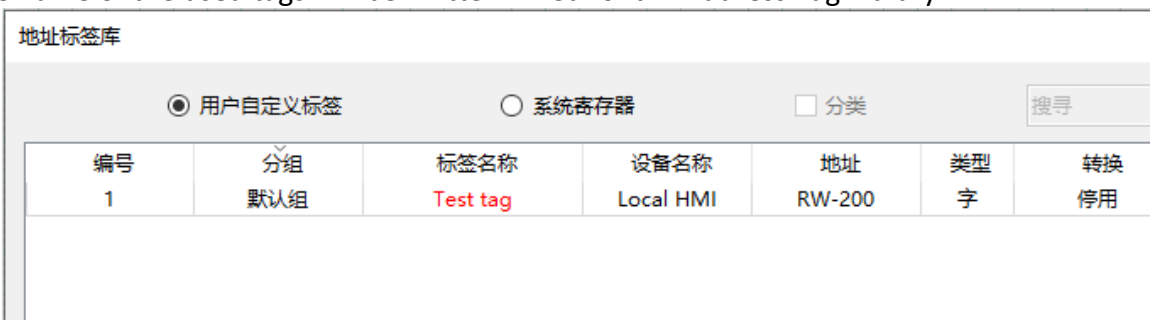


7. When finished, the window tree will show the address tag name used by the object.



Note

- The name of the used tags will be written in red font in Address Tag Library.



Chapter 16 Transferring Recipe Data

This chapter explains how to transfer recipe data.

16.1 Overview

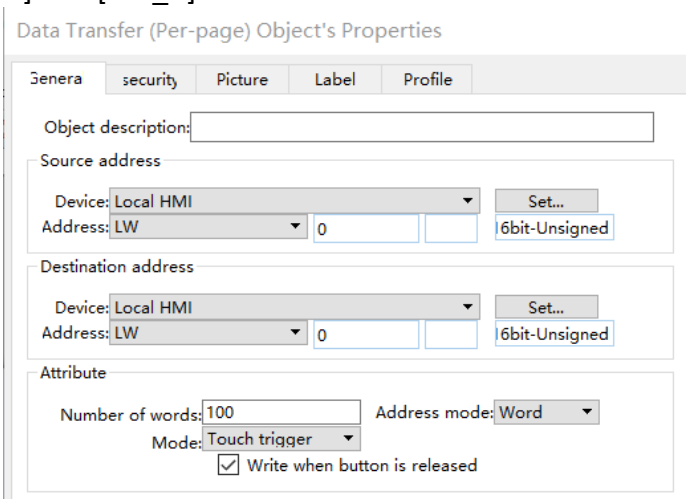
Recipe Data refers to the data stored in RW and RW_A addresses. The way of reading and writing these addresses is the same as operating a word register. The difference is that recipe data is stored in flash memory, when restarting HMI, the latest data records in RW and RW_A are kept.
The size of recipe 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.

16.2 Steps to Update Recipe Data with Ethernet or USB Cable (in developing)

16.3 Steps to Update Recipe Data with SD Card or USB Disk. (in developing)

16.4 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

16.5 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.

Chapter 17 Macro Reference

This chapter describes the syntax, programming methods and usage of macro commands.

17.1 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.

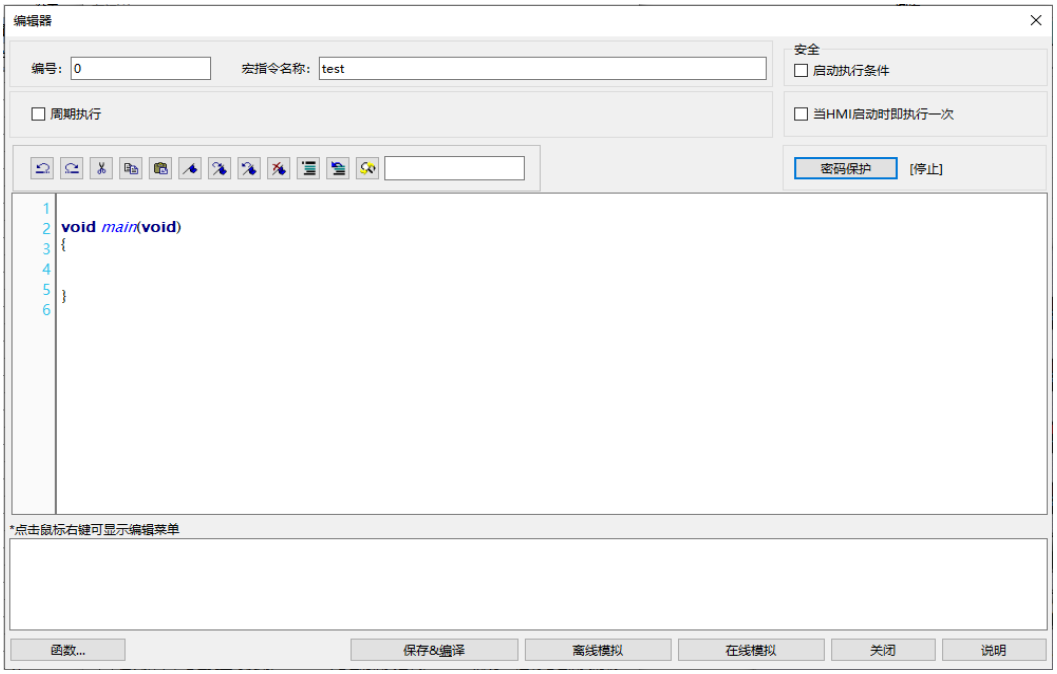
17.2 Instructions to use the Macro Editor

Macro editor provides the following functions:

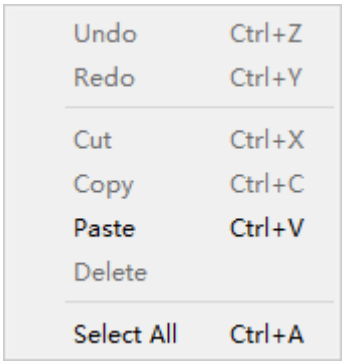
- Display line number
- Undo / Redo
- Cut / Copy / Paste
- Select All
- Toggle All Outlining
- Security -> Use execution condition
- Periodical execution
- Execute one time when HMI starts
- Password protection

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.



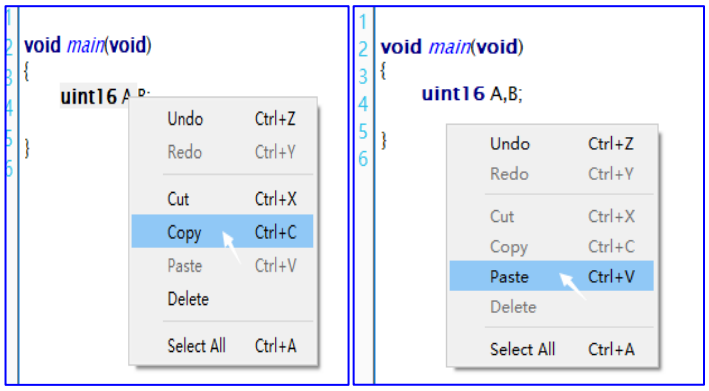
2. Right click on the edit area to open the pop-up menu as shown in the following figure. Disabled operations are colored grey, which indicates that it is not possible to use that function in the current status of the editor. For example, you should select some text to enable the copy function, otherwise it will be disabled. Keyboard shortcuts are also shown.



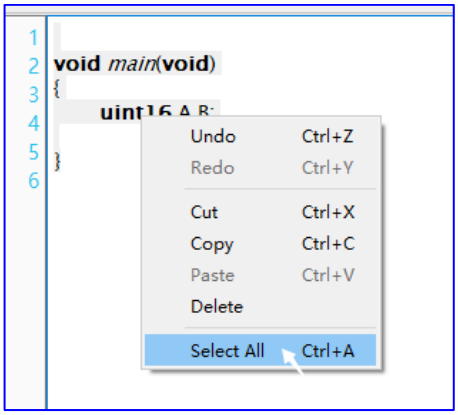
3. The toolbar provides [Undo], [Redo], [Cut], [Copy], [Paste], [Delete] and [Select all] buttons, and so on.

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. The statements enclosed in the following keywords are called a “block” of the macro code:

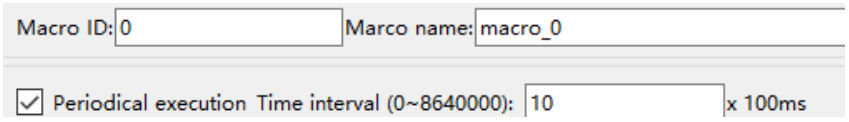
- Sub-Function: definitions outside the main function
- Iterative statements:

i. for

ii. while

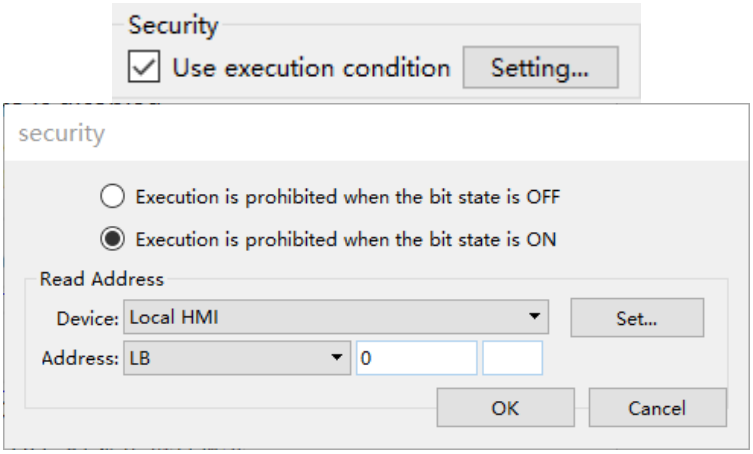
- Logical statements: a. if – else if – else
- Multiple judgment statement: Switch case

8. When [Periodical execution] is checked, this macro will be triggered periodically.

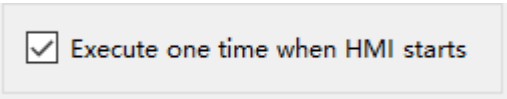


9. Select [Security] » [Use execution condition] » [Settings] to enable security settings:

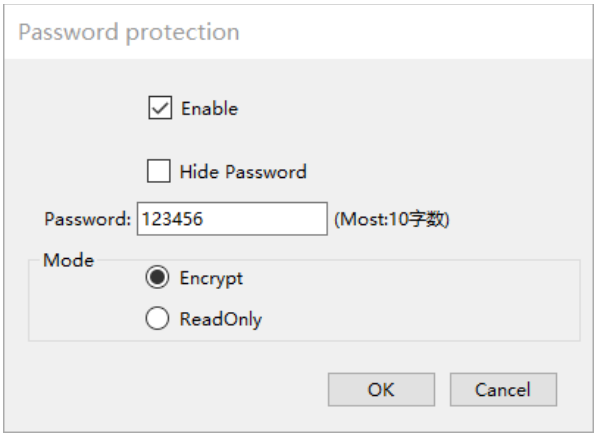
- [Disable when Bit is ON]: When Bit is ON, this macro is disabled.
- [Disable when Bit is OFF]: When Bit is OFF, this macro is disabled.



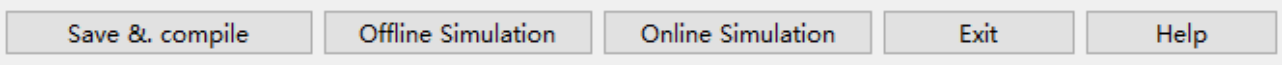
10. Select [Execute one time when HMI starts], this macro will be executed once when HMI starts up.



1. If enable [Password protection], users need to enter the correct password before modifying the script. When the password is entered incorrectly three times in a row, the system needs to be restarted in order to re-enter the password.



12. The macro editor is not a monopoly window. Returning to the main screen and editing the project with the Work Space window open is allowed.



13. The macro editor provides Find and Replace features.

17.3 Structure of Macro

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    ---------- Optional
      Local Variable Declarations
  End Sub

  macro_command main()               ---------- Required
      Local Variable Declarations
      [Statements]
  end macro_command                  ---------- Required
```

Macro must have one and only one main function which is the execution start point of macro. The format is:

```
void main(void)
{

}
```

The variable declaration must be placed before the macro instruction statement, otherwise if the statement is placed before the variable declaration, it will cause the macro instruction to fail compilation.
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. The double slash "//" represents the program annotation, and the text after it will not be executed.

```
void main(void)
{
    uint16   pressure = 10;                                  //   local variable declaration
SetData(pressure,0, "Allen-Bradley DF1", "N7", 0, 1) ; //   function calling end
}
```

17.4 Syntax of Macro

17.4.1 Constants and Variables

Constants

Constants are fixed values and can be directly written into statements. The formats are:

Constant type	Note	Example
Decimal integer		345,-213, 0, 23456
Hexadecimal	Must begin with 0x	0x3b,0xfff
ASCII	Single character must be enclosed in single quotation marks and a string (group of characters) must be enclosed in double quotation marks.	“hello”,”world”
Boolean		true, false

Here is an example using constants:

```
void main(void)
{
int16 A, B ; //    A and B are short integer
A= 1234 ;
B= 0x12;    //    1234 and 0x12 are constants
}
```

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 9 different Variable types, 6 for signed data types and 3 for unsigned data types:

Variable type	Description	Range
bool	1 bit	0, 1
string		
int8	8 bits	+128 ~-128
int16	16 bits	+32767 ~-32767
int32	32 bits	+2147418112 ~-2147418112
float	32 bits	

uint8	8 bits	0~255
uint16	16 bits	0~65535
uint32	32 bits	0~4294967295

Declaring Variables

Variables must be declared before being used. To declare a variable, specify the type before the variable name.
Example:
int a;

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 4096. (Macros only support at most 4096 variables per macro).
Example:
array<int8> a(10);

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:
  uint16 a ;
  array<int16> b(3);
  a = 10 ;
  b[0] = 1;
- During declaration assign values
  uint8 a = 6, b = 9 ;

Note: Each statement must end with a semicolon.

The declaration of arrays is a special case. Example as following,

array<int> C(10,0); //Defined a one-dimensional array variable C with a length of 10, type of double integer, and all default values of 0

array<float> D={1.2,3.14,5,10.69}; //The length of the one-dimensional array variable D is 4, the type is floating-point, and the initial data values are 1.2, 3.14, 5, and 10.69

17.4.2 Operators

Operators are used to designate how data is manipulated and calculated. In any statement, the variable result on the left side of the operator is obtained based on the condition on the right side of the operator. As following,

Operators	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){B = 5 ;}
<=	Less than or equal to	if (A <= 10){B = 5 ;}
>	Greater than	if(A > 10){B = 5 ;}
>=	Greater than or equal to	if (A >= 10){B = 5 ;}
==	Equal to	if (A == 10){B = 5 ;}
!=	not equal to	if(A != 10){B = 5 ;}
Logic Operators	Description	Example
&&	Logical AND	if(A < 10 && B > 5){ C = 10;}
||	Logical OR	If(A >= 10 || B > 5){ C = 10;}
!	Logical NOT	if(!A){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 bit set to the left a specified number of positions	A = B << 8

>>	Shifts the bits in a bit set 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

Note

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.

+,-, *, /, %, >=, >, <=, <, ==, &&, ||, !=, !, <<, >>,=, &, |, ^, ~ , for, to, down, step, next, return, bool, uint, int, string, float, void, if, else,switch, break, continue, set, sub, end, while, 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, 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。

17.5 Statement

17.5.1 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

17.5.2 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
uint8 A = 2 //where a variable A is assigned to 2

17.5.3 Logical Statement

Logical statements perform actions depending on the condition of a boolean expression. The syntax is as follows:

Single-Line Format

```
if(<Condition> )
{
    [Statements]
}
```

Block Format

```
if(<Condition> )
{
    [Statements]
}else if(<Condition-n> ){
    [Statements]
}else{

}
```

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

[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.

17.5.4 Selective Statements

The switch-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:

```
switch([variable]){
    case [constant-expression]:
        [statement]
        break;
    default:
        [statement]
}
```

For example,

```
switch (A){
        case 1:
        B=1;
        break;
        case 2:
        B=2;
        break;
        default:
        B=0;
}
```

Multiple cases in the same block

```
switch([variable])
{
    case [constant-expression]:
    case [constant-expression]:
        [statement]
        break;
    default:
        [statement]
}
```

For example,

```
switch (A)
```

```
{
    case 1:
    case 2:
    B=1;
    break;
    default:
    B=0;
}
```

Syntax description

switch()	Must be used to begin the statement.
“variable”	Required. The value of this variable will be compared to the value of each case.
default	Optional. Represents a preset case. When the value of 'variable' does not match any of the cases, this preset block will be executed.
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.
constant-expression	Represents the value of each case and needs to be prefixed with a “case”.

17.5.5 Iterative Statements

Iterative statements control loops and repetitive tasks depending on condition. There are two types of iterative statements.

For Statement

The For 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( init;condition;increment) {
      statement(s);
}
```

For example,

```
for(int a=0;a<20;a++)
{
    B+=a;
}
```

Syntax description

for	Must be used to begin the statement
init	Required. It will be executed at first and only one time. This step allows you to declare and initialize any loop control variables. You can also write nothing but a semicolon.

condition	Required. To determine the conditions for continuing the loop. If true, the loop is executed. If it is false, the loop will not be executed, and will jump to the next statement of the For Statement.
incremen	Required. It allows you to update loop control variables. It also can be left blank as long as a semicolon appears after the condition.

While

The While 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.

```
while( <Condition>)
{
[Statements]
}
```

For example

```
while(a<10)
{
    a++;
}
```

Syntax Description

while	Must be used to begin the statement.
Condition	Required. This is the controlling statement. When it is TRUE, the loop begins execution. When it is FALSE, the loop terminates.

Note

- Other control commands

break	Used in For and While Statements. It skips immediately to the end of the iterative statement.
continue	Used in For and While Statements. 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 and also can be used to describe the return value of a custom function.

17.6 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:

1. Function Blocks with return value.

```
type <name>(parameters)
{
    Local variable declarations
    [Statements]
    [return [value]]
}
```

For example,

```
int getValue()
{
    return 15;
}
```

2. Function Blocks without return value

```
void <name > (parameters)
{
        Local variable declarations
    [Statements]
}
```

For example,

```
int x=8,y=10;
void getValue(int x,int y)
{
    for(int i= 0;i<x;i++){
        y += i;
    }
}
```

Syntax Description

void	Optional. Used at the beginning of Function Blocks. If the Function Blocks have no returned value, it is mandatory.
type	Optional. This is the data type of value that the function returns.
(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.

17.7. Built-in Function Block

Our Marco has many built-in functions for retrieving and transferring data to the devices, data management and mathematical functions.

17.7.1

Name	SQRT
Syntax	SQRT(source, result)
Description	Calculates the square root of *source* and stores the result into *result*. *source* can be a constant or a variable. *result* must be a variable. *source* must be a non-negative value.

Example	void main(void)
	{
	float source = 9, result;
	
	SQRT(source, result);// result == 3
	}

Name	CUBERT
Syntax	CUBERT (source, result)
Description	Calculates the cube root of source and stores the result into *result*. *source* can be a constant or a variable. *result* must be a variable. *source* must be a nonnegative value.
Example	void main(void) {

	float source = 27, result;
	
	CUBERT(source, result);// result == 3
	}

Name	POW
Syntax	POW (source1, source2, result)
Description	Calculates *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	void main(void)
	{
	float y, result;
	y = 0.5;
	POW(25, y, result);// result == 5
	}

Name	SIN
Syntax	SIN(source, result)
Description	Calculates the sine of *source* (degree) into *result*. *source* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	float source = 30, result;
	
	SIN(source, result);// result == 0.5
	
	SIN(90, result);// result == 1
	}

Name	COS
Syntax	COS(source, result)
Description	Calculates the cosine of *source* (degree) into *result*. *source* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	float source = 60, result;
	COS(source, result);// result == 0.5
	COS(90, result);// result == 0
	}

Name	TAN
Syntax	TAN(source, result)
Description	Calculates the tangent of *source* (degree) into *result. source* can be a constant or a variable. *result* must be a variable.

Example
举例
	void main(void)
	{
	float source = 45, result;
	TAN(source, result);// result == 1
	}

Name	COT
Syntax	COT(source, result)
Description	Calculates the cotangent of *source* (degree) into *result*. *source* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	float source = 45, result;
	COT(source, result);// result == 1
	}

Name	SEC
Syntax	SEC(source, result)
Description	Calculates the secant of *source* (degree) into *result. source* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	float source = 60, result;
	SEC(source, result);// result == 2
	}

Name	CSC
Syntax	CSC(source, result)
Description	Calculates the cosecant of *source* (degree) into *result. source* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	float source = 30, result;
	CSC(source, result);// result == 2
	}

Name	ASIN
Syntax	ASIN(source, result)
Description	Calculates the arc sine of *source* into *result* (degree). *source* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	float source = 0.5, result;
	ASIN(source, result);// result == 30
	}

Name	ACOS
Syntax	ACOS(source, result)
Description	Calculates the arc cosine of *source* into *result. source* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	float source = 0.5, result;
	ACOS(source, result);// result == 60
	}

Name	ATAN
Syntax	ATAN(source, result)
Description	Calculates the arc tangent of *source* into *result.* *source* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	float source = 1, result;
	ATAN(source, result);// result == 45
	}

Name	LOG
Syntax	LOG (source, result)
Description	Calculates the natural logarithm of a number and saves into *result.* *source* can be either a variable or a constant. *result* must be a variable.

Example	void main(void)
	{
	float source = 100, result;
	LOG(source, result);// result == 4.6052
	}

Name	LOG10
Syntax	LOG10 (source, result)
Description	Calculates the base-10 logarithm of a number and saves into *result.* *source* can be either a variable or a constant. *result* must be a variable.

Example	void main(void)
	{
	float source = 100, result;
	LOG10(source, result);// result == 2
	}

Name	RAND
Syntax	RAND(result)
Description	Calculates a random integer and saves into *result.* (Range: 0 ~ 32766) *result* must be a variable

Example	void main(void)
	{
	int16 wRandom
	RAND(wRandom);// save a pseudorandom number to wRandom.
	}

17.7.2. Data Type Conversion Functions

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	void main(void)
	{
	int16 source = 1234, result;
	BIN2BCD(source, result);// result == 0x1234
	BIN2BCD(3456, result);// result == 0x3456
	}

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	void main(void)
	{
	int16 source = 0x1234, result;
	BCD2BIN(source, result);// result == 1234
	BCD2BIN(0x3456, result);// result == 3456
	}

Name	DEC2ASCII
Syntax	DEC2ASCII(source, result,start, len)
Description	Transforms a decimal value (*source*) into an ASCII string and saves 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	void main(void)
	{
	int16 source = 5678;
	array<uint8> result(4);
	DEC2ASCII(source, result, 0, 4);
	// result[0] == '5', result[1] == '6', result[2] == '7', result[3] == '8'
	}

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	void main(void)
	{
	int16 source = 0x5678;
	array<uint8> result(4);
	HEX2ASCII(source, result, 0, 4);
	// result[0] == '5', result[1] == '6', result[2] == '7', result[3] == '8'
	}

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	void main(void)
	{
	float source = 56.8;
	array<uint8> result(4);
	FLOAT2ASCII(source, result, 0, 4);
	// result[0] == '5', result[1] == '6', result[2] == '.', result[3] == '8'
	}

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	void main(void)
	{
	array<uint8> source(4);
	source[0] = 0x35;
	source[1] = 0x36;
	source[2] = 0x37;
	source[3] = 0x38;
	int16 result;
	ASCII2DEC(source, 0, result, 4);// result == 5678
	}

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	void main(void)
	{
	array<uint8> source(4);
	source[0] = 0x35;
	source[1] = 0x36;
	source[2] = 0x37;
	source[3] = 0x38;
	uint16 result;
	ASCII2HEX(source, 0, result, 4);// result == 0x5678
	}

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 *ource[start]*. *source* and *len* can be a constant or a variable. *result* must be a variable. *start* must be a constant.

Example	void main(void)
	{
	array<uint8> source(4);
	source[0] = 0x35;
	source[1] = 0x36;
	source[2] = 0x37;
	source[3] = 0x38;
	float result;
	ASCII2FLOAT(source, 0, result, 4);// result == 5678
	}

17.7.3. Data Operation Functions

Name	FILL
Syntax	FILL(source,start, preset, count)
Description	Sets array elements from '*source[start]* to '*source[start + count – 1]* to the specified value (*preset*). *source* and *start* must be a variable, and *preset* can be a constant or variable.

Example	void main(void)
	{
	array<int8> byCommand(32);
	FILL(byCommand, 0, 2, 32);// set buffers to 2
	}

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	void main(void)
	{
	int16 source = 0x1234, result;
	SWAPB(source, result);// result == 0x3412
	}

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	void main(void)
	{
	int32 source = 0x12345678, result;
	SWAPW(source, result)// result == 0x56781234
	}

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	void main(void)
	{
	int16 source = 0x1234, result;
	LOBYTE(source, result);// result == 0x34
	LOBYTE(0x5678, result);// result == 0x78
	}

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	void main(void)
	{
	int16 source = 0x1234, result;
	HIBYTE(source, result);// result == 0x12
	HIBYTE(0x1234, result);// result == 0x12
	}

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	void main(void)
	{
	int32 source = 0x12345678, result;
	LOWORD(source, result);// result == 0x5678
	}

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	void main(void)
	{
	int32 source = 0x12345678, result;
	HIWORD(source, result);// result == 0x1234
	}

17.7.4. Bit State Conversion

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	void main(void)
	{
	int16 source = 0x4, result, bit_pos = 2;
	GETBIT(source, result, bit_pos);// result == 1
	GETBIT(source, result, 1);// result == 0
	}

Name	SETBITON
Syntax	SETBITON(source, result, bit_pos)
Description	Changes the state of designated bit position of a data (*source*) to 1, and puts changed data into *result*. *source* and *bit_pos* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	int16 source = 0x4, result, bit_pos = 1;
	SETBITON(source, result, bit_pos);// result == 6
	}

Name	SETBITOFF
Syntax	SETBITOFF(source, result, bit_pos)
Description	Changes the state of designated bit position of a data (*source*) to 0, and puts changed data into *result. source* and *bit_pos* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	int16 source = 0x6, result, bit_pos = 1;
	SETBITOFF(source, result, bit_pos);// result == 4
	}

Name	INVBIT
Syntax	INVBIT(source, result, bit_pos)
Description	Inverts the state of designated bit position of a data (*source*), and puts changed data into *result*. *source* and *bit_pos* can be a constant or a variable. *result* must be a variable.

Example	void main(void)
	{
	int16 source = 0x6, result, bit_pos = 1;
	INVBIT(source, result, bit_pos);// result == 4
	}

17.7.5. Related Protocol Functions

Name	DELAY
Syntax	DELAY(time)
Description	Pause the execution of Macro for the specified time. The unit of time is milliseconds. time can be a constant or a variable.

Example	void main(void)
	{
	DELAY(100);// delay 100ms
	}

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	void main(void)
	{
	array<uint8> data(5);
	data[0] = 1;
	data[1] = 2;
	data[2] = 3;
	data[3] = 4;
	data[4] = 5;
	uint16 checksum = 0;
	ADDSUM(data, 0, checksum, 5);
	}

Name	XORSUM
Syntax	XORSUM(source,start, result, data_count)
Description	Uses XOR 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	void main(void)
	{
	array<uint8> data(5);
	data[0] = 1;
	data[1] = 2;
	data[2] = 3;
	data[3] = 4;
	data[4] = 5;
	uint16 checksum = 0;
	XORSUM(data, 0, checksum, 5);
	}

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	void main(void)
	{
	array<uint8> data(5);
	data[0] = 1;
	data[1] = 2;
	data[2] = 3;
	data[3] = 4;
	data[4] = 5;
	int16 result;
	CRC(data, 0, result, 5);
	}

Name	OUTPORT (in developing)
Syntax	OUTPORT(source,start, device_name, data_count)
Description	Sends out the specified data from source[start] to source [*start + data_count - 1*] to the device via a COM port or an Ethernet port. *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.

Example	void main(void)
	{
	array<int8> byCommand(32);
	FILL(byCommand, 0, 0, 32); // set buffers to a specified value
	OUTPORT(byCommand, 0, "Free Protocol", 32); // send 32 bytes
	}

Name	INPORT (in developing)
Syntax	INPORT(read_data,start, device_name, read_count, return_value)
Description	Reads data from a COM port or the Ethernet port. The 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	void main(void)
	{
	array<int16> wResponse(6);
	int16 return_value;
	INPORT(wResponse, 0, "Free Protocol", 6, return_value); // read 6 words
	if (return_value > 0)
	{
	SetData(wResponse, 0, "Local HMI", "LW", 0, 6); // set responses to LW0
	}
	}

Name	INPORT2 (in developing)
Syntax	INPORT2(response,start, device_name, receive_len, wait_time)
Description	Reads data from a COM port or the Ethernet port. The data read will be saved in the response array. *device_name* is the name of a device defined in the device table and the device must be a "Free Protocol"-type device. *receive_len* stores the length of the data received. It must be a variable. *receive_len* can't exceed the size of response array. *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	void main(void)
	{
	array<int16> wResponse(6);
	int16 receive_len, wait_time=20;
	INPORT2(wResponse, 0, "Free Protocol", receive_len, wait_time); // wait_time's unit : millisecond
	if (receive_len > 0)
	{
	SetData(wResponse, 0, "Local HMI", "LW", 0, 6); // set responses to LW0
	}
	}

<table>
<tr><td>Name</td><td>GetData</td></tr>
<tr><td>Syntax</td><td>GetData(read_data, start, device_name, device_type, address_offset, data_count)
or
GetData(read_data, 0, device_name, device_type, address_offset, 1)</td></tr>
<tr><td>Description</td><td>Receives data from the PLC. When the data is not read successfully, the function will not continue executing the next command. 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.
Device_name is the device 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 device 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 device. 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 device’s station number is N. AAAAA represents the address offset. This format is used while multiple devices 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 device’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.
<table>
<tr><th>read_data 的类型</th><th>data_count 的值</th><th>读取16位数据的个数</th></tr>
<tr><td>int8 (8-bit)</td><td>1</td><td>1</td></tr>
<tr><td>uint8 (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>int16 (16-bit)</td><td>1</td><td>1</td></tr>
<tr><td>int16 (16-bit)</td><td>2</td><td>2</td></tr>
<tr><td>int32 (32-bit)</td><td>1</td><td>2</td></tr>
<tr><td>int32 (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 GetData() is executed using a 32-bit data type (int or float), the function will automatically convert the data.</td></tr>
<tr><td>Example</td><td>void main(void)
{
bool a ;
array<bool> b(30);</td></tr>
</table>

```
        // get the state of LB2 to the variable a
        GetDataEx (a, 0,"Local HMI", "LB", 2, 1);
        // get 30 states of LB0 ~ LB29 to the variables b_array[0] ~ b_array[29]
        GetDataEx (b,0, "Local HMI","LB", 0, 30) ;
}
```

Name	GetDataEx
Syntax	GetDataEx(read_data, start, device_name, device_type, address_offset, data_count) or GetDataEx(read_data, 0, device_name, device_type, address_offset, 1)
Description	Receives data from the device and continues executing next command even when the read operation fails. Descriptions of *read_data*, *device_name*, *device_type*, *address_offset* and *data_count* are the same as GetData.

Example	void main(void)
	{
	array<int8> byData(10);
	array<int16> wData(6);
	GetDataEx(byData, 0, "Local HMI", "LW", 0, 10); // read 10 bytes = 5 words
	GetDataEx(wData, 0, "Local HMI", "LW", 0, 6); // read 6 words
	GetDataEx(wData, 0, "Local HMI", "Pressure", 6);
	// use user-defined tag- "Pressure" to indicate device type and address.
	}

Name	SetData
Syntax	SetData(send_data, start, device_name, device_type, address_offset, data_count) or SetData(send_data, 0, device_name, device_type, address_offset, 1)
Description	Sends data to the device. When the data is not written successfully, the function will not continue executing the next command. 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. *device_name* is the device 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 device 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 device. 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 device's station number is N. AAAAA represents the address offset. This format is used while multiple devices 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 device'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.*

send_data 的类型	data_count 的值	写入16位数据的个数
uint8 (8-bit)	1	1
uint8 (8-bit)	2	1
bool (8-bit)	1	1
bool (8-bit)	2	1
int16 (16-bit)	1	1
int16 (16-bit)	2	2
int32 (32-bit)	1	2
int32 (32-bit)	2	4
float (32-bit)	1	2
float (32-bit)	2	4

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.

Example	void main(void)
	{
	array<int8> byData(10);
	array<int16> wData(6);
	FILL(byData, 0, 0, 10);// set buffers to a specified value
	FILL(wData, 0, 0, 6);
	SetData(byData, 0, "Local HMI", "LW", 0, 10);// send 10 bytes = 5 words
	SetData(wData, 0, "Local HMI", "LW", 0, 6);// send 6 words
	SetData(wData, 0, "Local HMI", "Pressure", 6);
	// use user-defined tag- "Pressure" to indicate device type and address.
	}

Name	SetDataEx
Syntax	SetDataEx(send_data, start, device_name, device_type, address_offset, data_count) or SetDataEx(send_data, 0, device_name, device_type, address_offset, 1)
Description	Sends data to the device and continues executing next command even when the write operation fails. Descriptions of *send_data*, *device_name*, *device_type*, *address_offset* and *data_count* are the same as SetData.

Example	void main(void)
	{
	array<int8> byData(10);
	array<int16> wData(6);
	FILL(byData, 0, 0, 10);// set buffers to a specified value
	FILL(wData, 0, 0, 6);
	SetData(byData, 0, "Local HMI", "LW", 0, 10);// send 10 bytes = 5 words
	SetData(wData, 0, "Local HMI", "LW", 0, 6);// send 6 words
	SetData(wData, 0, "Local HMI", "Pressure", 6);
	// use user-defined tag- "Pressure" to indicate device type and address.
	}

Name	GetError
Syntax	GetError(err)
Description	Gets an error code.

Example	void main(void)
	{
	int16 err;
	array<int16> byData(10);
	GetDataEx(byData, 0, "MODBUS RTU", "4x", 1, 10); // read 10 bytes = 5 words
	// if err is equal to 0, it is succesful to execute GetDataEx().
	GetError(err);// save a error code to err
	}

Name	PURGE (in developing)
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	void main(void)
	{
	PURGE(1);// clear COM 1's buffers
	}

17.7.6. String Operation Functions

Name	StringGet (in developing)
Syntax	StringGet(read_data,start, device_ name , device_type, address_offset, data_count)
Description	Receives data from the device. 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 device 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 device 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 device. 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 device's station number is N. AAAAA represents the address offset. This format is used while multiple devices 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 device'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. 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.

	<table><tr><th>type of read_data</th><th>data_count</th><th>actual number of 16-bit register read</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></table>

Example	void main(void)
	{
	array<uint8> byData(20);
	StringGet(byData,0, "Local HMI", "LW", 0, 10);// read 10 bytes (equals to 5 words)
	StringGet(byData,0, "Local HMI", "Pressure", 10);// user-defined tag- "Pressure" to indicate device type and address.
	}

Name	StringGetEx (in developing)
Syntax	StringGetEx (read_data,start, device_ name , device_type, address_offset, data_count)
Description	Receives data from the device and continues executing next command even if there's no response from this device. Descriptions of *read_data*, *device_name*, *device_type*, *address_offset* and *data_count* are the same as GetData.

Example	void main(void)
	{
	array<uint16> byData(20);
	StringGet(byData,0, "Local HMI", "LW", 0, 10);// read 10 bytes (equals to 5 words)
	StringGet(byData,0, "Local HMI", "Pressure", 10);// user-defined tag- "Pressure" to indicate device type and address.
	}

Name	StringSet (in developing)
Syntax	StringSet (send_data,start, device_ name, device_type, address_offset, data_count)
Description	Sends data to the device. 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 device 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 device 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 device. For example, StringSet (read_data_1[0], "FATEK KB Series", RT, 5, 1) represents that the address offset is 5.

<table>
<tr><td></td><td>If address_offset uses the format –"N#AAAAA", N indicates that device's station number is N. AAAAA represents the address offset. This format is used while multiple devices 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 device'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.
<table>
<tr><th>type of read_data</th><th>data_count</th><th>actual number of 16-bit register read</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>
</table>
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.</td></tr>
<tr><td>Example</td><td>array<uint8> byData(20);
StringSet(byData,0, "Local HMI", “LW”, 0, 10)// send 10 bytes (equals to 5 words)
StringSet(byData,0, "Local HMI", "Pressure", 0,10)
// use user-defined tag- "Pressure" to indicate device type and address.</td></tr>
</table>

Name	StringSetEx (in developing)
Syntax	StringSetEx(send_data,start, device_name , device_type, address_offset, data_count)
Description	Sends data to the device and continues executing next command even if there's no response from this device. Descriptions of *send_data*, *device_name*, *device_type*, *address_offset* and *data_count* are the same as StringSet.

Example	Array<uint8> byData(10) ;
	StringSet(byData,0, "Local HMI",” LW,” 0, 10)// send 10 bytes (equals to 5 words)
	StringSet(byData,0, "Local HMI", "Pressure",0, 10)
	// use user-defined tag- "Pressure" to indicate device type and address.

Name	StringCopy (in developing)
Syntax	success = StringCopy ([source], destination,start) or success = StringCopy (source,start, destination,start)

Description	Copies 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 has been successfully completed. If so, 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	void main(void)
	{
	Array<uint8> strSrc(10) ;
	Array<uint8> strDest(10) ;
	bool result;
	result = StringCopy(strSrc,0, strDest,0); // "result" will be set to true.
	result = StringCopy("abcde", strDest,0) ;// "result" will be set to true.
	result = StringCopy("1234567890123", strDest,0) ;// "result" will be set to false.
	// "result" will be set to false if the size of source string exceeds the size of destination string.
	
	}

Name	StringDecAsc2Bin (in developing)
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 has been successfully completed. If so, it returns true; otherwise it returns false. The string can only contain these characters: +, -, and 0 to 9. If the string contains other characters, it returns false. The success field is optional.

Example	void main(void)
	{
	Array<uint8> strSrc(10)="1234";
	Int16 dest;
	bool result;
	result = StringDecAsc2Bin(strSrc,0, dest); // the function succeeds.
	// "result" will be set to true and "dest" is equal to 1234.
	result = StringDecAsc2Bin("003c", dest); // the function fails.
	
	}

Name	StringBin2DecAsc (in developing)
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 has been successfully completed. If so, 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	void main(void)
	{
	Int16 source=1234;
	Array<uint8> strDest(4);
	bool result;
	result = StringBin2DecAsc(source, strDest,0) ;// the function succeeds.
	// "result" will be set to true and "strDest" will be set to "1234".
	source = 12345;
	result = StringBin2DecAsc(source, strDest,0) ;// the function fails
	// "result" will be set to false due to insufficient destination buffer size
	}

Name	StringDecAsc2Float(in developing)
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 has been successfully completed. If so, 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	void main(void)
	{
	Array<uint8> strSrc(10)="12.3";
	float dest;
	bool result;
	result = StringDecAsc2Float (strSrc,0, dest) ;// the function succeeds.
	// "result" will be set to true and "dest" is equal to 12.3..
	result = StringDecAsc2Float ("003c",dest) ;// the function fails
	// "result" will be set to false if the source string contains
	}

Name	StringFloat2DecAsc (in developing)
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 has been successfully completed. If so, 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	void main(void)
	{
	float source=123.45;
	Array<uint8> strDest(6);
	bool result;
	result = StringFloat2DecAsc(source, strDest,0) ;// the function succeeds.
	// "result" will be set to true and "strDest" will be set to "123.45".
	source = 123.456;
	result = StringFloat2DecAsc(source, strDest,0) ;// the function fails
	// "result" will be set to false due to insufficient
	}

Name	StringHexAsc2Bin(in developing)
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 has been successfully completed. If so, 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	void main(void)
	{
	array<int8> strSrc(10);
	strSrc[0]='0';
	strSrc[1]='0';
	strSrc[2]='3';
	strSrc[3]='c';
	int dest;
	bool result;
	result = StringHexAsc2Bin(strSrc,0, dest); // the function succeeds.
	// "result" will be set to true and "dest" is equal to 0x003C.
	result = StringHexAsc2Bin("003G", dest); // the function fails.
	// "result" will be set to false if the source string contains characters other than '0' to '9' or 'a' to 'f' or 'A' to 'F'.
	}

Name	StringBin2HexAsc(in developing)
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 has been successfully completed. If so, 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. Please note that this function cannot convert negative values.

Example	void main(void)
	{
	int source=0x123c;
	array<int8> strDest(4);
	
	bool result;
	result = StringBin2HexAsc(source, strDest,0); // the function succeeds.
	// "result" will be set to true and "strDest" will be set to "123C".
	source = 0x1a2b3c4d;
	result = StringBin2HexAsc(source, strDest,0); // the function fails.
	// "result" will be set to false due to insufficient destination buffer size.
	}

Name	StringMid(in developing)
Syntax	success = StringMid (source,start, count, destination,start) or success = StringMid (string, start, count, destination,start)
Description	Retrieves a character sequence from the specified offset of the source string and stores 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 has been successfully completed. If so, 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	void main(void)
	{
	array<int8> strSrc(5);
	strSrc[0]=’A’;
	strSrc[1]=’B’;
	strSrc[2]=’C’;
	strSrc[3]=’D’;
	strSrc[4]=’E’;
	array<int8> strDest(5);
	bool result;
	result = StringMid(strSrc,1, 3, strDest,0); // the function succeeds.
	// "result" will be set to true and "strDest" will be set to "BCD".
	result = StringMid(strSrc,0, 5, strDest,3); // the function fails.
	// "result" will be set to false due to insufficient destination buffer size.
	}

Name	StringLength(in developing)
Syntax	length = StringLength (source,start) or length = StringLength ([source])
Description	Obtains 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	void main(void)
	{
	array<int8> strSrc(5);
	strSrc[0]='A';
	strSrc[1]='B';
	strSrc[2]='C';
	strSrc[3]='D';
	strSrc[4]='E';
	int result;
	result = StringLength(strSrc,0); // "result" is equal to 5.
	result = StringLength("1234567890") ;// "result" is equal to 10.
	}

Name	StringCat(in developing)
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 has been successfully completed. If so, 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	void main(void)
	{
	array<int8> strSrc(4);
	strSrc[0]='A';
	strSrc[1]='B';
	strSrc[2]='C';
	strSrc[3]='D';
	array<int8> strDest(4);
	strDest[0]=1';
	strDest[1]=2;
	strDest[2]=3;
	strDest[3]=4;
	
	bool result;
	result = StringCat(strSrc,0, strDest,0); // the function succeeds.
	// "result" will be set to true and "strDest" will be set to "1234ABCD".
	result = StringCat("EFG", strDest,0) ;// the function fails.
	// "result" will be set to false due to insufficient destination buffer size. In this case, the content of "strDest" remains the same.
	}

Name	StringCompare(in developing)
Syntax	ret = StringCompare (str1,start, str2,start) ret = StringCompare (string1, str2,start) ret = StringCompare (str1,start, string2) ret = StringCompare (string1, string2)
Description	Performs 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	void main(void)
	{
	array<int8> str1(3);
	str1[0]='A';
	str1[1]='B';
	str1[2]='C';
	array<int8> str2(4);
	str2[0]='A';
	str2[1]='B';
	str2[2]='C';
	str2[3]='D';
	bool result;
	result = StringCompare(str1,0, str2,0); // "result" will be set to false.
	result = StringCompare(str1,0, "ABC") ;// "result" will be set to true.
	result = StringCompare(str1,0, "abc") ;// "result" will be set to false.
	}

Name	StringCompareNoCase(in developing)
Syntax	ret = StringCompareNoCase(str1,start, str2,start) ret = StringCompareNoCase(string1, str2,start) ret = StringCompareNoCase(str1,start, string2) ret = StringCompareNoCase(string1 ,string2)
Description	Performs 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	void main(void)
	{
	array<int8> str1(3);
	str1[0]='A';
	str1[1]='B';
	str1[2]='C';
	array<int8> str2(4);
	str2[0]='A';
	str2[1]='B';
	str2[2]='C';
	str2[3]='D';
	bool result;
	result = StringCompareNoCase(str1,0, str2,0) ;// "result" will be set to false.
	result = StringCompareNoCase(str1,0, "ABC"); // "result" will be set to true.
	result = StringCompareNoCase(str1,0, "abc"); // "result" will be set to true.
	}

Name	StringFind(in developing)
Syntax	position = StringFind (source,start, target,start) position = StringFind (source, target,start) position = StringFind (source,start,,target) position = StringFind (source, ,target`)
Description	Returns 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	void main(void)
	{
	array<int8> source(5);
	source[0]='a';
	source[1]='b';
	source[2]='c';
	source[3]='d';
	source[4]='e';
	array<int8> target(2);
	target[0]='d';
	target[1]='e';
	int result;
	result = StringFind(source,0, target,0); // the function succeeds.
	// "result" will be set to 3.
	result = StringFind(source,0, "fg"); // the function fails.
	// "result" will be set to-1 if there is no match.
	}

Name	StringReverseFind(in developing)
Syntax	position = StringReverseFind (source,start, target,start) or position = StringReverseFind (source, target,start) or position = StringReverseFind (source,start, target) or position = StringReverseFind (source,target)
Description	Returns 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	void main(void)
	{
	array<int8> source(6);
	source[0]='a';
	source[1]='b';
	source[2]='c';
	source[3]='a';
	source[4]='b';
	source[5]='c';
	array<int8> target(2);
	target[0]='b';
	int result;
	result = StringReverseFind(source,0, target,0); // the function succeeds.
	// "result" will be set to 4.
	result = StringReverseFind(source,0, "fg"); // the function fails.
	//"result" will be set to-1 if there is no match.
	}

Name	StringFindOneOf (in developing)
Syntax	position = StringFindOneOf (source,start, target,start) or position = StringFindOneOf (source, target,start) or position = StringFindOneOf (source,start, target) or position = StringFindOneOf (source, target)
Description	Returns 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	void main(void)
	{
	array<int8> source(5);
	source[0]='a';
	source[1]='b';
	source[2]='c';
	source[3]='d';
	source[4]='e';
	array<int8> target(3);
	target[0]='x';
	target[1]='d';
	target[2]='e';
	int result;
	result = StringFindOneOf(source,0, target,0) ;// the function succeeds.
	//"result" will be set to 3.
	result = StringFindOneOf(source,0, "xy"); // the function fails.
	//"result" will be set to-1 if there is no match.
	}

Name	StringIncluding(in developing)
Syntax	success = StringIncluding (source,start, set,start, destination,start) or success = StringIncluding (source, set,start, destination,start) or success = StringIncluding (source,start, set, destination,start) or success = StringIncluding (source, set, destination,start)
Description	Retrieves 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 has been successfully completed. If so, it returns true; otherwise it returns false. If the length of retrieved substring exceeds the size of destination buffer, it returns false.

Example	void main(void)
	{
	array<int8> source(7),dest(7);
	source[0]='c';
	source[1]='a';
	source[2]='b';
	source[3]='b';
	source[4]='a';
	source[5]='g';
	source[6]='e';
	
	array<int8> target(3);
	target[0]='a';
	target[1]='b';
	target[2]='c';
	
	bool result;
	result = StringIncluding(source,0, target,0, dest,0); // the function succeeds.
	// "result" will be set to true and "dest" will be set to "cabba".
	result = StringIncluding(source,0, target,0, dest,5); // the function fails.
	// "result" will be set to false due to insufficient destination buffer size.
	}

Name	StringExcluding(in developing)
Syntax	success = StringExcluding (source,start, set,start, destination,start) or success = StringExcluding (source, set,start, destination,start) or success = StringExcluding (source,start, set, destination,start) or success = StringExcluding (source ,set, destination,start)
Description	Retrieves 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 has been successfully completed. If so, it returns true; otherwise it returns false. If the length of retrieved substring exceeds the size of destination buffer, it returns false

Example	void main(void)
	{
	array<int8> source(12),dest(5);
	source[0]='z';
	source[1]='x';
	source[2]='y';
	source[3]='c';
	source[4]='a';
	source[5]='b';
	source[6]='b';
	source[7]='a';
	source[8]='g';
	source[9='e';
	source[10]='z';
	source[11]='z';
	array<int8> target(3);
	target[0]='a';
	target[1]='b';
	target[2]='c';
	bool result;
	result = StringExcluding(source,0, target,0, dest,0) ;// the function succeeds.
	// "result" will be set to true and "dest" will be set to "zxyzz".
	result = StringExcluding(source,0, target,0, dest,3); // the function fails.
	// "result" will be set to false due to insufficient destination buffer size.
	}

Name	StringToUpper(in developing)
Syntax	success = StringToUpper (source,start, destination,start) or success = StringToUpper (source, destination,start)
Description	Converts all the characters in the source string to uppercase characters and stores 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 has been successfully completed. If so, 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	void main(void)
	{
	array<int8> source(5),dest(5);
	source[0]='a';
	source[1]='B';
	source[2]='C';
	source[3]='d';
	source[4]='E';
	bool result;
	result = StringToUpper(source,0, dest,0); // the function succeeds.
	// "result" will be set to true and "dest" will be set to "ABCDE".
	result = StringToUpper("abcdefghij", dest,0); // the function fails.
	// "result" will be set to false.
	}

Name	StringToLower(in developing)
Syntax	success = StringToLower (source,start, destination,start) or success = StringToLower (source, destination,start)
Description	Converts all the characters in the source string to lowercase characters and stores 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 has been successfully completed. If so, 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	void main(void)
	{
	array<int8> source(5),dest(5);
	source[0]='a';
	source[1]='B';
	source[2]='C';
	source[3]='d';
	source[4]='E';
	bool result;
	result = StringToLower(source,0, dest,0); // the function succeeds.
	// "result" will be set to true and "dest" will be set to "abcde".
	result = StringToLower("ABCDEFGHIJ", dest,0) ;// the function fails.
	// "result" will be set to false.
	}

Name	StringToReverse(in developing)
Syntax	success = StringToReverse (source,start, destination,start) success = StringToReverse (source, destination,start)
Description	Reverses the characters in the source string and stores 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 has been successfully completed. If so, it returns true; otherwise it returns false. If the length of reversed string exceeds the size of destination buffer, it returns false.

Example	void main(void)
	{
	array<int8> source(5),dest(5);
	source[0]='a';
	source[1]='b';
	source[2]='c';
	source[3]='d';
	source[4]='e';
	bool result;
	result = StringToReverse(source,0, dest,0); // the function succeeds.
	// "result" will be set to true and "dest" will be set to "edcba".
	result = StringToReverse("abcdefghij", dest,0); // the function fails.
	// "result" will be set to false.
	}

Name	StringTrimLeft(in developing)
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	Trims 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 has been successfully completed. If so, it returns true; otherwise it returns false. If the length of trimmed string exceeds the size of destination buffer, it returns false.

Example	void main(void)
	{
	array<int8> source(10),dest(5);
	source[0]='\t';
	source[1]='\t';
	source[2]=' ';
	source[3]='*';
	source[4]='*';
	source[5]='*';
	source[6]='a';
	source[7]='b';
	source[8]='c';
	source[9='d';
	array<int8> target(3);
	target[0]='\t';
	target[1]=' ';
	target[2]='*';
	
	bool result;
	result = StringTrimLeft(source,0, target,0, dest,0); // the function succeeds.
	// "result" will be set to true and "dest" will be set to "abcd".
	result = StringTrimLeft("\t\t ***abcdefg", target,0, dest,0) ;// the function fails.
	// "result" will be set to false.
	}

Name	StringTrimRight(in developing)
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	Trims 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 has been successfully completed. If so, it returns true; otherwise it returns false. If the length of trimmed string exceeds the size of destination buffer, it returns false.

Example	void main(void)
	{
	array<int8> source(10),dest(5);
	source[0]='a;
	source[1]='b';
	source[2]='c';
	source[3]='d';
	source[4]='!';
	source[5]='!';
	source[6]='!';
	source[7]='?';
	source[8]='?';
	source[9='?';
	array<int8> target(3);
	target[0]='?';
	target[1]='!';
	
	bool result;
	result = StringTrimRight(source,0, target,0, dest,0); // the function succeeds.
	// "result" will be set to true and "dest" will be set to "abcd".
	result = StringTrimRight("abcdefg!!!???", target,0, dest,0);
	// the function fails.
	// "result" will be set to false.
	}

Name	StringInsert(in developing)
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	Inserts 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 has been successfully completed. If so, it returns true; otherwise it returns false. If the length of string after insertion exceeds the size of destination buffer, it returns false.

Example	void main(void)
	{
	array<uint8> str1(20);
	str1[0]="b";
	str1[1]="u";
	str1[2]="t";
	str1[3]=" ";
	str1[4]="t";
	str1[5]="h";
	str1[6]="e";
	str1[7]=" ";
	str1[8]="q";
	str1[9]="u";
	str1[10]="e";
	str1[11]="s";
	str1[12]="t";
	str1[13]="i";
	str1[14]="o";
	str1[15]="n";
	str1[16]=" ";
	str1[17]="i";
	str1[18]="s";
	array<uint8> str2(10);
	str2[0]="t";
	str2[1]="h";
	str2[2]="a";
	str2[3]="t";
	str2[4]=" ";
	str2[5]="i";
	str2[6]="s";
	array<uint8> dest(40);
	dest[0]="t";
	dest[1]="o";
	dest[2]=" ";
	dest[3]="b";
	dest[4]="e";
	dest[5]=" ";
	dest[6]="o";
	dest[7]="r";
	dest[8]=" ";
	dest[9]="n";
	dest[10]="o";
	dest[11]="t";
	dest[12]=" ";

	dest[13]="t";
	dest[14]="o";
	dest[15]=" ";
	dest[16]="b";
	dest[17]="e";
	bool result;
	result = StringInsert(18, str1,3, 13, dest,0); //the function succeeds.
	// "result" will be set to true and "dest" will be set to "to be or not to be the question".
	result = StringInsert(18, str2,0, dest,0); //the function succeeds.
	// "result" will be set to true and "dest" will be set to "to be or not to be that is the question".
	result = StringInsert(0, "Hamlet: ", dest,0); //the function fails.
	// "result" will be set to false and "dest" will be set to "to be or not to be that is the question".}

17.7.7. Recipe Database Functions

Name	RecipeGetData (in developing)
Syntax	RecipeGetData (destination, recipe_address, record_ID)
Description	Gets 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	void main(void)
	{
	int data=0;
	array<uint8> str(20);
	int recordID;
	bool result;
	recordID = 0;
	result = RecipeGetData(data, "TypeA.item_weight", recordID);
	// get data from recipe "TypeA", where item name is "item_weight" and the record ID is 0.
	recordID = 1;
	result = RecipeGetData(str,0, "TypeB.item_name", recordID);
	// get data from recipe "TypeB", where item name is "item_name" and the record ID is 1.
	}

Name	RecipeQuery (in developing)
Syntax	RecipeQuery (SQL command, destination)
Description	Uses 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	void main(void)
	{
	int total_row=0;
	bool result;
	result = RecipeQuery("SELECT * FROM TypeA", total_row);
	// Query "TypeA". The total number of rows of query result is written to total_row.
	}

Name	RecipeQueryGetData (in developing)
Syntax	RecipeQueryGetData (destination, recipe_address, result_row_no)
Description	Gets 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	void main(void)
	{
	
	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 "TypeA". The total number of rows of query result is written to total_row.
	if (result_query){
	for(row_number=0; row_number<total_row-1;row_number++){
	result_data = RecipeQueryGetData(data, "TypeA.item_weight", row_number);
	}
	}
	}

Name	RecipeQueryGetRecordID (in developing)
Syntax	RecipeQueryGetRecordID (destination, result_row_no)
Description	Gets 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	void main(void)
	{
	int recordID=0,total_row=0,row_number=0;
	bool result_query;
	bool result_id;
	result_query=RecipeQuery("SELECT*FROMTypeA",total_row);
	// Query "TypeA". The total number of rows of query result is written to total_row.
	if (result_query){
	for(row_number=0;row_number<total_row1;row_number++){
	result_id = RecipeQueryGetRecordID(recordID,row_number);
	}
	}
	}

17.7.8. Miscellaneous Functions

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	void main(void)
	{
	Beep();
	}

Name	ASYNC_TRIG_MACRO (in developing)
Syntax	ASYNC_TRIG_MACRO(macro_id)
Description	Triggers 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	void main(void)
	{
	ASYNC_TRIG_MACRO(5);// execute MACRO 5
	}

Name	SYNC_TRIG_MACRO (in developing)
Syntax	SYNC_TRIG_MACRO(macro_id)
Description	Triggers 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	void main(void)
	{
	SYNC_TRIG_MACRO(5);// execute MACRO 5
	}

Name	FindDataSamplingDate (in developing)
Syntax	return_value = FindDataSamplingDate(data_log_number, index, year, month, day) or FindDataSamplingDate(data_log_number, index, year, month, day)
Description	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. *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.

Example	void main(void)
	{
	short data_log_number=1, index=2, year, month, day;
	short result;
	// "result" will be set to 1 if referred data log file is successfully found, ohterwise 0.
	// If successful, then save the date to "year", "month" and "day".
	result = FindDataSamplingDate(data_log_number, index, year, month, day);
	}

Name	FindDataSamplingIndex (in developing)
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. *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	void main(void)
	{
	short data_log_number=1, year=2010, month=12, day=10, index;
	short result;
	// "result" == 1 if referred data log file is successfully found, ohterwise 0.
	// If successful, then save the file index to "index".-1 indicates not found.
	result = FindDataSamplingIndex(data_log_number, year, month, day, index);
	}

Name	FindEventLogDate (in developing)
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. *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	void main(void)
	{
	short index=1, year, month, day;
	short result;
	// "result" will be set to 1 if referred event log file is successfully found, ohterwise 0.
	// If successful, then save the date to "year", "month" and "day".
	result = FindEventLogDate(index, year, month, day);
	}

Name	FindEventLogIndex (in developing)
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. *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.

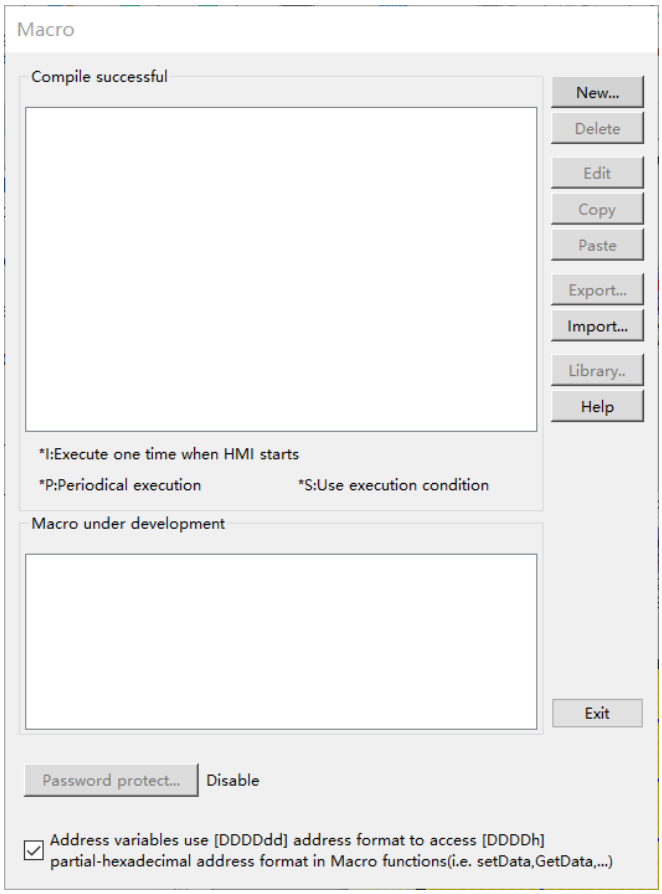
Example	void main(void)
	{
	short index, year=2010, month=12, day=10;
	short result;
	// "result" will be set to 1 if referred event log file is successfully found, ohterwise 0.
	// If successful, then save the file index to "index".-1 indicates not found.
	result = FindEventLogIndex(year, month, day, index);
	}

17.8 How to Create and Execute a Macro

17.8.1 How to Create a Macro

Please follow the steps below to create a macro.

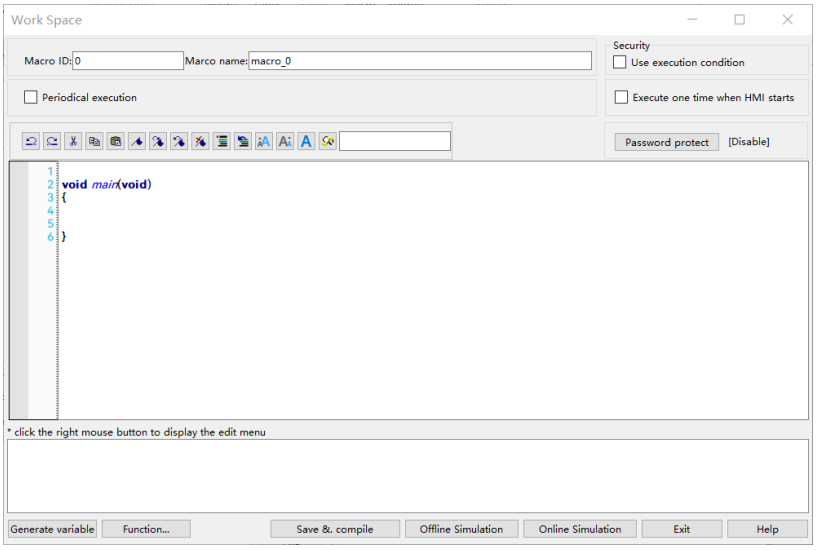
1. Click [Project] » [Macro]  to open Macro Manager dialog box.



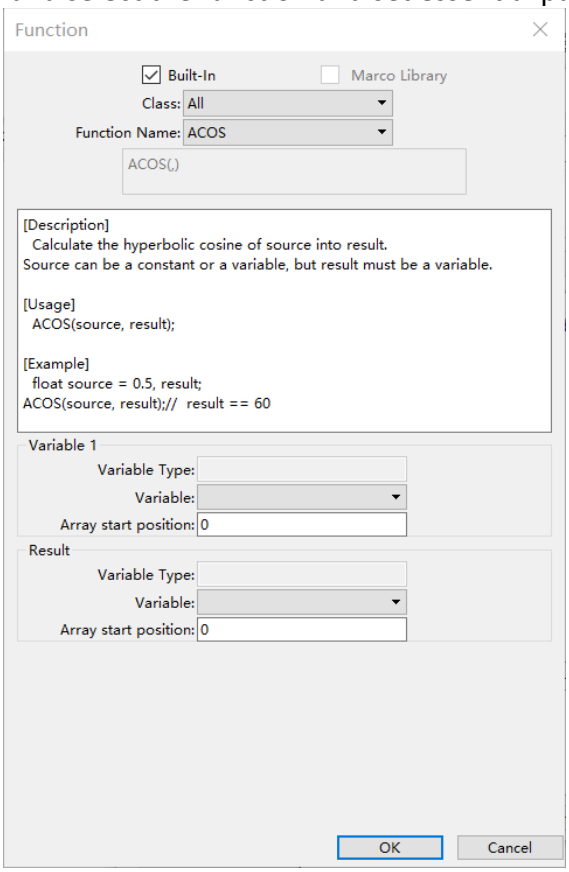
In Macro Manager, all macros compiled successfully are displayed in “Macro list”, and all macros under development or cannot be compiled are displayed in “Macro under development”. The following is a description of the various buttons.

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.
Exit	Confirm all macro instructions compiled then leave this Macro dialog box.
Library	Open Macro Function Library managing dialog.

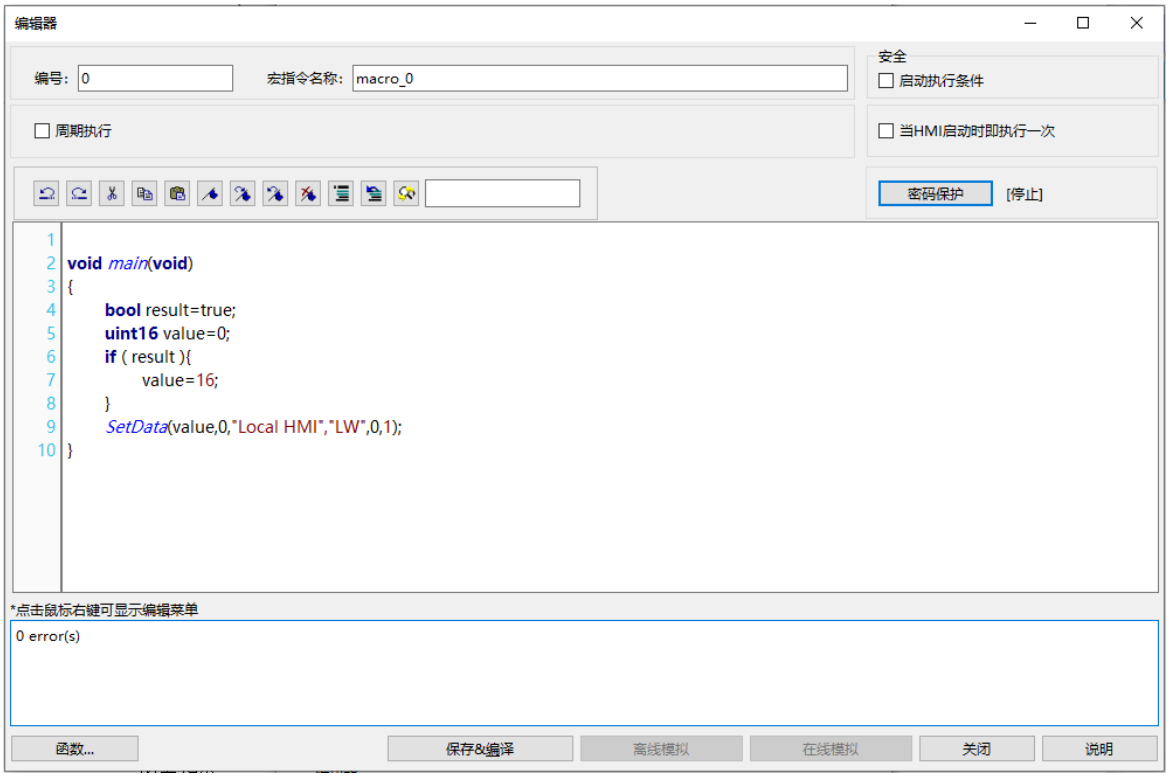
2. Press the [New] button to create an empty macro and open the macro editor. Every macro has a unique number defined at [Macro ID], and must have a macro name, otherwise an error will appear while compiling.



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_0” will be in “Compile successful! ”.

17.8.2 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.

- 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.

- In Window Settings, Macro group box

1. [Open]: When the window opens, run the selected macro once.
2. [Cycle]: When the window opens, run the selected macro every 0.5 second.
3. [Close]: When the window closes, run the selected macro once.

17.9 User Defined Macro Function（in developing）

17.10

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:

```
array<uint8> a(4096);
array<bool> b(4096);
array<uint16> c(2048);
array<int> d(1024);
array<float> f(1024);
```

2. A maximum of 255 macros are allowed in a project. However.
3. A macro may cause the HMI to be unresponsive. Possible reasons may include:

- It contains an undesired infinite loop.
- Array size exceeds the available variable storage space in a macro.

4. The device communication speed may affects execution speed of the macro . Similarly, having too many macros may slow down the communication between an HMI and a device.

17.11 Use the free Protocol to control a device（in developing）

17.12 Compile Error Message

- Error Message Format

[Row x Column y] Error: Description
(x, y help users quickly identify error lines, n represents the number of errors)
For example:
[R5 C4] Error: Data format error
1 error(s)
When there are compile errors, the description of the error can be found by the compiler error message number

- Error Description

① syntax error：'identifier'

There are many possibilities to cause compiler error.

For example,

```
void main(void)
{
    char i;//   this is an unsupported variable name
}
```

② 'identifier' used without having been initialized

Macro must define the size of an array during declaration.

For example,

```
void main(void)
{
    uint16 i;
    array<uint16> b(i);//   i must be a numeric constant
}
```

③ Error: 'g' is already declared (redefinition error: 'identifier') The name of variable and function within its scope must be unique.

For example,

```
void main(void)
{
    uint8 g;
    array<uint8> g;//   'g' is already declared
}
```

④ Error: Expected identifier (Function name error)

Reserved keywords and constant cannot be the name of a function.

For example,

```
int if(){//   error

    return 1;
}
```

⑤ Error：Wrong data format (for example: parentheses have not come in pairs)

For example,

```
void main(void)
{
    uint16 result=0;
    GetData(result,0,"Local HMI","LB",0,1;      // Missing right parenthesis)
}
```

⑥ Error: Condition/loop statement format error （illegal expression in if/for statements.)

For example,

```
void main(void)
{
    uint16 result=0;
    GetData(result,0,"Local HMI","LB",0,1);
    if(){   // illegal expression in if statement.

    }
}
```

⑦ Error: Conditional statement missing curly braces (missing curly brace in if/for statements. Or curly braces have not come in pairs.)

For example,

```
void main(void)
{
    uint16 result=0;
    GetData(result,0,"Local HMI","LB",0,1);
    if(result){

    // Missing curly brace here. }
}
```

⑧ Error: Conditional statement missing parentheses. (Missing 'if' before else if or else statement.)

For example,

```
void main(void)
{
    uint16 result=0;
    GetData(result,0,"Local HMI","LB",0,1);
    // missing if(){} here.
    else if(result){
    }
}
```

⑨ Error: Invalid 'break'. Break Statement can only be used in For & While Statement.

For example,

```
void main(void)
{
    if(true){
        break;// invalid “break”
    }
}
```

⑩ Error: Invalid 'continue'. Continue Statement can only be used in For & While Statement.

For example,

```
void main(void)
{
    if(true){
        continue;// Invalid Continue Statement.
    }
}
```

11 Error: Variable hasn't been defined. Missing assign statement in expression.

For example,

```
void main(void)
{
    int a,b;
    a=b+x;//    x is undefined.

}
```

12 Error: Identifier 'vo' is not a data type in global namespace。

13 Error: Not all paths return a value。

For example,

```
int GetValue(){
      //no return to int value here.
}
void main(void)
{
}
```

parameter type is incorrect

14 Error: 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.

For example,

```
int GetValue(int x,int y){
    return x+y;
}
void main(void)
{
    int a=10;
    float x=3.5;
    int result=GetValue(a,x);// x should be float, not int.
}
```

15 Error: Multiple matching signatures to 'GetData(uint8, const int, const string, const string, const int, const int)' Indicates that the format of the data parameter is incorrect. Prefix with const is correct, and vice versa is incorrect

The format of functions are,

Declaration: GetData(uint16,const int,const string,const string,const int,const int);

Usage: GetData(uint8,const int,const string,const string,const int,const int);

Any format other than this format will cause a compile error.

16 float variables cannot be contained in shift calculation

17 Error: A function with the same name and parameters already exists

17.13 Sample Marco code

- "for" statement and other expressions (arithmetic, bitwise shift, logic and comparison)

```
void main(void)
{
		array<int> a(10),b(10);
		b[0] = ( 400 + 400 << 2 ) / 401;
		b[1] = 22 * 2- 30 %7;
		b[2] = 111>>2 ;
		b[3] = 403 > 9 + 3 ? 1 : 0;
		b[4] = 8 + 1 & 2 + 1 | 0 + 1 ^ 2;
		b[5] = 405 & 3 & 0 ;
		b[6] = 4 + 4 & 4 + 8 | 4 + 8 ^ 4 ;
		b[7] = 6- ( ~4 ) ;
		b[8] = 0x11 ;
		b[9] = 409 ;
		for(int i = 0 ; i < 10 ; i++)
		{
			if(b[0] > 1)
			{
				GetData(a,i,"Local HMI","LW",100+i,1);
				GetData(b,i,"Local HMI","LW",150+i,1);
			}
		}
}
```

- While, if, break Statements.

```
void main(void)
{
	array<int> b(10);
	int i=5;
	while(i== 5- 20 %3)
	{
		GetData(b,1,"Local HMI","LW",11,1);
		if(b[1] == 100) break;
	}
}
```

- Global variables and function call

```
int GetValue(int x,int y)
{
		return x+y;
}
void main(void)
{
		int value=8;
		int z=2;
		GetValue(value,z) ;
}
```

- If Statement.

```
void main(void)
{
		array<int> k(10);
		for(int i=0;i<10;i++)
```

```
        {
                k[i] = i;
        }
        if(k[0] == 0)
        {
                SetData(k[0],0,"Local HMI","LB",0,1);
        }
        if(k[1] ==0)
        {
                SetData(k[1],0,"Local HMI","LB",1,1);
        }else
        {
                SetData(k[1],0,"Local HMI","LB",2,1);
        }
        if(k[2] >3)
        {
                SetData(k[2],0,"Local HMI","LW",1,1);
        }else if(k[2] >0)
        {
                SetData(k[2],0,"Local HMI","LW",2,1);
        }
        if(k[5] > k[4])
        {
                SetData(k[5],0,"Local HMI","LW",3,1);
        }else if(k[4] > k[3])
        {
                SetData(k[4],0,"Local HMI","LW",4,1);
        }else
        {
                SetData(k[3],0,"Local HMI","LW",5,1);
        }
}
```

- “While” and “Wend” Statement.

```
void main(void)
{
        uint8 i=0;
        array<int> a(13),b(14);
        int c=4848;
        b[0]=13;
        while(b[0]>0)
        {
                a[i] = 20 + i * 10;
                if(a[i] == 120)
               {
                    c=200;
                    break;
                }
                i= i+1;
        }
        SetData(c,0,"Local HMI","LW",23,1);
}
```

- “break” and “continue” statements.

```
void main(void)
{
        uint8 i=0;
        array<int> a(13),b(14);
        int c=4848;
        b[0]=13;
        while(b[0]>0)
        {
            a[i] = 20 + i * 10;
            if(a[i] == 120)
            {
                c=200;
                i= i+1;
                continue;
            }
            i= i+1;
            if(c==200)
            {
                SetData(c,0,"Local HMI","LW",43,1);
                break;
            }
        }
}
```

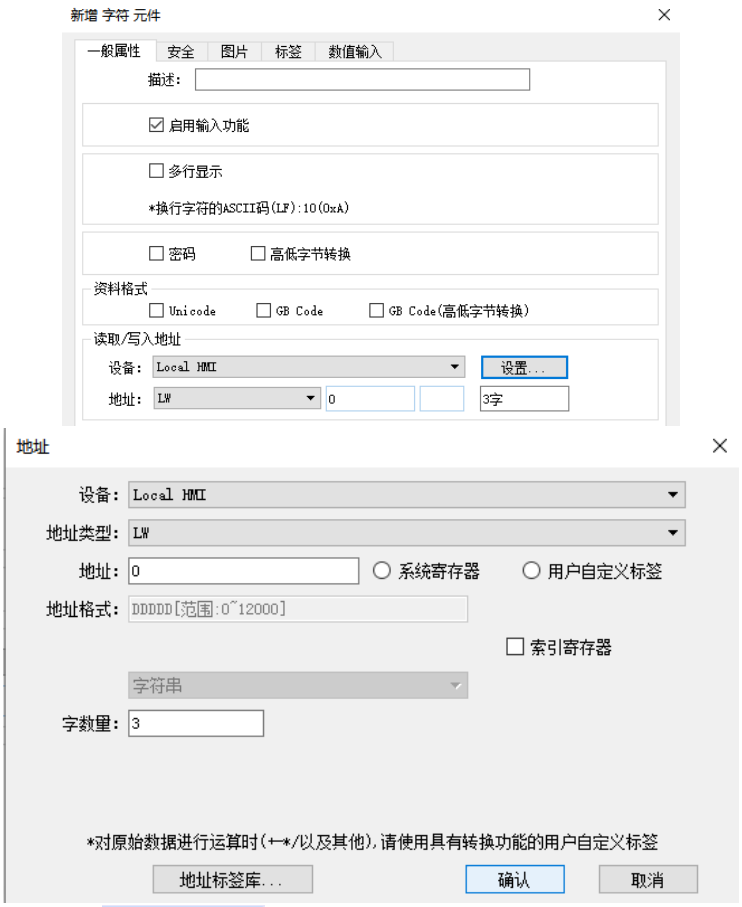
- Array

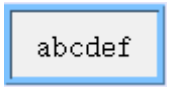
```
void main(void)
{
        array<int> a(25),b(25);
        b[0] = 13;
        for(int i=0;i<b[0];i++){
            a[i] = 20 + i * 10;
        }
        SetData(a,0,"Local HMI","LW",20,13);
}
```

17.14 Marco TRACE Function (in developing)

17.15 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:



Input “abcdef” in this ASCII object, 

The string “abcdef” is stored in LW-0~LW-2 as follows (LB represents low byte and HB represents high byte):

	HB	LB
LW-0	B	A
LW-1	D	C
LW-2	F	E

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.

Function Name	Description
StringGet	Receive the string data from device (PLC).
StringGetEx	Receives data from the device and continues executing next command even if there’s no response from the device.
StringSet	Sends string data to the device.
StringSetEx	Sends data to the device and continues executing next command even if there’s no response from the device.
StringCopy	Copies one string to another

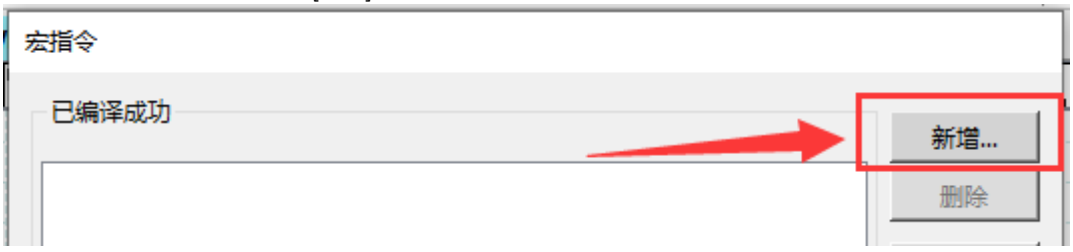
StringMid	Retrieves a character sequence from the specified offset of the source string.
StringDecAsc2Bin	Converts a decimal string to an integer
StringBin2DecAsc	Converts an integer to a decimal string.
StringDecAsc2Float	Converts a decimal string to floats.
StringFloat2DecAsc	Converts a float to a decimal string.
StringHexAsc2Bin	Converts a hexadecimal string to binary data.
StringBin2HexAsc	Converts binary data to a hexadecimal string.
StringLength	Obtains the length of a string.
StringCat	Appends source string to destination string.
StringCompare	Performs a case-sensitive comparison of two strings.
StringCompareNoCase	Performs a case-insensitive comparison of two strings
StringFind	Returns the zero-based index of the first character of substring in the source string that matches the target string.

As for specifications and usage of all String Operation Functions in above table, Please refer to Chapter 17.7 Built-in Function for detail. 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.

Example 1

This example to show how to read (or write) a string from device register.

1. Create a new Marco: [007]Go



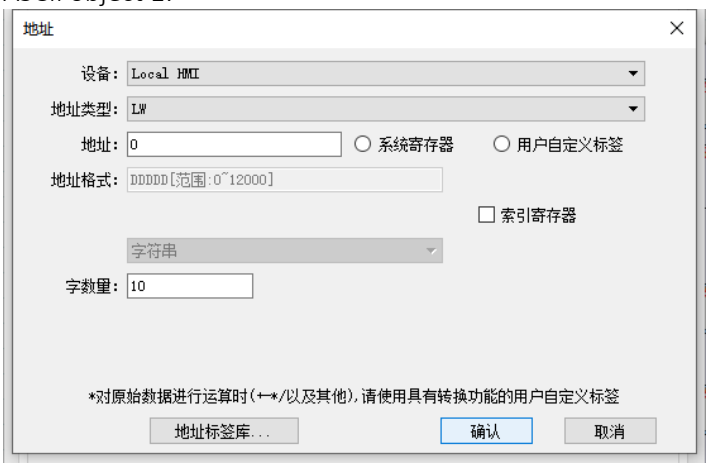
2. Edit the content as below:

```
1
2  void main(void)
3  {
4      array<uint8> str(20);
5
6      GetData(str,0,"Local HMI","LW",0,20);
7
8      SetData(str,0,"Local HMI","LW",50,20);
9  }
```

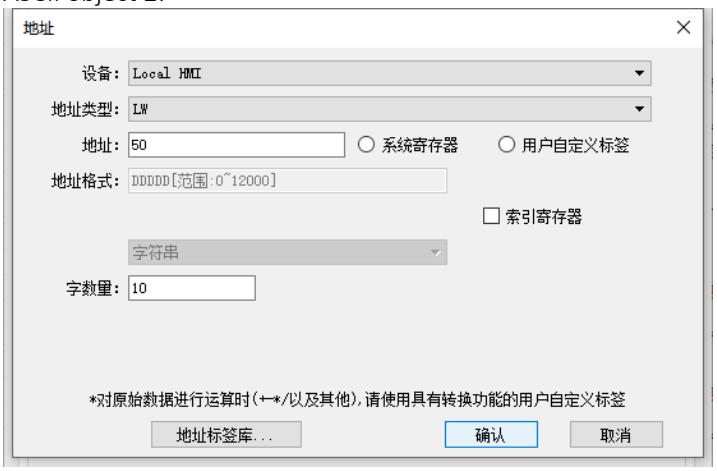
The first function “GetData” is used to read a string from LW-0~LW-19, and store it into the str array. The second function “SetData” is used to output the content of str array.

3. Add two “char” 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 [007]Go.

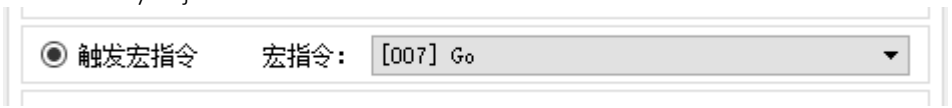
ASCII object 1:



ASCII object 2:



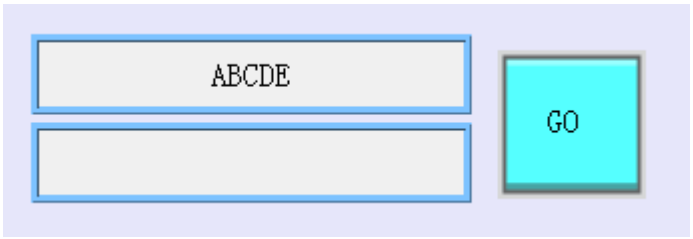
Function Key object:



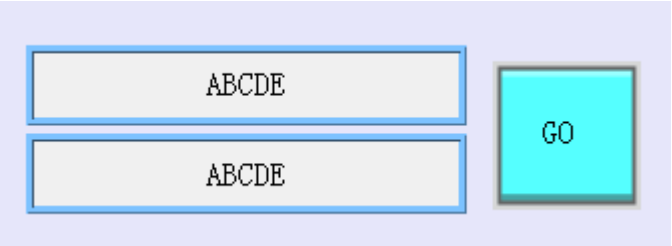
4. use [Compile] to compile the project and execute [Off-line simulation] or [On-line simulation]. Follow the steps below to operate the executing project:

a. Input string

b. Press ‘GO’ button

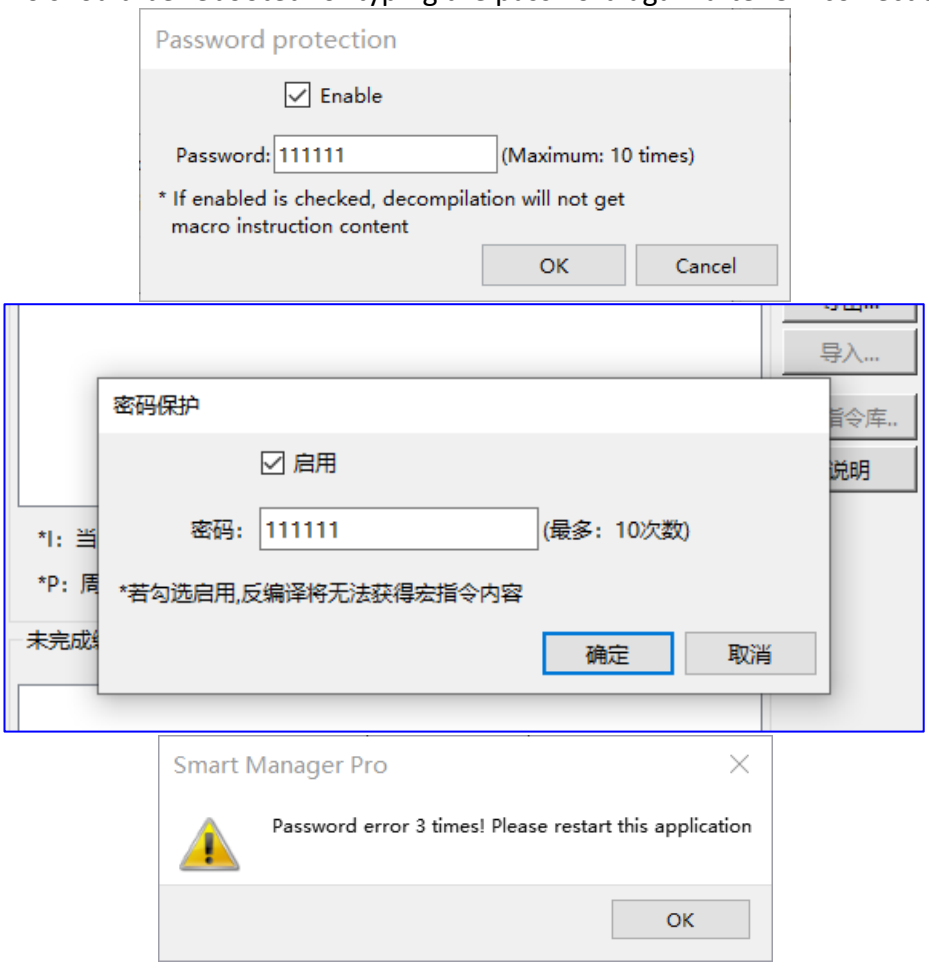


c. Output string and display the result



17.16 Macro Password Protection

In Macro Manager window there's the [Password Protect…] button, click it and then click [Enable] to set a password less than or equals to 10 characters (support ASCII character only, e.g. "a$#*hFds"). After setting the password, users will have to enter correct password when opening Macro Manager. SmartManager Pro should be rebooted for typing the password again after 3 incorrect attempts.



Note

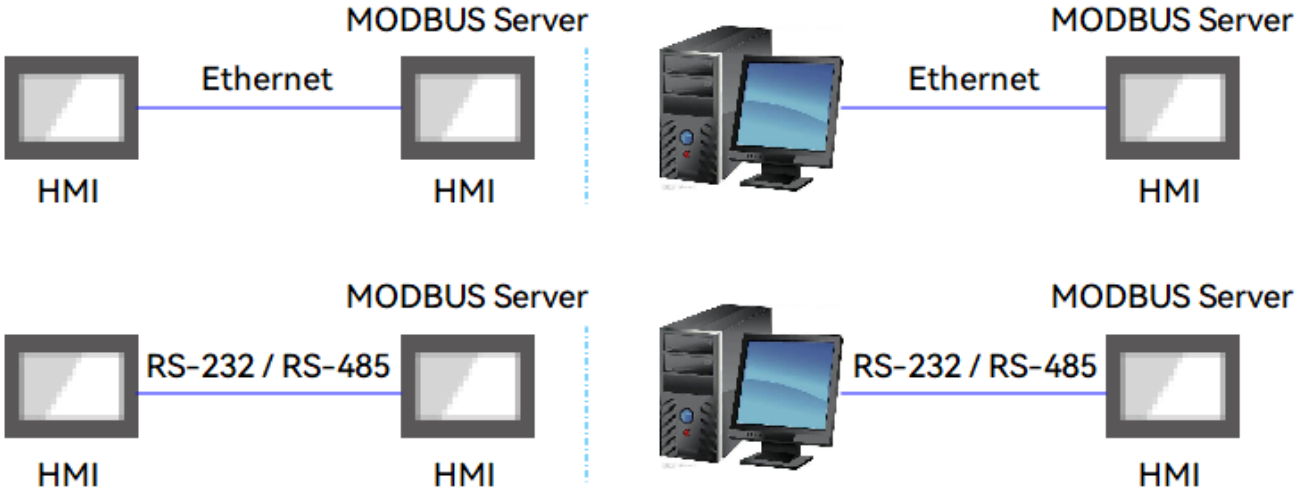
- When the macro password protection function is enabled, the macro content cannot be obtained by decompilation.

Chapter 18 Configure HMI as a MODBUS Server

This chapter explains how to configure HMI as a MODBUS Server.

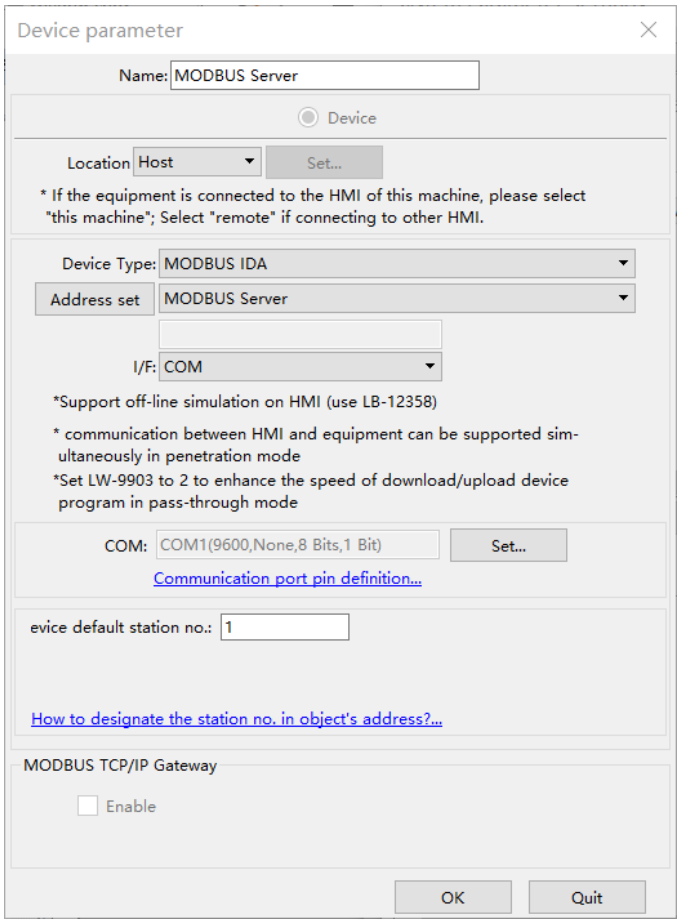
18.1 Overview

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 RS-232 / RS-485 interface.

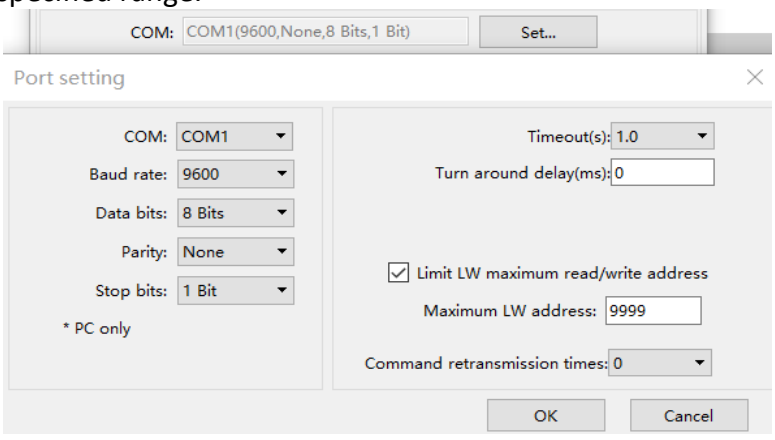


18.2 Steps to Create a MODBUS Server

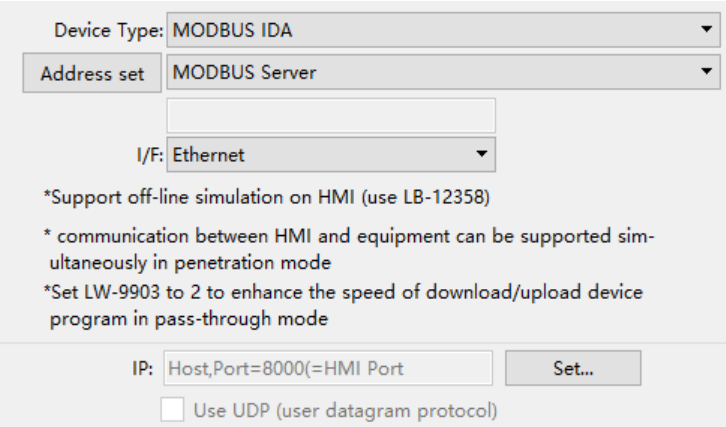
1. To configure the HMI as a MODBUS device, add a new device to the device list in [System Parameters] first. Click [PLC type] drop-down box and select “MODBUS Server” driver. [PLC I/F] includes: RS-232 / RS-485 2W / RS-485 4W / Ethernet / USB / CAN BUS. Choose the PLC interface appropriate for your PLC model.



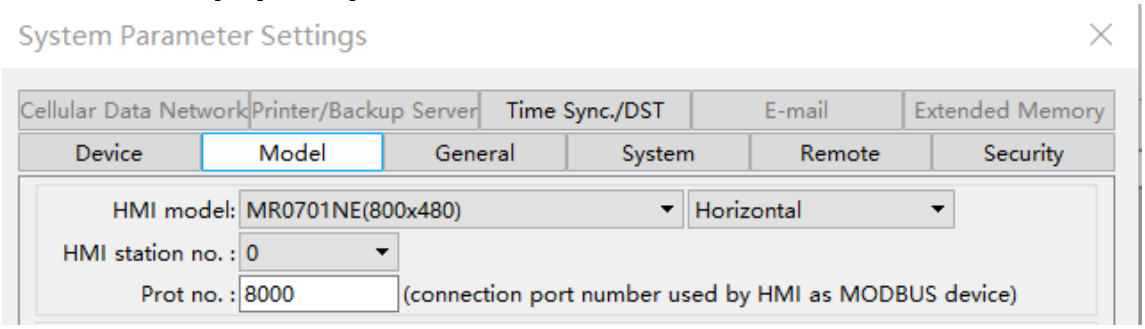
2. If [PLC I/F] is set to [RS-232] or [RS-485], please select [COM] (COM 1 ~ COM 3) and set correct communication parameters as shown in the following figure. MODBUS Server [Station no.] is set to 1. Click [Settings], the maximum LW address range read / written by Modbus Client can be set. When the object 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.



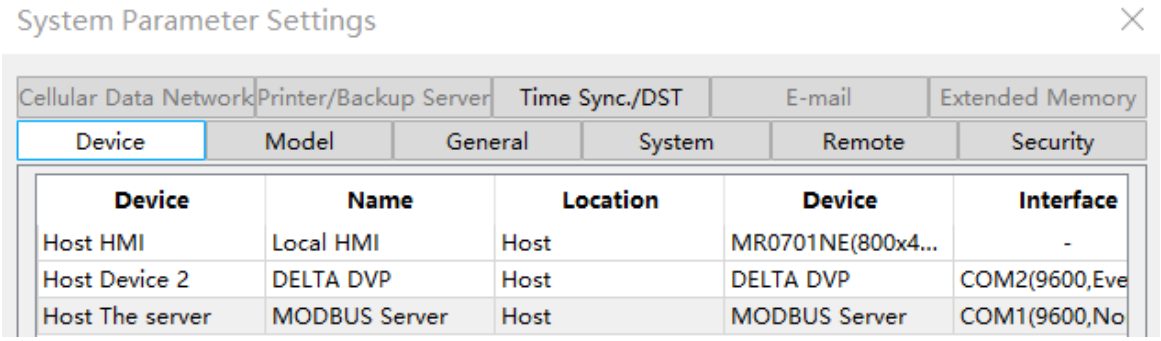
If [PLC I/F] is set to [Ethernet], please set [Port no.].



The [Port no.] of MODBUS Server and HMI must be the same. To change the port number, please set in the [System Parameters] » [Model] tab.



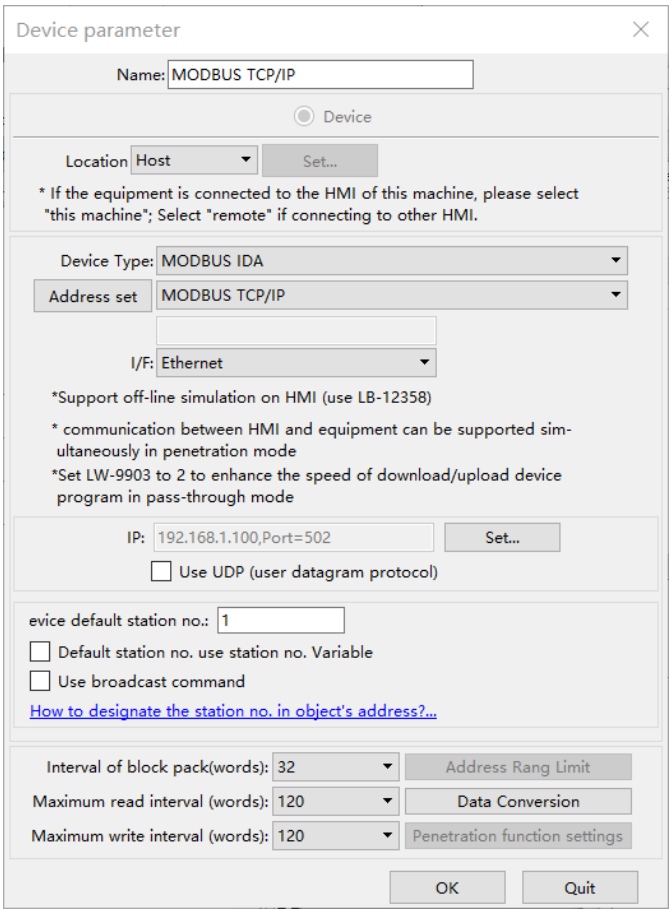
3. When finished, MODBUS Server is listed in [Device] tab. The configuration of MODBUS device is completed. Compile the .hmi file and download the compiled .exhmi file to the HMI, then HMI data can be read or written by using MODBUS protocol.



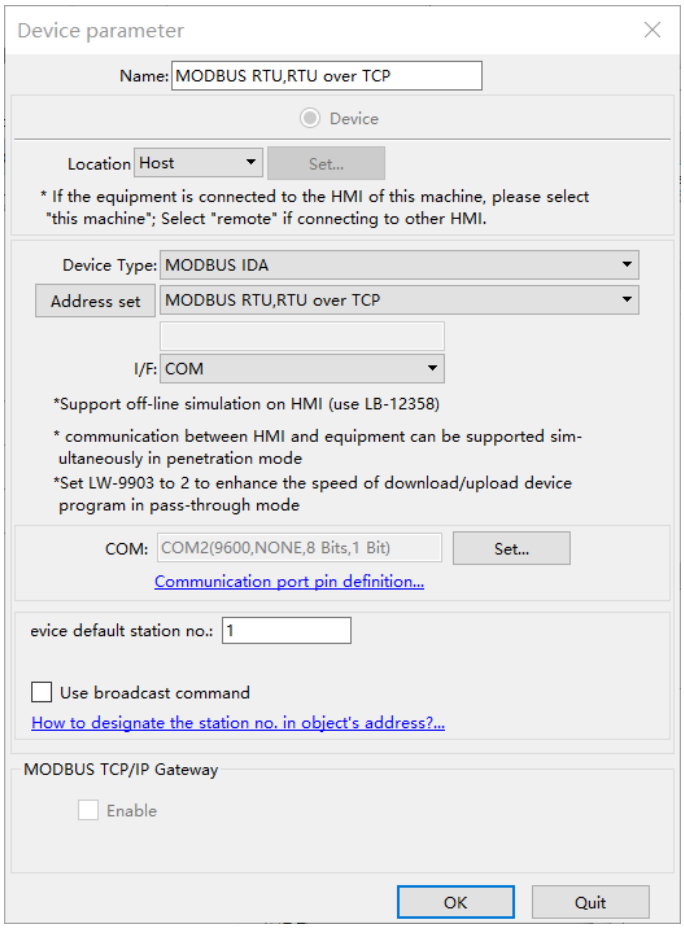
18.3 Steps to Access a MODBUS Server

Two HMIs can be configured as one MODBUS client and one MODBUS server to communicate and exchange data.

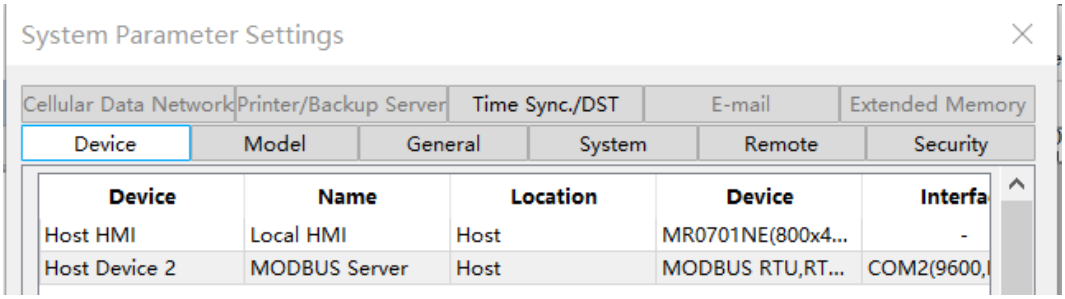
1. Add a new device in client's device list. If the client chooses [Ethernet] PLC interface, set [PLC type] to "MODBUS TCP/IP" and fill in the correct [IP address] (the IP of MODBUS Server), [Port no.], and [Station no.].



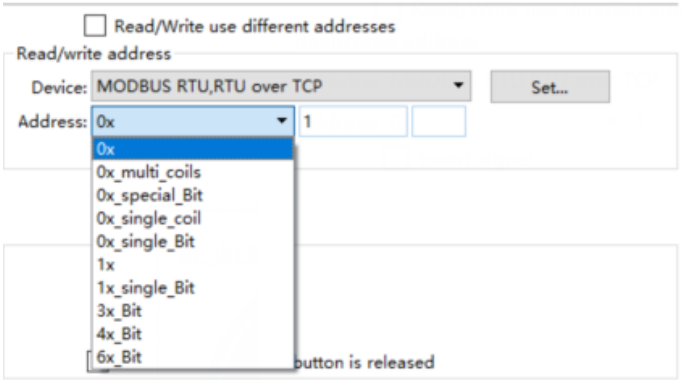
If the client chooses [RS-232] or [RS-485] PLC interface, the [PLC type] must be set to “MODBUS RTU”, and its communication parameters also must be configured correctly.



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：

Read/write 0x/1x (1 ~ 12800)	Corresponds to read and write LB (0 ~ 12799)
Read/write 3x/4x/5x (1 ~ 9999)	Corresponds to read and write LW (0 ~ 9998)
Read/write 3x/4x/5x (10000 ~ 65535)	Corresponds to read and write RW (0 ~ 55535)

18.4 Changing MODBUS Server Station Number Online

SmartManager Pro provides the following system registers to change MODBUS Server station number online.

LW-9541	MODBUS/ASCII server Station Number	(COM 1)
LW-9542	MODBUS/ASCII server Station Number	(COM 2)
LW-9543	MODBUS/ASCII server Station Number	(COM 3)
LW-9544	MODBUS/ASCII server Station Number	(Ethernet)

18.5 MODBUS Address Types

In the SmartManager 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	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	A read only device type. When reading a bit the function code is 02H.
3x	A read only device type. When reading data, the function code is 04H.
4x	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.
6x_bit	The function code is the same as 6x. The difference is that 6x_bit reads a single bit in the data.

Chapter 19 How to Connect to Barcode Scanner (in developing)

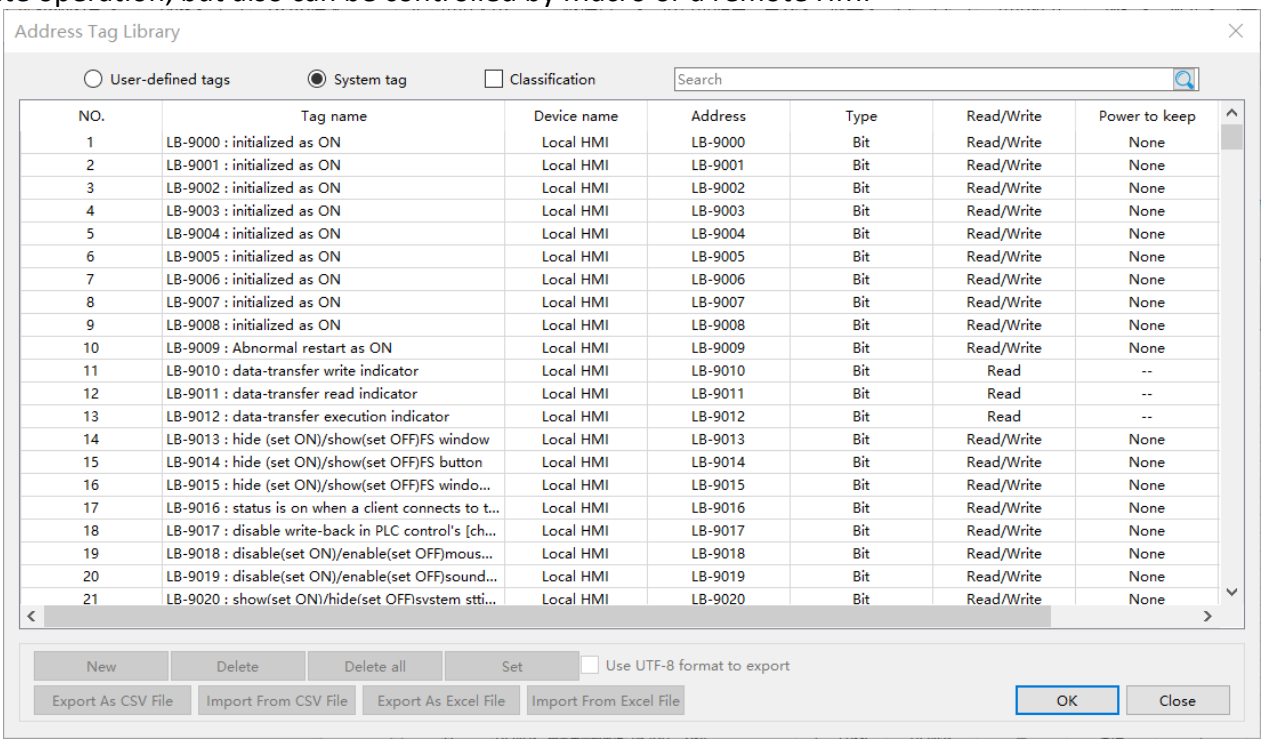
Chapter 20 Ethernet Communication & Multi-HMI Connection (in developing)

Chapter 21 System Registers

This chapter introduces different types of registers.

21.1 Overview

Some Word and Bit addresses are reserved in SmartManager 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



21.2 The Address Ranges of Local HMI

21.2.1 Bits

Register	Device Type	Range	Format
Local Bits	LB	0 ~ 12399	DDDDD
Local ASCII address taking address	LW_Bit	0 ~ 1200015	DDDDDdd DDDDD: Address dd: Bit number (00 ~ 15)
Retentive Bit Index	RBI	0 ~ 65535f	DDDDDh

			DDDDD: Address h: bit number (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 number (0 ~ f)
Retentive A Word Bits	RW_A_Bit	0 ~ 65535f	DDDDDh DDDDD: Address h: bit number (0 ~ f)

21.2.2 Words

Register	Device Type	Range	Format
Local words	LW	0 ~ 12000	DDDDD
Retentive Words	RW	0 ~ 524287	DDDDDD
Retentive A Words	RW_A	0 ~ 65535	DDDDD
Retentive Word Index	RWI	0 ~ 65535	DDDDD Use LW-9000 as Index Register, and correspond to RW
Extended Memory Words	EM0~EM9	0 ~ 1073741823	DDDDDDDDDD

21.3 System Registers (the items in red is in developing)

21.3.1 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 读/控制
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) : Time (12-hour format)	R/W	R/C	R/C

Note

1. Value range: 2000 ~ 2037
2. Value range: 0 ~ 6, stand for Sunday ~ Saturday
3. When using LW-9010 to LW-9023 to update RTC time and an invalid value is set, the time before the update will be restored and the system register [LB-11958: time setting error] will be set ON. Updating time on PC during simulation by using LW-9010 to LW-9023 is ineffective.

21.3.2 HMI Operation

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9018	disable(set ON)/enable (set OFF) mouse cursor	R/W	R/C	R/C
LB-9019	disable(set ON)/enable (set OFF) sound output	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) popping information dialog while finding an USB disk	R/W	R/C	R/C
LB-9064	enable USB barcode device (disable keyboard) (when ON)	R/W	R/C	R
LB-11959	LED indicator control	R/W	R/C	R/C
LB-12042	open/close [System information] dialog (set ON/set OFF)	R/W	R/C	R/C
LB-12051	buzzer control (active when ON)	R/W	R/C	R/C

LB-12360	CPU loading alarm (> 95%) *Note 3	R	R	R
LB-12364	show (set ON)/hide (set OFF) [Reset HMI to default] button in calibration mode	R/W	R/C	R/C
LW-9007	(16bit) : hardware index	R	R	R
LW-9008	(32bit-float) : battery voltage	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-9054	(32bit) : HMI model ID	R	R	R
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-9141	(16bit) : HMI station no.	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 (K bytes)	R	R	R
LW-11157	(32bit) : the free size of HMI memory (K bytes)	R	R	R
LW-11159	(16bit) : memory loading (x 100%)	R	R	R
LW-11382	(16bit) : DIP switch status (bit 0 : DIP 1, bit 1 : DIP 2, bit 2 : DIP 3)	R	R	R

Note

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. This register will be set to ON if CPU loading remains over 95% for 30 seconds.

21.3.3 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

21.3.4 Local HMI Network Information

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-12041	refresh HMI ethernet information (DHCP, gateway, netmask, IP) (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,) (set ON)	R/W	R/C	R/C
LW-9125	(16bit): HMI Ethernet 1 gateway 0 (for HMI use only)	R/W	R/C	R/C
LW-9126	(16bit): HMI Ethernet 1 gateway 1 (for HMI use only)	R/W	R/C	R/C
LW-9127	(16bit): HMI Ethernet 1 gateway 2 (for HMI use only)	R/W	R/C	R/C
LW-9128	(16bit): HMI Ethernet 1 gateway 3 (for HMI use only)	R/W	R/C	R/C
LW-9129	(16bit): HMI Ethernet 1 IP 0 (for HMI use only)	R/W	R/C	R/C
LW-9130	(16bit): HMI Ethernet 1 IP 1 (for HMI use only)	R/W	R/C	R/C
LW-9131	(16bit): HMI Ethernet 1 IP 2 (for HMI use only)	R/W	R/C	R/C
LW-9132	(16bit): HMI Ethernet 1 IP 3 (for HMI use only)	R/W	R/C	R/C
LW-9133	(16bit): Ethernet port no. (for HMI use only)	R	R	R
LW-9135	(16bit): HMI media access control (MAC)address 0	R	R	R
LW-9136	(16bit): HMI media access control (MAC)address 1	R	R	R
LW-9137	(16bit): HMI media access control (MAC)address 2	R	R	R
LW-9138	(16bit): HMI media access control (MAC)address 3	R	R	R
LW-9139	(16bit): HMI media access control (MAC)address 4	R	R	R
LW-9140	(16bit): HMI media access control (MAC)address 5	R	R	R
LW-10750	(16bit): HMI Ethernet 1 Mask 0 (for HMI use only)	R/W	R/C	R/C
LW-10751	(16bit): HMI Ethernet 1 Mask 1 (for HMI use only)	R/W	R/C	R/C
LW-10752	(16bit): HMI Ethernet 1 Mask 2 (for HMI use only)	R/W	R/C	R/C
LW-10753	(16bit): HMI Ethernet 1 Mask 3 (for HMI use only)	R/W	R/C	R/C
LW-10786	(16bit): HMI Ethernet 2 IP 0 (for HMI use only)	R/W	R/C	R/C

LW-10787	(16bit): HMI Ethernet 2 IP 1 (for HMI use only)	R/W	R/C	R/C
LW-10788	(16bit): HMI Ethernet 2 IP 2 (for HMI use only)	R/W	R/C	R/C
LW-10789	(16bit): HMI Ethernet 2 IP 3 (for HMI use only)	R/W	R/C	R/C
LW-10790	(16bit): HMI Ethernet 2 Mask 0 (for HMI use only)	R/W	R/C	R/C
LW-10791	(16bit): HMI Ethernet 2 Mask 1 (for HMI use only)	R/W	R/C	R/C
LW-10792	(16bit): HMI Ethernet 2 Mask 2 (for HMI use only)	R/W	R/C	R/C
LW-10793	(16bit): HMI Ethernet 2 Mask 3 (for HMI use only)	R/W	R/C	R/C
LW-10794	(16bit): HMI Ethernet 2 Gateway 0 (for HMI use only)	R/W	R/C	R/C
LW-10795	(16bit): HMI Ethernet 2 Gateway 1 (for HMI use only)	R/W	R/C	R/C
LW-10796	(16bit): HMI Ethernet 2 Gateway 2 (for HMI use only)	R/W	R/C	R/C
LW-10797	(16bit): HMI Ethernet 2 Gateway 3 (for HMI use 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
LW-10813	(16bit): obtain an Ethernet 2 IP address automatically (DHCP => 0 : off, 1 : on)	R/W	R/C	R/C
LW-10815	(16bit): link speed of Ethernet 1 (0:failure, 10 (10M), 100 (100M), 1000 (1G))	R	R	R
LW-10816	(16bit): link speed of Ethernet 2 (0:failure, 10 (10M), 100 (100M), 1000 (1G))	R	R	R

21.3.5 Project File Information

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-9100	(16bit): project name (14 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): SmartManager Pro 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
LW-11440	(16bit): project compiled time [hour] (24-hour format)	R	R	R
LW-11441	(16bit): project compiled time [minute]	R	R	R
LW-11442	(16bit): project compiled time [second]	R	R	R

21.3.6 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 1 current free space (K bytes)	R	R	R
LW-11458	(32bit): HMI total space for history data (K bytes)	R	R	R
LW-11460	(32bit): HMI current free space for history data (K bytes)	R	R	R

21.3.7 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
LB-12363	Prohibit recipe database update from remote HMI (set ON)	R/W	R/C	R/C

21.3.8 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) *Note 1	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 (bytes)	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 (bytes)	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 (bytes)	R	R	R

Note

1. The shortest interval between two successful executions is 2 seconds.
2. The registers for deleting or updating data samplings do not work during simulation on PC.

21.3.9 Event Log

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9021	reset current event log (OFF->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) *Note 2	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 1 (set ON)	W	C	C
LB-11945	refresh event log information on USB disk (set ON)	W	C	C
LB-12399	status is on when alarm exists in any category	R	R	R
LB-12400	status is on when alarm exists in category 0	R	R	R
LB-12401	status is on when alarm exists in category 1	R	R	R
LB-12402	status is on when alarm exists in category 2	R	R	R
LB-12403	status is on when alarm exists in category 3	R	R	R
LB-12404	status is on when alarm exists in category 4	R	R	R
LB-12405	status is on when alarm exists in category 5	R	R	R
LB-12406	status is on when alarm exists in category 6	R	R	R
LB-12407	status is on when alarm exists in category 7	R	R	R
LB-12655	status is on when alarm exists in category 255	R	R	R
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 (bytes)	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 (bytes)	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 (bytes)	R	R	R
LW-11443	(16bit) : push notification alarm status (0 : none; 1 : green; 2 : yellow; 3 : red)	R	R	R
LW-11499	total no. of alarms	R	R	R
LW-11500	no. of alarms in category 0	R	R	R
LW-11501	no. of alarms in category 1	R	R	R

LW-11502	no. of alarms in category 2	R	R	R
LW-11503	no. of alarms in category 3	R	R	R
LW-11504	no. of alarms in category 4	R	R	R
LW-11505	no. of alarms in category 5	R	R	R
LW-11506	no. of alarms in category 6	R	R	R
LW-11507	no. of alarms in category 7	R	R	R
LW-11755	no. of alarms in category 255	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 shortest interval between two successful executions is 2 seconds.
3. The registers for deleting or updating event logs do not work during simulation on PC.

21.3.10 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

21.3.11 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

21.3.12 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

21.3.13 Communication Parameter Settings

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9030	activate COM 1 new communication settings (LW-9550~9554) (set ON)	R/W	R/C	R/C
LB-9031	activate COM 2 new communication settings (LW-9555~9559) (set ON)	R/W	R/C	R/C
LB-9032	activate COM 3 new communication settings (LW-9560~9564) (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) (use LB-9030 to activate all communication settings)	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,..) *Note 1	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) (use LB-9031 to activate all communication settings)	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,..) *Note 1	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) (use LB-9032 to activate all communication settings)	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,..) *Note 1	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): device 1 timeout (unit : 100ms, 0 : 50ms)	R/W	R/C	R/C
LW-10501	(16bit): device 1 turn around delay (unit : ms)	R/W	R/C	R/C
LW-10502	(16bit): device 1 send ACK delay (unit : ms)	R/W	R/C	R/C
LW-10503	(16bit): device 1 parameter 1	R/W	R/C	R/C
LW-10504	(16bit): device 1 parameter 2	R/W	R/C	R/C
LW-10505	(16bit): device 2 timeout (unit : 100ms, 0 : 50ms)	R/W	R/C	R/C
LW-10506	(16bit): device 2 turn around delay (unit : ms)	R/W	R/C	R/C
LW-10507	(16bit): device 2 send ACK delay (unit : ms)	R/W	R/C	R/C
LW-10508	(16bit): device 2 parameter 1	R/W	R/C	R/C
LW-10509	(16bit): device 2 parameter 2	R/W	R/C	R/C
LW-10510	(16bit): device 3 timeout (unit : 100ms, 0 : 50ms)	R/W	R/C	R/C
LW-10511	(16bit): device 3 turn around delay (unit : ms)	R/W	R/C	R/C
LW-10512	(16bit): device 3 send ACK delay (unit : ms)	R/W	R/C	R/C
LW-10513	(16bit): device 3 parameter 1	R/W	R/C	R/C
LW-10514	(16bit): device 3 parameter 2	R/W	R/C	R/C
LW-10515	(16bit): device 4 timeout (unit : 100ms)	R/W	R/C	R/C
LW-10516	(16bit): device 4 turn around delay (unit : ms)	R/W	R/C	R/C
LW-10517	(16bit): device 4 send ACK delay (unit : ms) (SIEMENS S7/400 Link type)	R/W	R/C	R/C
LW-10518	(16bit): device 4 parameter 1 (SIEMENS S7/400 rack)	R/W	R/C	R/C
LW-10519	(16bit): device 4 parameter 2 (SIEMENS S7/400 CPU slot)	R/W	R/C	R/C
LW-10520	(16bit): device 5 timeout (unit : 100ms)	R/W	R/C	R/C

LW-10521	(16bit): device 5 turn around delay (unit : ms)	R/W	R/C	R/C
LW-10522	(16bit): device 5 send ACK delay (unit : ms) (SIEMENS S7/400 Link type)	R/W	R/C	R/C
LW-10523	(16bit): device 5 parameter 1 (SIEMENS S7/400 rack)	R/W	R/C	R/C
LW-10524	(16bit): device 5 parameter 2 (SIEMENS S7/400 CPU slot)	R/W	R/C	R/C
LW-10525	(16bit): device 6 timeout (unit : 100ms)	R/W	R/C	R/C
LW-10526	(16bit): device 6 turn around delay (unit : ms)	R/W	R/C	R/C
LW-10527	(16bit): device 6 send ACK delay (unit : ms) (SIEMENS S7/400 Link type)	R/W	R/C	R/C
LW-10528	(16bit): device 6 parameter 1 (SIEMENS S7/400 rack)	R/W	R/C	R/C
LW-10529	(16bit): device 6 parameter 2 (SIEMENS S7/400 CPU slot)	R/W	R/C	R/C
LW-10530	(16bit): device 7 timeout (unit : 100ms)	R/W	R/C	R/C
LW-10531	(16bit): device 7 turn around delay (unit : ms)	R/W	R/C	R/C
LW-10532	(16bit): device 7 send ACK delay (unit : ms) (SIEMENS S7/400 Link type)	R/W	R/C	R/C
LW-10533	(16bit): device 7 parameter 1 (SIEMENS S7/400 rack)	R/W	R/C	R/C
LW-10534	(16bit): device 7 parameter 2 (SIEMENS S7/400 CPU slot)	R/W	R/C	R/C
LW-10535	(16bit): device 8 timeout (unit : 100ms)	R/W	R/C	R/C
LW-10536	(16bit): device 8 turn around delay (unit : ms)	R/W	R/C	R/C
LW-10537	(16bit): device 8 send ACK delay (unit : ms) (SIEMENS S7/400 Link type)	R/W	R/C	R/C
LW-10538	(16bit): device 8 parameter 1 (SIEMENS S7/400 rack)	R/W	R/C	R/C
LW-10539	(16bit): device 8 parameter 2 (SIEMENS S7/400 CPU slot)	R/W	R/C	R/C
LW-10655	(16bit): device 32 timeout (unit : 100ms)	R/W	R/C	R/C
LW-10656	(16bit): device 32 turn around delay (unit : ms)	R/W	R/C	R/C
LW-10657	(16bit): device 32 send ACK delay (unit : ms)	R/W	R/C	R/C
LW-10658	(16bit): device 32 parameter 1	R/W	R/C	R/C
LW-10659	(16bit): device 32 parameter 2	R/W	R/C	R/C

Note

1. The Baud Rates are: 0:4800, 1:9600, 2:19200, 3:38400, 4:57600, 5:115200, 6:187.5K, 7:1200, 8:2400, 10:14400, 11:28800, 12:76800.

21.3.14 Communication Status and Control with Device (COM)

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9150	auto. connection for device 1 (COM 1) (when ON)	R/W	R/C	R/C
LB-9151	auto. connection for device 2 (COM 2) (when ON)	R/W	R/C	R/C
LB-9152	auto. connection for device 3 (COM 3) (when ON)	R/W	R/C	R/C
LB-9200	device 1 status (SN0, COM 1), set on to retry connection.	R/W	R/C	R/C
LB-9201	device 1 status (SN1, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9202	device 1 status (SN2, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9203	device 1 status (SN3, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9204	device 1 status (SN4, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9205	device 1 status (SN5, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9206	device 1 status (SN6, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9207	device 1 status (SN7, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9455	device 1 status (SN255, COM 1), set on to retry connection	R/W	R/C	R/C
LB-9500	device 2 status (SN0, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9501	device 2 status (SN1, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9502	device 2 status (SN2, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9503	device 2 status (SN3, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9504	device 2 status (SN4, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9505	device 2 status (SN5, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9506	device 2 status (SN6, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9507	device 2 status (SN7, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9755	device 2 status (SN255, COM 2), set on to retry connection	R/W	R/C	R/C
LB-9800	device 3 status (SN0, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9801	device 3 status (SN1, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9802	device 3 status (SN2, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9803	device 3 status (SN3, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9804	device 3 status (SN4, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9805	device 3 status (SN5, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9806	device 3 status (SN6, COM 3), set on to retry connection	R/W	R/C	R/C
LB-9807	device 3 status (SN7, COM 3), set on to retry connection	R/W	R/C	R/C

LB-10055	device 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 device 1 (COM1)	R	R	R
LW-9352	(16bit): pending command no. in device 2 (COM2)	R	R	R
LW-9353	(16bit): pending command no. in device 3 (COM3)	R	R	R

Note

- The ON state of COM is for checking if COM is occupied by other program during simulation on PC.

21.3.15 Communication Status and Control with Device (Ethernet)

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9153	auto. connection for device 4 (Ethernet) (when ON)	R/W	R/C	R/C
LB-9154	auto. connection for device 5 (Ethernet) (when ON)	R/W	R/C	R/C
LB-9155	auto. connection for device 6 (Ethernet) (when ON)	R/W	R/C	R/C
LB-9156	auto. connection for device 7 (Ethernet) (when ON)	R/W	R/C	R/C
LB-9157	auto. connection for device 8 (Ethernet) (when ON)	R/W	R/C	R/C
LB-9158	auto. connection for device 9 (Ethernet) (when ON)	R/W	R/C	R/C
LB-9189	auto. connection for device 40 (Ethernet) (when ON)	R/W	R/C	R/C
LB-10070	forced to reconnect device 4 (Ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10071	forced to reconnect device 5 (Ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10072	forced to reconnect device 6 (Ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10073	forced to reconnect device 7 (Ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C

LB-10074	forced to reconnect device 8 (Ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10075	forced to reconnect device 9 (Ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10099	forced to reconnect device 33 (Ethernet) when IP or system parameters changed on-line (set ON)	R/W	R/C	R/C
LB-10100	device 4 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10101	device 4 status (SN0, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10102	device 4 status (SN1, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10103	device 4 status (SN2, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10104	device 4 status (SN3, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10105	device 4 status (SN4, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10106	device 4 status (SN5, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10107	device 4 status (SN6, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10108	device 4 status (SN7, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10356	device 4 status (SN255, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10400	device 5 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10401	device 5 status (SN0, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10402	device 5 status (SN1, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10403	device 5 status (SN2, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10404	device 5 status (SN3, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10405	device 5 status (SN4, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10406	device 5 status (SN5, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10407	device 5 status (SN6, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10408	device 5 status (SN7, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10656	device 5 status (SN255, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10700	device 6 status (Ethernet), set on to retry connection	R/W	R/C	R/C

LB-10701	device 6 status (SN0, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10702	device 6 status (SN1, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10703	device 6 status (SN2, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10704	device 6 status (SN3, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10705	device 6 status (SN4, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10706	device 6 status (SN5, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10707	device 6 status (SN6, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10708	device 6 status (SN7, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-10956	device 6 status (SN255, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11000	device 7 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11001	device 7 status (SN0, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11002	device 7 status (SN1, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11003	device 7 status (SN2, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11004	device 7 status (SN3, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11005	device 7 status (SN4, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11006	device 7 status (SN5, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11007	device 7 status (SN6, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11008	device 7 status (SN7, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11256	device 7 status (SN255, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11300	Device8 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11301	device 8 status (SN0, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11302	device 8 status (SN1, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11303	device 8 status (SN2, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11304	device 8 status (SN3, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11305	device 8 status (SN4, Ethernet), set on to retry connection	R/W	R/C	R/C

LB-11306	device 8 status (SN5, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11307	device 8 status (SN6, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11308	device 8 status (SN7, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11556	device 8 status (SN255, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11600	device 9 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11601	device 9 status (SN0, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11602	device 9 status (SN1, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11603	device 9 status (SN2, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11604	device 9 status (SN3, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11605	device 9 status (SN4, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11606	device 9 status (SN5, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11607	device 9 status (SN6, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11608	device 9 status (SN7, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11856	device 9 status (SN255, Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11900	device 10 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11901	device 11 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11902	device 12 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11903	device 13 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11904	device 14 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11905	device 15 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11906	device 16 status (Ethernet), set on to retry connection	R/W	R/C	R/C
LB-11939	device 49 status (Ethernet), set on to retry connection	R/W	R/C	R/C

LW-9354	(16bit): pending command no. in device 4 (Ethernet)	R	R	R
LW-9355	(16bit): pending command no. in device 5 (Ethernet)	R	R	R
LW-9356	(16bit): pending command no. in device 6 (Ethernet)	R	R	R
LW-9357	(16bit): pending command no. in device 7 (Ethernet)	R	R	R
LW-9389	(16bit): pending command no. in device 39 (Ethernet)	R	R	R
LW-9600	(16bit): device 4's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9601	(16bit): device 4's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9602	(16bit): device 4's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9603	(16bit): device 4's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9604	(16bit): device 4's port no.	R/W	R/C	R/C
LW-9605	(16bit): device 5's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9606	(16bit): device 5's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9607	(16bit): device 5's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9608	(16bit): device 5's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9609	(16bit): device 5's port no.	R/W	R/C	R/C
LW-9610	(16bit): device 6's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9611	(16bit): device 6's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9612	(16bit): device 6's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C

LW-9613	(16bit): device 6's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9614	(16bit): device 6's port no.	R/W	R/C	R/C
LW-9615	(16bit): device 7's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9616	(16bit): device 7's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9617	(16bit): device 7's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9618	(16bit): device 7's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9619	(16bit): device 7's port no.	R/W	R/C	R/C
LW-9620	(16bit): device 8's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9621	(16bit): device 8's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9622	(16bit): device 8's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9623	(16bit): device 8's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9624	(16bit): device 8's port no.	R/W	R/C	R/C
LW-9625	(16bit): device 9's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9626	(16bit): device 9's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9627	(16bit): device 9's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9628	(16bit): device 9's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9629	(16bit): device 9's port no.	R/W	R/C	R/C
LW-9765	(16bit): device 37's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C

LW-9766	(16bit): device 37's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9767	(16bit): device 37's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9768	(16bit): device 37's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-9769	(16bit): device 37's port no.	R/W	R/C	R/C
LW-11472	(16bit): device 4's ID0 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11473	(16bit): device 4's ID1 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11474	(16bit): device 4's ID2 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11475	(16bit): device 4's ID3 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11476	(16bit): device 4's ID4 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11477	(16bit): device 4's ID5 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11478	(16bit): device 5's ID0 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11479	(16bit): device 5's ID1 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11480	(16bit): device 5's ID2 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11481	(16bit): device 5's ID3 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11482	(16bit): device 5's ID4 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11483	(16bit): device 5's ID5 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11484	(16bit): device 6's ID0 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11485	(16bit): device 6's ID1 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C

LW-11486	(16bit): device 6's ID2 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11487	(16bit): device 6's ID3 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11488	(16bit): device 6's ID4 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11489	(16bit): device 6's ID5 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11490	(16bit): device 7's ID0 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11491	(16bit): device 7's ID1 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11492	(16bit): device 7's ID2 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11493	(16bit): device 7's ID3 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11494	(16bit): device 7's ID4 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C
LW-11495	(16bit): device 7's ID5 (Beckhoff AMS NetId = ID0:ID1:ID2:ID3:ID4:ID5)	R/W	R/C	R/C

21.3.16 Communication Status and Control with Device (USB)

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9190	auto. connection for device (USB) (when ON)	R/W	R/C	R/C
LB-9191	device status (USB), set on to retry connection	R/W	R/C	R/C
LW-9390	(16bit): pending command no. in device (USB)	R	R	R

21.3.17 Communication Status and Control with Device (CAN Bus)

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-12080	auto. connection for device (CAN Bus) (when ON)	R/W	R/C	R/C

LB-12081	device 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 device (CAN Bus)	R	R	R

21.3.18 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

21.3.19 Communication Status and Control with Remote Device

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-10050	(16bit): IP0 of the HMI connecting to remote device 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10051	(16bit): IP1 of the HMI connecting to remote device 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10052	(16bit): IP2 of the HMI connecting to remote device 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10053	(16bit): IP3 of the HMI connecting to remote device 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10054	(16bit): port no. of the HMI connecting to remote device 1	R/W	R/C	R/C
LW-10055	(16bit): IP0 of the HMI connecting to remote device 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10056	(16bit): IP1 of the HMI connecting to remote device 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10057	(16bit): IP2 of the HMI connecting to remote device 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10058	(16bit): IP3 of the HMI connecting to remote device 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10059	(16bit): port no. of the HMI connecting to remote device 2	R/W	R/C	R/C
LW-10060	(16bit): IP0 of the HMI connecting to remote device 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10061	(16bit): IP1 of the HMI connecting to remote device 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C

LW-10062	(16bit): IP2 of the HMI connecting to remote device 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10063	(16bit): IP3 of the HMI connecting to remote device 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10064	(16bit): port no. of the HMI connecting to remote device 3	R/W	R/C	R/C
LW-10065	(16bit): IP0 of the HMI connecting to remote device 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10066	(16bit): IP1 of the HMI connecting to remote device 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10067	(16bit): IP2 of the HMI connecting to remote device 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10068	(16bit): IP3 of the HMI connecting to remote device 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10069	(16bit): port no. of the HMI connecting to remote device 4	R/W	R/C	R/C
LW-10205	(16bit): IP0 of the HMI connecting to remote device 32 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10206	(16bit): IP1 of the HMI connecting to remote device 32 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10207	(16bit): IP2 of the HMI connecting to remote device 32 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10208	(16bit): IP3 of the HMI connecting to remote device 32 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10209	(16bit): port no. of the HMI connecting to remote device 32	R/W	R/C	R/C
LW-10300	(16bit): remote device 1's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10301	(16bit): remote device 1's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10302	(16bit): remote device 1's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10303	(16bit): remote device 1's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10304	(16bit): remote device 1's port no.	R/W	R/C	R/C
LW-10305	(16bit): remote device 2's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10306	(16bit): remote device 2's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10307	(16bit): remote device 2's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10308	(16bit): remote device 2's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10309	(16bit): remote device 2's port no.	R/W	R/C	R/C
LW-10310	(16bit): remote device 3's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10311	(16bit): remote device 3's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10312	(16bit): remote device 3's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10313	(16bit): remote device 3's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10314	(16bit): remote device 3's port no.	R/W	R/C	R/C
LW-10315	(16bit): remote device 4's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10316	(16bit): remote device 4's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10317	(16bit): remote device 4's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10318	(16bit): remote device 4's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C

LW-10319	(16bit): remote device 4's port no.	R/W	R/C	R/C
LW-10455	(16bit): remote device 32's IP0 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10456	(16bit): remote device 32's IP1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10457	(16bit): remote device 32's IP2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10458	(16bit): remote device 32's IP3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-10459	(16bit): remote device 32's port no.	R/W	R/C	R/C

21.3.20 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

21.3.21 Communication Error Codes

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-9400	(16bit): error code for device 1	R	R	R
LW-9401	(16bit): error code for device 2	R	R	R
LW-9402	(16bit): error code for device 3	R	R	R
LW-9403	(16bit): error code for device 4	R	R	R
LW-9404	(16bit): error code for device 5	R	R	R
LW-9405	(16bit): error code for device 6	R	R	R
LW-9406	(16bit): error code for device 7	R	R	R
LW-9407	(16bit): error code for device 8	R	R	R
LW-9449	(16bit): error code for device 50	R	R	R

LW-9490	(16bit): error code for USB device	R	R	R
LW-9491	(16bit): error code for CAN-Bus device	R	R	R

Note

- A list of the explanation of device communication error codes:

Error Code	Cause of Communication Error
0	Normal
1	The device is busy and not yet ready to process a command.
2	Communication error due to unexpected reason.
3	The device does not exist.
4	The device using the specified station number does not exist.
5	Incorrect address format.
6	Read/Write unsupported address.
7	The driver of the device does not exist.
8	The COM port does not exist.
9	Incorrect IP address or unable to connect the device.
10	Checksum error.
11	Unidentified command.
12	Ignore
20	The USB device is improperly connected.
21	The CAN Bus device is improperly connected.
22	No reply from the device.
23	Insufficient data read from the device before timeout.
24	The Conversion Tag used by the object does not exist or the content is incorrect.
25	HMI is not accepting any commands from a remote HMI.
251	Read/Write exceeding number of words from/to the register of the MODBUS device.
252	MODBUS device replies incorrect data format.
253	MODBUS device checksum error.

21.3.22 Driver ID

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-9300	(16bit): driver ID of local device 1	R	R	R
LW-9301	(16bit): driver ID of local device 2	R	R	R
LW-9302	(16bit): driver ID of local device 3	R	R	R
LW-9303	(16bit): driver ID of local device 4	R	R	R
LW-9331	(16bit): driver ID of local device 32	R	R	R

21.3.23 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

21.3.24 [Device No Response] Window Control

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9192	disable USB device's "Device No Response" dialog (when ON)	R/W	R/C	R/C
LB-11960	disable device 1's "Device No Response" dialog (when ON)	R/W	R/C	R/C

LB-11961	disable device 2's "Device No Response" dialog (when ON)	R/W	R/C	R/C
LB-11962	disable device 3's "Device No Response" dialog (when ON)	R/W	R/C	R/C
LB-11963	disable device 4's "Device No Response" dialog (when ON)	R/W	R/C	R/C
LB-11964	disable device 5's "Device No Response" dialog (when ON)	R/W	R/C	R/C
LB-11965	disable device 6's "Device No Response" dialog (when ON)	R/W	R/C	R/C
LB-11966	disable device 7's "Device No Response" dialog (when ON)	R/W	R/C	R/C
LB-11967	disable device 8's "Device No Response" dialog (when ON)	R/W	R/C	R/C
LB-12023	disable device 64's "Device No Response" dialog (when ON)	R/W	R/C	R/C
LB-12082	disable CAN Bus device's "Device No Response" dialog (when ON)	R/W	R/C	R/C

21.3.25 [Fast Selection] Window Control

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9013	hide (set ON)/show (set OFF) FS window	R/W	R/C	R/C
LB-9014	hide (set ON)/show (set OFF) FS button	R/W	R/C	R/C
LB-9015	hide (set ON)/show (set OFF) FS window/button	R/W	R/C	R/C

21.3.26 Server

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9051	disconnect (set OFF)/connect (set ON) Server	R/W	R/C	R/C
LB-9052	status of connecting to Server (when ON)	R	R	R

21.3.27 Cloud Service

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-10820	(16bit): disable (set 0)/enable (set 1) (Cloud Service 1.0)	R/W	R/C	R/C
LW-10821	(5 words): session ID (Cloud Service 1.0)	R/W	R/C	R/C
LW-10826	(2 words): password (Cloud Service 1.0)	R/W	R/C	R/C
LW-10828	(16bit): execution status (Cloud Service 1.0)	R	R	R
LW-10829	(16bit): the last error code (Cloud Service 1.0)	R	R	R
LW-11170	(16bit): Proxy Disable/Enable (0:disable, 1:enable) (Cloud Service 1.0)	R/W	R/C	R/C
LW-11171	(16bit): Proxy Type (0:HTTP, 1:SOCKSv4, 2:SOCKSv5) (Cloud Service 1.0)	R/W	R/C	R/C
LW-11172	(16bit): Proxy Server IP0 (Cloud Service 1.0)	R/W	R/C	R/C
LW-11173	(16bit): Proxy Server IP1 (Cloud Service 1.0)	R/W	R/C	R/C
LW-11174	(16bit): Proxy Server IP2 (Cloud Service 1.0)	R/W	R/C	R/C
LW-11175	(16bit): Proxy Server IP3 (Cloud Service 1.0)	R/W	R/C	R/C
LW-11176	(16bit): Proxy Server Port (Cloud Service 1.0)	R/W	R/C	R/C
LW-11177	(16bit): Proxy authentication (0:disable, 1:enable) (Cloud Service 1.0)	R/W	R/C	R/C
LW-11178	(16 words): Proxy username (Cloud Service 1.0)	R/W	R/C	R/C
LW-11194	(16 words): Proxy password (Cloud Service 1.0)	R/W	R/C	R/C
LW-11210	(20 words): Hardware key (Cloud Service 1.0)	R	R	R
LW-11296	(16bit): Location of Cloud Service 1.0 Server (0 : Global, 1 : China)	R	R	R

21.3.28 Remote Printer/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.

21.3.29 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 device 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 (S(16bit): the last error (Siemens pass-through)	R	R	R

21.3.30 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) *Note1	R/W	R/C	R/C
LW-9530	(4 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.

21.3.31 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.

21.3.32 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

21.3.33 User Name and Password

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9050	user logout	R	C	C
LB-9060	password error	R	R	R
LB-9061	update password (set ON)	R	C	C
LB-12056	the user touches an unauthorized object (when ON)	R	R	R

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): object 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 (on tablet) *Note 1	R	R	R

Note

1. Only for [Security] » [Enhanced security mode].

21.3.34 Macro

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-9059	disable macro TRACE function (when ON)	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

21.3.35 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

21.3.36 Time Sync. / Daylight Saving Time

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-12055	failed to execute time synchronization (when ON)	R	R	R
LB-12355	daylight saving time period (when ON)	R	R	R
LW-11260	(16bit): enable/disable the daylight saving time (DST) (0:disable, 1:enable)	R/W	R/C	R/C
LW-11261	(16bit): hour of the DST bias	R/W	R/C	R/C
LW-11262	(16bit): minute of the DST bias	R/W	R/C	R/C
LW-11263	(16bit): month of the year when DST starts	R/W	R/C	R/C
LW-11264	(16bit): week of the month when DST starts (1~5)	R/W	R/C	R/C
LW-11265	(16bit): day of the week when DST starts (0~ 6)	R/W	R/C	R/C
LW-11266	(16bit): hour of local time when DST starts	R/W	R/C	R/C
LW-11267	(16bit): minute of local time when DST starts	R/W	R/C	R/C
LW-11268	(16bit): month of the year when DST ends	R/W	R/C	R/C
LW-11269	(16bit): week of the month when DST ends (1~5)	R/W	R/C	R/C
LW-11270	(16bit): day of the week when DST ends (0~6)	R/W	R/C	R/C
LW-11271	(16bit): hour of local time when DST ends	R/W	R/C	R/C
LW-11272	(16bit): minute of local time when DST ends	R/W	R/C	R/C
LW-11273	(16bit): enable/disable time synchronization via NTP (Network Time Protocol) server (0:disable, 1:enable)	R/W	R/C	R/C
LW-11274	(16bit): execute time synchronization when HMI starts (0:disable, 1:enable)	R/W	R/C	R/C
LW-11275	(16bit): server response time has been adjusted in accordance with DST (0:disable, 1:enable)	R/W	R/C	R/C
LW-11276	(16bit): HMI time zone (unit : minute)	R/W	R/C	R/C
LW-11277	(16bit): server response time (server time zone) (unit : minute)	R/W	R/C	R/C
LW-11278	(16bit): IP 0 of network time server 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11279	(16bit): IP 1 of network time server 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11280	(16bit): IP 2 of network time server 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11281	(16bit): IP 3 of network time server 1 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11282	(16bit): IP 0 of network time server 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11283	(16bit): IP 1 of network time server 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11284	(16bit): IP 2 of network time server 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C

LW-11285	(16bit): IP 3 of network time server 2 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11286	(16bit): IP 0 of network time server 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11287	(16bit): IP 1 of network time server 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11288	(16bit): IP 2 of network time server 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11289	(16bit): IP 3 of network time server 3 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11290	(16bit): IP 0 of network time server 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11291	(16bit): IP 1 of network time server 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11292	(16bit): IP 2 of network time server 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11293	(16bit): IP 3 of network time server 4 (IP address = IP0:IP1:IP2:IP3)	R/W	R/C	R/C
LW-11294	(32bit): update interval (time synchronization interval) (10 ~ 86400, unit : second)	R/W	R/C	R/C

21.3.37 Cellular Data Network

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-11297	(16 words): PIN code of SIM card (cellular network)	R/W	R/C	R/C
LW-11313	(16 words): Access Point Name (cellular network)	R/W	R/C	R/C
LW-11329	(16 words): username (cellular network)	R/W	R/C	R/C
LW-11345	(16 words): password (cellular network)	R/W	R/C	R/C
LW-11361	(16 words): dial number (cellular network)	R/W	R/C	R/C
LW-11377	(16bit): stop (set 0)/start (set 1) connection (cellular network)	R/W	R/C	R/C
LW-11378	(16bit): last error code (0:success, 1:incorrect PIN code, 2:no SIM, 3:no device, 4:puk locked, 5:other) (cellular network)	R	R	R
LW-11379	(16bit): connection status (0:no device, 1:disconnect, 2:connecting, 3:connected) (cellular network)	R	R	R
LW-11380	(16bit): stop (set 0)/start (set 1) connection (USB tethering)	R/W	R/C	R/C
LW-11381	(16bit): connection status (0:no device, 1:disconnect, 2: connected, 3:fail, 4:OS not support, 5:HMI not support) (USB tethering)	R	R	R

21.3.38 WiFi Settings

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LB-12365	update wifi setting (IP, netmask, gateway, DNS) (set ON)	R/W	R/C	R/C
LW-11383	(16bit): WiFi connection (1: disconnect, 2: connect, 3: popup setting dialog, (Wi-Fi)	R/W	R/C	R/C
LW-11384	(16bit): WiFi error code (0: no error, 1: no such device, 2. Wi-Fi radio is off) (Wi-Fi)	R	R	R
LW-11385	(16bit): WiFi status (0: stopped; 1: connecting; 2: connected) (Wi-Fi)	R	R	R
LW-11386	(16 words): WiFi connected SSID	R	R	R
LW-11402	(16bit): WiFi signal level (0: none, 1: weak, 2: fair, 3: good, 4: excellent) *Note2	R	R	R
LW-11403	(16bit): WiFi country code *Note1	R/W	R/C	R/C
LW-11404	(16bit): WiFi radio (0: Off, 1: On)	R/W	R/C	R/C
LW-11405	(16bit): WiFi signal strength (dBm) (0, 1, 2: failed, others: signal strength)	R	R	R
LW-11410	(16bit): HMI WiFi IP 0 (for HMI use only)	R/W	R/C	R/C
LW-11411	(16bit): HMI WiFi IP 1 (for HMI use only)	R/W	R/C	R/C
LW-11412	(16bit): HMI WiFi IP 2 (for HMI use only)	R/W	R/C	R/C
LW-11413	(16bit): HMI WiFi IP 3 (for HMI use only)	R/W	R/C	R/C
LW-11414	(16bit): HMI WiFi netmask 0 (for HMI use only)	R/W	R/C	R/C
LW-11415	(16bit): HMI WiFi netmask 1 (for HMI use only)	R/W	R/C	R/C
LW-11416	(16bit): HMI WiFi netmask 2 (for HMI use only)	R/W	R/C	R/C
LW-11417	(16bit): HMI WiFi netmask 3 (for HMI use only)	R/W	R/C	R/C
LW-11418	(16bit): HMI WiFi gateway 0 (for HMI use only)	R/W	R/C	R/C
LW-11419	(16bit): HMI WiFi gateway 1 (for HMI use only)	R/W	R/C	R/C
LW-11420	(16bit): HMI WiFi gateway 2 (for HMI use only)	R/W	R/C	R/C

LW-11421	(16bit): HMI WiFi gateway 3 (for HMI use only)	R/W	R/C	R/C
LW-11422	(16bit): HMI WiFi media access control (MAC) address 0	R	R	R
LW-11423	(16bit): HMI WiFi media access control (MAC) address 1	R	R	R
LW-11424	(16bit): HMI WiFi media access control (MAC) address 2	R	R	R
LW-11425	(16bit): HMI WiFi media access control (MAC) address 3	R	R	R
LW-11426	(16bit): HMI WiFi media access control (MAC) address 4	R	R	R
LW-11427	(16bit): HMI WiFi media access control (MAC) address 5	R	R	R
LW-11428	(16bit): HMI WiFi domain name system (DNS) server IP 0	R/W	R/C	R/C
LW-11429	(16bit): HMI WiFi domain name system (DNS) server IP 1	R/W	R/C	R/C
LW-11430	(16bit): HMI WiFi domain name system (DNS) server IP 2	R/W	R/C	R/C
LW-11431	(16bit): HMI WiFi domain name system (DNS) server IP 3	R/W	R/C	R/C
LW-11432	(16bit): obtain an WiFi IP address automatically (DHCP => 0 : off, 1 : on)	R/W	R/C	R/C

Note

1. Please enter WiFi country code in ASCII for uppercase letters, the setting will take effect after rebooting HMI. Wireless regulations vary from country to country. The country code selection affects the list of channels of the wireless radio.
2. The signal strength is classified into four levels: 1. Weak (<-70 dBm), 2. Fair (-60 ~ -70 dBm), 3. Good (-50 ~ -60 dBm), 4. Excellent (>-50 dBm)

21.3.39 OPC UA Server

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
LW-11435	(16bit): OPC UA Server status (0: Stopped, 1: Started)	R	R	R
LW-11436	(16bit): OPC UA Server error code (0: Success, 1 or more: Error)		R	R
LW-11437	(16bit): OPC UA Server control command (0: None, 1: Start, 2: Stop)	R/W	R/C	R/C

21.3.40 e-Mail

Address	Description	Read(R)/Write(W)/Control(C)		
		Local HMI	Macro	Remote HMI
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-9216	(16bit): the result of importing email data *Note 1	R	R	R
LW-11444	(16bit): failed step (e-Mail) *Note 2	R	R	R
LW-11445	(16bit): error code (e-Mail) *Note 3	R	R	R

Note

1. 1: import succeeded, 2: import failed (file doesn't exist) , 4: incorrect file format or file cannot be decrypted.
2. Error codes of failed steps include:

Code	Cause of Error
0	CSMTP_NO_ERROR
100	WSA_STARTUP = Unable to initialise winsock2
101	WSA_VER = Wrong version of the winsock2
102	WSA_SEND = Function send() failed
103	WSA_RECV = Function recv() failed
104	WSA_CONNECT = Function connect failed
105	WSA_GETHOSTBY_NAME_ADDR = Unable to determine remote server
106	WSA_INVALID_SOCKET = Invalid winsock2 socket
107	WSA_HOSTNAME = Function hostname() failed
108	WSA_IOCTLSOCKET = Function ioctlsocket() failed
109	WSA_SELECT
110	BAD_IPV4_ADDR = Improper IPv4 address
200	UNDEF_MSG_HEADER = Undefined message header
201	UNDEF_MAIL_FROM = Undefined mail sender
202	UNDEF_SUBJECT = Undefined message subject
203	UNDEF_RECIPIENTS = Undefined at least one reciepent
204	UNDEF_RECIPIENT_MAIL = Undefined recipent mail

205	UNDEF_LOGIN = Undefined user login
206	UNDEF_PASSWORD = Undefined user password
207	BAD_LOGIN_PASSWORD = Invalid user login or password
208	BAD_DIGEST_RESPONSE = Server returned a bad digest MD5 response
209	BAD_SERVER_NAME = Unable to determine server name for digest MD5 response
300	COMMAND_MAIL_FROM = Server returned error after sending MAIL FROM
301	COMMAND_EHLO = Server returned error after sending EHLO
302	COMMAND_AUTH_PLAIN = Server returned error after sending AUTH PLAIN
303	COMMAND_AUTH_LOGIN = Server returned error after sending AUTH LOGIN
304	COMMAND_AUTH_CRAMMD5 = Server returned error after sending AUTH CRAM-MD5
305	COMMAND_AUTH_DIGESTMD5 = Server returned error after sending AUTH DIGEST-MD5
306	COMMAND_DIGESTMD5 = Server returned error after sending MD5 DIGEST
307	COMMAND_DATA = Server returned error after sending DATA
308	COMMAND_QUIT = Server returned error after sending QUIT
309	COMMAND_RCPT_TO = Server returned error after sending RCPT TO
310	MSG_BODY_ERROR = Error in message body
400	CONNECTION_CLOSED = Server has closed the connection
401	SERVER_NOT_READY = Server is not ready
402	SERVER_NOT_RESPONDING = Server not responding
403	SELECT_TIMEOUT =
404	FILE_NOT_EXIST = File not exist
405	MSG_TOO_BIG = Message is too big
406	BAD_LOGIN_PASS = Bad login or password
407	UNDEF_XYZ_RESPONSE = Undefined xyz SMTP response
408	LACK_OF_MEMORY = Lack of memory
409	TIME_ERROR = time() error
410	RECVBUF_IS_EMPTY = RecvBuf is empty
411	SENDBUF_IS_EMPTY = SendBuf is empty
412	OUT_OF_MSG_RANGE = Specified line number is out of message size
413	COMMAND_EHLO_STARTTLS = Server returned error after sending STARTTLS

414	SSL_PROBLEM = SSL problem
415	COMMAND_DATABLOCK = Failed to send data block
416	STARTTLS_NOT_SUPPORTED = The STARTTLS command is not supported by the server
417	LOGIN_NOT_SUPPORTED = AUTH LOGIN is not supported by the server

3. Error messages sent from mail server can be shown by designating a word address (length adjustable) in System Parameter Settings » e-Mail tab » [Error message].

21.3.41 Other functions

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-transfer write indicator	R	R	R
LB-9011	data-transfer read indicator	R	R	R
LB-9012	data-transfer execution 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	disable/enable watch dog (use LW-11456 set watch dog timeout) *Note 1	R/W	R/C	R/C
LB-12356	enable(set on)/disable(set off) web streaming	R/W	R/C	R/C
LB-12357	web streaming status (on: enabled / off: disabled)	R	R	R
LB-12358	enable (when ON) / disable (when OFF) off-line simulation on HMI *Note 5	R/W	R/C	R/C
LB-12361	status of operation log function (OFF : disabled, ON : enabled)	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 (on tablet)	R	R	R
LW-9134	(16bit): language mode (on tablet) *Note 2	R/W	R/C	R/C

LW-9900	(16bit): HMI run mode (0 : normal mode, 1-3 : test mode (COM 1-COM 3)	R/W	R/C	R/C
LW-10762	(8 words): slot1 user name	R/W	R/C	R/C
LW-10770	(8 words): slot2 user name	R/W	R/C	R/C
LW-10778	(8 words): slot3 user name	R/W	R/C	R/C
LW-10814	(16bit): connecting to a Weintek HMI (0:none, 1:connecting) *Note 4	R	R	R
LW-11456	(16bit): watch dog timeout (3 ~ 10), unit : second *Note 1	R/W	R/C	R/C

Note

1. When LB-9049 watch dog function is enabled, watch dog automatically reboots the system after the HMI stops functioning for a specified period of time.

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. 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 system will use the HMI name as the default backup folder name.

4. When the USB Host is connected to the HMI, the address is set to 1, for testing if the USB Client of another HMI works normally.

5. This register allows switching to off-line mode on HMI. In off-line mode, “Device No Response” message will not show even when HMI is not properly connected. In this case, device related objects can still operate, however, the values are not read by / written to the device.

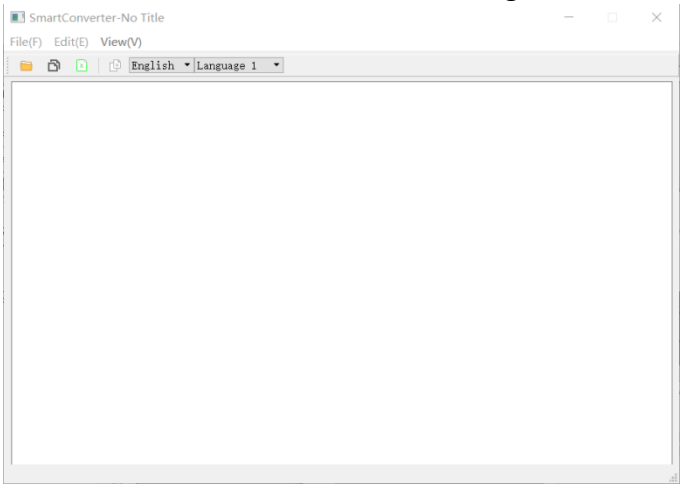
Chapter 22 HMI Supported Printers (in developing)

Chapter 23 SmartConverter

This Chapter explains how to use SmartConverter.

23.1 Overview

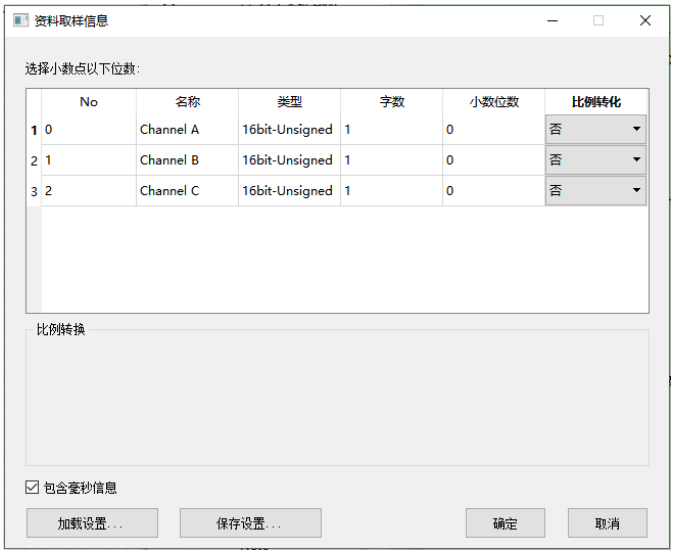
SmartConverter reads the Data Log file, Event Log file, and Operation Log file in HMI and convert the files to Excel format. From SmartManager Pro menu select [Tool] » [Data/Event Log Converter].



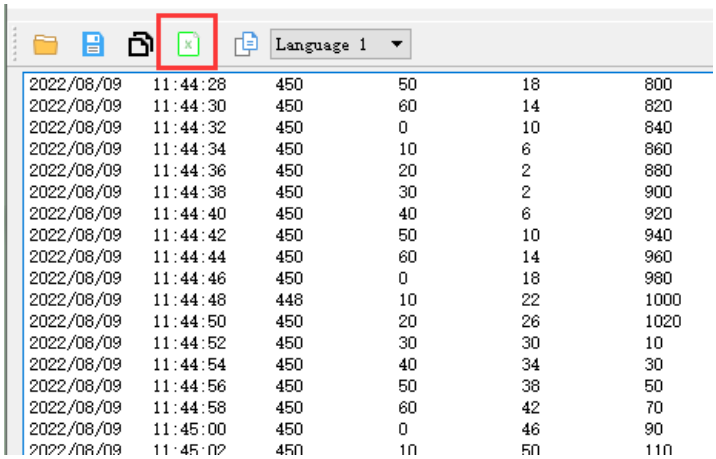
23.2 Converting Data Log File to Excel File

1. If the Data 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.

2. The following is the setting dialog box, please set based on actual needs.



3. Click [OK], the Data Log layout is shown in the following figure. Click [Export to Excel]. The file will be converted to Excel format.



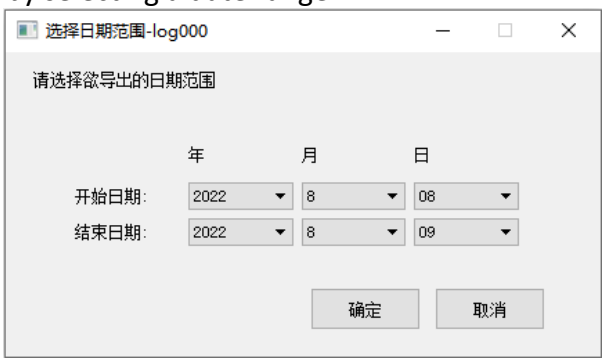
4. The exported Excel file will be saved in the Excel folder under the installation directory. The Excel file displays as follows.

日期	时间	毫秒	16bit-Unsigned	16bit-Unsigned	16bit-Unsigned
2022/8/8	11:51:49	0	10	4	20
2022/8/8	11:51:51	0	20	8	40
2022/8/8	11:51:54	0	30	12	60
2022/8/8	11:51:56	0	40	16	80
2022/8/8	11:51:58	0	50	20	100
2022/8/8	11:52:00	0	60	24	120
2022/8/8	11:52:02	0	0	28	140
2022/8/8	11:52:04	0	10	32	160
2022/8/8	11:52:06	0	20	36	180
2022/8/8	11:52:08	0	30	40	200
2022/8/9	11:43:30	448	40	44	220
2022/8/9	11:43:32	450	50	48	240
2022/8/9	11:43:34	450	60	52	260
2022/8/9	11:43:36	450	0	56	280
2022/8/9	11:43:38	450	10	60	300
2022/8/9	11:43:40	450	20	64	320
2022/8/9	11:43:42	450	30	68	340
2022/8/9	11:43:44	450	40	72	360
2022/8/9	11:43:46	450	50	76	380

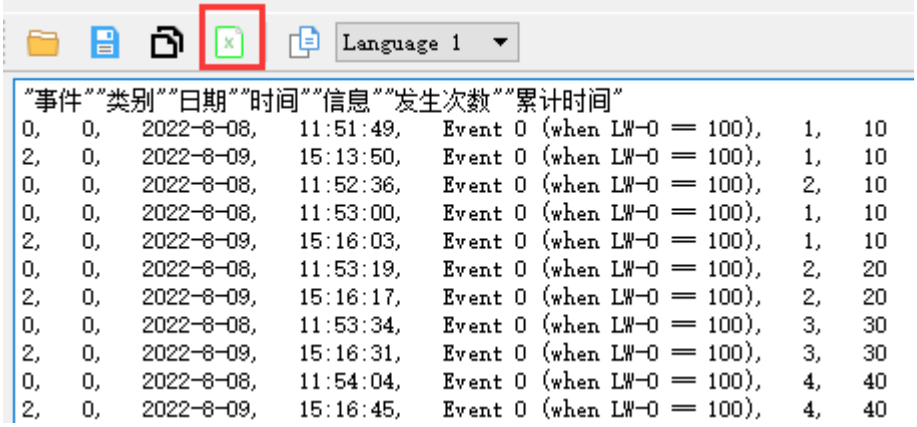
23.3 Converting Event Log File to Excel File

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.



2. Click [OK], the Event Log layout is shown in the following figure. Click [Export to Excel]. The file will be converted to Excel format.



3. The exported Excel file will be saved in the Excel folder under the installation directory. The Excel file displays as follows.

"事件"	"类别"	"日期"	"时间"	"信息"	"发生次数"	"累计时间"
0	0	2022/8/8	11:51:49	Event 0 (when LW-0 == 100)	1	10
2	0	2022/8/9	15:13:50	Event 0 (when LW-0 == 100)	1	10
0	0	2022/8/8	11:52:36	Event 0 (when LW-0 == 100)	2	10
0	0	2022/8/8	11:53:00	Event 0 (when LW-0 == 100)	1	10
2	0	2022/8/9	15:16:03	Event 0 (when LW-0 == 100)	1	10
0	0	2022/8/8	11:53:19	Event 0 (when LW-0 == 100)	2	20
2	0	2022/8/9	15:16:17	Event 0 (when LW-0 == 100)	2	20
0	0	2022/8/8	11:53:34	Event 0 (when LW-0 == 100)	3	30
2	0	2022/8/9	15:16:31	Event 0 (when LW-0 == 100)	3	30
0	0	2022/8/8	11:54:04	Event 0 (when LW-0 == 100)	4	40
2	0	2022/8/9	15:16:45	Event 0 (when LW-0 == 100)	4	40
0	0	2022/8/8	11:54:19	Event 0 (when LW-0 == 100)	5	50
2	0	2022/8/9	15:16:59	Event 0 (when LW-0 == 100)	5	50
0	0	2022/8/8	11:54:39	Event 0 (when LW-0 == 100)	6	61
2	0	2022/8/9	15:17:13	Event 0 (when LW-0 == 100)	6	61
0	0	2022/8/8	11:55:00	Event 0 (when LW-0 == 100)	7	71
2	0	2022/8/9	15:17:27	Event 0 (when LW-0 == 100)	7	71
0	0	2022/8/8	11:55:02	Event 0 (when LW-0 == 100)	8	81
2	0	2022/8/9	15:17:41	Event 0 (when LW-0 == 100)	8	81
0	0	2022/8/8	11:55:30	Event 0 (when LW-0 == 100)	9	91
2	0	2022/8/9	15:17:55	Event 0 (when LW-0 == 100)	9	91
0	0	2022/8/9	15:17:59	Event 0 (when LW-0 == 100)	10	102
2	0	2022/8/9	15:18:09	Event 0 (when LW-0 == 100)	10	102
0	0	2022/8/9	15:18:13	Event 0 (when LW-0 == 100)	11	112
2	0	2022/8/9	15:18:23	Event 0 (when LW-0 == 100)	11	112
0	0	2022/8/9	15:18:27	Event 0 (when LW-0 == 100)	12	122

Note

■ The "Event" column can be found. 0-> Event triggered; 1-> Event acknowledged; 2-> Event returns to normal.

23.4 Converting Operation Log File to Excel File (in developing)

23.5 Converting Multiple Files (in developing)

23.6 Scaling Function

When opening a Data Log file, the scaling function can be set.
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).

Chapter 24 Project Protection

This chapter explains the settings relevant to project protection.

24.1 Overview

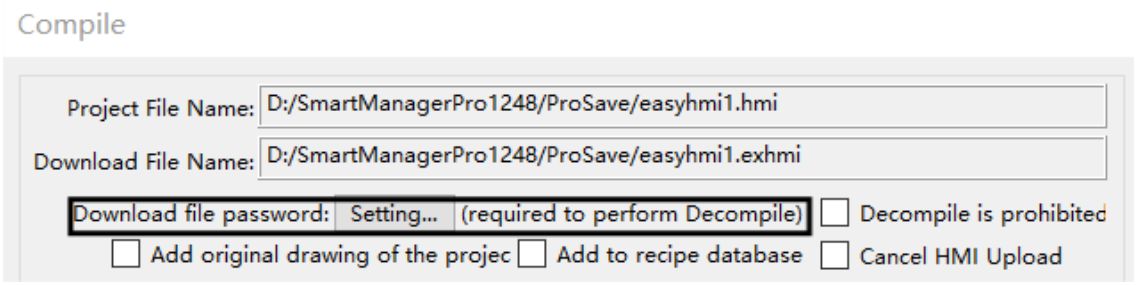
SmartManager Pro provides a number of features to protect a project file.

Note

- The protected projects cannot be decrypted by the factory since they are encrypted by users; therefore, please remember your password.

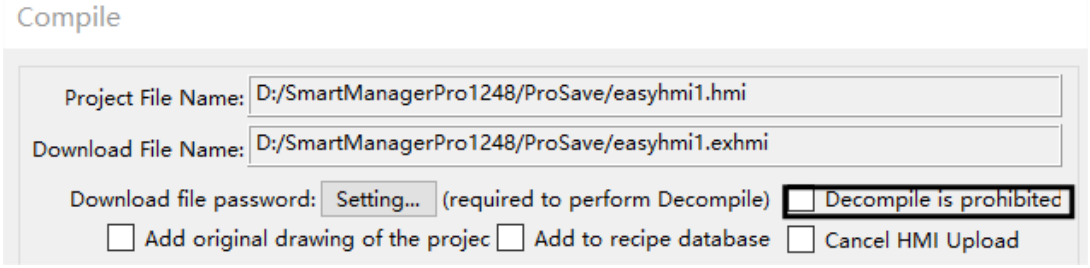
24.2 exhmi Password

After editing a project (.hmi), users can compile the project to .exhmi format. The .exhmi file can be downloaded to HMI. Password can be set to protect the .exhmi file in [EXHMI password] when compiling. (Password range: 0 ~ 4294967295). A password is needed when attempting to decompile the .exhmi file back to .hmi file. If the password is entered incorrectly for three times, please restart the configuration file.



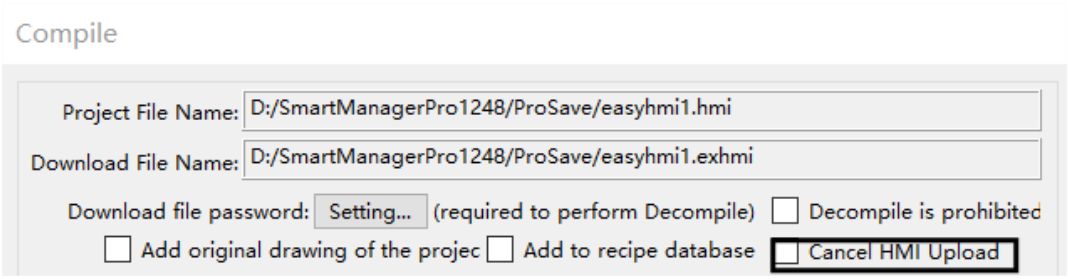
24.3 Decompilation is Prohibited

After a project (.hmi) is edited, users can compile the project to .exhmi format. The .exhmi file can be downloaded to HMI. If [Decompilation is prohibited] check box is selected when compiling, the setting in [EXHMI password] will be ignored. Furthermore, the .exhmi file cannot be decompiled to .hmi file.

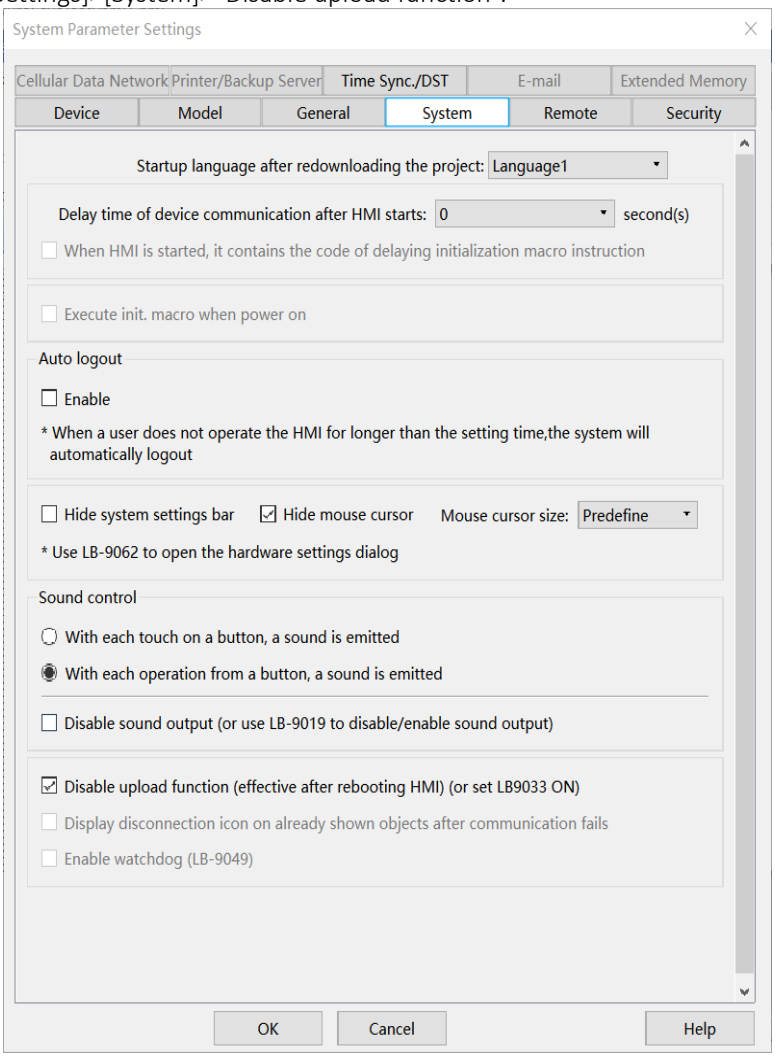


24.4 Disable exhmi Upload Function

This software provides three methods to disable HMI upload function as below.
1. Check the option “Cancel HMI Upload” in Compile dialog window.



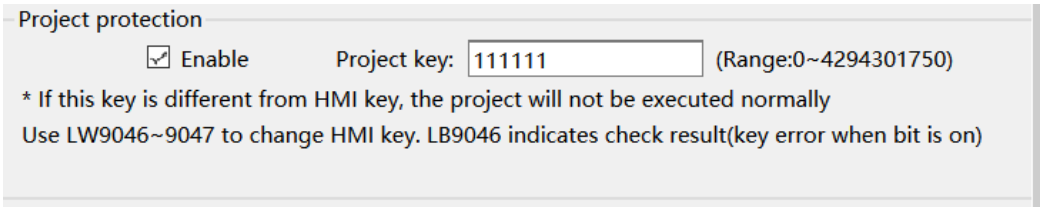
2. [System Parameter Settings]>[System]>”Disable upload function”.



3. provides a system reserved register [LB-9033]. When this register is set ON, the .exhmi file cannot be uploaded. When attempting to upload an .exhmi 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.

24.5 Project Key

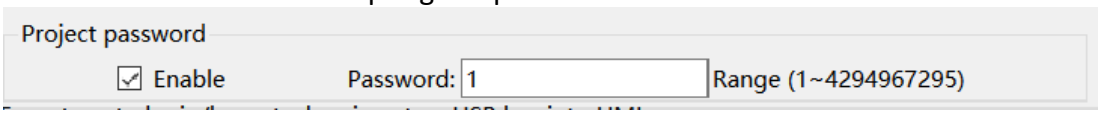
Projects can be restricted to run on a specific HMI.
The setting is in [System Parameters] » [General] » [Project Protection].



Enable and set the [Project Protection] (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 accessed by a remote device. The .exhmi file obtained after compiling can only be executed on HMI when [HMI key] and [Project Protection] match. If the keys don't match, LB-9046 is set ON. To change [HMI key], please reboot HMI.

24.6 HMI Password

After a project (.hmi) is edited, a password can be set to protect the .hmi file. In [System Parameters] » [Security] tab, enable [Project password] and click [Settings] (password range: 1 ~ 4294967295). The password will be needed when attempting to open the .hmi file.



Note

- When using "Window Copy" function, if the source file is protected by Project password, please enter the correct password for the system to execute window copy.

Chapter 25 MODBUS TCP/IP Gateway (in developing)

Chapter 26 Data Security (in developing)

Chapter 27 IIOT

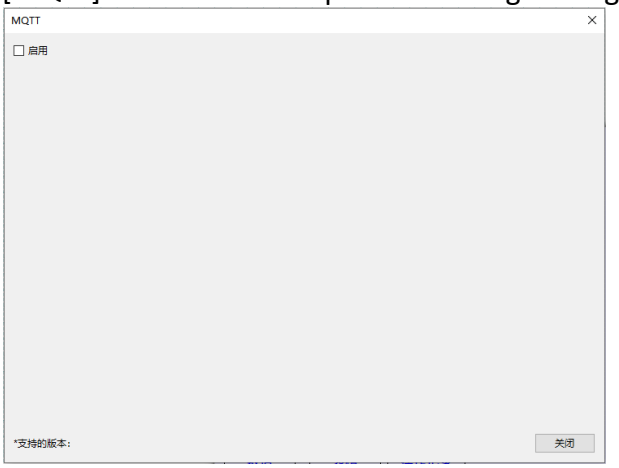
27.1 MQTT

27.1.1 Overview

MQTT object can publish messages to an MQTT server, or subscribe to topics to receive messages from an MQTT server. HMI can serve as an MQTT server as well. When HMI serves as an MQTT server, it does not send message to another MQTT server. Supported MQTT version v3.1.

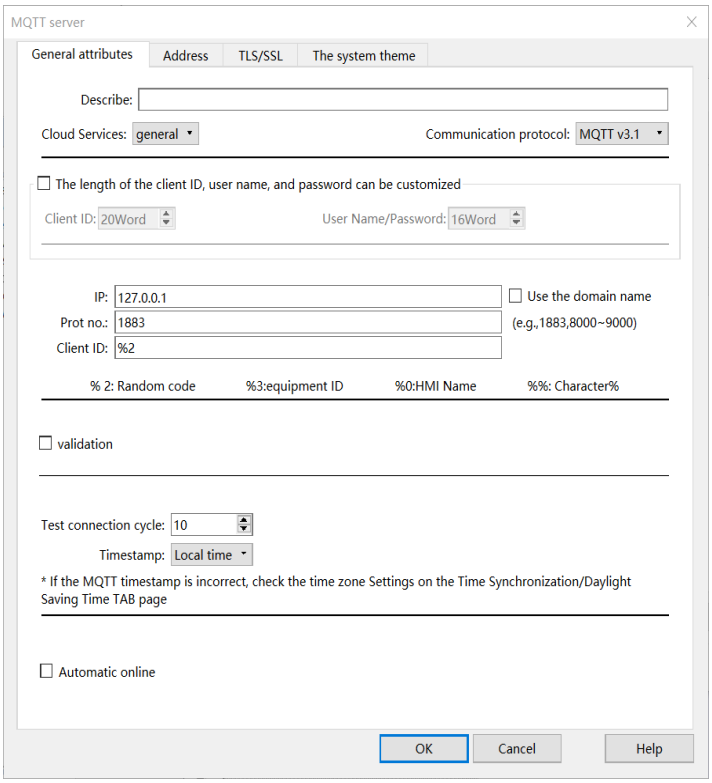
27.1.2 Configuration

Click [Object] » [IIoT] » [MQTT] in the menu to open the settings dialog box.



Server Settings

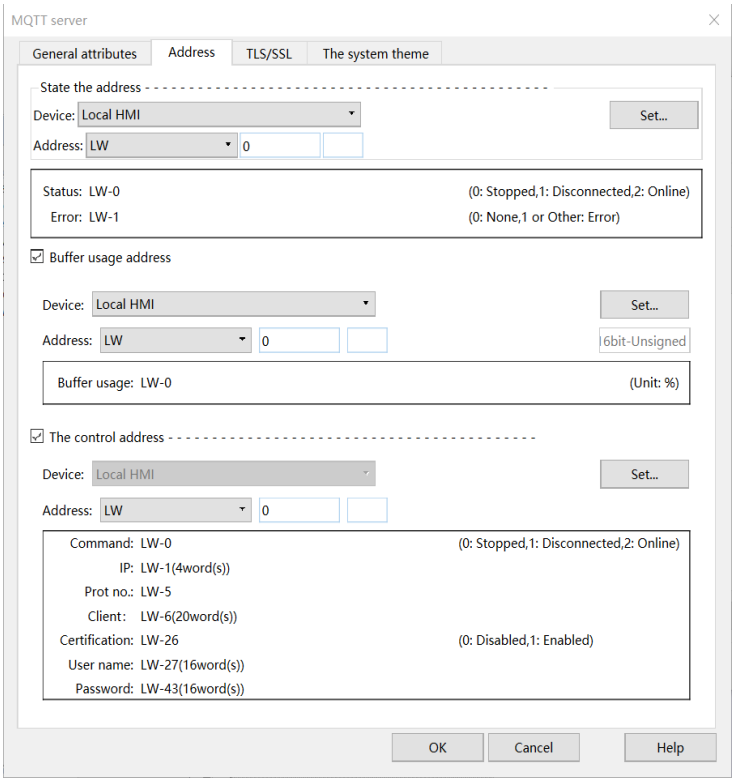
General Tab



Setting	Description
Cloud service	Use general MQTT publish-subscribe service.
IP	Enter the MQTT Server IP address for receiving the message. If select the option “Use domain name”, HMI will run a MQTT server locally.
Use domain name	A domain name can be used as MQTT server’s IP address.
Port	Enter the MQTT Server port number for receiving the message.
Client ID	Login name. %2 represents random code, %0 represents HMI name, and %% represents characters
Validation	If selected, connecting MQTT Server will require [User name] and [Password].
User Name	Enter the username for connecting MQTT Server
Password	Enter the password for connecting MQTT Server.
Test connection cycle	When MQTT Server does not receive the message from HMI passing the specified time, the HMI will be identified as disconnected. Note: When running simulation, the message may be delayed, but the delay will not exceed the [Test connection cycle]. The message from the HMI will be sent immediately.
Automation online	In this mode, the connection will be automatically terminated if there’s no data update for a specified period of time. The connection will resume once any data update occurs.

	The user can choose to publish initial values / topic list only at the first connection. In this mode, the start and stop commands are disabled.

Address Tab



Setting	Description
State the address	LW-n: Displays the connection status to MQTT Server. Value / Description 0 / Not attempting to connect to MQTT Server. 1 / Disconnected and can't connect to MQTT Server 2 / Connection succeeded. LW-n+1: Error indicator. Value / Description 0 / No error 1 or more / Error occurs
Buffer usage address	Messages that have not been sent are stored in the buffer. The maximum buffer capacity is 10000 messages. The buffer capacity is measured in percentage (%), rounded up. LW-n: Shows buffer usage.
The Control Address	LW-n: Controls the operation of MQTT Server. Value / Description 0 / Ready 1 / Start

2	Stop
3	Update

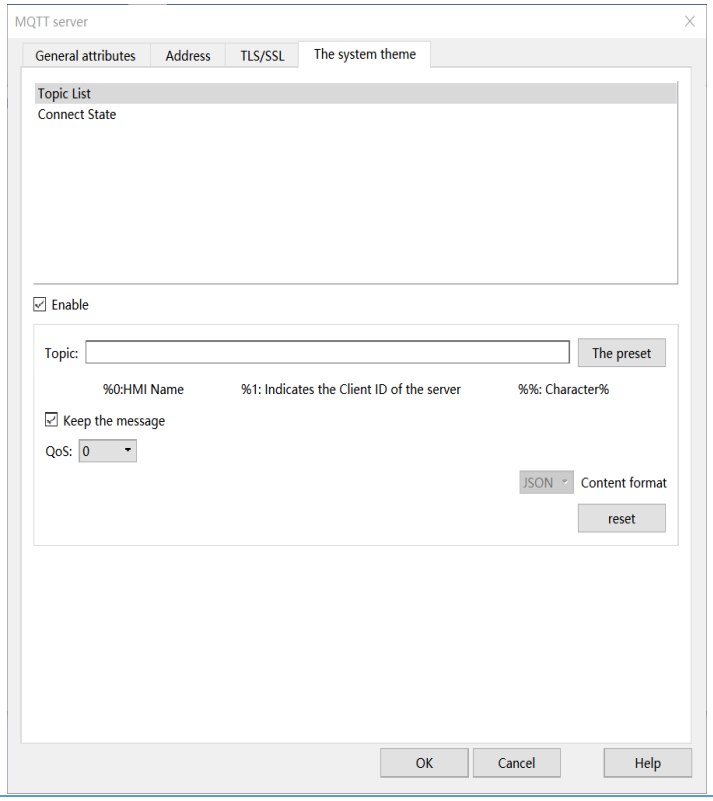
LW-n+1: Sets the IP address of MQTT Server.
LW-n+5: Sets the port number of MQTT Server.
LW-n+6: Sets the Client ID for connecting MQTT Server.
LW-n+26: Enables / Disables Validation

Value	Description
0	Disable
1	Enable

LW-n+27: Sets the username for connecting MQTT Server.
LW-n+43: Sets the password for connecting MQTT Server.

TLS/SSL Tab (in developing)

System Topic



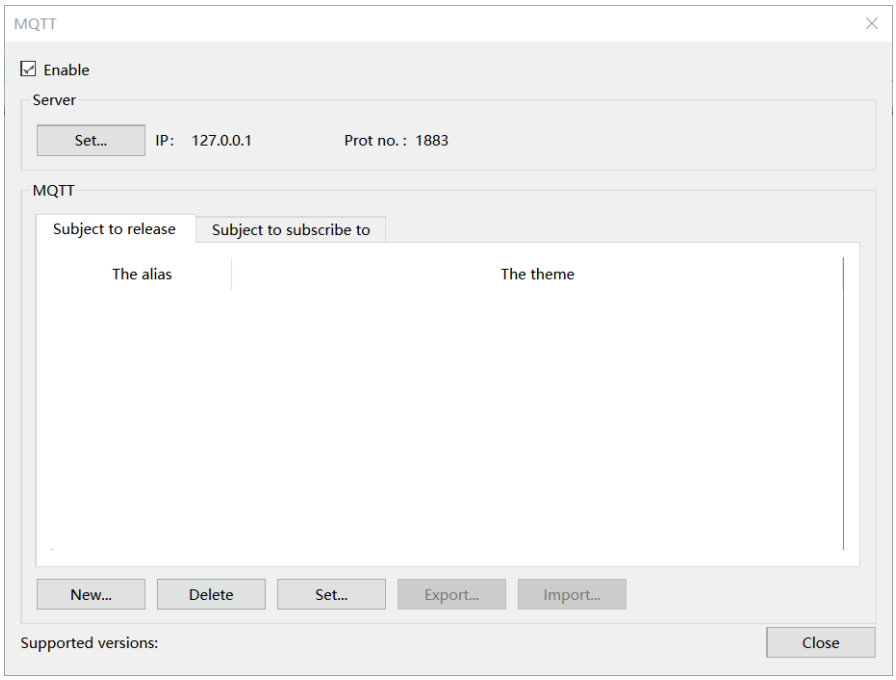
Setting	Description
Topic List	Enable When HMI is the publisher, the Topic List will be included in the server and can also forward all topics published by other HMI. When subscribers comes online to the server firstly, the server will actively transmit this Topic List to them or let they to subscribe this topic to know which topics from publishers are available on the server for subscription. Keep the message **When this checkbox is selected, the MQTT server will save the latest message.**
Connect State	Enable

Indicate the connection status between MQTT server and the Publisher HMI. When subscribers comes online to the server firstly, the server will actively transmit this Topic to them or let they to subscribe this topic to know the connection status between MQTT server and publishers (HMI).

Keep the message

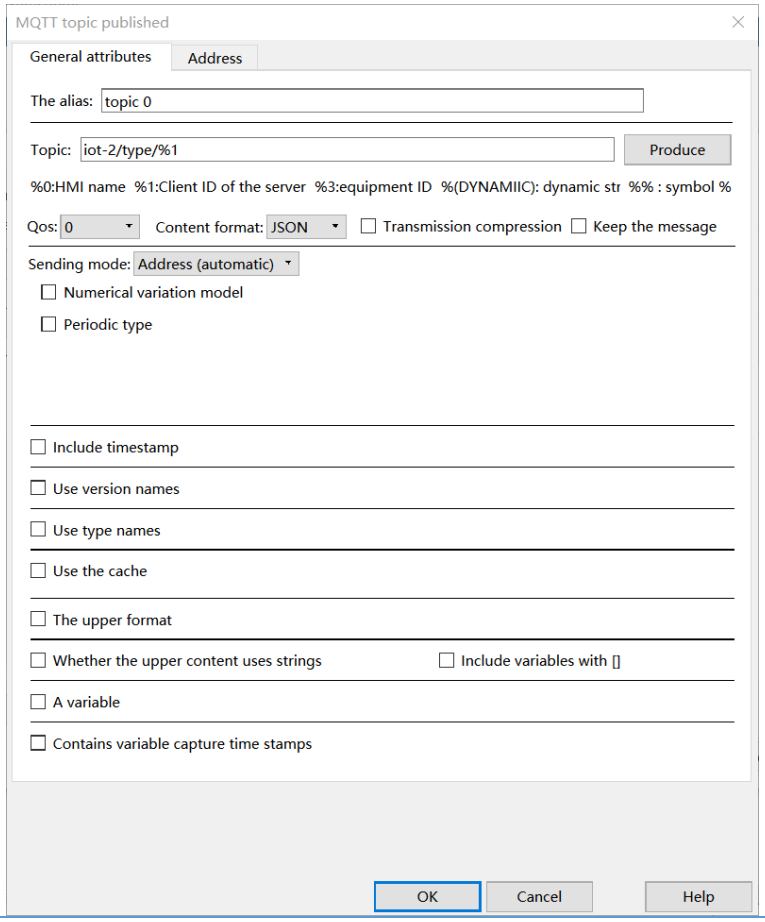
When this checkbox is selected, the MQTT server will save the latest message.

MQTT Subject Release

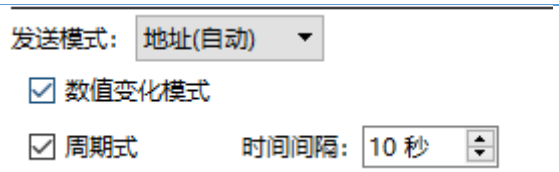


Click [New] to open General and Address settings, or click [Import] / [Export] to import or export an existing *.csv file. The maximum allowable number of topics is 255.

General Tab

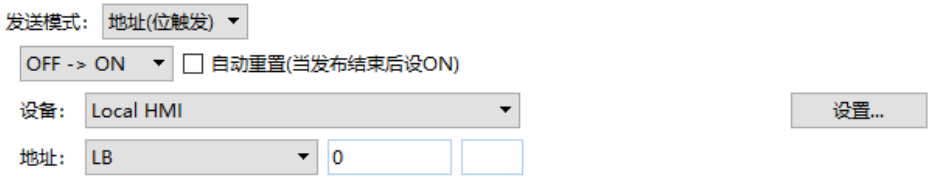


Setting	Description
The alias	Enter the alias of the MQTT Topic for easier reference.
Topic	Specify the format of the message topic sent to MQTT Server.
QoS	MQTT provides three levels of reliability, which are known as quality of service (QoS). The reliability of the message determines the persistence of the message. QoS 0: At most once, messages are not persistent. QoS 1: At least once. QoS 2: Exactly once.
Content format	The supported formats are JSON and Raw data.
Transmission Compression	The message must be compressed firstly before transmitting. And the compressed message needs to be decompressed before being read by the MQTT client. MQTT uses the DEFLATE algorithm to compress or decompress messages.
Keep the message	If selected, the MQTT server will save the latest message.
Sending mode	**Address (Auto.)** Numerical variation model: Sends MQTT message when any value changes. Periodic type: Sends MQTT message in a time-based interval, range: 1~3600 second.



Address (Bit trigger)

Sends MQTT message when a designated bit is triggered.

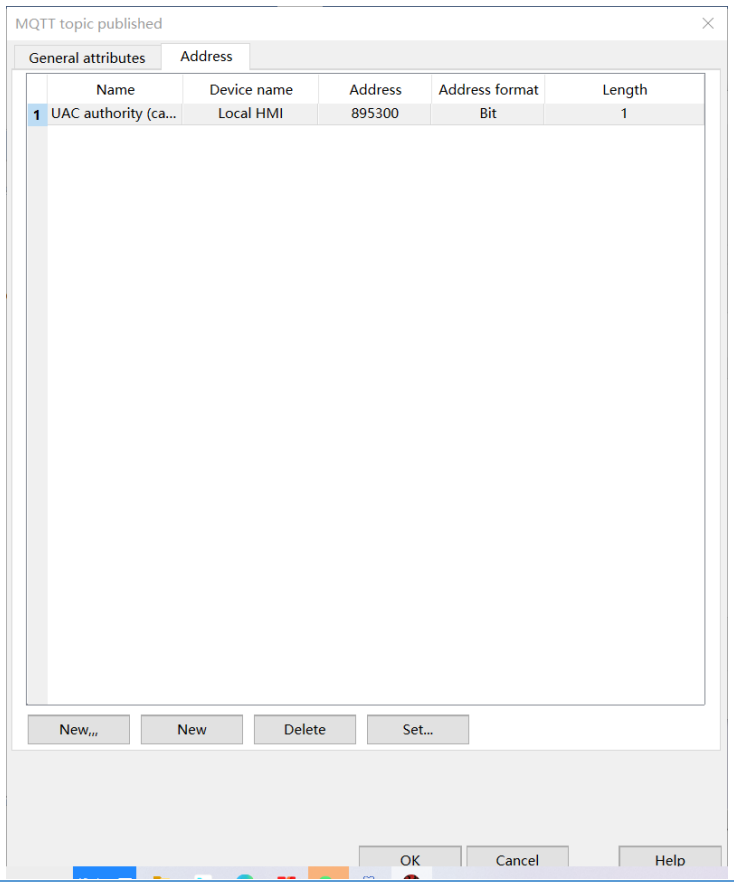


Event (Alarm) Log

The topic source can be an Event Log. MQTT message can be sent when a single event or any event in a specific category occurs.

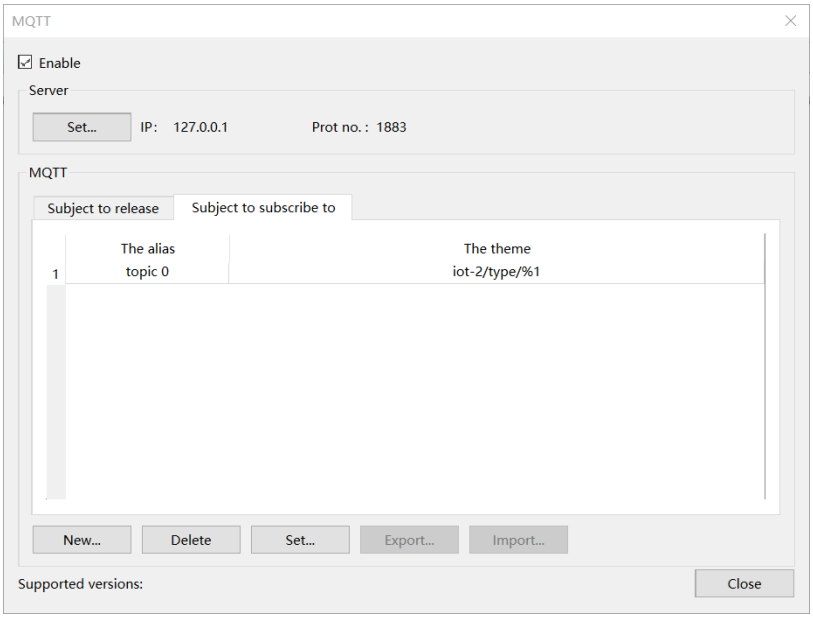
Include timestamp	This option is available only when the format used is [JSON]. And people can include timestamp in the message by manual.
Use version names	
Use type names	
Use the Cache	
The upper format	
Whether the upper content use strings	
A variable	
Contains variable capture time stamps	

Address Settings



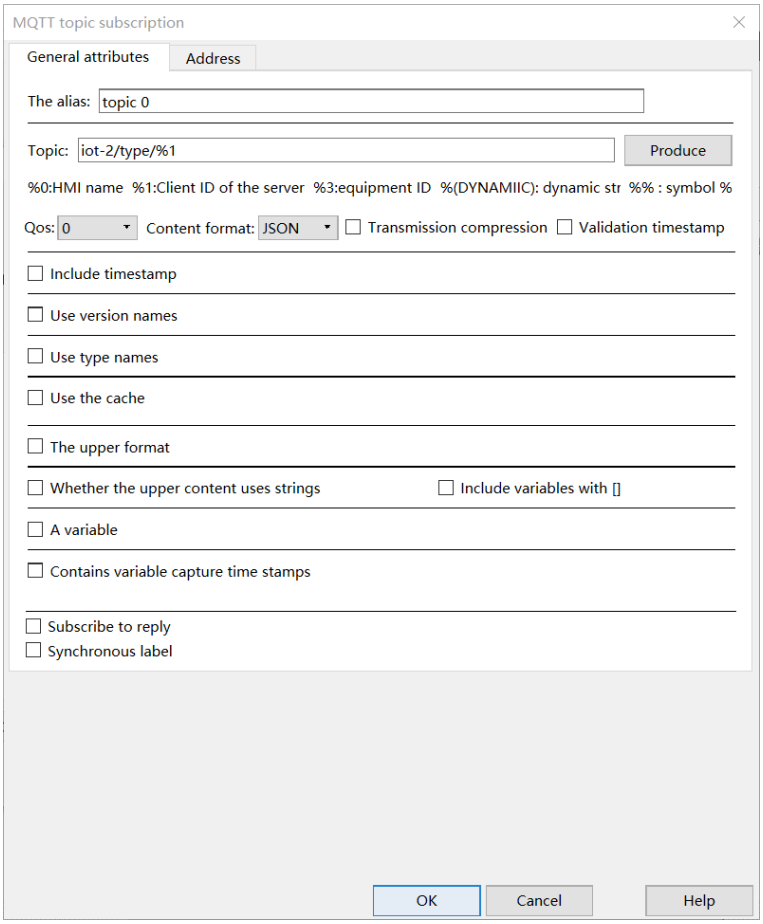
Setting	Description
New	Add the source address of the topic. Each address can be specified respectively by user.
Delete	Delete the address
Setting	Change the name and address

MQTT Topic Subscription



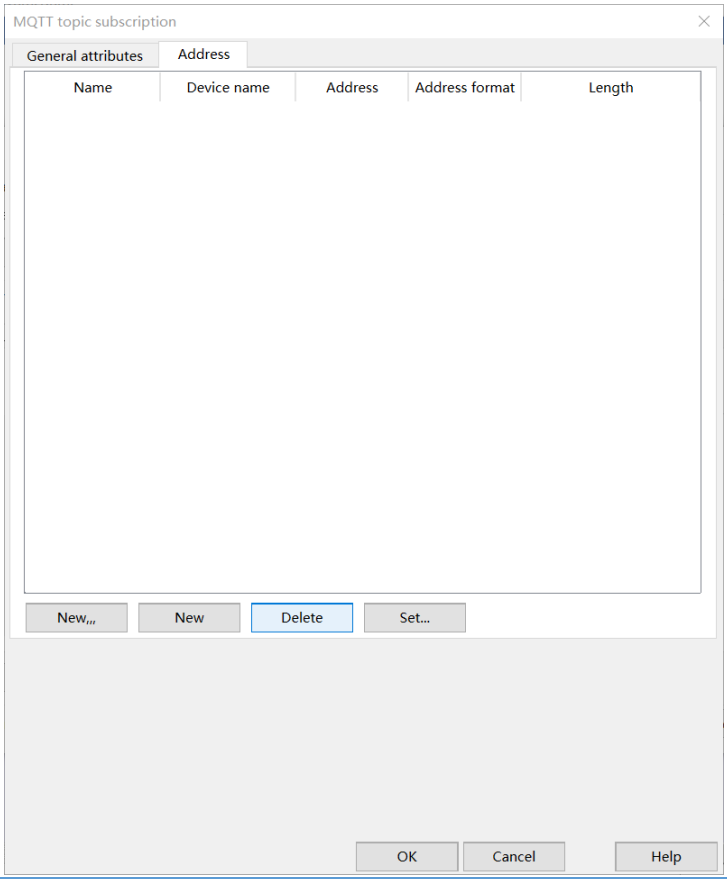
Click [New] to open General and Address settings, or click [Import] / [Export] to import or export an existing *.csv file. The maximum allowable number of topics is 255.

General Tab



Setting	Description
The alias	Enter the nickname of the MQTT Topic for easier reference.
Topic	Subscribe to a topic in MQTT Server. The topic name can be dynamic strings.
QoS	MQTT provides three levels of reliability, which are known as qualities of service (QoS). The reliability of the message determines the persistence of the message. 0: At most once, messages are not persistent. 1: At least once. 2: Exactly once.
Content format	The supported formats are JSON and Raw data.
Transmission Compression	Configure with the same setting as MQTT Topic Publisher
Validation timestamp	When timestamp is included in the message, selecting this option will verify whether the timestamp is increasing, and update will occur when the timestamp does increase; otherwise, the message will be treated as expired message and update will not occur.
Include timestamp	This option is available only when the format used is [JSON]. And people can include timestamp in the message by manual.

Address Tab



Setting	Description
New	Add the destination address of the subscribed topic. Each address can be specified respectively by user.
Delete	Delete the address.

Setting	Change the name and address.

27.2 OPC UA Server (in developing)

Chapter 28 How to Make Boot Logo

28.1 Overview

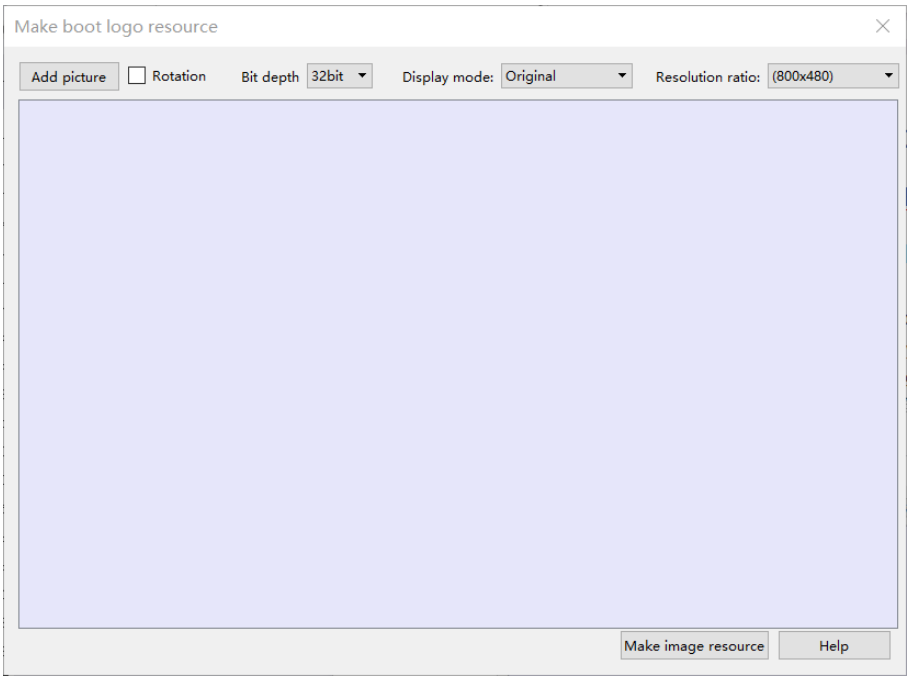
This Chapter mainly introduces how users can customize the startup screen.

28.2 Boot Logo Page Creation and Formatting Guidelines

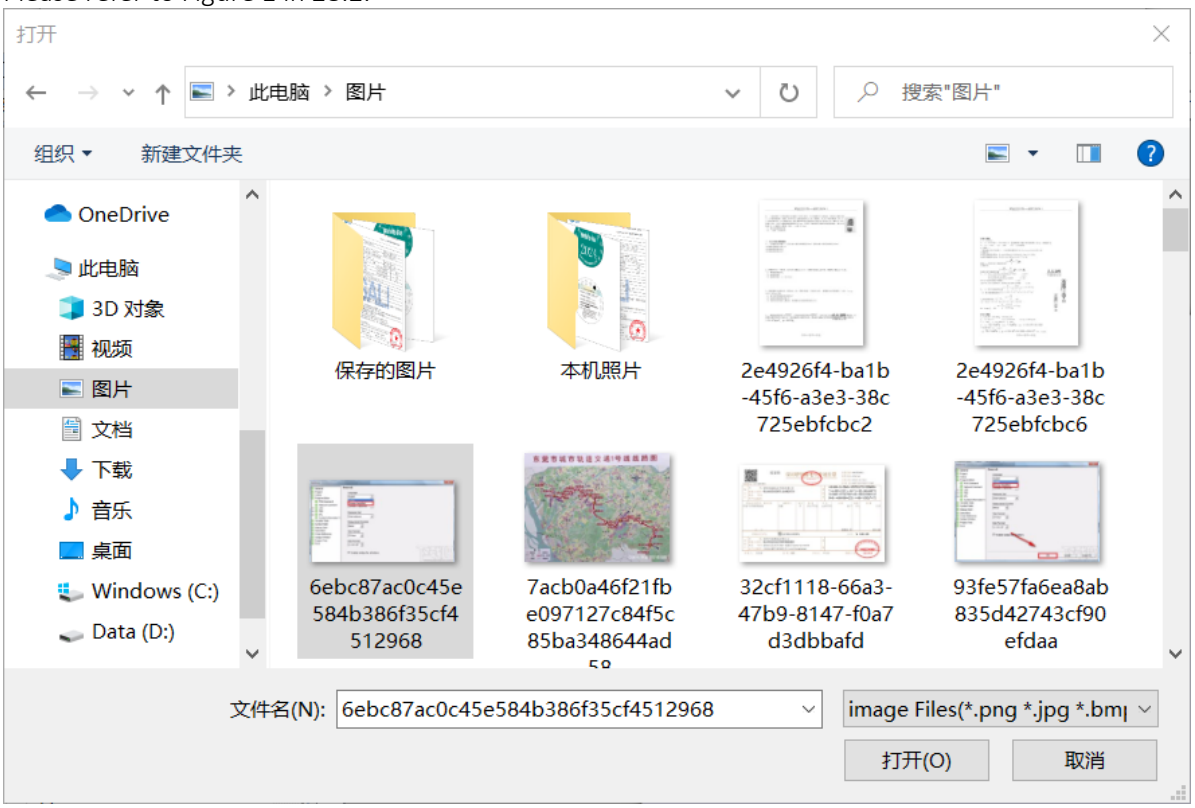
Users are required to create image files that showcase their company's logo, software name, and other relevant elements. These images should be saved in the format of PNG， PMG, JPG or JPEG. To accommodate various screen sizes and resolutions, it is recommended that users generate images in different resolutions, name them accordingly, and store them within the same folder. For more details, please refer to the image below.

28.3 Modify the Image File from 28.2 into a Suitable Resource for Importing into various sizes of HMIs

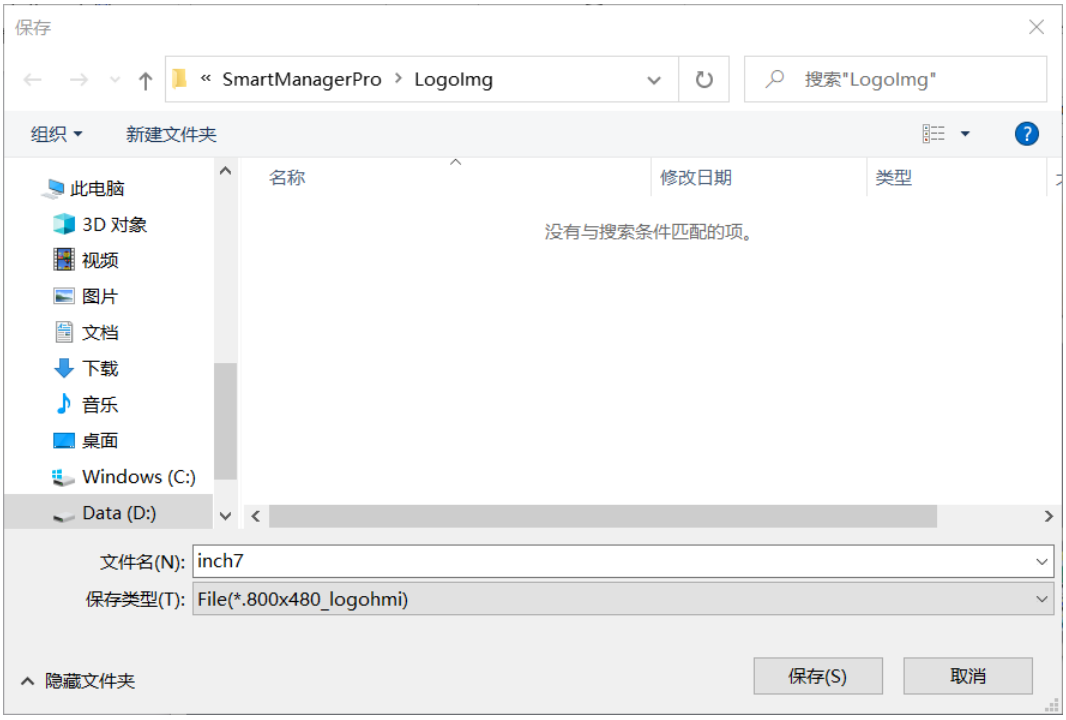
Process 1: Click [Tool] » [Make boot logo] in the menu to open the settings dialog box [Make boot logo resource] as below.



Process 2: Click "Add picture" and the following dialog box will appear. Please select and open the picture file generated in 28.2. At this time, Please remember the file must correspond to the resolution of the HMI to be placed. Please refer to Figure 1 in 28.2.



Process 3: To select the appropriate display mode: Scale, and Resolution, then click "Make image resource ", a Save dialog box will appear, name it, and click "Save" - "OK"



Repeat the above process to prepare corresponding "image resource files" for each size and resolution of HMI for backup.

28.4 Import image resource files into the HMI

There are three ways: USB disk, Ethernet, and Cloud Service, to import image resource files into the HMI.

28.4.1 Import by USB Disk.

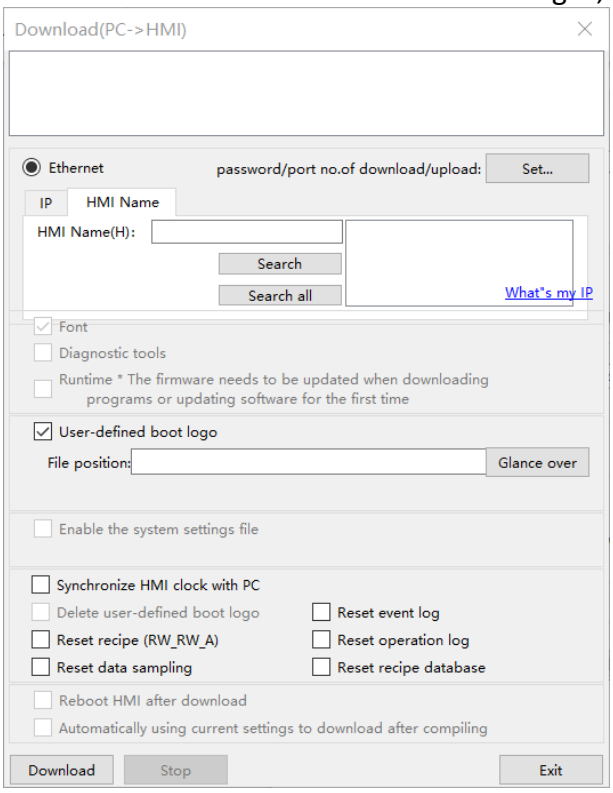
The first step is to store the image resource files created in 28.3 in a USB disk and then insert the USB disk into the HMI.

Step 2, power on the HMI click n the bottom-right corner to pop up System Setting toolbar：  Click the far left Gear, then enter the password "111111", click "OK" to enter the System Settings window. Click "Next" until the "OS setting" screen appears.

Step 3, on the "OS setting" page, click "Update boot image". The image resource file stored in the USB flash drive will automatically appear. Select the file, click "OK", and the HMI will restart. After restarting, the boot image will be changed.

28.4.2 Ethernet import mode

First, ensure that the PC and HMI are connected via Ethernet cable, then press the [Project] » [Download (PC -> HMI)]. After compiling, the following dialog box appears. Check "Ethernet", and connect to the HMI screen that you want to add a custom boot image using the "IP" address method or "HMI name" method. Then check "User-defined boot logo", and confirm the "File location". Finally, click "Download".



28.4.3 Cloud Service Import Mode

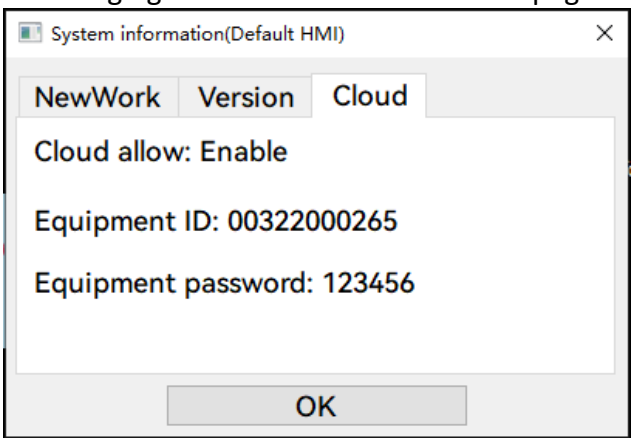
1.Process for Cloud Service.dc

First, you need to activate Cloud services. Please refer to the attachment for details，

2. User Setting for Cloud Service

Then, set up the user settings, please refer to the attachment for details，

Confirm to open the cloud service of the HMI that intends to update the startup screen, as shown in the following figure. Click "OK" on the Cloud page of the system information.



Return to the configuration software on the PC, press [Cloud Services] - [Cloud Services] on the taskbar, enter the HMI ID and password in the dialog box on the upper left, and the hardware in the cloud under your account will appear. Select the HMI to update the startup screen, check "User defined startup screen" on the right, confirm "File location", and finally click "Download".