HMI

User Manual

Important Information

Information in this manual is subject to change without notice and does not represent a commitment on the part of Everest Software LLC. Permission is granted to duplicate this material without modification only for your use or the internal use of other members of your company or your agents to assist you. **No permission is granted to modify this material.**

The drawings, screen captures, diagrams, charts, scripts, configurations, etc., shown in this document are intended solely as an example.

Other products and companies referred to herein are trademarks or registered trademarks of their respective companies or mark holders.

Thanks to “Project JEDI Code Library (JCL)”.

Copyright @2025 Everest Software LLC All rights reserved.

Online help files

The help files included as part of the software installation will always contain the most complete and up to date information about the software.

This manual was intended to provide an overview of the software as well as provide useful information prior to installation of the software.

Manual layout

This user manual starts with the first menu item in configuration and proceeds to cover all areas. Use the table of contents to access those parts of the manual of interest.

See the "Sections" area for more layout information.

Definitions

Point: A point is a data object that contains many attributes to provide access to the program. For example, a pressure switch or a pressure transmitter would be a point.

Tagname: In the HMI a tagname is the attribute used to distinguish one point from another point. Each tagname must be unique. Point and tagname are used interchangeable.

Port: Access to external devices for data collection, to connect to points, is normally via a serial connection or Ethernet, using a communication protocol. In some cases, for example ODBC (Open Database Connectivity), the connection is software only. "Port" is used to refer to one of these connection types.

Graphic element: An icon/image/representation/table/glyph the user places in a graphic window.

Graphic: A user created window that contains graphic elements.

Contact support with any questions or needed assistance.

Table of Contents

Overview

The HMI is divided into two main parts. The first part is named "configuration" and the second part is named "runtime".

When the HMI is first started it begins operation in configuration mode. The user can stay in configuration mode, select to begin runtime mode or via a setting, the runtime mode will automatically launch.

Configuration mode is used to create graphics, configure communication drivers, set up alarms, create data points to monitor, etc.

Runtime mode is used to communicate with the external devices and "monitor" the devices per the "configuration", display graphics, log values, etc.

The configuration settings are maintained via "projects". The number of projects is unlimited. Only one project can be active at a time. Each project is a directory and all the files in the directory comprise the project. The project name is the same as the directory name that contains all the project configuration files.

The HMI is very easy to use and is designed to not require any scripting or extensive programming experience before the user can create a project.

The design of the program is compartmented. For example, if the project does not need the HTTP web server features, those parts of the HMI, help files and this manual that cover the HTTP web server can be skipped over.

Sections

This user manual is divided into several sections.

Projects covers what comprises a project and how to create and save projects.

Configuration covers all areas of the HMI that are not covered by other sections of the manual.

Points covers point types, configuration, copying, deleting, exporting, importing, etc.

Graphics covers the graphic editor.

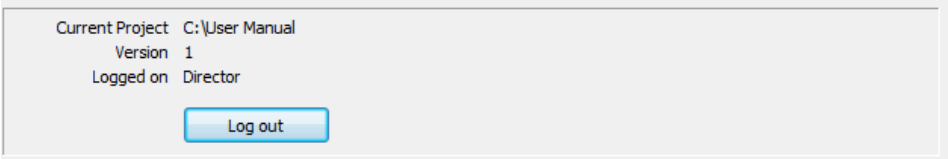
Scripting covers the scripting editor, script commands/functions, etc.

Communication covers all the protocols the HMI supports.

Runtime covers runtime windows and operations.

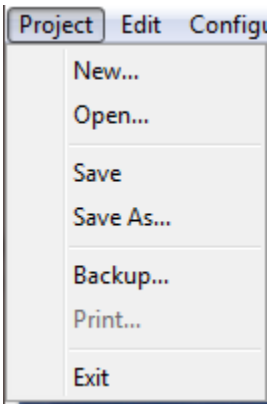
PROJECTS

A project is a collection of files, in a directory, that fully defines the operations of the program during runtime or monitoring mode.



PROJECT MENU

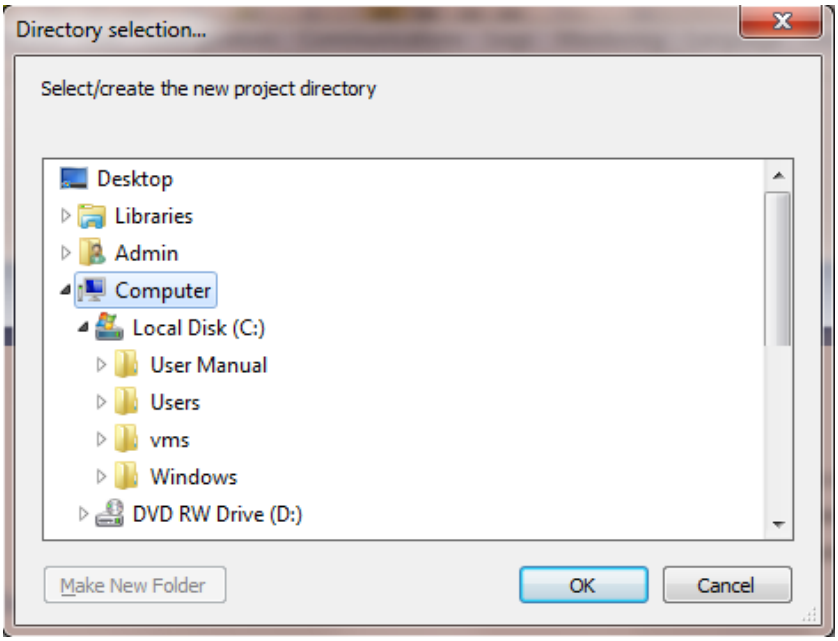
The current project is listed on the main window.



NEW

This menu is used to create a new project. A directory dialog will be displayed.

Note: DO NOT select a protected directory (e.g. c:\program files (x86)) or create a project directory in a protected directory.



Select a directory to store the project files or create a new directory. It is recommended to save/store each project in a separate directory, not a protected directory/path. Mixing project directories will cause confusion and possible loss of data.

Open

This menu is used to open an existing project. The same directory dialog as the “New” menu item will be displayed.

Save

The “Save” menu item will be enabled when a change to the project has been made and the change has not been saved. Selecting the menu item will save the change and the menu item will become disabled.

Save as

The “Save as” menu item will save the current project to the selected directory and make the directory the current project. The same directory dialog as the “New” menu item will be displayed.

Backup

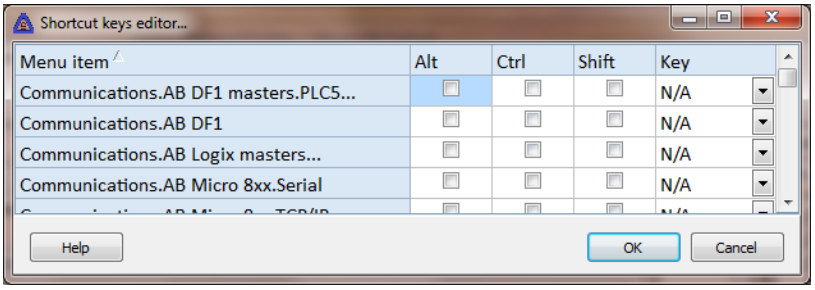
The “Backup” menu item presents the standard file save dialog and prompts with the name of the project as the filename and “zip” as the extension. The backup menu function creates a compressed zip file of the current project directory and saves the data to the file name supplied. The project is not modified.

Exit

The “Exit” menu function ends the configuration program.

Edit

Short cut keys…



This feature provides a method to customize menu shortcut keys.

Cross reference

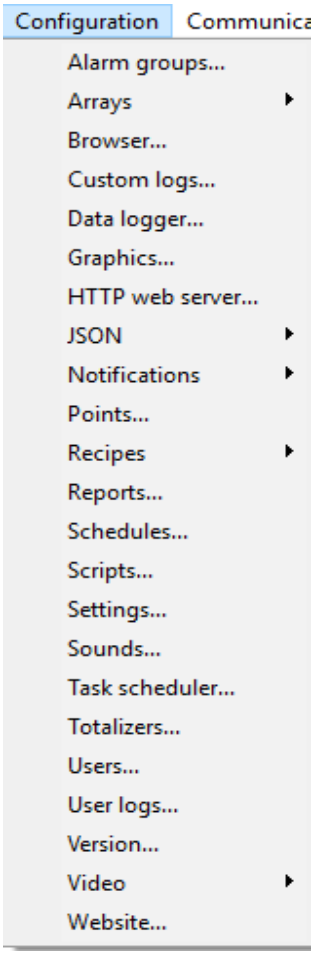
Point/tagnames are used in many sections of the HMI. This feature list all tagnames/port names and any section the name is used. e.g. graphic, logger, script. See the script command "CrossReference" for runtime usage.

Configuration

The configuration section comprises all settings that are not part of a communication driver, points, scripting or graphics.

Configuration menu

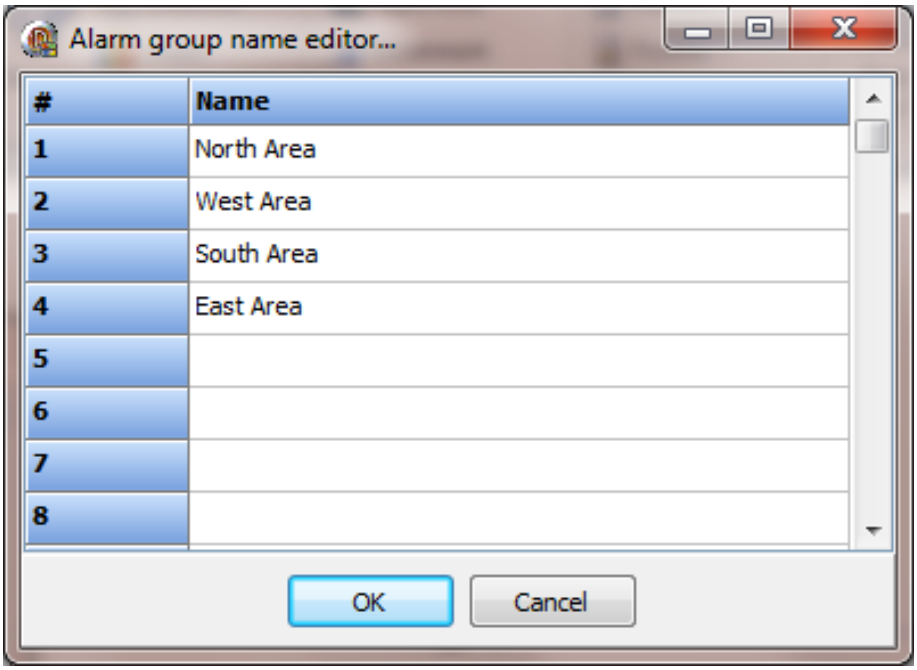
The configuration menu has many items.



Some of the items are defined in this section and the remaining items are given separate sections. For example, "Graphics" has a section.

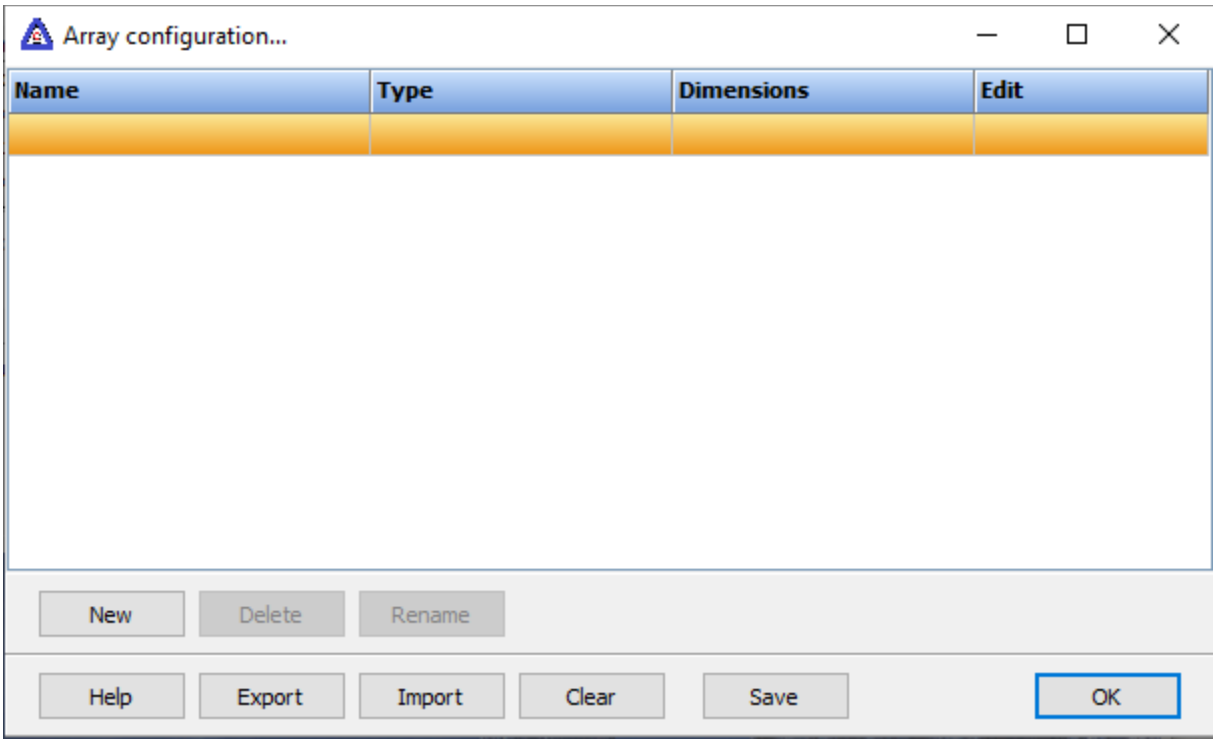
ALARM GROUPS

"Alarm groups" provide a method to partition alarm data. This could be by area, type, machinery, etc. Other elements of the program allow for displaying all groups or a selected group. For example, a graphic screen may show the status of the fire pumps. On that screen could be an "Alarm panel" and the panel only displays the alarm group for fire pumps.



The alarm group names are not required to be unique.

Arrays



"User defined arrays" are defined by a type e.g. byte, word, boolean, one to three dimensions and a name. Each array name must be unique.

Examples

A one-dimension array of 5 elements: (total 5)

0	1	2	3	4	X

A two-dimension array of 2 elements each containing 5 elements: (total 10)

0	1	2	3	4	0	1	2	3	4	X
0					1					Y

A three-dimension array of 2 elements, containing 3 elements, containing 5 elements. (total 30)

0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	X
0					1					2					0					1					2					Y
0															1															Z

When defining array dimensions, the "index" position is determined by the number of dimensions. The "X" index is always present.

A two-dimension array adds the “Y” index and a three-dimension array adds the “Z” index.

	0	1	2
Single	X		
Double	Y	X	
Triple	Z	Y	X

In scripting the array dimension count, defines the syntax for array element access.

Single	[X]
Double	[Y, X]
Triple	[Z, Y, X]

The low/high in the three examples above:

One-dimension	[0] - [4]	(X)
Two-dimension	[0,0] - [1,4]	(Y, X)
Three-dimension	[0,0,0] - [1,2,4]	(Z, Y, X)

FYI, array data is stored sequential. The three-dimension array:

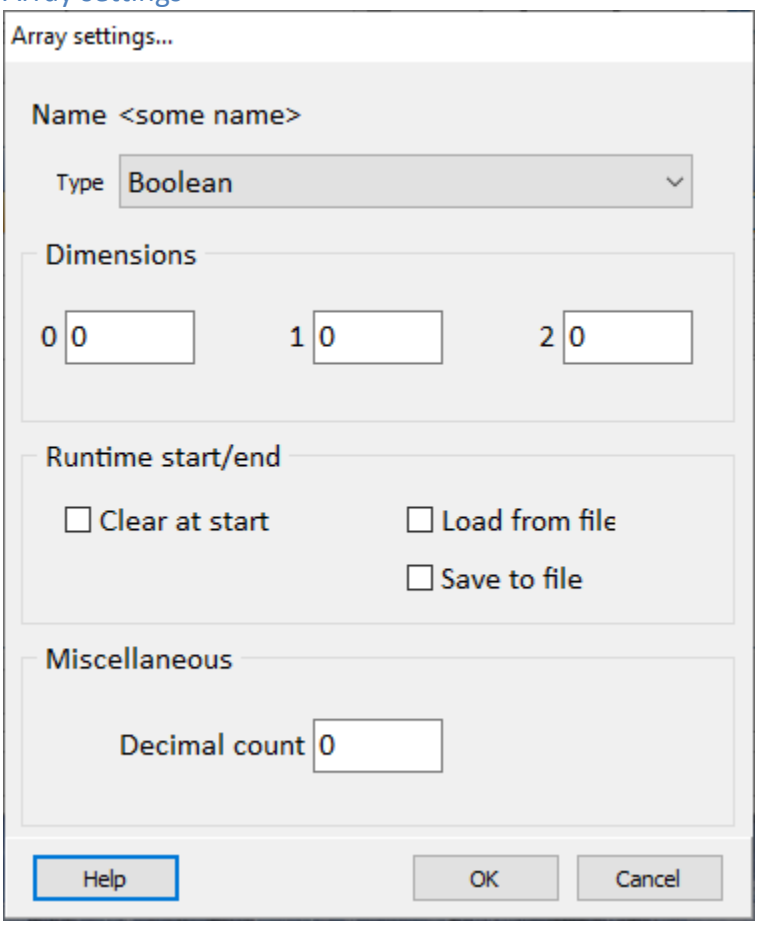
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
0					1					2					0					1					2				
0															1														

When writing/copying array data the HMI will attempt to convert the data to/from the correct type. That is not always possible. Example, the string “ABC” cannot be converted to a number.

See the UArray function for scripting access to arrays.

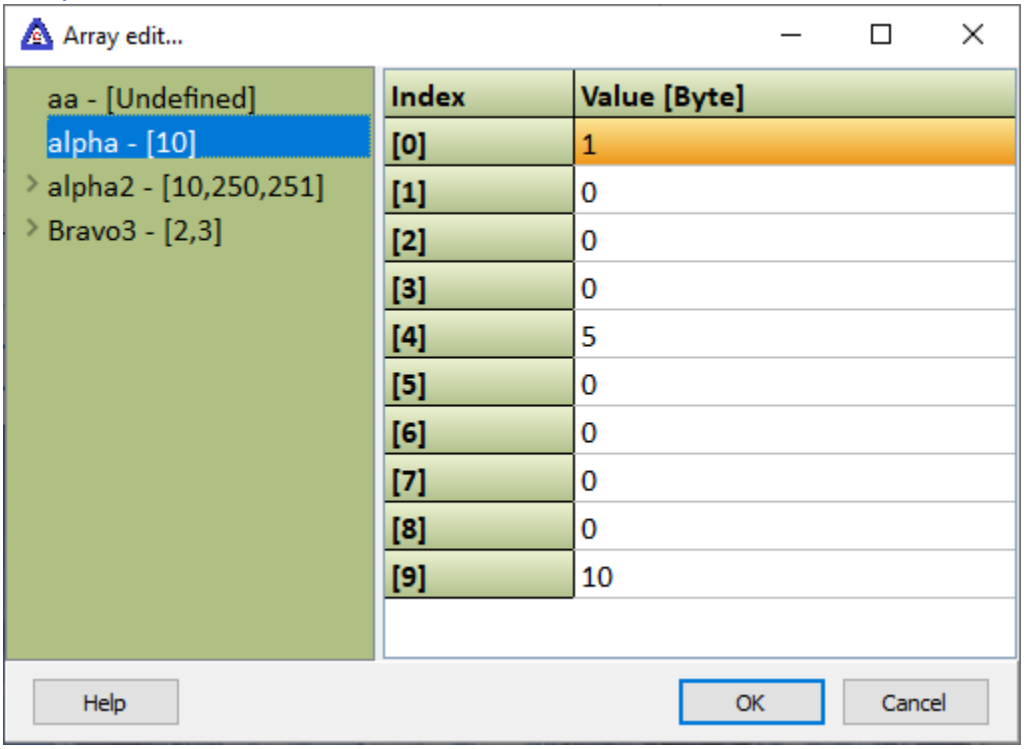
Data type	Low	High	Byte count
Boolean	False (0)	True (<> 0)	1
Byte	0	255	1
Float	2.23E-308	1.79e+308	8
Integer	-2,147,483,648	2,147,483,647	4
Integer (64 bit)	-9,223,372,036,854,775,808	9,223,372,036,854,775,807	8
Long word	0	4,294,967,295	4
Short integer	-128	127	1
Small integer	-32,768	32,767	2
String	Empty	2 GB	n
Unsigned integer (64 bit)	0	18,446,744,073,709,551,615	8
Word	0	65,535	2

Array settings



Clear at start	The HMI loads the array elements data as defined when the array is edited. If enabled, the contains of the array will be set to zero (0) or for string type, a blank string when runtime monitoring starts.
Load from file	If enabled, at runtime start the HMI will search the project directory for a comma separated file (.csv) with the same name as the array, *<array name>.csv*, and load the data into the array. The file can be a link (.lnk) file to the actual CSV file. If a link file, the file name must be *<array name>.lnk*.
Save to file	The reverse of “load from file”.
Decimal count	When viewing the array contents in the configuration editor or the runtime monitor, the number of decimal places used when the type is “float”.

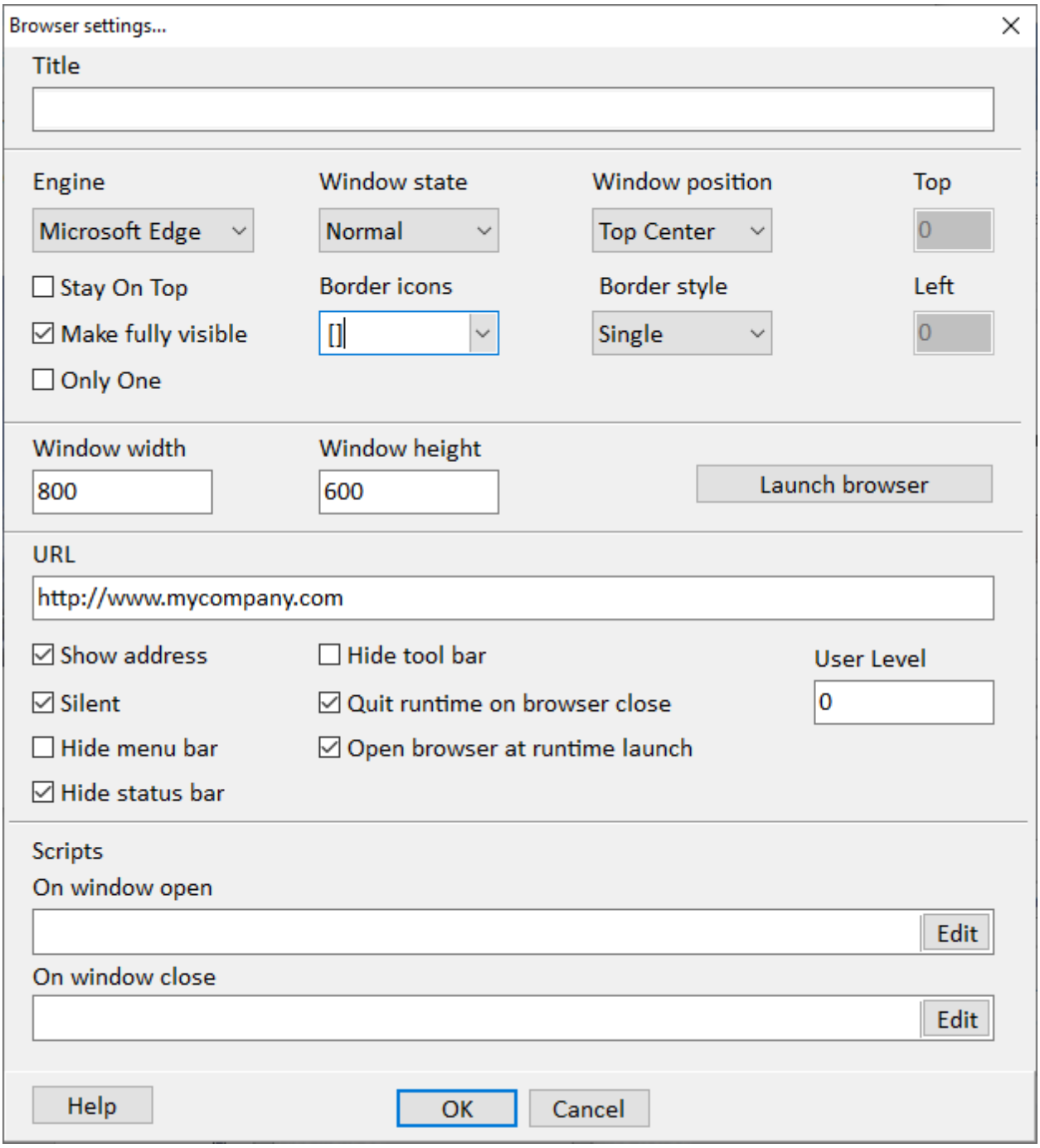
Array edit



Select the array/dimension in the tree and enter a value in the grid.

Browser

The HMI can display a browser window. The browser window uses Internet Explorer (IE) or Microsoft Edge (version 79 or greater) to render the window but, it does not use an IE/Edge window.



Title This string will appear in the title bar of the window if the window is configured to show the title bar.

Engine Internet Explorer or Microsoft Edge (version 79 or greater).

Window State

This determines how the window is to be displayed when opened. Normal is the configured size. Maximized, the window will fill the main screen. Minimized will show the window as a small bar on the bottom of the screen (as determined by Windows).

Window position

Select the position to open the window. As designed opens the window in the last position it was during configuration.

Border style Windows provides for several window border styles. None, single, sizable and dialog. These styles can be viewed by selecting the "Show Window" button in the middle of the screen.

Border icons These are the normal "Windows" icons in the top right corner of the window. If the "Maximum" or Minimize" is enabled the "System" must be enabled.

Stay on top This instructs the window to stay on top off all other windows. Having more than one window open with this attribute set will cause unpredictable window behavior.

Make fully visible

On multi-monitor systems when the window is positioned the window may span several monitors. If this attribute is enabled the form will be repositioned to fully fit on the monitor, if possible. Note: The main monitor is the target monitor for the window when this attribute is enabled.

Only one If this attribute is enabled the program will allow only one window with this name to be opened.

Window width/height

This is the "normal" size of the window, including any border and title bar. The border style and the "normal" size of the window determine the client width and height.

Show window

View the window with the selected settings. If the window configuration does not provide a close button, use ALT+F4 to close the window.

URL

The URL to use when the browser window is opened.

Open browser at runtime launch

When the runtime application is started the browser window will open and navigate to the specified URL.

Silent

This setting specifies if the browser can display dialog boxes. (This does not disable all dialogs).

Quit runtime on browser close

When the last browser window is closed runtime monitoring will stop and the application will quit.

Show address bar

Hide or show the URL address bar.

Hide menu bar

Hide or show the main menu bar.

Hide tool bar

Hide or show the tool bar, the bar below the main menu bar.

Hide status bar

Hide or show the status bar, the bar at the bottom of the window.

User Level

At runtime the logged on user must have at least the level entered to view the window.

Scripts

WARNING: The scripts apply to all browser windows. The 'On Window Open' and 'On Window Close' will execute each time a browser window is opened or closed.

On Window Open

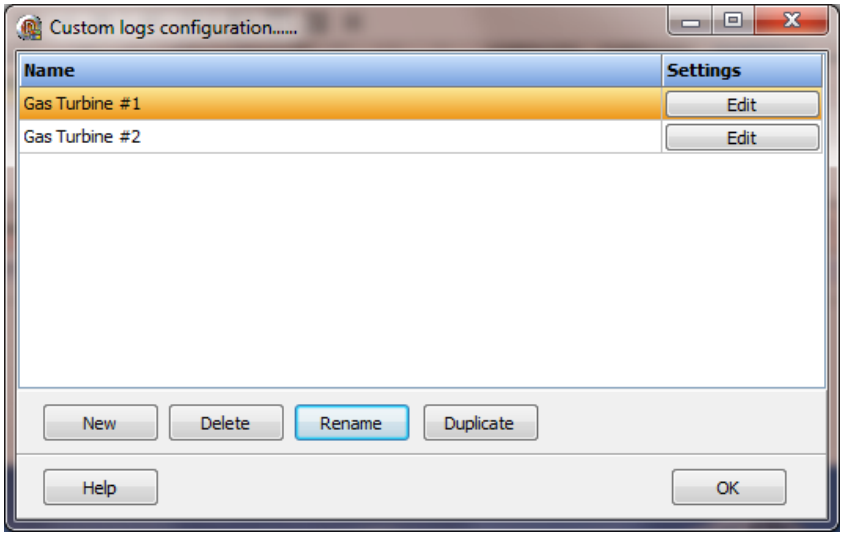
If selected, the script will run once when the window is opened. It will run each time the window is opened. The script is executed as the next to last action for opening a window. The last action is to check the user level. If the logged on user level is not equal to or greater than the user level the window is destroyed.

On Window Close

If selected, the script will run once when the window is closed. It will run each time the window is closed. Warning: The window may be closed before the script executes.

CUSTOM LOGS

The HMI has several logging options. On occasion, one of the built-in logging options does not provide a level of control a user needs for a project. "Custom logs" provides a method to log text data to a file in a format that is customized by the user.



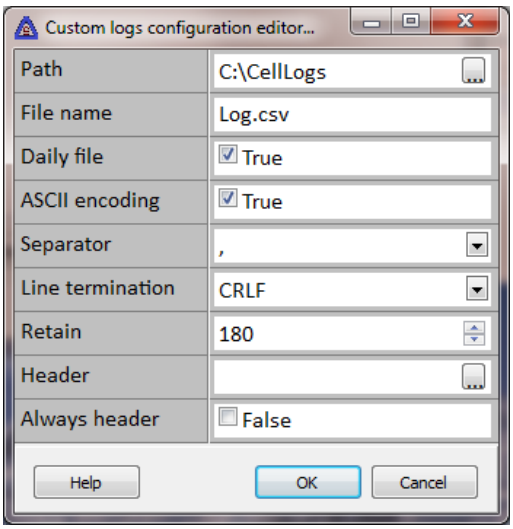
Each custom log object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a custom log object select the "Delete" button.

Custom log configuration



Path

This is the path of the log file. If the path does not exist, the HMI will attempt to create the path. Note: Saving the log to the root of the OS drive, normally “C” (C:\), may cause a failure to save the log. Save the log to a directory on “C” or another drive.

Filename

This is the name of the log file, with extension, in the path selected.

Daily file

If enabled the HMI will append the date to the end of the file name. <file name>-<month>-<day>-<year>.<ext> Example. LogMountain-12-9-2013.txt
The order of the day/month/year is determined by the OS setting.
A new file is created, if needed, at runtime start and at midnight.

Notes:
1) The OS has a limit on the maximum file size. Verify that the limit is not exceeded or failure of the HMI could result.
2) If the log file is empty the file will be deleted.

ASCII encoding

The HMI uses Unicode text encoding by default. If enabled the text encoding will be set to ASCII.

Separator

When one of the log commands is used to log text in columns, this is the text that will be inserted between the columns.

Name	Text	Description
Comma	,	Comma
Tab	#9	Tab
Custom	<user>	The text entered.

Line termination

When a line is added to the file, terminate the line with the value.

Name	Value	Description
CR	#13	Carriage return
LF	#10	Line feed
CRLF	#13#10	Carriage return and line feed. (Default)

Retain

This is the number of days to retain the log files. If the setting is 0 (Zero) old log files are not deleted. Be aware the destination storage device might become full and cause errors.

Header

If a header value is entered, this text will be inserted as the first line of text in the file when the file is created by the HMI at runtime.

Always header

If this attribute is true and the "header" is not blank, the first line in the file will be the header value. If CustomLogFlush is called the file will be flushed and the header line will be inserted. When the file is opened for logging, the first line will be compared to the header. If the text does not match, the header text will be inserted into the first line of the file.

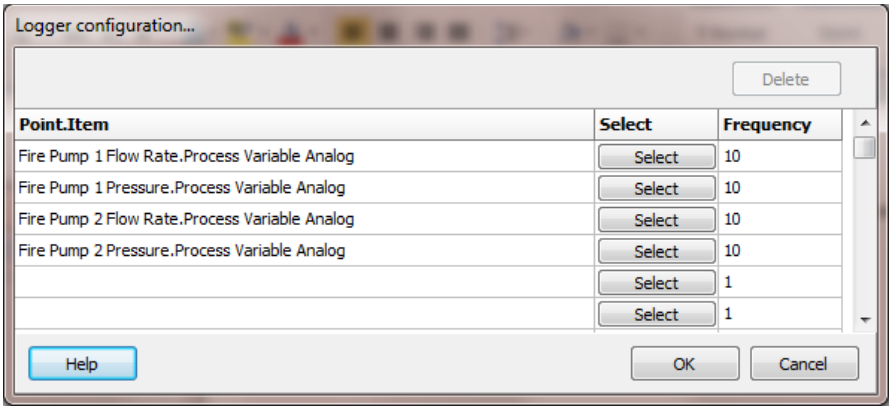
Scripting commands

Command	Description
CustomLogCol	Log the text using the separator (columns).
CustomLogCopy	The active log is copied to the name supplied. The existing file is unchanged.
CustomLogFlush	The active log is emptied.
CustomLogLineCount	The log line count is returned.
CustomLogLog	Log the text (no separator).
CustomLogPoint	The value of the point(s) is logged using the separator (columns).

CustomLogPoint2	Column 1 can be anything and the remaining columns are the value of the point(s) using the separator (columns).
CustomLogSave	The active log is saved to disk.
CustomLogView	View the log in a window.

DATA LOGGER

The data logger is used to store the value of a point at a specified frequency. This data logger is used for trending. The data logs can be exported but, using the ODBC Data Logger or Custom Logs could be a more useful data logger.



A point and item are selected and the frequency is set. The frequency is in seconds. The frequency range is 1 - 32767 seconds.

Note: The frequency for data log points used in the "Round trend chart" must be set to one (1) second.

A file is created for each point/item logged. A point/item can only be selected once.

The log file holds a maximum of one day of data values.

The log file is created at runtime start. If the log file already exists for the day, the new data is appended to the log file.

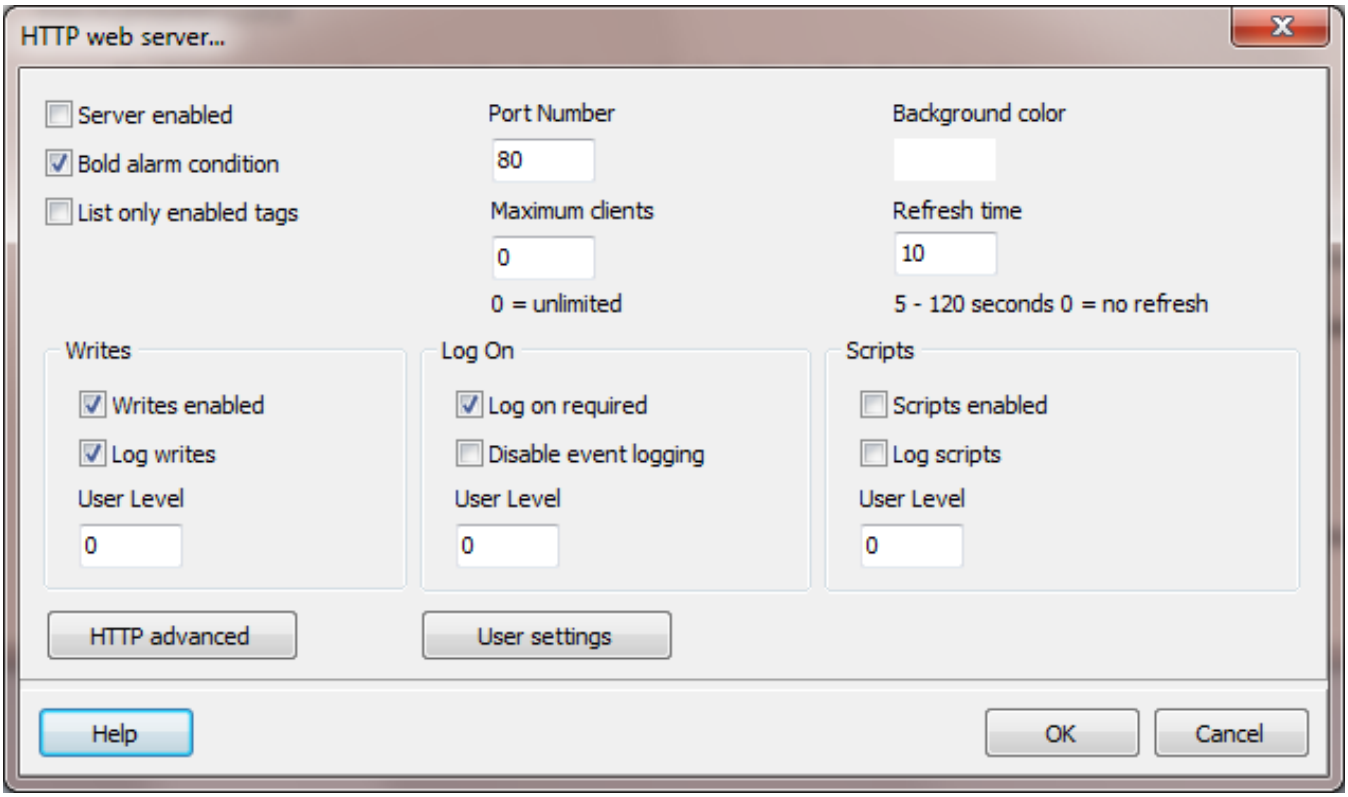
At midnight a new log file is created and the old log file is closed.

The log files are stored in the path defined in the dialog.

If data logger trending is to be used in a graphic window the point(s) to be trended must be configured for logging. ODBC trending is covered later in the manual.

HTTP WEB SERVER

The built in web server is 100% HTML code. User pages can include other code types. The pre-configured web server makes it easy to publish the data in the HMI.



Server enabled

This checkbox must be checked for the server to be active at runtime. The Website cannot share the same port number. Note: The addition of the Website feature may lead to the removal of the HTTP server in the future. We recommend using the Website feature for browser access.

Bold alarm condition

On pages displaying point (tag) data, if the alarm item is enabled and active the text will be bold.

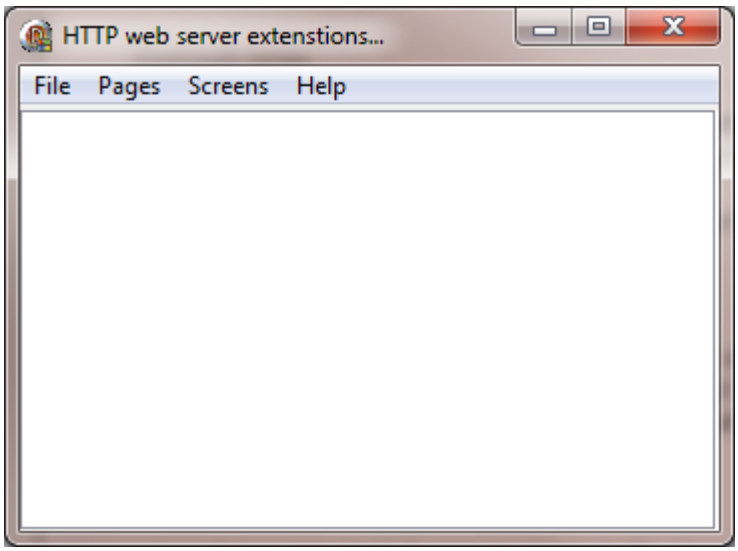
List only enabled tags

Some tags in the database may not be active. There are many possible reasons. One reason could be the tag was forced inactive via the runtime override file. If this attribute is enabled the inactive tags will not be displayed.

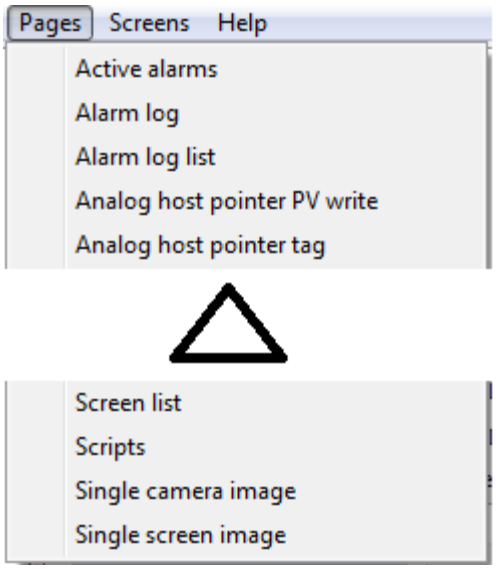
HTTP advanced

This is used to add custom HTML code to the various embedded screens. Adding HTML code may prevent the embedded pages from correct interaction with the client browser. This includes a total failure of the client to render the page.

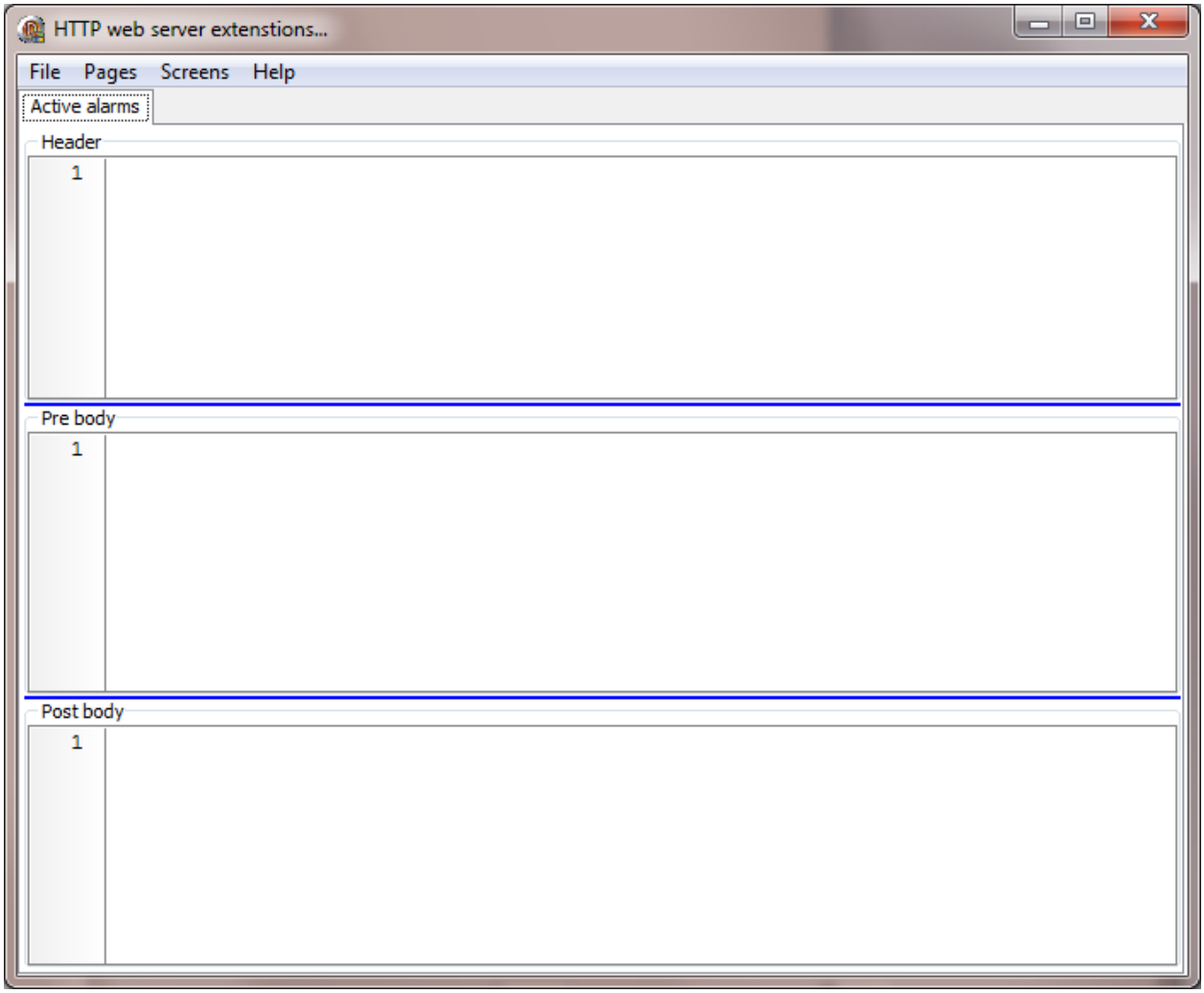
This is the HTTP advanced window.



The “Pages” menu lists all the built in screens. The “Screens” list all the graphic screens.



When selecting a page or screen, the screen displays the three areas that code can be added.



WARNING: Adding code to these sections may cause the feature to fail partially or completely in the web browser, rendering the pages unreadable by the client.

Each web page has three sections, none, one or all of the sections may be used.

Header

The text in this field is placed/inserted before the **</HEAD>** statement in the header section of the page. For screen pages, if the screen has a 'header' it is inserted following the page type 'header' section.

Pre body

The text in this field is inserted just after the first **BODY** statement. In most pages the first statement sets the background color. For other pages it is inserted after the second statement that sets the form action. For the index/home page it is inserted after the page title. For screen pages, if the screen has a 'pre body' it is inserted following the page type 'pre body' section.

Post body

The text in this field is inserted just before the **</BODY>** statement. For screen pages, if the screen has a 'post body' it is inserted before the page type 'post body' section.

Notes:
The sections for the pages are stored in separate files located in the project directory. Each file is loaded before the page is rendered. Changes made between page loads will reflect changes made to the text of the file.

For screens, the sections are loaded when the screen is opened for rendering and the sections are stored with the screen file. Changes made will only display if the window is closed and then opened.

Use <STYLE>H2 {FONT-SIZE: 1pt; COLOR: #FFFFFF};</STYLE> in the 'pre body' section to hide the page title.

Use <STYLE>H3 {FONT-SIZE: 0pt; COLOR: #FFFFFF};</STYLE> in the 'header' to hide the 'Home' link.

The following in the 'pre body' section, will hide the text at the top of the page. Really it changed all text with the 'A' tag.

<STYLE> A {FONT-SIZE: 0pt; COLOR: #FFFFFF};</STYLE>

These commands are case sensitive.

To redirect to another page when a page opens place this in the header:
<meta HTTP-EQUIV="REFRESH" content="0; url=/SCRN/ID=Screen name">

Example: <meta HTTP-EQUIV="REFRESH" content="0; url=/SCRN/ID=Pump Pressure Trend">

To open a screen via HTML use: "/SCRN/ID=Screen name". Replace 'Screen name' with the name of the screen.
Example: <A HREF="/SCRN/ID=Pump Pressure Trend">Pump Pressure Graph</A>

To open a **custom page** via HTML use: "/CP/Page name". Replace 'Page name' with the file name. Do not include the path.

Example: <A HREF="/CP/My only page.html">One custom page</A>

Custom HTML pages must be in the project directory '\HTTP Server\Custom pages\' path.

Each tag must use the head marker '##=' and the tail marker '$$?'.
Example: ##=PT(Tank.Process Variable Analog) $$?

Notes:
1) All text is case sensitive.
2) Files larger than 32K are not processed.

Tag commands

Prefix	**Operation**
PT	Collects the value of a point.item
SG	Collects the value of a script global
DT	Date time display

Points

The value of the point.item will be returned.
##=PT(Tank.Process Variable Analog)$$?

The value of the point.item will be returned with 3 decimal places. The default is 2 and does not need to be present.
##=PT(Tank.Process Variable Analog~D3)$$?

The value of the point.item will be returned using the supplied strings. The default is True/False and does not need to be present.
##=PT(Tank.Process Variable Digital~TOpen~FClosed)$$?

If the point.item is not found 'PTERROR' is returned.

Script globals

The value of the script global will be returned. (Section.item)
##=SG(Logged on.User)$$?

The value of the script global will be returned and limited to the first ten characters. Default is all characters and does not need to be present.
##=SG(Logged on.User~L10)$$?

If the script global is not found 'SGERROR' is returned

Date/Time

The date and/or time will be returned. No format specified will use the system default.

```
##=DT$$?                                  // as defined by the OS
##=DT(h:n:s d/m/yy)$$?                    // 13:6:36 24/5/11
##=DT(mm/dd/yyyy hh:mm:ss)$$?             // 05/24/2011 13:06:36
##=DT(mm-dd-yyyy hh:mm:ss)$$?             // 05-24-2011 13:06:36
##=DT(hh:mm:ss)$$?                        // 13:06:36
```

A tag or script global can be modified via a URL.

The URL is case sensitive.

If the command is received without error a '204 No content' is returned to the browser, otherwise, a 404 error will be returned.
If HTTP writes are not enabled a 403 error will be returned.
If HTTP logon required is enabled and the user's level is not sufficient, a 403 error will be returned.

To write a point

/CPW/PT?T=<tagname>&I=<item>&V=<value>

Examples:
/CPW/PT?T=Pump 1 Setpoint&I=Process Variable Analog&V=12.34
/CPW/PT?T=Pump 1 Run Command&I=Process Variable Digital&V=True
/CPW/PT?T=Pump 1 Run Command&I=Process Variable Digital&V=False
/CPW/PT?T=Pump 1 Run Command&I=Process Variable Digital&V=1
/CPW/PT?T=Pump 1 Run Command&I=Process Variable Digital&V=0

To write a script global

/CPW/PT?S=<section>&I=<item>&V=<value>

Example:
/CPW/PT?STrip report&I=Cause&V=Water entered motor from roof leak

To force a log out of the user use: "/Logout".

Example: <A HREF="/logout">Log out</A>

This example places a button on the form and when selected the user will log out.

```
<form
action="/logout.html" method="get">
<input type="submit" value="Log out">
</form>
```

Custom page example

This is an example page containing one of each tag read/write.

Screen capture.

Item	Tag value, script global or date/time	Tag (head and tail marker removed)
A_Test_1	42.34	PT(A_Test_1.Process Variable Analog)
B_Test_1	True	PT(B_Test_1.Process Variable Digital)
Script global	1234	SG(excel.File name)
DateTime	7/21/2012 2:59:57 PM	DT

A_Test_1 12.34 onClick location.href="/CPW/PT?T=A_Test_1&I=Process Variable Analog&V=12.34"

Set B_Test_1 True onClick location.href="/CPW/PT?T=B_Test_1&I=Process Variable Digital&V=True"

Set B_Test_1 False onClick location.href="/CPW/PT?T=B_Test_1&I=Process Variable Digital&V=False"

Set SG to ABCD onClick location.href="/CPW/PT?S=excel&I=File name&V=ABCD"

Set SG to 1234 onClick location.href="/CPW/PT?S=excel&I=File name&V=1234"

The End

HTML code

Some of the code might be redundant (like setting font styles). The code was generated with an HTML generator.

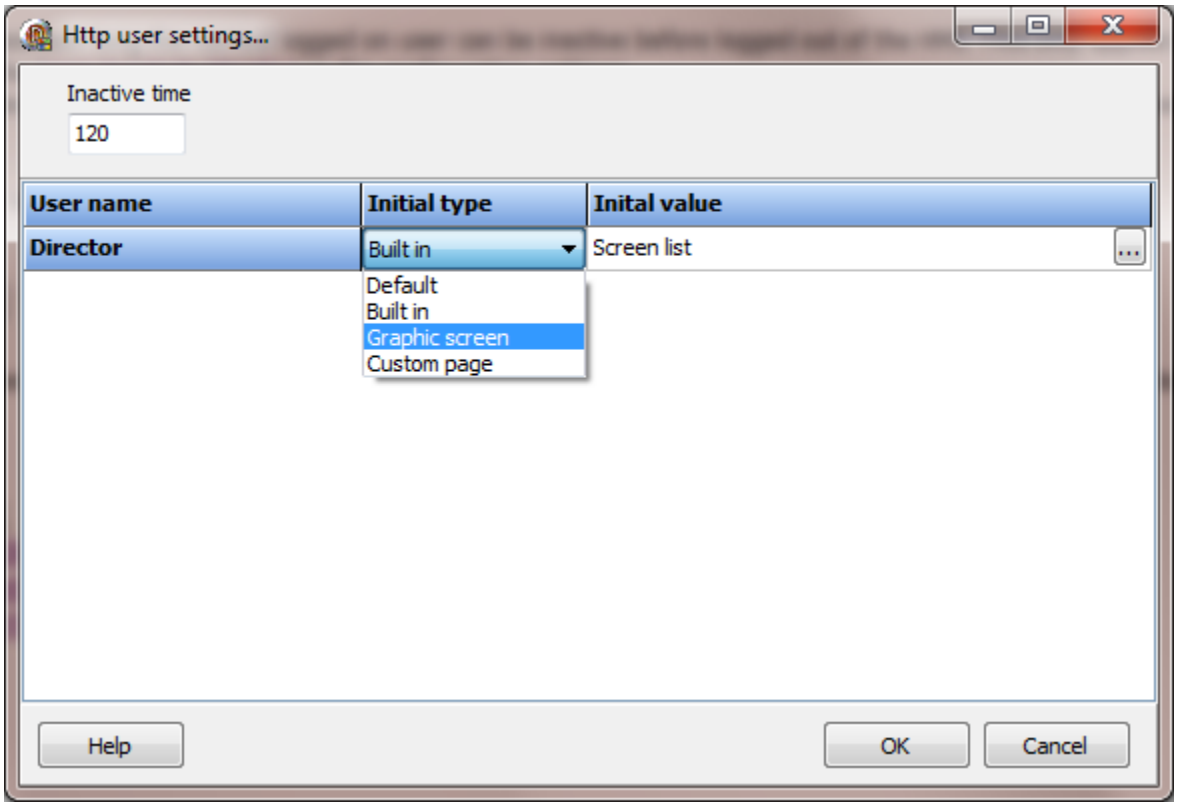
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
 <head>
  <title></title>
  <meta content="text/html; charset=unicode" http-equiv="Content-Type">
 </head>

 <body>
```

```
<table style="WIDTH: 889px; HEIGHT: 148px" border="1" cellspacing="1" cellpadding="1" width="889">
  <tr>
                <td style="WIDTH: 166px; HEIGHT: 28px">
                 <font color="#000000" face="Tahoma" size="2">Item</font>
                </td>
                <td style="WIDTH: 293px; HEIGHT: 28px">
                 <font size="2" face="Tahoma">Tag value, script global or date/time</font>
                </td>
                <td><font face="Tahoma" size="2">Tag (head and tail marker removed)</font>
      </td>
              </tr>
              <tr>
                <td style="WIDTH: 166px; HEIGHT: 28px">
                 <font color="#000000" face="Tahoma" size="2">A_Test_1</font>
                </td>
                <td style="WIDTH: 293px; HEIGHT: 28px">
                 <font size="2" face="Tahoma">##=PT(A_Test_1.Process Variable Analog)$$?</font>
                 </td>
                <td>
                 <font size="2" face="Tahoma">PT(A_Test_1.Process Variable Analog)</font>
                </td>
              </tr>
              <tr>
               <td style="WIDTH: 293px; HEIGHT: 28px">
                <font color="#000000" face="Tahoma" size="2">B_Test_1</font>
               </td>
               <td style="WIDTH: 293px; HEIGHT: 28px">
                <font size="2" face="Tahoma">##=PT(B_Test_1.Process Variable Digital)$$?</font>
               <td>
                <font size="2" face="Tahoma">PT(B_Test_1.Process Variable Digital)</font>
               </td>
              </tr>
              <tr>
                <td style="HEIGHT: 28px"><font face="Tahoma" size="2">Script global</font>
                </td>
                <td style="WIDTH: 293px; HEIGHT: 28px">
                 <font size="2" face="Tahoma">##=SG(excel.File name)$$?</font>
                </td>
                <td>
                 <font size="2" face="Tahoma">SG(excel.File name)</font>
                </td>
              </tr>
              <tr>
                <td><font face="Tahoma" size="2">DateTime</font>
                </td>
                <td style="WIDTH: 293px">
                 <font size="2" face="Tahoma">##=DT$$?</font>
                </td>
                <td>
                 <font size="2" face="Tahoma">DT</font>
                </td>
              </tr>
  </table>
  <p><input value="A_Test_1 12.34" type="button" name="A_Test_1 12.34"
onclick='location.href="/CPW/PT?T=A_Test_1&amp;I=Process Variable Analog&amp;V=12.34"' style="Z-INDEX: 0; TOP:
147px; LEFT: 10px"><font face="Tahoma" size="2">  onClick  
location.href="/CPW/PT?T=A_Test_1&amp;I=Process Variable Analog&amp;V=12.34"
```

```
</font>
</p>
    <p><input value="Set B_Test_1 True" type="button" name="SetB_Test_1True"
onclick='location.href="/CPW/PT?T=B_Test_1&amp;I=Process Variable Digital&amp;V=True"' style="Z-INDEX: 0; WIDTH:
135px; HEIGHT: 24px; TOP: 147px; LEFT: 10px" size="16"><font face="Tahoma" size="2">  onClick  
location.href="/CPW/PT?T=B_Test_1&amp;I=Process Variable Digital&amp;V=True"
</font>
</p>
    <p>
     <input value="Set B_Test_1 False" type="button" name="SetB_Test_1True"
onclick='location.href="/CPW/PT?T=B_Test_1&amp;I=Process Variable Digital&amp;V=False"' style="WIDTH: 135px; HEIGHT:
24px" size="15"><font face="Tahoma" size="2">  onClick  
location.href="/CPW/PT?T=B_Test_1&amp;I=Process Variable Digital&amp;V=False"
</font>
</p><font size="2" face="Tahoma">
    <p>
     <input value="Set SG to ABCD" type="button" name="SetB_Test_1True"
onclick='location.href="/CPW/PT?S=excel&amp;I=File name&amp;V=ABCD"' style="WIDTH: 135px; HEIGHT: 24px"
size="15"><font face="Tahoma" size="2">  onClick   location.href="/CPW/PT?S=excel&amp;I=File
name&amp;V=ABCD"
</font>
</p></font><font size="2" face="Tahoma">
    <p>
     <input value="Set SG to 1234" type="button" name="SetB_Test_1True"
onclick='location.href="/CPW/PT?S=excel&amp;I=File name&amp;V=1234"' style="WIDTH: 135px; HEIGHT: 24px"
size="15"><font face="Tahoma" size="2">  onClick   location.href="/CPW/PT?S=excel&amp;I=File
name&amp;V=1234"
</font>
</p></font>
    <p><font size="2" face="Tahoma"></font> 
</p>
    <p>
</p>
    <p><font face="Tahoma" size="2"></font></p>
    <p><font face="Tahoma" size="2"></font></p>
     <div style="POSITION: relative; WIDTH: 70px; DISPLAY: inline; HEIGHT: 15px" ms_positioning="FlowLayout"><font
face="Tahoma" size="2">The End
</font>
</div>
    <p><font face="Tahoma" size="2"></font></p>
    <p><font face="Tahoma" size="2"></font></p>
    <p><font face="Tahoma" size="2"></font>
</p>
    <p><font face="Tahoma" size="2"></font>
</p>

 </body>
</html>
```

User settings

If logon is required, and a user logs in, this configuration is used to display the first window the user access.



Note: The 'Log on required' attribute must be enabled for these features to function.

Inactive time

This defines how long the named logged on user can be inactive before the user is logged out of the HMI. This only applies to access via a web browser. For inactivity in the HMI, see the 'User Inactivity Monitor' in the configuration settings.

Note: If the refresh time is greater than zero (time > 0) the user must change pages to reset the inactivity timer.

Each user can have the following settings. If a setting is not used the default setting will be applied.

Initial type/value

This defines the first page the user can access after log on. It can be a graphic screen, a built in page or a custom page. The default page is the built in 'Home' page.

Port number

The TCP/IP port number the port is to use. 80 is the default.

Maximum clients

The maximum number of client connections. 0 = unlimited and is the default.

Background color

For browsers supporting a window background color this is the desired color.

Refresh time

The elapsed time the browser is to delay before performing a refresh on the page. This is applied to pages displaying dynamic data. 10 seconds is the default. 0 seconds is no refresh. The page can always be refreshed via a refresh button or command in the browser.

Note: Automatic refresh will cause the page to reload. If the user has scrolled the window, after reload the page will not be scrolled.

Writes

Enabled: Enables the display of windows to set a tag value.
Log writes: All writes are logged to the event log. IP Address:Tagname : item number: value
User level: The logged on users level required to initiate a write command. If the user level is below this level, the writable items will not be underlined.

Logon

Logon required: A user must log on the server before pages will be displayed.
User level: The logged on users level required to display pages.

If the user level is not high enough, the authentication will fail. If logon is successful an entry is created in the event log. IP Address:Logon : user name

Disable event logging: When "Logon required" is enabled and a user "logs on", an entry is added to the event log. If this attribute is enabled an entry <u>will not</u> be placed in the event log.

Note: Logon via HTTP or via the HMI screen does not affect the other logon. The logons are independent.

Scripts

Enabled: Enables the display of window to execute scripts.
Log scripts: All scripts execute commands are logged to the event log.
User level: The logged on users level required to execute a script.

Note: These three groups of settings have interaction. If logon required is false, "writes enabled", will determine if write commands will be executed. If logon required is true and writes enabled is true, the logged on user level and writes user level determine if write commands will be executed. This also applies to executing scripts.

Anyone can browse without writes/scripts: Logon required = false, Writes enabled = false, Scripts enable = false

Anyone can browse with writes/scripts: Logon required = false, Writes enabled = true, Scripts enable = true *least restrictive

Logon with browse only: Logon required = true, Writes enabled = false, Scripts enable = false **most restrictive

Logon with browse writes/scripts: Logon required = true, Writes enabled = true, Scripts enable = true

Notes:
1. Event logs are displayed latest entry at the top and in descending time.
2. Custom HTML pages must be in the project directory '\HTTP Server\Custom pages\' path.

JSON

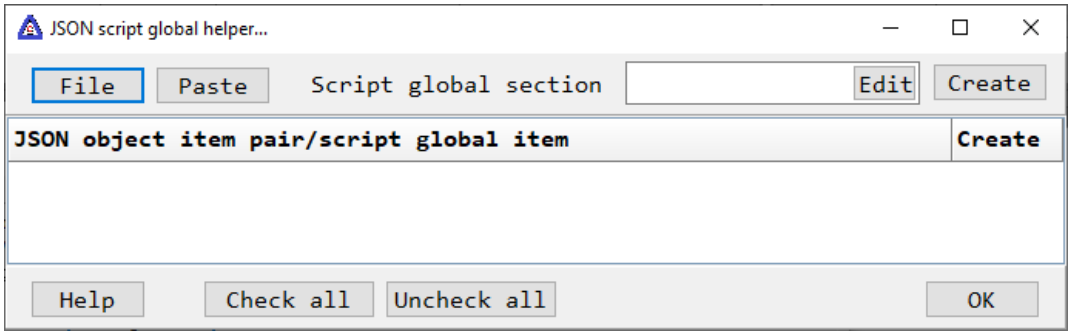
JSON (JavaScript Object Notation) parsing can be time consuming. The HMI provides several methods to process incoming JSON data. See the scripting commands "JSONToScriptGlobal" and "JSONToHostPoints." The JSON validator can be used to verify the JSON input if problems arise with parsing.

Note: The script global section requires two items to process the JSON string at runtime. When using the configuration JSON helpers these two items are automatically generated. Deleting either item will prevent the JSON runtime processing from executing. If manually creating a section, create these two items in the section, first.

HMIJSONSourceHMI: When JSON processing begins the contents of this item are compared against the JSON string of the command. If the strings are different (indicating a value change) the JSON parsing process is executed. To force JSON processing to execute, write a blank string to this item before the JSON processing command is executed.

HMIJSONParseErrorHMI: If an error is encountered while JSON parsing is executing, error information will be written in this item and parsing will cease.

Script global editor



File

Select a text file that contains a JSON object tree (text), the file will be parsed for name/item pairs.

Paste

If the clipboard contains a JSON object tree (text) the text will be parsed for name/item pairs.

Script global section

Select or enter a name for the script global section to be the destination of the parsed name/value pairs. **Note:** If the script global section exists it will be overwritten.

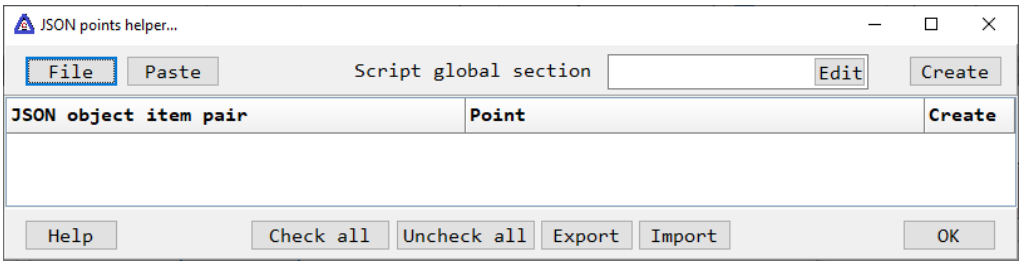
Create

Select the button and the parsed names will be added to the configured script global, if the checkbox for the name is enabled.

Check/Uncheck

Check or uncheck all the grid rows.

Host points editor



File

Select a text file that contains a JSON object tree (text), the file will be parsed for name/item pairs.

Paste

If the clipboard contains a JSON object tree (text), the text will be parsed for name/item pairs.

Script global section

Select or enter a name for the script global section to be the destination of the parsed name/value pairs. Note: If the script global section exists it will be overwritten.

Create

Select the button and the parsed names and points will be added to the configured script global, if the checkbox for the row is enabled.

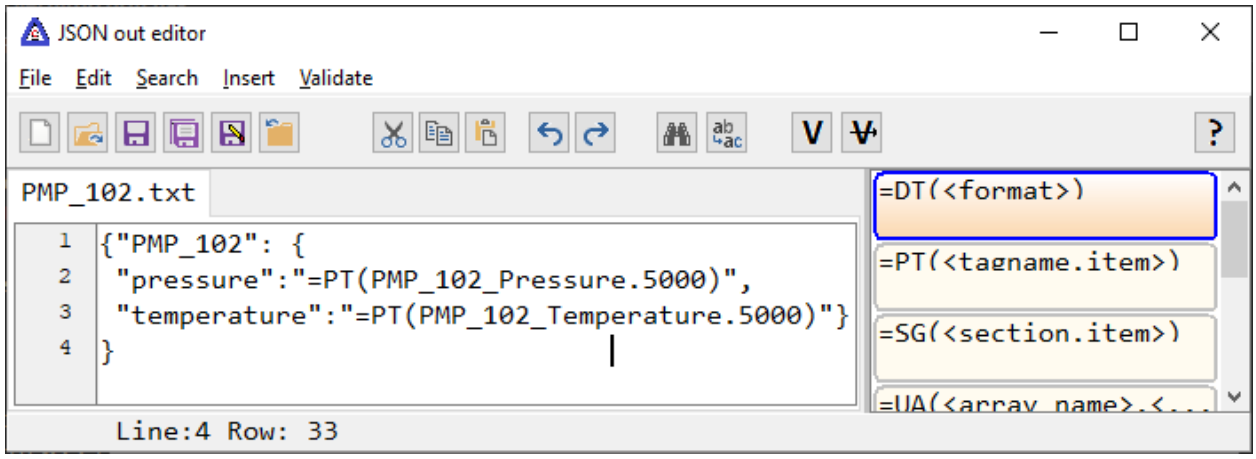
Check/Uncheck

Check or uncheck all the grid rows.

Export/Import

Export or import a CSV (command separated values) file.

JSON out



Using a script to create a formatted JSON string to save or transmit to a remote device, (e.g. MQTT broker) is workable but, can be made easier by using the editor and the script command "JSONOutToString".

Using the example in the above image, when the script command is called:

```
value:=JSONOutToString('PMP_102.txt');
```

the function is executed and "=PT(PMP_102_Pressure.5000)" and "=PT(PMP_102_Temperature.5000)" are evaluated, the text is replaced with the value of the referenced points. An example result would be:

```
{"PMP_102": {
 "pressure":"300.12",
 "temperature":"142.45"}
}
```

Helpers

Four data helpers are provided. Three, helpers **Point value**, **Date/time** and **Script global**, are the same as in the report generator and are defined here.

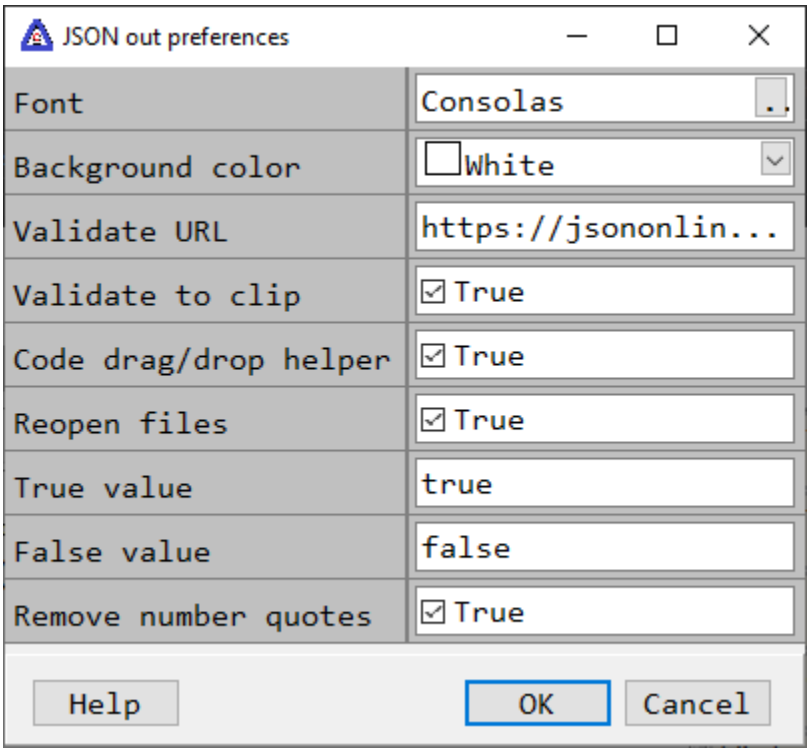
The fourth is for "Array" data access.

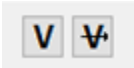
User Array

The value of array elements can be place in the JSON string as a JSON array type i.e. [element 1, element 2, element n]. Note: See "Arrays" for "<dimension>" format.

The helper format is : =UA(<array name>, [<dimension>], <count>);

Preferences



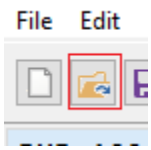
Font	Text appearance in the text memo field.
Background color	The background color of the text memo field.
Validate URL	 Two methods are provided to validate the JSON format. The first method uses HMI validation. Using the currently selected file, selecting the left "V" icon will display a dialog with the validation results. The second method provides for using a web based validator. A default validator URL is supplied.
Validate to clip	When the right "V" is selected the contents of the editor are placed on the clipboard before the URL is launched in the default browser.
Code drag/drop Helper	When dragging and dropping code prototypes from the code list (right side, =DT, =PT, =SG, and =UA) of the window, if this is enabled a window will appear to help configure the code syntax.
Reopen files	When enabled the files that were open when the editor window was closed will be reopened when the window is opened.

True/false value	The value to place in the JSON string at runtime for digital values. Examples: A JSON recipient might require quotes (“) around the true/false text or 1 for true and 0 for false.
Remove number quotes	The JSON specifications does not use quotes (“) around numbers. For the validation to be successful, the code helpers (e.g. “=PT(<tagname.item>)”) must be in quotes. If this property is enabled and the helper evaluates to a number, at runtime, the quotes will be omitted from the string result of the JSONOutToString function call.

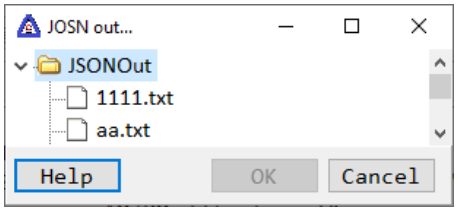
Notes on validation

1) { is required as the first character and } as the last character. The HMI does not add the characters.
2) JSON is based on name:item pairs.
3) : separates the name from the item.
4) , is required for separation of “name:item pairs”. In the example image above line 2 is a name:item pair followed by a command and line 3 is another name:item pair.
5) { or [require a balancing end } or].
6) “ quotes are required for strings.

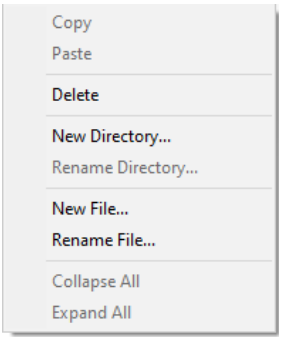
Selecting menu “File/Open” or the file open icon:



Will display the “JSON out...” dialog.



Right click the “tree” and the popup menu will appear:

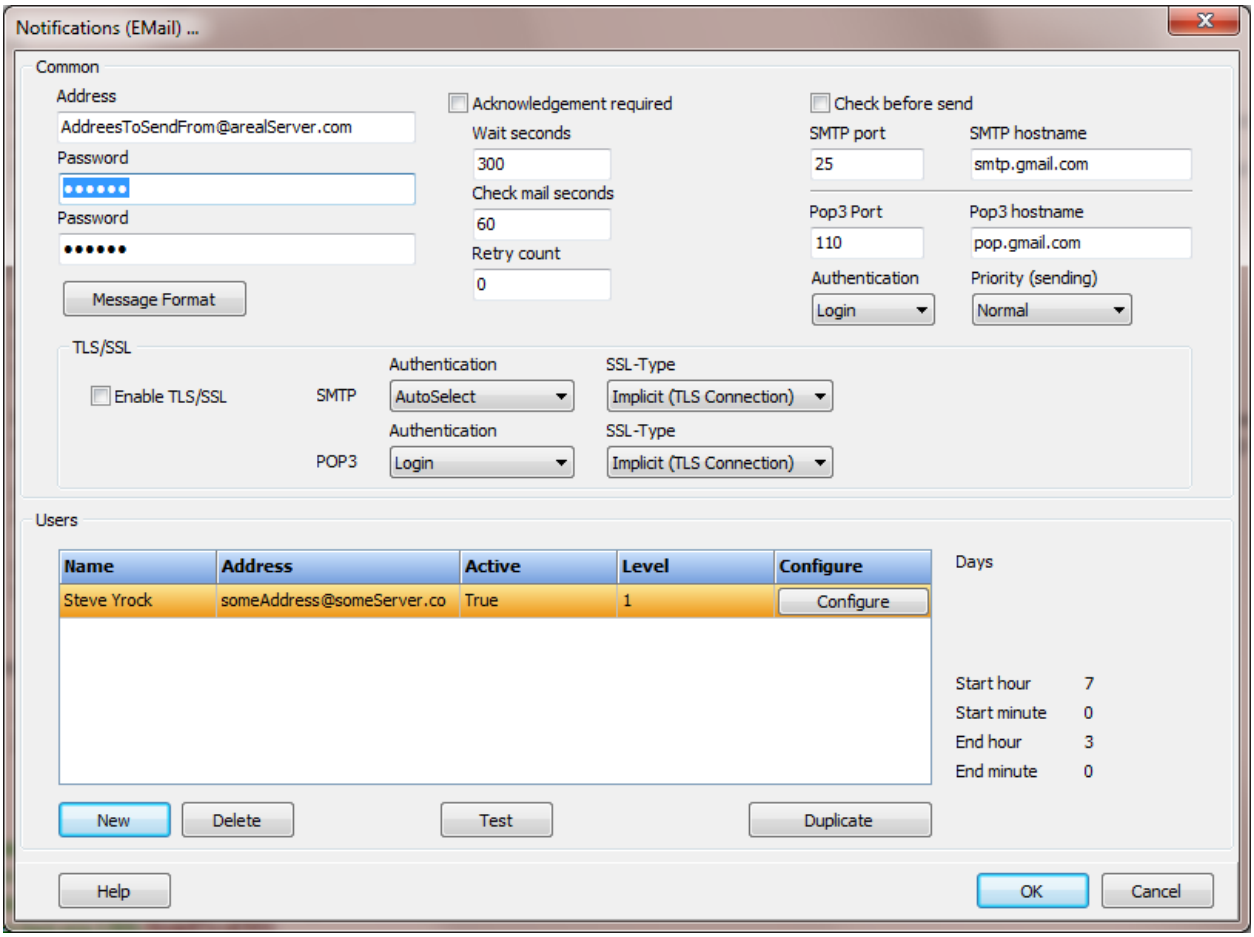


NOTIFICATIONS

The HMI offers two methods, Email and SMS (Short message service), to send notifications when an alarm condition is detected.
Email notifications require an email account and connection to the email server.

SMS notifications require a cell phone/modem that can be accessed via a serial driver, appears (in the Device Manager) as a modem that supports the AT command set and an active cell phone account that includes SMS messaging.

EMAIL



Address

This is the email address used as the sender of the message. This will be the address/name used to connect with the mail server for transmitting and receiving messages. A blank address will disable notifications.

Password

This is the password for the sending/receiving email address. Both password fields must match. (Password can be blank)

Acknowledgment required

If enabled, the notification is sent to the next user with a higher level configured to be notified. If 'acknowledgments required' is not enabled all configured users are notified as configured. If 'acknowledgments required' is enabled the configured users are notified in configuration order and level order until the first user acknowledges the message.

Note: The acknowledgment must be a reply containing the message ID line as the last line. The best method to reply is; select the reply command in the client and then select send without any additions or changes. If the email parser cannot determine which notification message the reply acknowledges, the reply will be ignored.

Acknowledge wait time

When a notification is sent and acknowledgments are required this is the amount of time to wait, in seconds, for an acknowledgment to be received before sending a notification to the next level user. An acknowledgment ends the sending of notifications for the alarm. If the wait time is zero the user level setting is ignored and a notification is sent to all configured users for the alarm. If that is the configuration and the retry count is not zero the notification will immediately be sent again up to the value of the retry count.

Check mail received

This is the amount of time, in seconds, to wait between polling the mail server for new mail.

Retry count

This is the number of additional times to attempt a notification. If an acknowledgment is not received from any user configured for the notification, after all users have been processed, the process is restarted if the count has not been reached.

Note: If Gmail is the email host, the default settings in Gmail need to be changed to:
"2. When messages are accessed with POP delete Gmail's copy."
This allows the HMI to properly read/track and delete acknowledgments.

Check before sending

Some mail servers require the sender to check for incoming mail before the sender can transmit mail.

SMTP port

The mail server SMTP port number. (Simple Mail Transfer Protocol (SMTP) used for transmissions of emails across the internet.)

SMTP hostname

This is the mail servers SMTP name. This field must be valid.

Pop3 port

This is the mail server POP3 port number. (Post Office Protocol version 3 (POP3), used to retrieve e-mail from a remote server.)

Pop3 hostname

This is the mail server POP3 server name. If 'check before send' or 'acknowledgment required' is enabled, this field must be valid.

Pop3 Authentication

The “Authentication” protocol used to validate the address/user with the pop3 server. (SMTP server authentication is automatic.)

Priority (sending)

This is a flag to indicate the message priority.

TLS/SSL (Transport Layer Security (TLS) protocol Secure Sockets Layer (SSL) protocol)

Enable TLS/SSL

The server must support TLS/SSL protocol.

SMTP

Authentication

When TLS/SSL is enabled, this is the authentication type used for email sending.

SSL-Type

When TLS/SSL is enabled, this is the connection type.

Priority

When TLS/SSL is enabled, a flag to indicate the message priority.

POP3

Authentication

When TLS/SSL is enabled, the type of authentication to for email receiving.

SSL-Type

When TLS/SSL is enabled, this is the connection type.

Test button

The program will attempt to send a test email to the selected user, using the common settings. In the results, near the bottom, a string "****Notify SMTP test complete**** Error: 0" will be inserted. If the error is zero (0) the test was successful.

TLS/SSL testing

The settings used for testing TLS/SSL with Gmail:

SMTP host: smtp.gmail.com
SMTP port: 465
Authentication: Autoselect
SSL-Type: Implicit (TLS Connection)
Priority: Normal

POP3 (if required)
POP3 host: pop.gmail.com
SMTP port: 995
Authentication: Login
SSL-Type: Implicit (TLS Connection)

Address and password, the account values.

Notes:
Gmail might return errors. If so try the below steps.

1. Log into your google email account and go to this link: https://www.google.com/settings/security/lesssecureapps and set "Access for less secure apps" to ON. Test if the issue is resolved. If the issue is not resolved, continue to #2.

2. Go to https://support.google.com/accounts/answer/6009563 (Titled: "Password incorrect error"). This page says "There are several reasons why you might see a “Password incorrect” error (aka 534-5.7.14) when signing in to Google using third-party apps. In some cases even if you type your password correctly." This page gives 4 suggestions of things to try.

The settings used for the testing TLS/SSL with CenturyLink:

SMTP host: smtp.centurylink.net
SMTP port: 587
Authentication: Autoselect
SSL-Type: Explicit (STARTTLS)
Priority: Normal

POP3 (if required)
POP3 host: pop.centurylink.net
SMTP port: 995
Authentication: Login
SSL-Type: Implicit (TLS Connection)

Address and password, the account values.

Message Format

The message header and body settings are configured to display alarm data fields. All messages are sent as plain text. Email messages have few restrictions on size. SMS messages have severe limits. Some providers limit the character count to 100 characters. Normally, the limit is 160 characters for the Latin alphabets and 70 for non-Latin alphabets.

Note: The header settings are not used for SMS notifications.

Users

Each address, to be notified, has several settings. A user/address can be in the list more than once.

Name

This is a name to make viewing and working with the list easier. **Note:** When using a script to send a message via a user name, the first name in the list to match the name is selected.

Active

If true the processing of this user is included when an alarm arrives. This can be used to prevent notifications to the user without changing other settings. Example: The user goes on vacation, sick days, etc.

Address

The address for emails or telephone number for SMS, of the recipient of the message.

Level

This is the level (1 - 5000) used to determine the order of notification for an alarm when 'acknowledgments required' is enabled.
If Joe is level 3 and Bob is level 4, a notification is first sent to Joe. If an acknowledgment is not received after the wait time has elapsed a notification is sent to Bob. If 'acknowledgments required' is not enabled this field is not evaluated.

Day

The day of the week the user is to be included in notifications.

Start hour, start minute (24hour)

The beginning time of the day the user is to be included in notifications.

End hour, end minute (24hour)

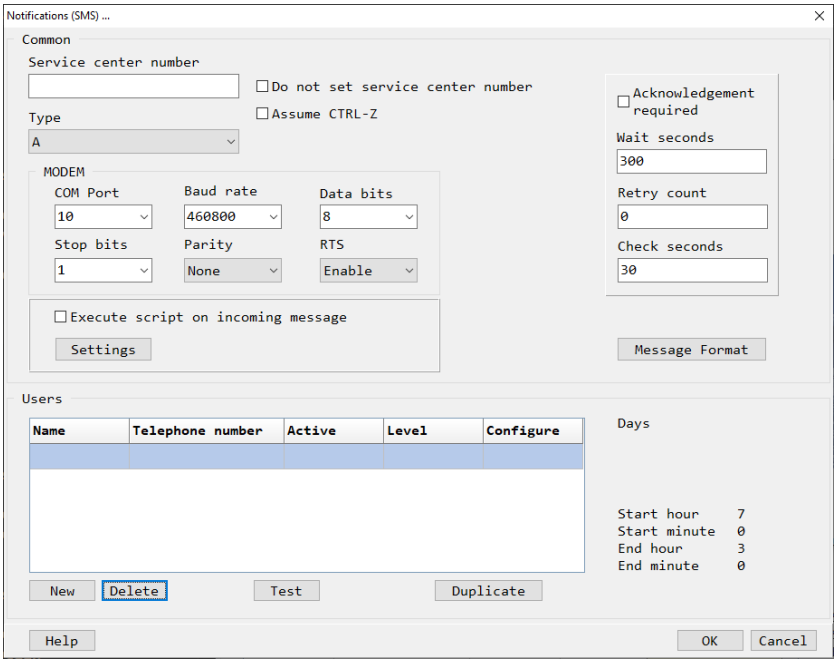
The ending time of the day the user is to be included in notifications.

Note: Setting all the time fields to zero (0) selects 'All' time during the day.

Alarm groups

These are the alarm groups this user is to be included for notifications.

SMS



Note: If the connection to the cell phone is lost, (e.g. HMI execution is terminated, USB cable is disconnected, etc.) during the transfer of the text portion of the message to the cell phone/modem, the cell phone/modem can become unresponsive. The solution; turn off the cell phone/modem and then turn the cell phone back on.

Common

Service center number

This is the telephone number, supplied by the cell service provider, to access the SMS network. Using AT&T, it is listed as the "Text message center address".

Type

A: Samsung A887
B: Nokia C2-01 (SMS send only, note 1)

Both types should work with all phones from the same manufacture. If using a cell phone of a different manufacture/type and have tried both types without success, please contact support. There is no standard for cell phone/modem access to send/receive SMS messages. We want to support as many cell phone/modem as possible.

Note:
The Nokia C2-01 does not support reading SMS messages via the AT command set. "Acknowledgment required" and “Execute script on incoming message” will not function.

Do not set service center number

A specification for sending SMS message does not exist. Some phone/devices have implemented sending SMS message via the AT command set, using a common but, not identical method. Some devices return an error when setting the service center number. Enable this attribute, if when testing, a watchdog timeout occurs when setting the service center number or an error occurs.

Assume CTRL-Z

A specification for sending SMS message does not exist. Some phone/devices have implemented sending SMS message via the AT command set, using a common but, not identical method. Some devices do not respond when a CTRL-Z command is sent. Enable this attribute, if when testing, a watchdog timeout occurs when waiting for a CTRL-Z response or contact support.

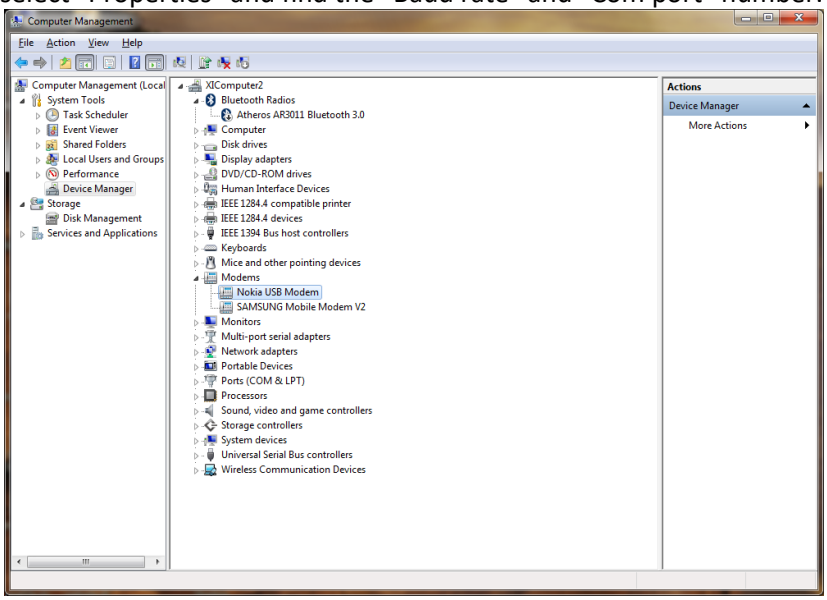
MODEM

Some cell phones install drivers for USB that appear as serial ports and MODEMS. The HMI needs the MODEM driver.

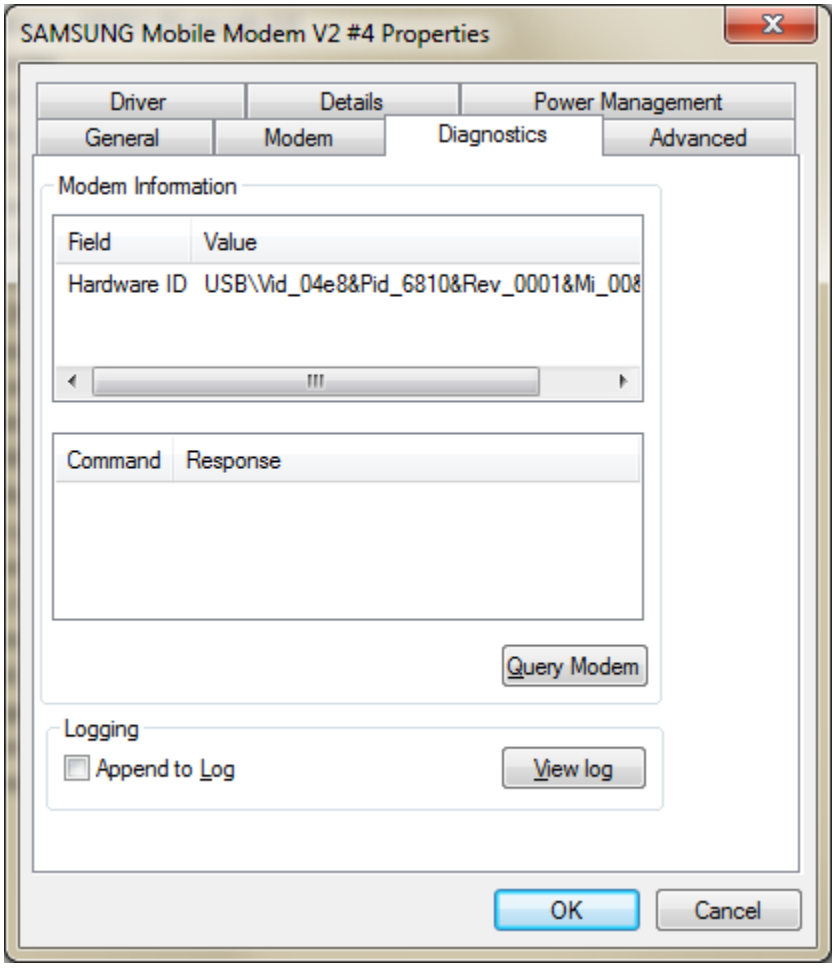
For Samsung, installing the "PC Studio" installed the drivers. "PC Studio" can be removed, if not needed, after the drivers are installed. Another method to install the drivers might exist.
On another computer running Windows 7 and connected to the internet, the serial and modem drivers were installed when the cell phone was connected via the USB port to the computer.

For Nokia, the "Nokia_Connectivity_Cable_Driver_eng.msi" was used to install the drivers.

Below is a picture of the MODEM drivers in the "Device manager". Right click on the MODEM, select "Properties" and find the "Baud rate" and "Com port" number.



Sometimes after accessing the cell phone with the manufactures application, the cell phone will not respond. Query the modem, in the driver settings, to reset the communications link.



Acknowledgment required

If enabled, the notification is sent to the next person with a higher level configured to be notified. If 'acknowledgments required' is not enabled all configured users are notified as configured. If 'acknowledgments required' is enabled the configured users are notified in the configuration order and level order until the first address acknowledges the message. **Note:** Any acknowledgment from the user will acknowledge all notifications for that user.

Wait seconds

When a notification is sent and acknowledgments are required, this attribute is the amount of time to wait, in seconds, for an acknowledgment to be received before sending a notification to the next higher level user. An acknowledgment from a user stops the sending of notifications for the alarm. If the wait time is zero, the user level setting is ignored and a notification is sent to all configured users for the alarm.

Check seconds

This is the amount of time, in seconds, to wait between polling the cell phone for a new message. It is used when “Acknowledgment required” or “Execute script on incoming message” is enabled and properly configured. **Note:** Setting this value too low can cause failure of the system.

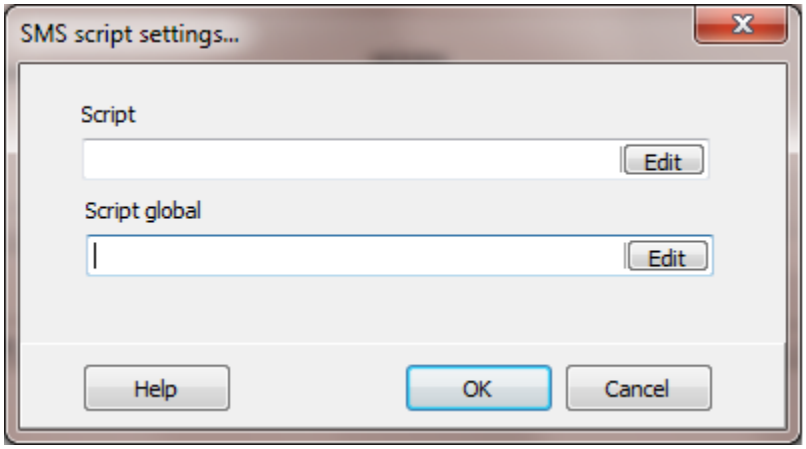
Retry count

This is the number of additional times to attempt a notification. If an acknowledgment is not received from any user configured for the notification after all users have been processed the process is restarted if the count has not been reached.

Execute script on incoming message

If enabled, and an incoming SMS message is returned from the phone/device, the message will be a parsed, saved in a script global and the script is executed. See below.

Settings



Script

The script to be executed when an incoming SMS message is processed.

Script global

The script global to save the SMS message information. The message is parsed and saved in several items of the selected script global section.

TimeStamp	The date and time the message was received.
SendingNumber	The phone number attached to the message. *1
User	The user name, if found, that matches the “SendingNumber”. *1
Text	The text of the message.

Note:

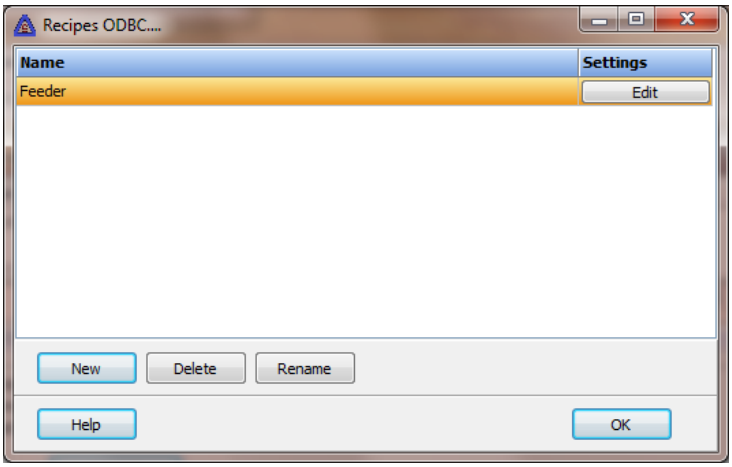
1) The phone number attached to the incoming message may or may not match the cell phone number used to send a message to the same phone. For example: Sending a message the phone number is 1713xxxXXXX and incoming number could be 713xxxXXXX. A published standard does not exist and each carrier might be different.

Message format

Refer to the Email message format for information.

RECIPES

The HMI supports recipes using XLS (Excel) recipe files and/or connections to a database via ODBC.



Each ODBC recipe is listed in the window.

To create a new recipe select the "New" button and enter a name. Each name must be unique across all recipe names.

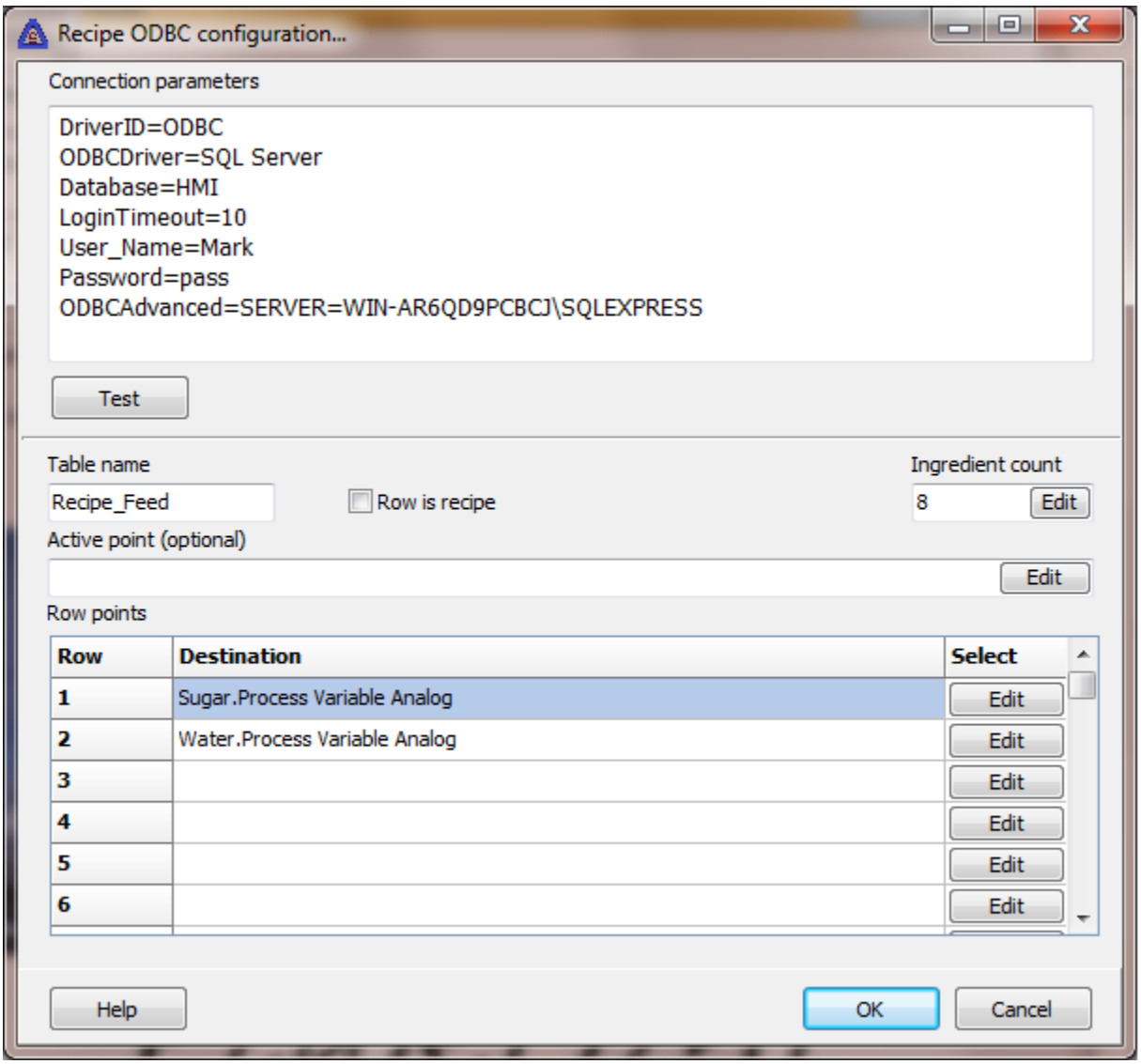
To rename a recipe select the "Rename" button and supply a new name.

To delete a recipe select the "Delete" button.

Notes:

1) Use the Recipe XLS if Excel file access is needed.
2) Some versions of Excel allow the spreadsheet to be open before the ODBC connection is established without adverse effect. But, that is not always the case with each attempt. It is advised to **not** open or have open the spreadsheet when the ODBC connection is started or active.

ODBC



Connection parameters

Refer to the connection parameters for ODBC.

Table name

This is the table name containing the recipe in the database. The database name is specified in the connection parameters.

Row is recipe

This property defines the ingredient orientation.

When this property is false, each row is an ingredient and each column is a recipe. Use LoadRecipe to write the recipe.

When this property is true, each row is a recipe and each column is an ingredient. Use LoadRecipe2 to write the recipe.

Ingredient count

This is the number of rows or columns in the table. Each row or column is an ingredient.

Active point (optional)

This is a point that will be set true when an operation for the recipe is active. Once the operation is complete the point will be set false.

These are the operations that will operate on the point at runtime.

Writing recipe values to the points

Writes are added to the queue for the 'port'. The active point is only active while the 'write' processing is performed to add the command to the queue. This will normally be a very short time. It does not indicate the data has been transferred to the external device.

Row points

Note: If only 4 rows/columns are needed, set this value for 4. This value can be increased later without loss of configuration data.

Each row/column has a point reference. At runtime when the LoadRecipe/ LoadRecipe2 function is called the program will write the value, of each row/column, from the column/row specified to the point.

If the point reference is blank or the value from the database table column/row reference is empty, a write will not be performed.

Depending on the destination of the point reference and the number of rows/columns, the amount of time to write the values is variable.

Monitor the 'Active point' above if completion of the writes is required.

If the external device needs to know when the writes are complete, set up the last ingredient as a flag. Clear the flag before the LoadRecipe/LoadRecipe2 function is called

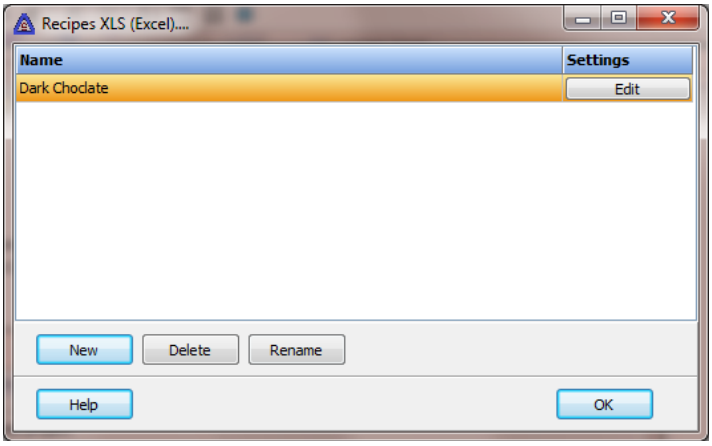
and use the last row to set the flag. The flag could be an integer checksum of all the rows for the column. The external device can then validate against the flag to verify all the ingredient values are correct for the recipe.

Note: Due to possible write errors or the order the destination device processes the write commands, the write order is not guaranteed.

Test button

Selecting the test button will attempt to connect to the database and read the table; a test window will be displayed. Use the test button to verify the connection parameters are correct and the table exists in the database.

XLS/XLSX (Excel)



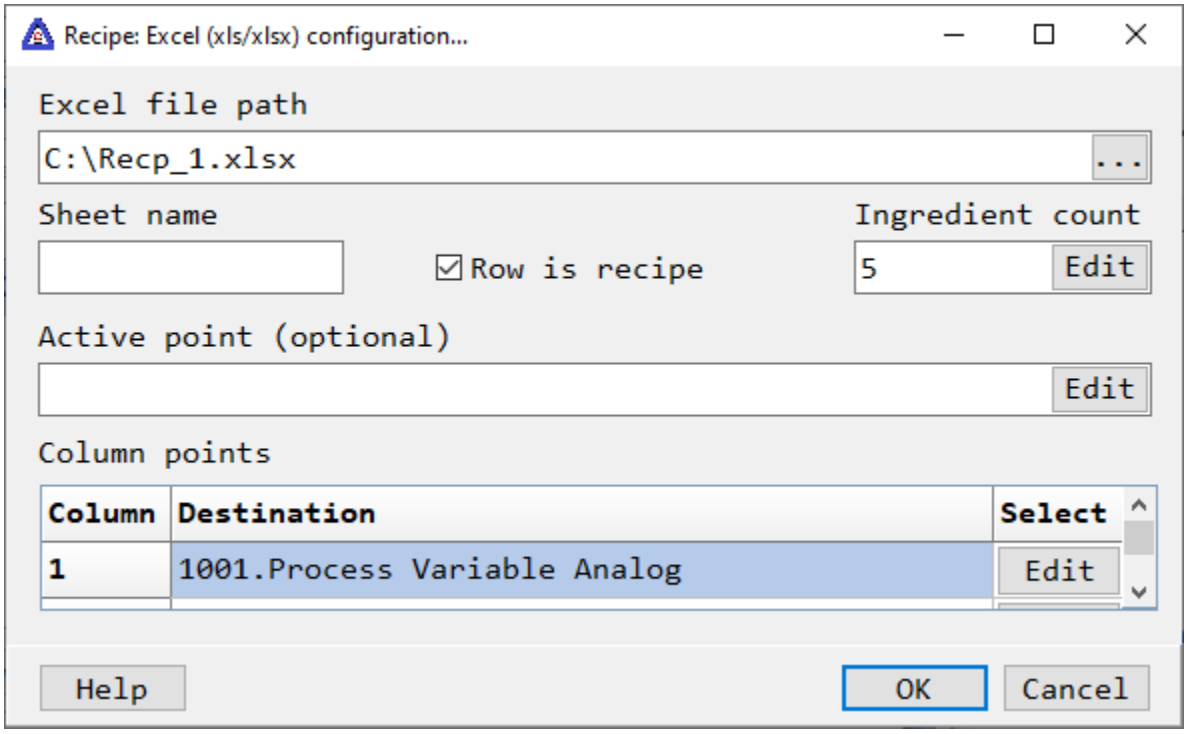
Each XLS/XLSX recipe is listed in the window.

To create a new recipe select the "New" button and enter a name. Each name must be unique across all recipe names.

To rename a recipe select the "Rename" button and supply a new name.

To delete a recipe select the "Delete" button.

Notes:
1) Excel does not need to be installed on the computer for the HMI to read the 'XLS/XLSX' sheet.
2) To save processing time, the sheets are loaded at program start and held in memory. To reload a sheet without stopping and restarting monitoring see the ReloadRecipeSheet.
3) It is **advised** to mark all cells in the spreadsheet as "**TEXT**".



All the ingredients are in the rows. Each column is a recipe.
Excel does not need to be installed on the computer for the program to access the file.

Excel file path

This is the path to the Excel spreadsheet.

Sheet name

This is the name of the sheet containing the recipes. A blank name will use the default sheet.

Row is recipe

This property defines the ingredient orientation.

When this property is false, each row is an ingredient and each column is a recipe. Use LoadRecipe to write the recipe.

When this property is true, each row is a recipe and each column is an ingredient. Use LoadRecipe2 to write the recipe.

Ingredient count

This is the number of rows in the spreadsheets. Each row is an ingredient.

Active point (optional)

This is a point that will be set true when an operation for the recipe is active. Once the operation is complete the point will be set false.

These are the operations that will operate on the point at runtime.

Writing recipe values to the points.

Writes are added to the queue for the 'port'. The active point is only active while the 'write' processing is performed to add the command to the queue. This will normally be a very short time. It does not indicate the data has been transferred to the external device.

Row points

Note: If only 4 rows are needed, set this value for 4 rows. This value can be increased later without loss of configuration data.

Each row has a point reference. At runtime when the LoadRecipe/LoadRecipe2 function is called the program will write the value of each row from the column specified to the point.

If the point reference is blank or the value from the Excel spreadsheet column/row reference is empty a write will not be performed.

Depending on the destination of the point reference and the number of rows, the amount of time to write the values is variable.

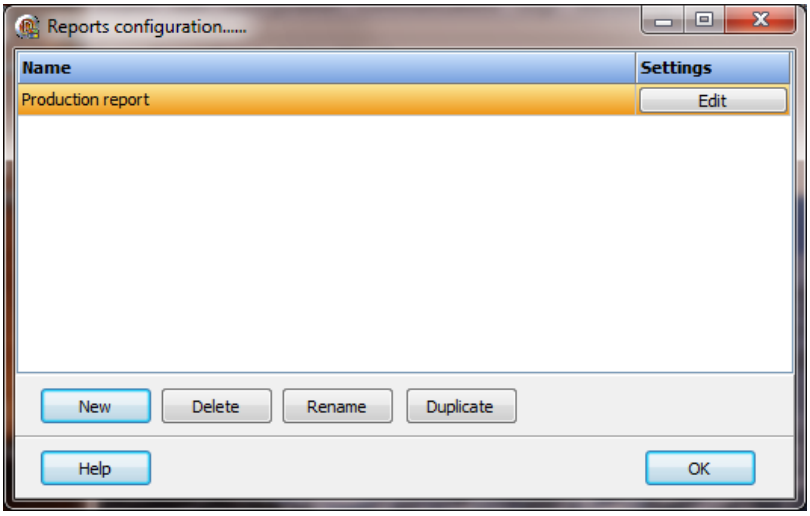
Monitor the 'Active point' above if completion of the writes is required.

If the external device needs to know when the writes are complete, set up the last ingredient as a flag. Clear the flag before the LoadRecipe/LoadRecipe2 function is called and use the last row to set the flag. The flag could be an integer checksum of all the rows for the column. The external device can then validate against the flag to verify all the ingredient values are correct for the recipe.

Note: Due to possible write errors or the order the destination device processes the write commands, the write order is not guaranteed.

REPORTS

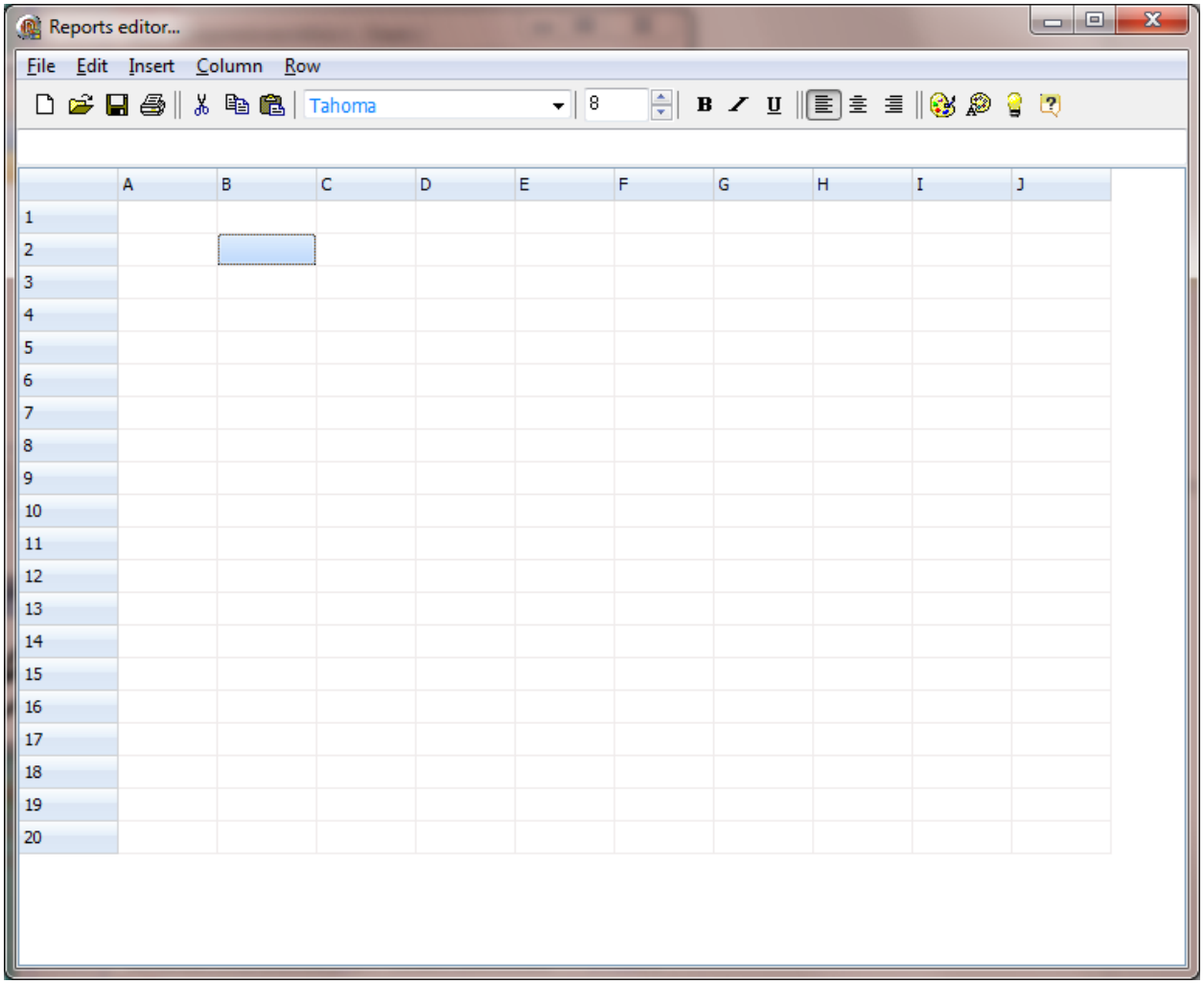
The HMI supports an unlimited number of reports.



Each report is listed in the window.

Each report is comprised of columns and rows. The cells can contain plain text, the value of a point, the value of a script global, the date/time or a picture. The text font, color, attributes, etc. can be configured for each cell.

At runtime a report is executed via a mouse command or script command.



File menu

Import

The import menu is used to import a report or CSV (command separated values) file.

Save

Save the report. Each report is saved in a separate file.

Preview

This will print the report applying the settings and configuration values that are possible at design time.

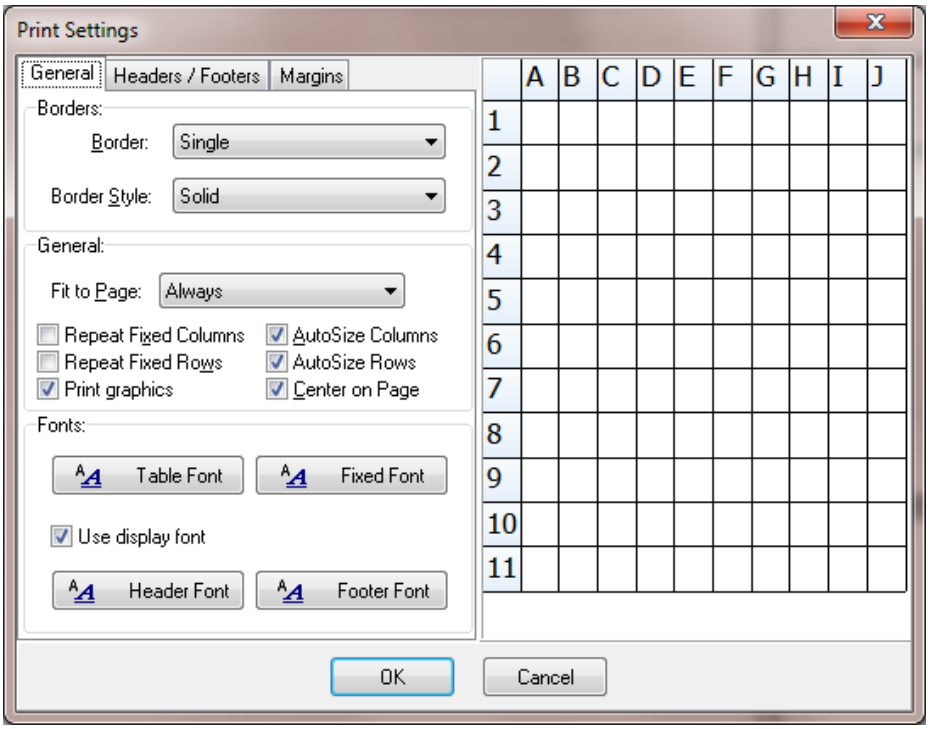
Test report

This will export the report command as if at runtime. All settings will be applied to creating the report(s).

EDIT MENU

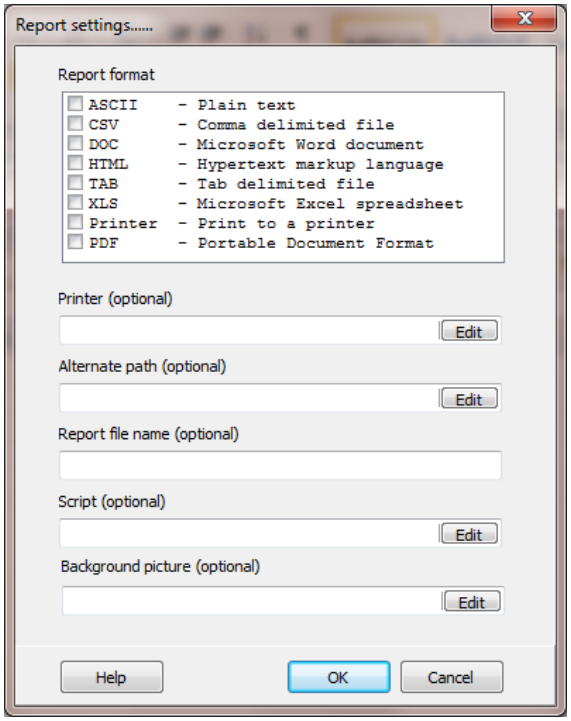
PRINT SETTINGS

Various print settings.



Settings

Settings used to determine the output type when the report is generated at runtime.
Note: XLS, PDF and printing produce good results. All other formats can produce good results within the limits of the file format.



Each report can be created in several formats.

Format	**Description**
ASCII	Plain text
CSV	Comma delimited file
DOC	Microsoft Word document
HTML	Hypertext markup language
TAB	Tab delimited file
XLS	Microsoft Excel spreadsheet
Printer	Print to the selected printer
PDF	Portable Document Format

DOC and XLS format do not require Excel or Word to be installed to create the report.

Printer name (optional)

This is the name of the printer to print the report. If this field is blank and the 'Printer' format is selected the default printer will be used.

Alternate path (optional)

This is the path to save the report. If this field is empty, the settings in the project configuration settings will be used (Log File Settings).

Report file name (optional)

This is an alternate file name for the report. If this field is empty, the name with the date and time of the report will be used. The date and time format configured for the computer will be used.
Example: Gas Flow Report 07~15~2011 11~08~07 AM

Note: Do not include a file extension. The correct extension will be added to the file name.

Script (optional)

This is a script to execute when the report is generated. The script is executed as the first step in generating the report.
There are two script commands specifically for report generation. ReportSetCell and ReportSetCellColor.

Background picture (optional)

This is the path to a file that will be used as the background for the report. The picture is not exported. It was added to allow easier alignment of the cells to a report format.

Script

This is the script to run before the report is generated. Two script functions are provided to change the cell value or color, ReportSetCell and ReportSetCellColor.

Merge/Unmerge

Use these commands to merge and unmerge cells. Use the "SHIFT" key and "ARROW" keys to select cells to merge. Merged cells cannot be merged with other cells. Unmerge the merged cell, select all the cells to be merged and merge.

Notes:
1) Copying of merged cells is disabled.
2) Merging is not supported by all file formats.

Insert menu

If the first character of the cell is an '=' (equal sign) the cells is considered to be a calculation. When the report is rendered the cells contents will determine the value in the report for the cells location.
Note: Press the mouse right button on a cell to display the insert popup menu.

Cell commands

Prefix	Operation
PT	Collects the value of a point.item
SG	Collects the value of a script global
DT	Date time display
GF	Display a picture from a file

Date/Time

Date/Time

The date and/or time will be placed in the cell. This command uses the same format as the graphic engine. No format specified will use the system default. **Note:** This is a 'text' field type.

=DT	// as defined by the OS
=DT(h:n:s d/m/yy)	// 13:6:36 24/5/11
=DT(mm/dd/yyyy hh:mm:ss)	// 05/24/2011 13:06:36
=DT(mm-dd-yyyy hh:mm:ss)	// 05-24-2011 13:06:36
=DT(hh:mm:ss)	// 13:06:36

Global string

Script globals

The value of the script global will be placed in the cell.
(Section.item)
=SG(Logged on.User)

The value of the script global will be placed in the cell and limited to the first ten characters. Default is all characters and does not need to be present.
=SG(Logged on.User~L10)

Picture

Picture file

The picture is aligned to the top and left of the source cell. The picture will be stretched/shrunk to fit the cell. Merge cells for the picture to be viewed across many cells. In '=GF', change the F to N and the picture will not be stretched /shrunk to fit the cell.
=GF(C:\test.bmp) //file name (complete path)
=GN(C:\test.bmp) //file name (complete path), no alter

Notes:
1) Exporting of pictures is only supported for XLS, DOC, HTML, PDF and printing.
2) XLS, PDF and printing produce good results. All other formats can produce good results within the limits of the file format.

Point value

Points

The value of the point.item will be placed in the cell.
=PT(Tank.5000)

The value of the point.item will be placed in the cell with 3 decimal places. The default is 2 and does not need to be present.
=PT(Tank.5000~D3)

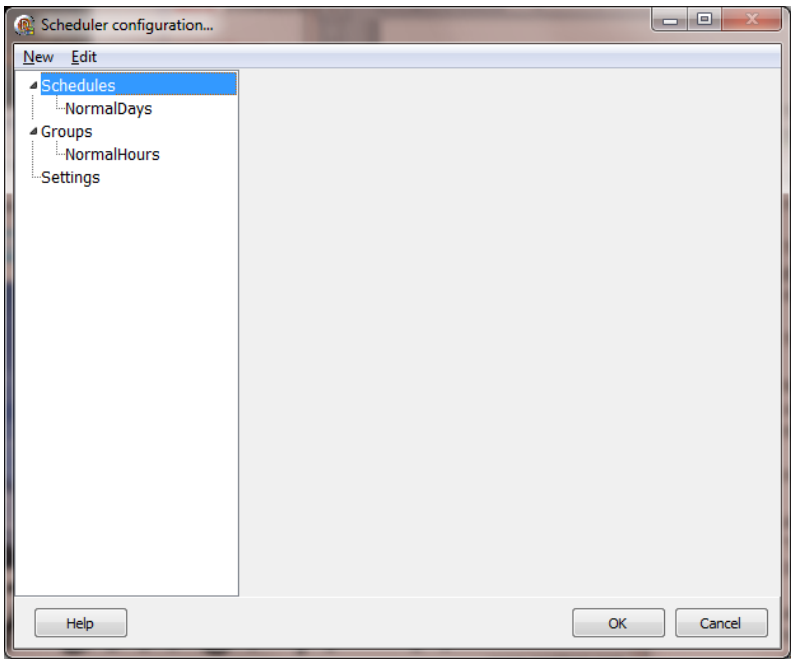
The value of the point.item will be placed in the cell using the supplied strings. The default is True/False and does not need to be present.
=PT(Tank.5007~TOpen~FClosed)

Note:
The commands listed are automatically formatted when using the report builder. The cell contents can be imported/exported and on import the format must be followed or the runtime report generator and the report builder will not function correctly.

Columns/Rows

These menus are used to add, insert, delete and change the column width/row height.

Schedules



Overview

The scheduler consists of "schedules" and "groups".

Schedules define a start and optional end day/time.

Hourly

A selected minute and second in an hour of the day.

Daily

Every day at the selected time.

Day

On the selected day of the week (Sunday - Saturday) at the selected time.

Date

The selected date(s) (October 12) at the selected time.

Monthly

The selected day(s) of the month at the selected time.

The configured schedule creates event "triggers". One trigger for the start time and one for the optional end time.

The start time is a trigger to operate on a group. The start trigger must have at least one group selected.

The optional end time is also a trigger to operate on a group. If a group(s) is not configured for the end trigger, the end trigger will operate on the group(s) configured for the start trigger.

The time is entered in 24 hour notation.

Multiple schedules can concurrently be enabled. Each enabled schedule will execute in the order configured. The order is listed in the tree view at configuration.

For example:

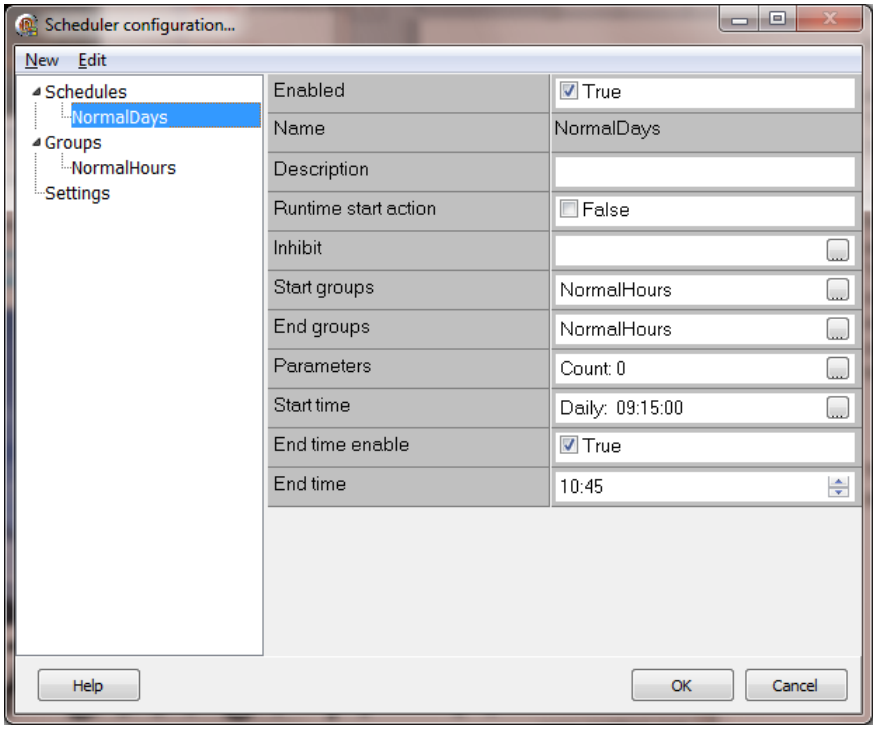
A schedule is created and named "All days" and configured for "Daily" operation. At 7:00:00 the lights are commanded on and at 19:00:00 the lights are commanded off.

Another schedule is created and named "Holidays" and configured for "Date" operation. All the holidays are selected for the year. The start and end times are the same as the "All days" schedule. In the "Holidays" schedule all the lights are commanded off. If non combining is not enabled the lights will be command off (if they are on). In this example, the "Holidays" schedule is only used to command the lights off, to override the "All days" schedule start time, and the "end time" in the "Holidays" schedule is not required.

NEW MENU

The new menu is used to create new schedules, group and group items.

SCHEDULE



Enabled

The schedule must be enabled to be active.

Name

The name of the schedule. The name must be unique across all schedules.

Description

User defined string for the schedule.

Runtime start action enabled

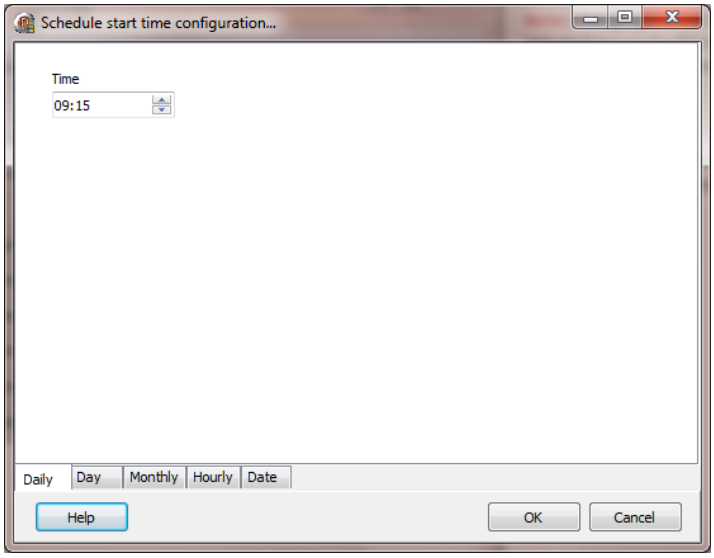
When runtime monitoring is started;
If the current time is after the start time and before the end time (if enabled) the start trigger event will execute.

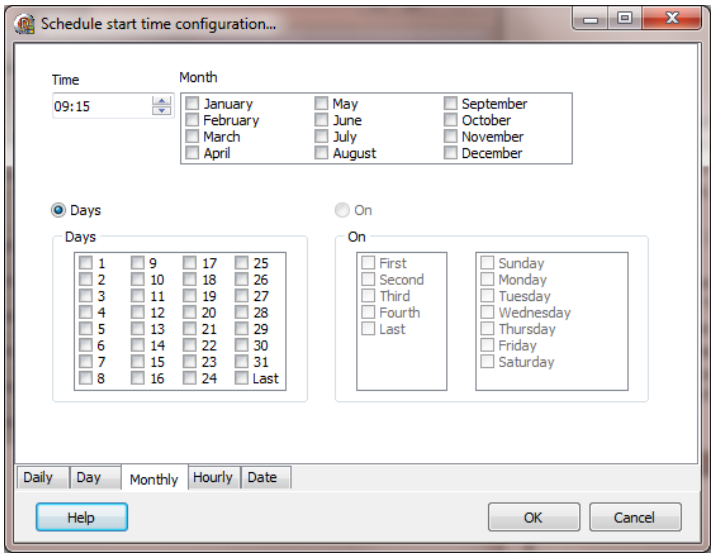
Inhibit (optional)

If a point is selected and true the schedule will not execute, the triggers will not execute, until the point is false. If the start or end time has passed while the inhibit point is false, when the point becomes true, the elapsed triggers will not execute.

Start time and end time (optional)

The start and optional end time for the schedule.
If the start time type is "hourly" and the optional end time (minute) is enabled, the end time (minute) must be greater than the start time. If required to have an end time less than the start time, use two schedules with only a start time in each schedule. "A" schedule for the start time and "B" schedule as the end time.

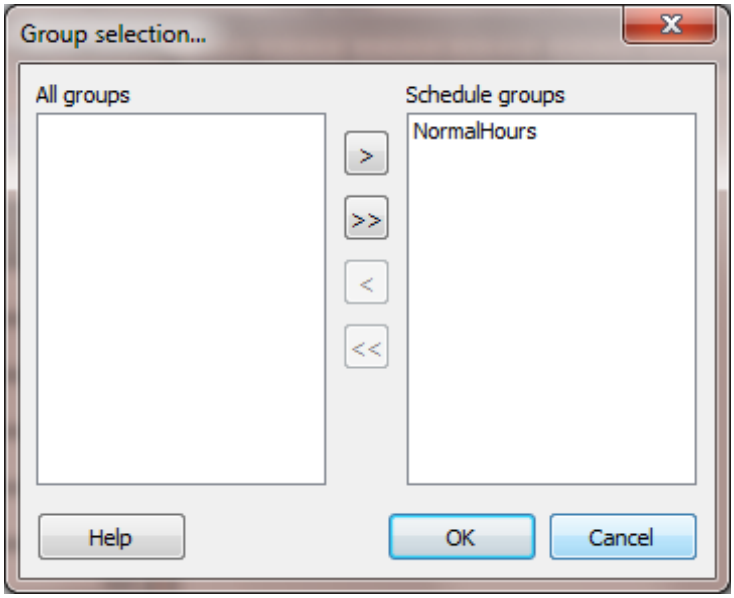




Groups (Start/End)

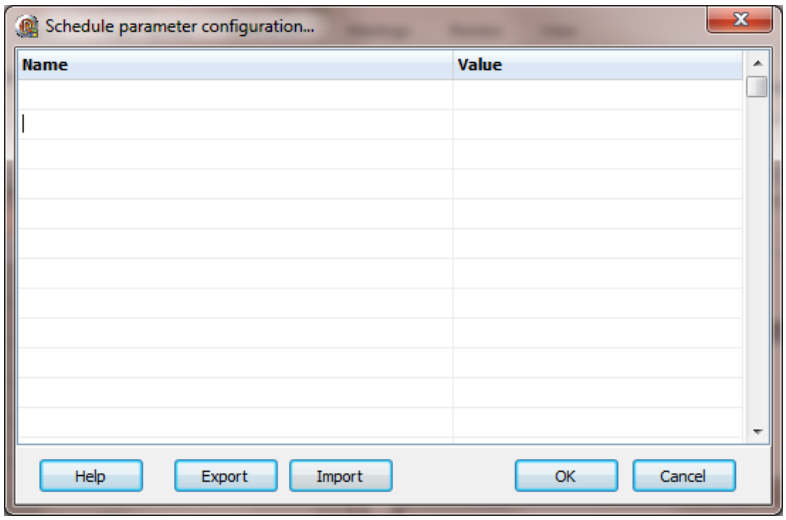
This is the list of groups that will be acted upon when this schedule is active and a trigger event occurs. The order in the schedule is used for runtime execution, not the order in the "tree view" of the main configuration window.

Note: Changing the text in the edit field does not change the selected groups for the trigger. Use the button in the text edit field to configure the groups for the trigger.



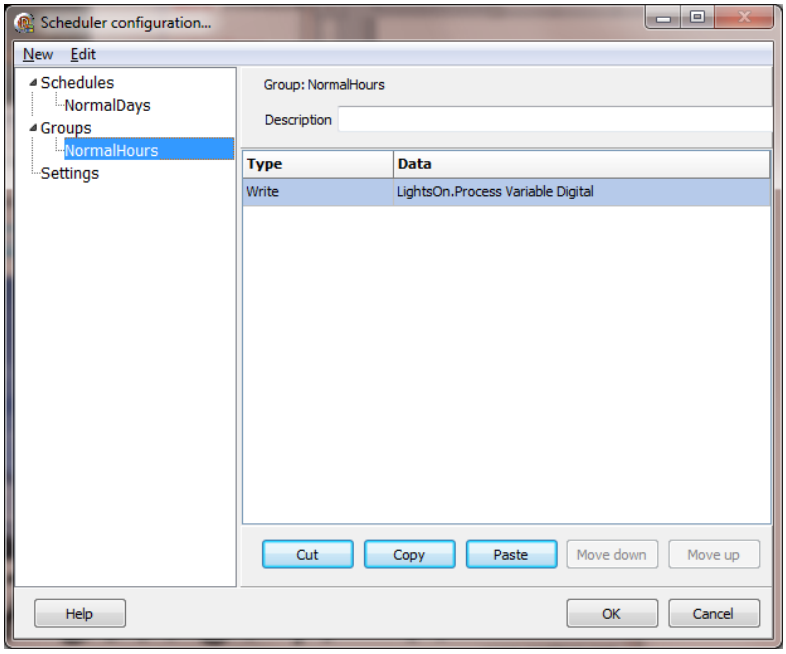
Parameters

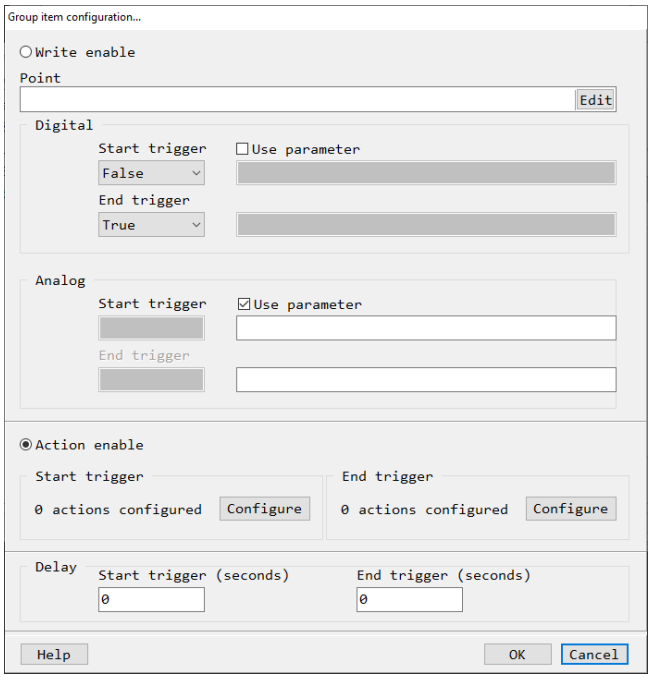
This is a list of named-value pairs. The name must be unique for the schedule. These parameters can be used in groups for an action value.



Group

Groups are collections of points and actions that execute commands.





Name

The name of the group. The name must be unique across all groups.

Description

User defined string for the group.

Write

All points must have an "Access Rights" of read/write.

Points have a start and end parameter.

Digital points, the value can be:

False	If the current value of the point is true, a write command will be issued to set the value false.
True	If the current value of the point is false, a write command will be issued to set the value true.
No action	The point will not be modified when the trigger event is executed.

Parameter

The value of a parameter in the schedule that called the group to perform the actions for the trigger is used. If the current value does not match the parameter a write command will be issued. Double click the left mouse button in the parameter field to open a selection dialog.

Analog points, the value can be any value the point data type allows or a parameter value of the schedule. If the current value does not match the parameter a write command will be issued. If the point quality is bad, a write command will not be issued.

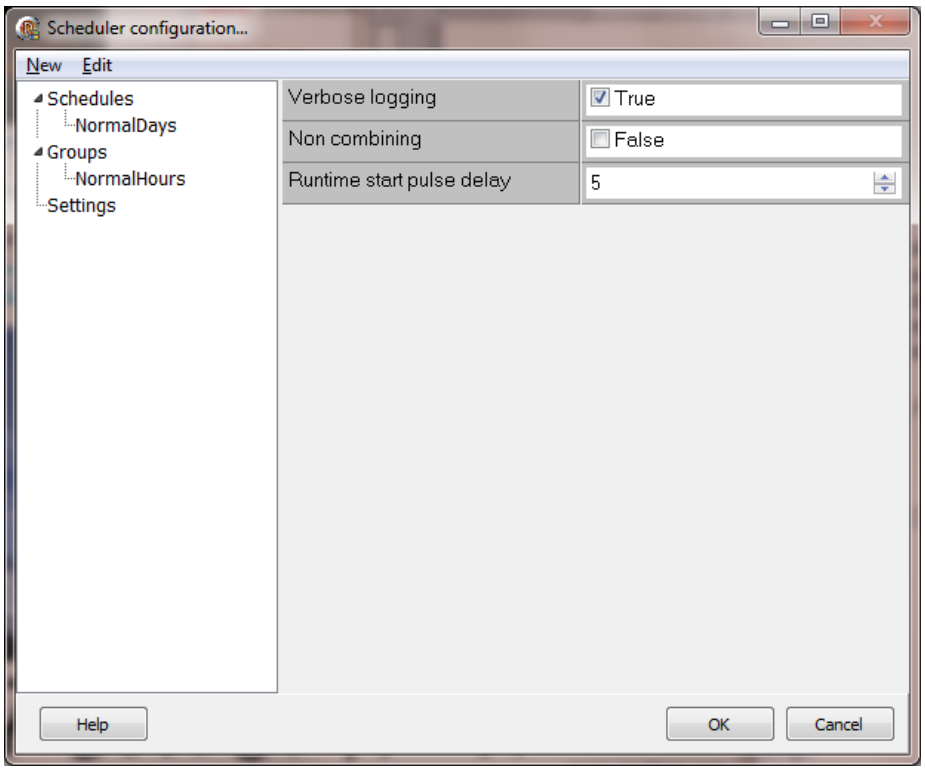
Action

See the mouse commands for a list of possible actions.

Delay

A start trigger and/or end trigger. After the command is executed the next action can be delayed. The delay is in seconds and can range from 0 - 300 seconds. If the value is 0 the next action/write in the list will be executed without delay.

SETTINGS



Verbose logging

When enabled logging to the event file will be much greater than normal operation.

Non combining

When a point.item is referenced in more than one enabled schedule and both schedules execute at the same time, the point.item will not be combined for one command.

Example

Group 1

LampSet_1.Process variable digital start trigger = true, end trigger = false
LampSet_2.Process variable digital start trigger = true, end trigger = false
LampSet_3.Process variable digital start trigger = true, end trigger = false
Execute script XYZ

Group 2

LampSet_1.Process variable digital start trigger = false
LampSet_2.Process variable digital start trigger = true
LampSet_3.Process variable digital start trigger = false

Schedule - Everyday (Sunday - Saturday)

This schedule sets all the lamps on at 7:00 and off at 17:00.
Start time 7:00, end time 17:00
Group 1
This schedule sets all the lamps on at 7:00 and off at 17:00.

Schedule - Stored closed (selected days) (January, 1 July 4 and November 4)
Start time 7:00, no end time
Group 2
This schedule sets LampSet_1 and LampSet_3 off and LampSet_2 one at 0:00.

Both schedules are enabled.

When the start trigger for schedule "Everyday" is fired, a command is placed in a queue to turn on the three lamps.
When the start trigger for schedule "Stored closed" is fired, it looks in the queue for any commands to each of the three items in the group and replaces the command with the command from this group.

After all the schedules execute the commands queue is executed.

For this example the queue would be:

LampSet_1.Process variable digital start trigger = false
LampSet_2.Process variable digital start trigger = true
LampSet_3.Process variable digital start trigger = false
Execute script XYZ

<u>If the attribute "Non combining" was enabled the queue would be:</u>

LampSet_1.Process variable digital start trigger = true
LampSet_2.Process variable digital start trigger = true
LampSet_3.Process variable digital start trigger = true
Execute script XYZ
LampSet_1.Process variable digital start trigger = false
LampSet_2.Process variable digital start trigger = true

LampSet_3.Process variable digital start trigger = false

Runtime start pulse delay

When runtime starts the program begins issuing commands to read data from the external devices. Depending on the protocol, communication delays, polling rate, etc., it can take several seconds to several minutes for all the external data to be collected and processed at least once. This attribute inhibits the scheduler logic from executing the 'runtime start pulse' until the time delay has elapsed. If the value is 0 the 'runtime start pulse' is not delayed. The range is 0 - 3600 seconds.

Settings

Program Start

On program start automatically log on

When the program starts a user can be configured for automatic “log on”. If no user is to be automatically “logged on”, this field must be blank. If the name in the field is not a valid user name, the field content will be ignored.

This attribute is also applied when a project is opened.

Launch runtime on program start

When the main program is started, start runtime monitoring. The runtime monitoring logic is a separate program and this will launch the program.

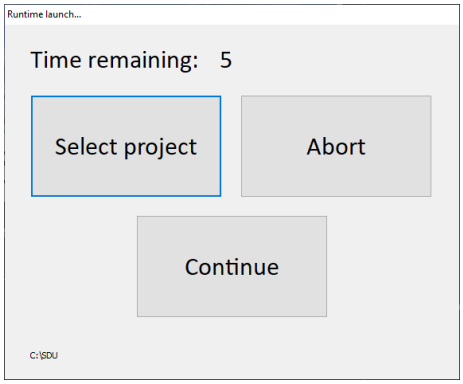
To make the program start when "Windows" starts, make a shortcut of the "Configure.exe" and place it in the "Startup" directory of Windows.

To start runtime monitoring at program startup, enable this option.

Note: If this option is enabled and need to prevent runtime monitoring from beginning, ***immediately*** after the main program is launched, press and hold the "CTRL" key.

Runtime license

If the detected license type is “Runtime”, the runtime launch panel will be displayed.

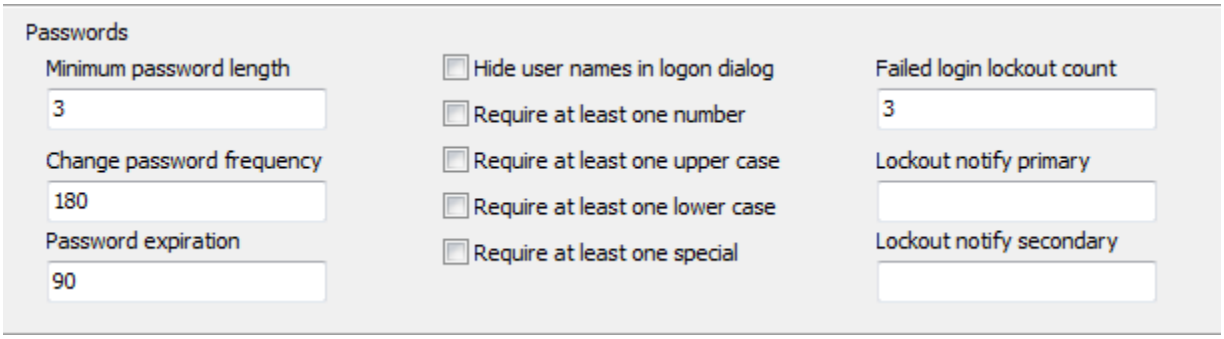


Hide launch panel	The runtime launch panel will not be displayed and the runtime program will immediately launch.
Disable abort button	If enabled, the “Abort” button will not be enabled.
Disable project select button	If enabled, the “Project select” button will not be enabled.
Wait time	The number of seconds to display the launch panel before launching the runtime program.

User level to quit runtime

After runtime monitoring has begun, the logged on user must have a user level equal to or greater than this value to terminate runtime monitoring.

PASSWORDS



Minimum password length

The minimum length of the password.
0 = no password required.

Change password frequency

This is the number of days that can elapse before the user must change the password.
0 = the password does not require changing.
The password "change" is checked each time the user logs in. When the user is flagged to change the password, the user must change the password. The password dialog will appear. If the user fails to change the password, the user will be locked out.

Password expiration

The number of days before the password expires.
This is used to automatically lockout an inactive account.
0 = never expire.

Hide user names in logon dialog

By default, the user names appear in the logon dialog so the user can use the drop down list and select the desired user name. If this attribute is enabled, the user names will not be listed and the user will be required to type in the desired user name. User names are case sensitive.

Require at least one number

If enabled, each password must include at least one number character.
0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Require at least one upper case

If enabled, each password must include at least one upper case character.
A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z

Require at least one lower case

If enabled, each password must include at least one lower case character.
a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z

Require at least one special character

If enabled, each password must include at least one special character.
! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [\ ] ^ _ ` { | } ~

Failed login lockout count

If a user attempting to "log in", enters an incorrect password <count> times the user is locked out from logging in. The "lock out" must be cleared by a logged in user that has access rights.
0 = never lockout a user on password entry failure.

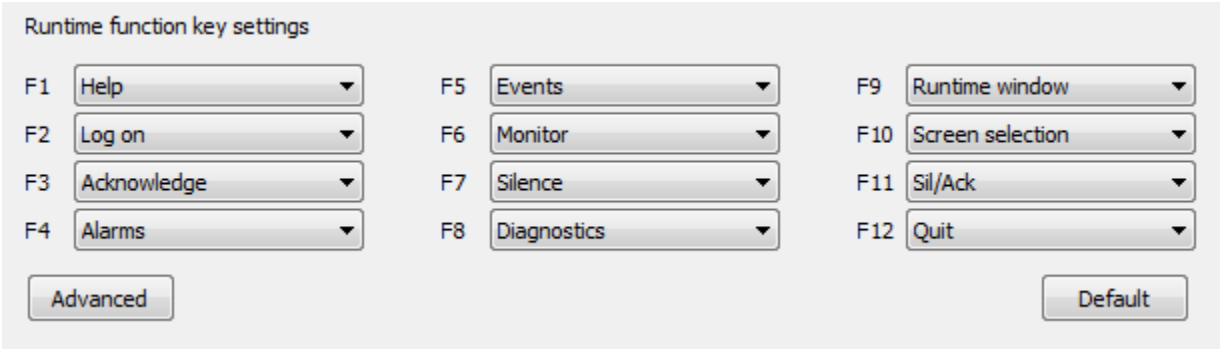
Notes:
1. If only one user is configured, the user cannot be locked out.
2. If only one user is configured with configuration rights, that user cannot be locked out.
3. Changes made via the configuration program while runtime monitoring is active, do alter runtime operations and may be overwritten via changes in the runtime environment.

Lockout notify primary/secondary

If a notification user name is supplied and a user is locked out via a password entry failure, an email will be sent to the name entered. The email settings for email notifications must be valid. Note: If the “Lockout notify primary” field is blank, a notification will not be sent.

Note:
The password requirements are applied when the password is set. For example, if the “require at least one special” is disabled when the password is set and later the requirement is enabled, existing passwords are not flagged as nonconforming until the password is set.

RUNTIME FUNCTION KEY SETTINGS



This provides for the remapping of function keys to actions.

The defaults assignments are:

F1 = Help
F2 = Log on
F3 = Acknowledge
F4 = Alarms
F5 = Events
F6 = Monitor
F7 = Silence
F8 = Diagnostics
F9 = Hide/show the runtime window
F10 = Screen selection
F11 = Silence/Acknowledge
F12 = Quit

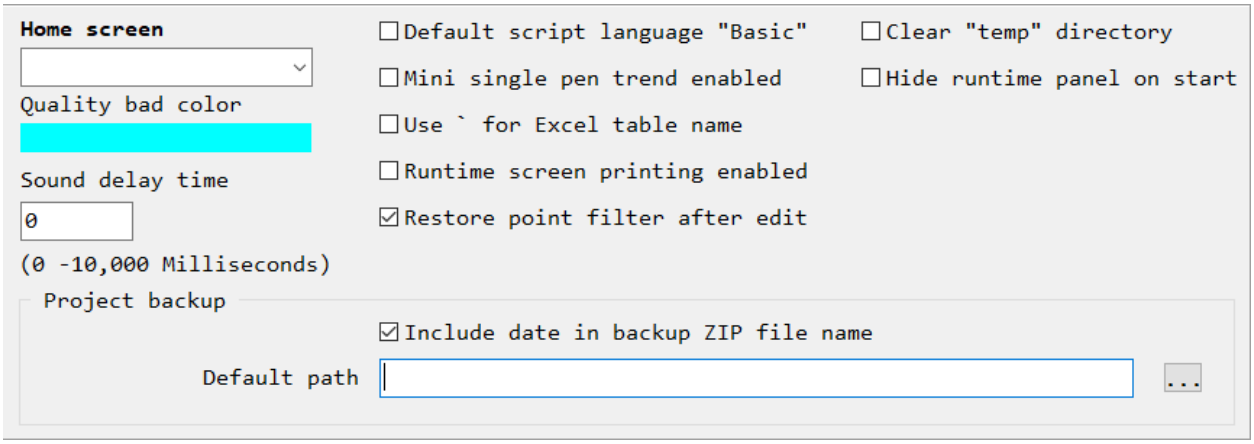
When a script editor window is open these function keys are disabled.

Advanced

The advanced settings provide a method to assign a user level to each function key and an optional script to execute after the function key command is executed.

Note: Function keys mapped to “N/A” will execute based on the user level and if a script is assigned.

MISCELLANEOUS



Home screen

If this field is not blank, when runtime monitoring begins, the program will attempt to open the screen for operation.

Quality bad color

Graphic elements containing animations are connected to data points. Data points have either a good quality or bad quality. Good quality indicates the connection to the data source is established and active. Bad quality indicates that the connection to the data source has been interrupted or the underlying data point has become unavailable, corrupted, etc..

When the quality is bad the graphic element will be rendered in the color selected. When the quality is good the graphic element will be rendered as it is configured.

Sound delay time

When runtime monitoring is active and a sound is commanded to play, it is placed in a queue. The sounds are played one after another until the end of the queue and then the first sound in the queue is played. As long as the queue contains sounds, the program continues to play the sounds in a "round robin" method. This value instructs the "sound player" to pause "X" milliseconds between playing each sound.

Example:
Setting is 2000. (2 seconds)
Three sounds are in the queue.

Sound 1 is played
Wait 2 seconds
Sound 2 is played
Wait 2 seconds
Sound 3 is played
Wait 2 seconds

Return to top and begin

Default script language “Basic”

The HMI supports “Pascal” and “Basic” scripting and both can be used, as needed, in the same project. The default script language is “Pascal”. When enabled the default script language will be “Basic”. In the script editor window the language can be changed to the other via a menu.
Warning: If the script is file based and the language is changed and the file is saved to the new language file, the existing file (original language) is not altered which could lead to unintended operations. Pascal files, “.psc”, Basic files, “.bsc”, file extension.

Mini single pen trend enable

The "Mini single pen trend" graphic elements display the previous 100 seconds of the point value. If this attribute is enabled, all the points will store the required values. If the project does not contain any "Mini single pen trend" graphic elements, this attribute should **not** be enabled.

Use ` for Excel table name

Depending on the version of Excel and how the OS language settings are configured, accessing Excel spreadsheets via ODBC can require two different table name qualifiers.

Runtime screen printing enabled

When runtime is active and the user presses the "Print Screen" key, a capture of the screen will be created and sent to the configured default printer. On systems with more than one monitor each monitor will be rendered on a separate page. A script command to print the screen is also provided.

Note: If the 'ALT' key is pressed when the "Print Screen" key is pressed or the script command is executed, only the active window will be captured and printed.

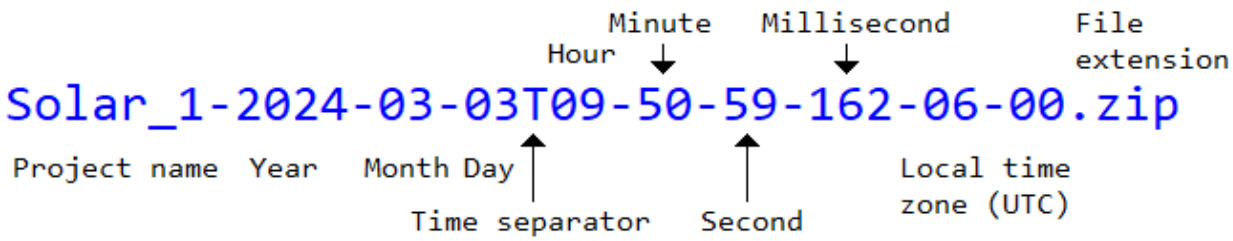
Restore point filter after edit

The default action, after editing a point, when a filter had been applied to the points configuration list, was to fully populate the points list. If this property is enabled, when a filter had been applied and the point editor window was opened/closed, the filter will be re-applied.

Include date in backup ZIP file name

If enabled the time and date will be added to the suggested file name (project name and time/date). The time/date format is ISO8601, with the separator characters (: and .) replaced with a dash (-) to create a valid MS Windows file name.

Example:



Default path (project backup)

The path displayed when the "Backup" menu item is selected.

Clean "temp directory"

The OS "temp" (C:\Users\Admin\AppData\Local\Temp) directory can consume a significant portion of a drive. If this property is enabled the HMI will delete files/directories in the "temp" directory at runtime start and once a day, a short time after midnight. Several logical actions can be configured with an "ini" file in the project directory.

The "ini" file name must be : **TempDirClean.ini**
Three sections of the "ini" file are provided. All sections are optional.

```
[Common]
DaysOld=7
LoggingEnabled=0
```

DaysOld	The file "last modified" date must be larger than current date – DaysOld. If not specified the default value is seven (7) days.
Logging enabled	If enabled, the HMI will log, to the "Event log", most actions of the logic. This is normally not needed. Default is no logging. To enable, LoggingEnabled=1 To disable, LoggingEnabled=0

This provides a method to prevent a file from being deleted by the filename extension.

```
[Extensions]
fe0=.asf
fe1=.txt
```

This provides a method to prevent a file from being deleted based on some text/file name. **Note:** Any text may be used to prevent a file from being deleted. In the example files below, both could be replaced with "fn0=Music". The logic examines the full path and file name for the text "Music". If found, the file is not deleted.

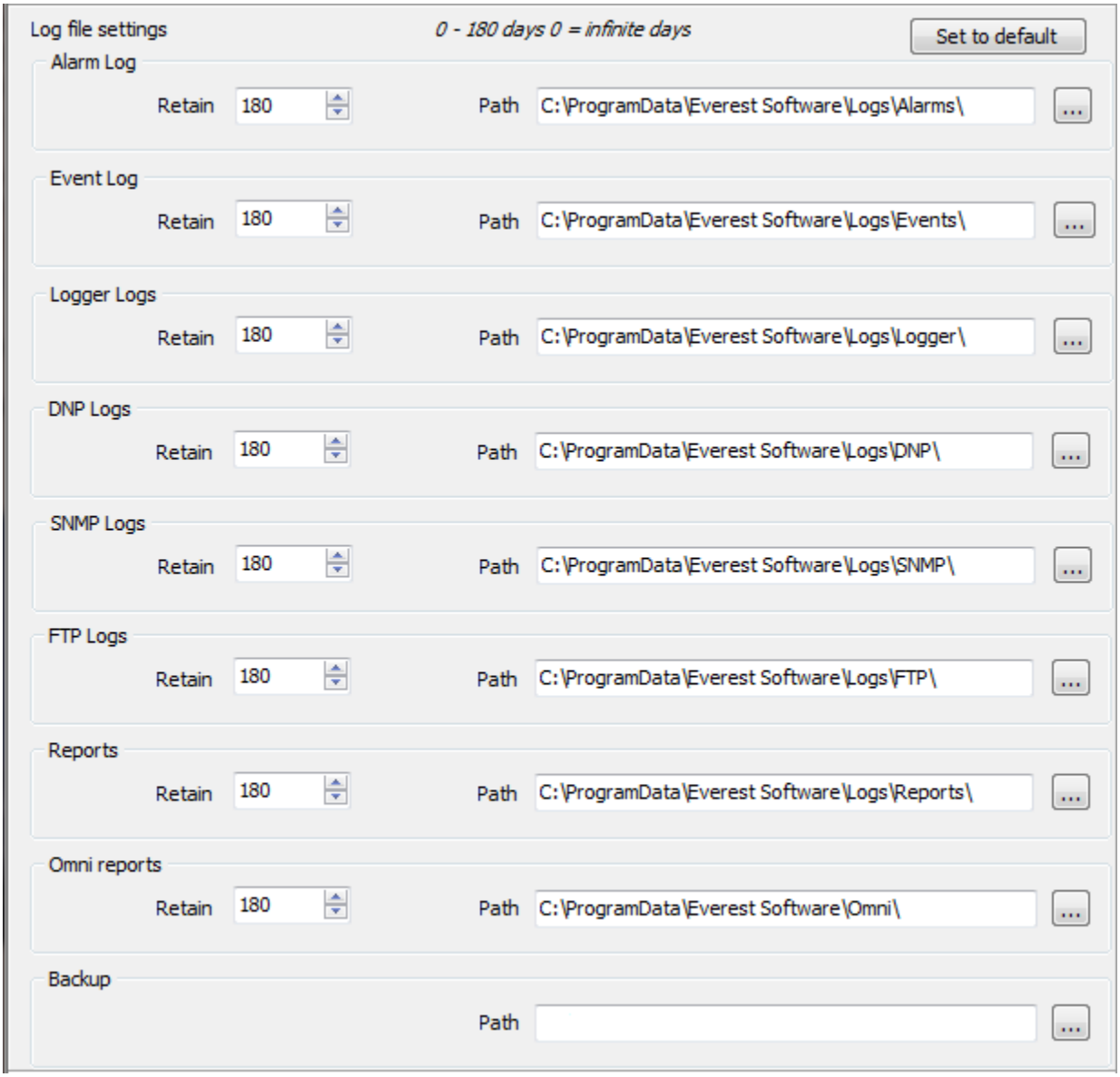
```
[FileNames]
```

```
fn0=HerMusic.bmp
fn1=HisMusic.bmp
```

Hide runtime panel on start

If enabled, the Runtime Panel will be hidden when “Runtime Monitoring” is started. See “Runtime function keys” and/or “SendKeys” script function.

Log File Settings



All log files are "daily" files. At midnight new files are created for each type (alarm, event, logger, etc.) and the previous days files are closed. When runtime monitoring begins logic is executed to determine if a file is present for the current day. If a file is found, the program appends any new data to the file. If a file is not found, a new file is created.

Omni files are stored in the Omni path. The file name will be the port name, the date, the time and the file extension will be the report type. All reports are in text (single byte ASCII) format. <Omni path>\<port name>~<day>~<time><ext>
Example: Meter1~10-29-2011~14-29-34.snr

Retain

This is the number of days to retain the log files. If the setting is 0 (zero) old log files are not deleted. Use caution, the destination drive might become full and cause errors. And the length of time for runtime to start might become extended. If long duration log file retention is required, this setting in conjunction with the "backup" feature below. might be a good solution to lower this setting.

Example:
Retain = 30
Each day when the new log file is created any files of the same type that are older than 30 days are deleted.
For Omni, this is directory based.

Note: If this value is set to 0 (zero) and do not provide an external method to manage the files retention, the "DeleteFilesAge" script command might be an option.

Path

This is the path to store the log files for each type. Each type must have a unique end destination.

Set to default

This sets the path for all the log files to the default destination.

Backup

At runtime, if a path is set, at approximately 03:00 a check is performed to verity a copy of all log files are in the backup path. Any file in the log file path, not in the backup path will be copied to the backup path. The full path will be the backup path configured and the last directory name specified in the path setting for each log file type.
Example:
Alarm log path: C:\ProgramData\HMI Software\Logs\Alarms\
Backup path: C:\logback\
Final path: C:\logback\Alarms\

Note:
1) The runtime program will attempt to create the final path. If the final path is a network drive and the path does not exist, the attempt to create the final path may fail and the path will need to be created outside the program for this feature to function properly.
2) Shortly after runtime is launched and a backup path is configured, the program will attempt to verify all the logs files have been copied to the backup path.
3) The backup operation copies all files in the log paths.

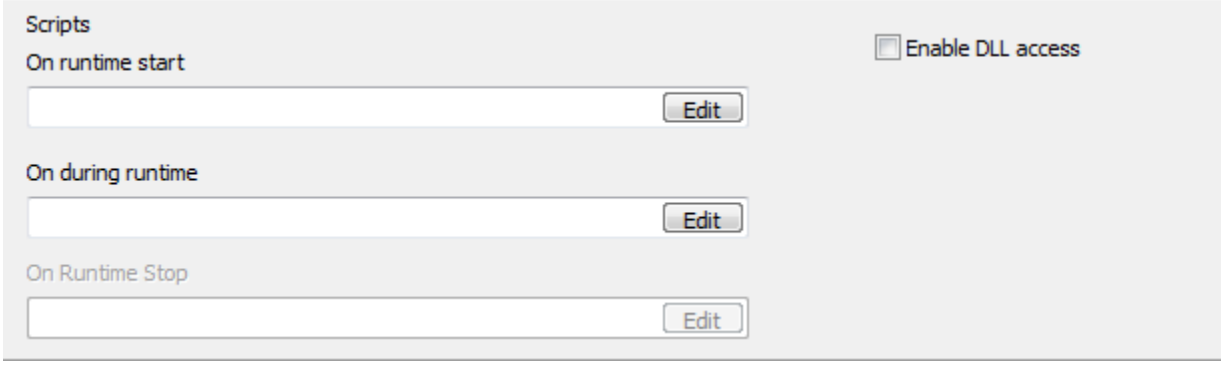
Disable path security setting

On occasion, years ago, an issue appeared where virus protection programs and/or the OS would alter the security level on a file, preventing the HMI programs from accessing a file, causing the program to fail. To solve this issue, when configure.exe starts it sets the security level so the program can access the file and runtime.exe does the same.

When the file count in the “Logs” directory becomes very large it can cause the HMI, both programs, to take a longer than expected time to start.

Enable this property and the security set function will not be performed. If it is later discovered the security level is altered and the HMI fails, enable this property. Contact support if assistance is needed.

Scripts



On runtime start

If a script is selected, the script will execute once when runtime monitoring starts. It executes after all initialization steps and before data request are issued to external devices, before data logging begins and before the home window, if assigned, is opened.

On during runtime

If a script is selected, the script will execute while runtime monitoring is active, about once per second. It will be placed in the queue. For more information see scripting.

On runtime stop (not supported)

The selected, if one is selected, script will execute when the command is executed to end runtime monitoring. This setting is disabled.

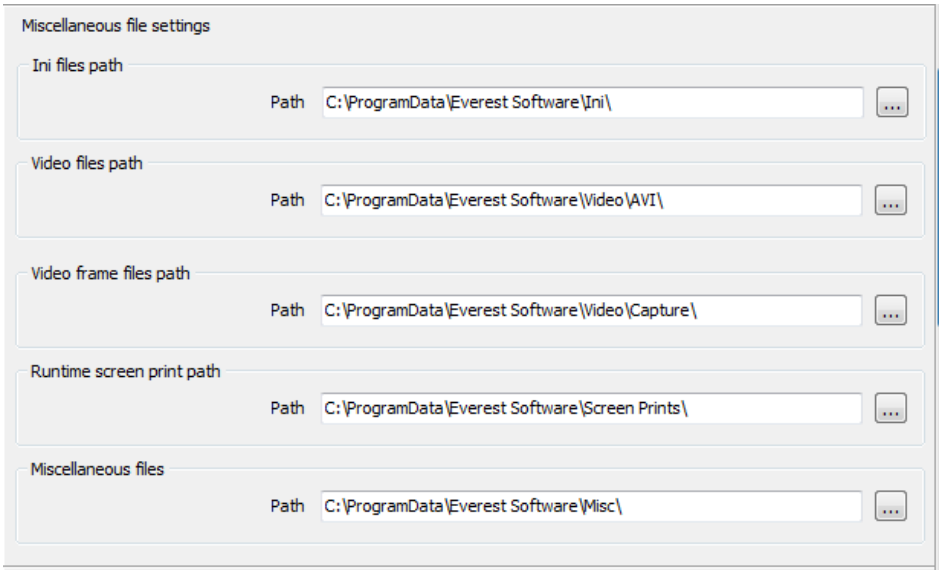
Enable DLL access

If enabled, the various scripting engines can call functions located in external DLLs. The DLL file must be in the same directory as the HMI program or in the “Windows” DLL search path.

Warning: Access to any external DLL may cause the HMI program to fail or degrade program operation.

See DLL access in the scripting section for more information.

Miscellaneous File Settings



Ini files path

Several "ini" files are used to store user preferences, window settings, etc.. This is the path to those files.

Video files path

This is the path to recorded video files. The file name format is <Configured camera name> ^ <date string> ~ <time string> Example: Camera1^02-27-2007~15-45-34. Day/month/year order per system setting. The time uses the 24 hour format.

Video frame files path

This is the path to captured video frames files. The file name format is <Configured camera name> ^ <date string> ~ <time string> Example: Camera1^02-27-2007~15-45-34. Day/month/year order per system setting. The time uses the 24 hour format.

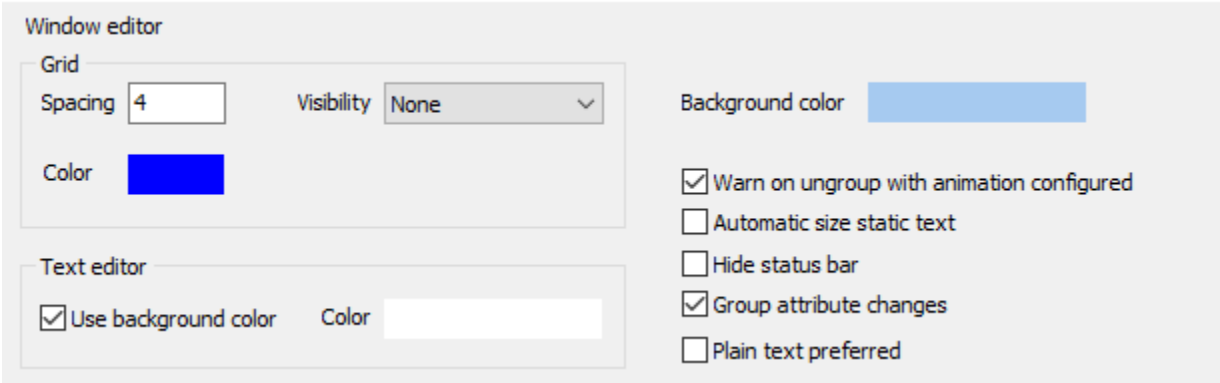
Runtime screen print path

This is the path to captured screen prints files. This directory is normally empty. When a screen print command is initiated the captured screen image(s) is saved in a file using this path. The file is then sent to the default configured printer and the file is deleted. At runtime start all the files in this directory are deleted. Do not store files in this directory. Sub directories and files in sub directories are not deleted.

Miscellaneous files path

Miscellaneous files that are used to store isolated data.

Window Editor



Grid

Spacing The default grid spacing in the window editor.
Color The default grid color setting in the window editor.
Visibility The default grid visibility setting.

Theses settings are the default settings for new windows. The settings for each window are saved with the window. Refer to graphics configuration for more information.

Background color

The editor window can be larger than the user window. For example, the user window can be 800 X 600 and the editor window can be much larger. This creates a blank area in the editor window. This attribute sets the color of the blank area.

Text editor

Use background color If the text element "transparency" is true the window's "foreground" color of the fill settings is used otherwise, the background color of the text element is used.
Color Sets the color when the "Use background color" is not enabled.

Note: If the font color and the selected color to be used as the background in the text editor are the same color, the background in the text editor will be light grey.

Warn on ungroup with animation configured

Grouped graphic elements can be configured for animation. If a grouped graphic element is "ungrouped" the grouped element no longer exists and the animation(s) for the group is lost. The animation for the elements of the group, if configured, are not lost.

Automatic size static text

When an attribute of a text element is changed, if the element does not have any animations, the bounding rectangle will be adjusted to fully enclose the text.

Note: Does not apply to rotated text.

Hide status bar

The bar at the bottom of the graphic editor that displays size, position and selected count for the graphic elements. When enabled the bar will not be displayed.

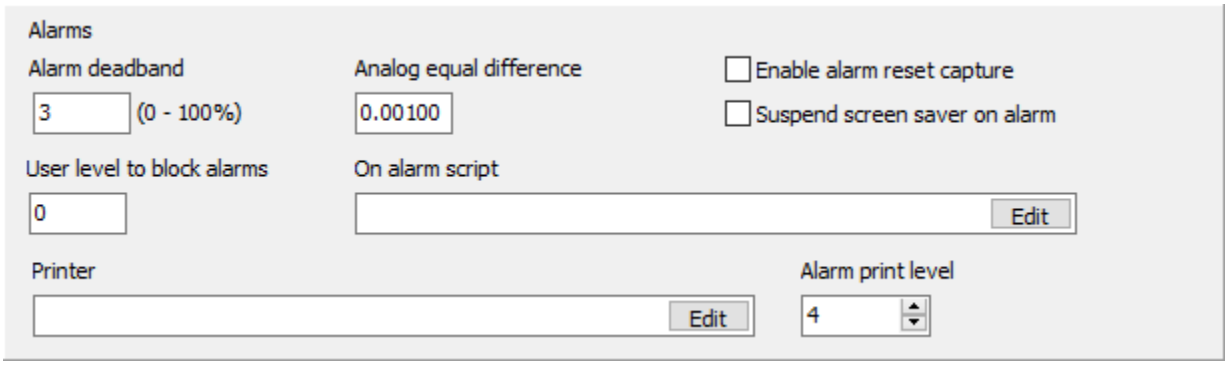
Group attribute change

A grouped element does not have a color or pen width, etc. If this attribute is enabled, changing the foreground, background or pen color will change the attribute in all the elements of the group that utilize the attribute. The same applies to brush style, pen style and pen width. This does not apply to bitmaps or WMF/EMF. This is for configuration mode. It does not apply to runtime animations.

Plain text preferred

When a program places text on the clipboard, if the program supports RTF (rich text format) and plain text (no text formatting), the program may place both text formats on the clipboard. If this property is enabled, when pasting the contents of the clipboard to the window, the plain text will be pasted on the window.

Alarms



Alarm deadband

This is the default value applied to alarm deadband settings when a point is created during configuration.

User level to block alarms

This is the required user level to block alarms via the runtime panel "Alarm Blocks" button.

Analog equal difference

This value is the maximum difference between two floating point values and to be "considered" equal.

Floating point numbers are represented in binary fractions and therefore are always an approximation of a decimal fraction.

That is why occasionally a value like 1.980000000001 is displayed for something that would be expected to have only a few decimal places of precision. When comparing two floating point values, the comparison should be on how close the two values are to each other rather than testing for equality.

AED = 0.001 (default)
Equal = ((a - b) <= AED)

On alarm script

If configured, when an alarm is added or removed from the master alarm list the configured script and script function is called. See OnAlarmEvent for scripting information.

Enable alarm reset capture

This enables the “life” of an alarm to be captured in one location. See the “FetchAlarmReset” script command.

Suspend screen saver on alarm

If enabled, when a new alarm occurs a command is sent to the OS to suspend the screen saver. See script function ScreenSaverSuspend for more information.

Printer

Select the name of the printer for runtime alarm printing. If the project is created on one computer and run on another computer, verify the correct printer is selected. If the names do not match alarm printing will be disabled. To disable runtime alarm printing, clear this field.

Alarm print level

This attribute is used to save paper on single sheet printers. Each alarm prints 4 lines. Setting this value to other than zero (0) causes the <count> alarms to be queued for printing.

Example: Setting this value to 0, each alarm prints as is determined.

Example: Setting this value to 3, each alarm is queued until 3 alarms are in the queue, then the alarms are printed.

If the selected printer is a single sheet printer and this setting is 0, each alarm would be printed on a separate page.

Print Format

Line 1:Tagname and right aligned date and time
Line 2: Condition status ":" value at alarm time
or
Line 2: Normal status "-" Condition status
Line 3: Point description
Line 4: Blank

Example:

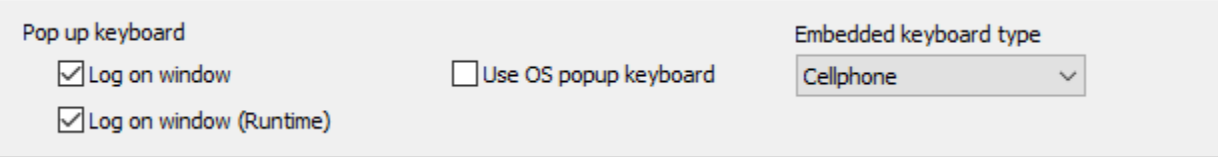
Tower_1_Temperature_4 7/4/2007 10:51:12 AM
Hi Temperature : 562.34
Temperature probe level 4

Tower_1_Temperature_4 7/4/2007 10:52:45 AM
Hi Hi Temperature : 582.10
Temperature probe level 4

Tower_1_Temperature_4 7/4/2007 11:15:00 AM
Normal - Hi Hi Temperature
Temperature probe level 4

Tower_1_Temperature_4 7/4/2007 11:20:37 AM
Normal - Hi Temperature
Temperature probe level 4

POP UP KEYBOARD



Log on window

If enabled the on screen virtual keyboard will be displayed when the log on window is displayed.

Log on window (Runtime)

If enabled the on screen virtual keyboard will be displayed when the log on window is displayed, at runtime.

Use OS popup keyboard

The HMI has an embedded keyboard. The size is fixed and it always appears below the “Log on” dialog.

The OS also provides a popup keyboard.

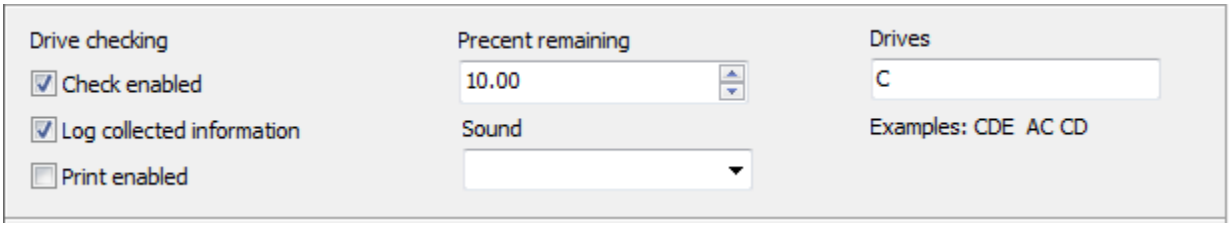
On Microsoft Windows 7, the keyboard size is adjustable and the keyboard appears in the last place the dialog was positioned before it was closed.

On Microsoft Windows XP, the keyboard size is smaller than the embedded keyboard and the keyboard appears in the last place the dialog was positioned before it was closed.

Embedded keyboard type

If the “Use OS popup keyboard” is not enabled the embedded keyboard type selection.

Drive checking



Check enabled

If enabled the runtime program will check the drive capacity once per day at midnight.

Log collected information

If enabled when the runtime program checks the drive the information collected from the drive will be logged to the event log. The format in the log file is one line per drive. The values are separated by a comma. The fields are:
Drive, Ready, Low space, Total space, Free space, Used space, Percent free, Percent used

Example:
C, True, False, 465.74 GB, 386.73 GB, 79.01 GB, 83.04%, 16.96%

Print enabled

If this is enabled and an alarm printer is configured, when a low percent remaining condition for a drive is detected, a message will be printed.

Percent remaining

If the amount of free space is below X percent of the drive total space an entry will be added to the event log. If the sound setting is configured the sound will be queued for playing.

Sound

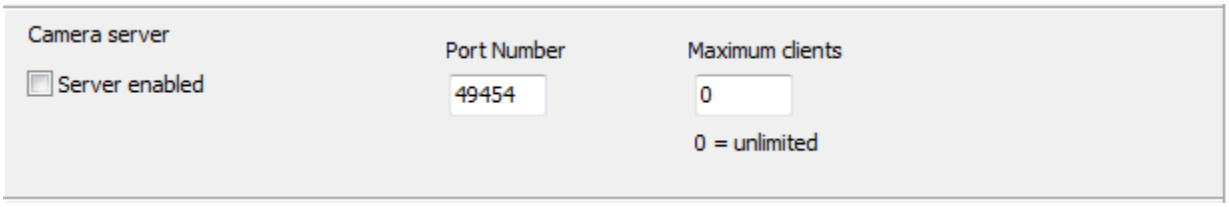
Select the sound to play when the drive free space is below the threshold.

Drives

Enter the drive letters to be examined. The allowable drive letters are A-Z. Via scripting the drive capacity can be viewed. This setting is used for the checking of the drive capacity and the information displayed in the window.

Note: If the drive is not ready, e.g. drive has been removed, power is off, etc., the check will only log that the drive is not ready and will not play a sound.

VIDEO SERVER



When an HMI is collecting images from an IP camera the images can be published. Enabling this checkbox allows a remote HMI to request camera images.

Server enabled

This checkbox must be checked for the server to be active at runtime.

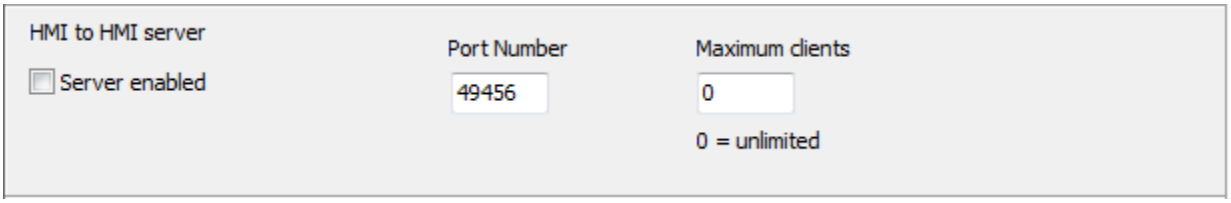
Port number

The TCP/IP port number the port is to use. 49454 is the default.

Maximum clients

The maximum number of client connections. 0 = unlimited and is the default.

HMI TO HMI SERVER



The points of this HMI can be published for use by remote HMIs. Enabling this checkbox allows a remote HMI to read/write the points.

Server enabled

This checkbox must be checked for the server to be active at runtime.

Port number

The TCP/IP port number the port is to use. 49456 is the default.

Maximum clients

The maximum number of client connections. 0 = unlimited and is the default.

User inactivity monitor

This is used to monitor if the user is interacting with the HMI and if not then perform some action.

Enabled

This checkbox must be enabled for the monitor to be active at runtime.

Minutes

If the mouse position is not changed for X minutes the user is considered inactive. Mouse position has a deadband of +-5 pixels. The time is checked at each new minute.

Log off user

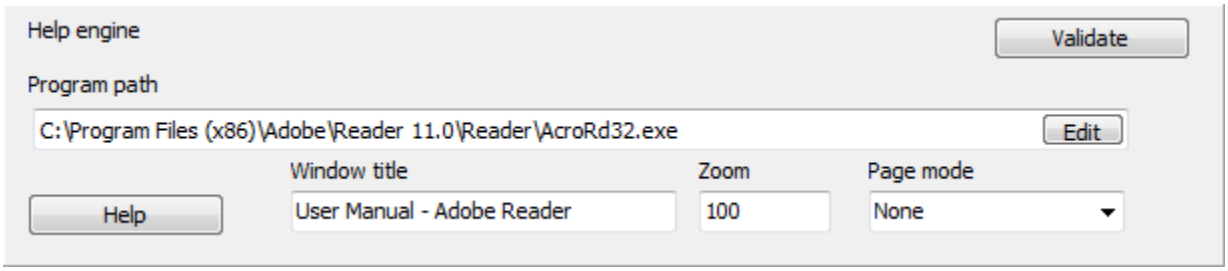
If enabled, the user will be logged off.

Note:
If other actions are needed when a log off occurs, use a task.

Queue script

If a script is selected it will be queued to execute.

HELP ENGINE



The HMI uses the PDF file format to document the HMI and provide context sensitive help. Three help viewers are supported. Select the "Help" button for more information.

Program path

This is the path to the program utilized to display the PDF file. Use the edit button to manual select a program.

Validate

Selecting this button will command the program to attempt to discover the external program that is associated with files that use the ".PDF" file extension. If one of the three known programs is found the "Window title" will be properly formatted.

Window title

This is the window title, of the window the HMI opens, to display the help file. The correct window title is required for the HMI to properly close the help file as needed.
Note: "Adobe Reader" changed the window name.
Old <filename.ext> - Adobe Reader Example: User Manual**.pdf** – Adobe Reader
New <filename> - Adobe Reader Example: User Manual – Adobe Reader

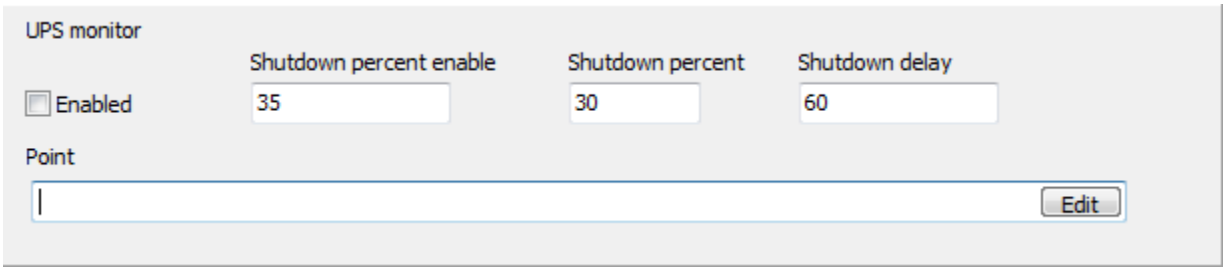
Zoom

This is the zoom level of the help window contents.

Page mode (Adobe Reader)

The "bookmarks" or the "thumbs" panel can be set "visible" when the HMI opens the help file.

UPS MONITOR



Notes:

1) Some devices may report incorrect values, not report any values or cycle between correct and incorrect values.
2) To only monitor the "Battery life remaining" value, set the enabled checkbox to true, set both the "shutdown percent enable" and "shutdown percent" fields to 0 (zero) and assign a point to the "Point" property.

Enabled

If this is true, the runtime program will read the "Battery life remaining" value from the OS once per second. The value represents the percent of battery life remaining. The value will be 0 – 100 or 255 if the status is unknown.

Shutdown percent enable

The logic to shut down the computer on a low "Battery life remaining" value is edge triggered. The battery life remaining value must be or rise above, this value (and not be 255) before the logic is executed to monitor for a falling battery life remaining value. This allows for the battery to be recharged when "mains" power is restored without the program commanding a "shut down" from a low battery life remaining value. This value must be greater than the "shutdown percent" property.

Shutdown percent

If the "Battery life remaining" value falls below this value a command will be sent to the computer to shut down. The "Shutdown delay" property could delay the command.

Shutdown delay

If the shutdown command is triggered and this value is greater than 0 (zero) a timer will begin and delay the shutdown command until the timer expires. When the timer expires, if the "Battery life remaining" value is above the "Shutdown percent", the command will not be executed and the logic will readied to monitor for a falling battery life remaining value.

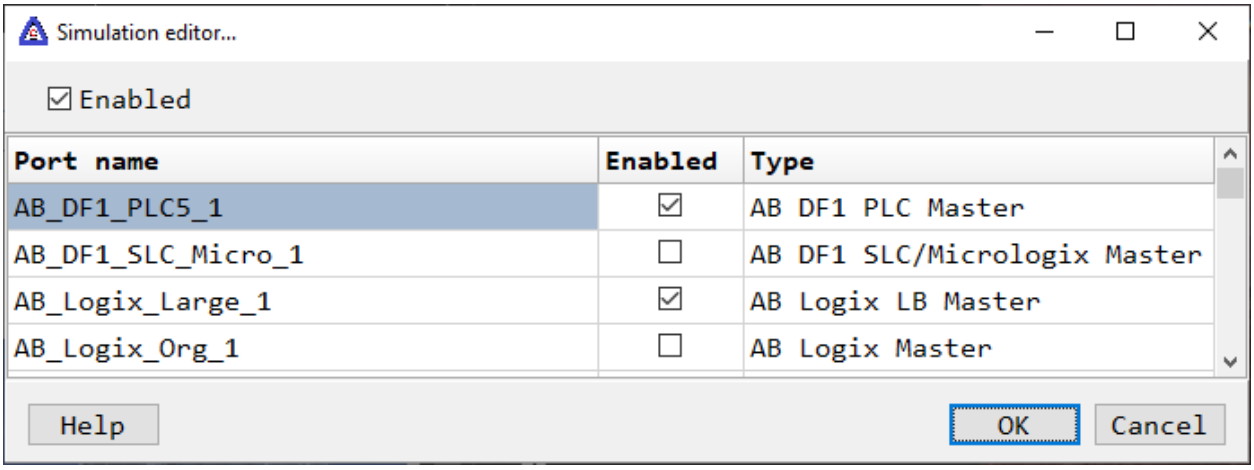
Point

The "Battery life remaining" value can be accessed via an analog host point.

SIMULATION

To facilitate **graphic testing** the "ports" (communication and video) can be simulated allowing the "points" to be set via the "All points monitor" window or scripting.

Which ports to simulate can be selected. At least one port must be simulated for simulation to be active.



Enabled The global "simulation" property. It must be enabled for simulation to be active at runtime. If all ports are disabled, this property will be set to disabled. This property can be examined at runtime via the "GetSystemVariable" script command. See the "Simulate" script command for additional script features.

Notes:

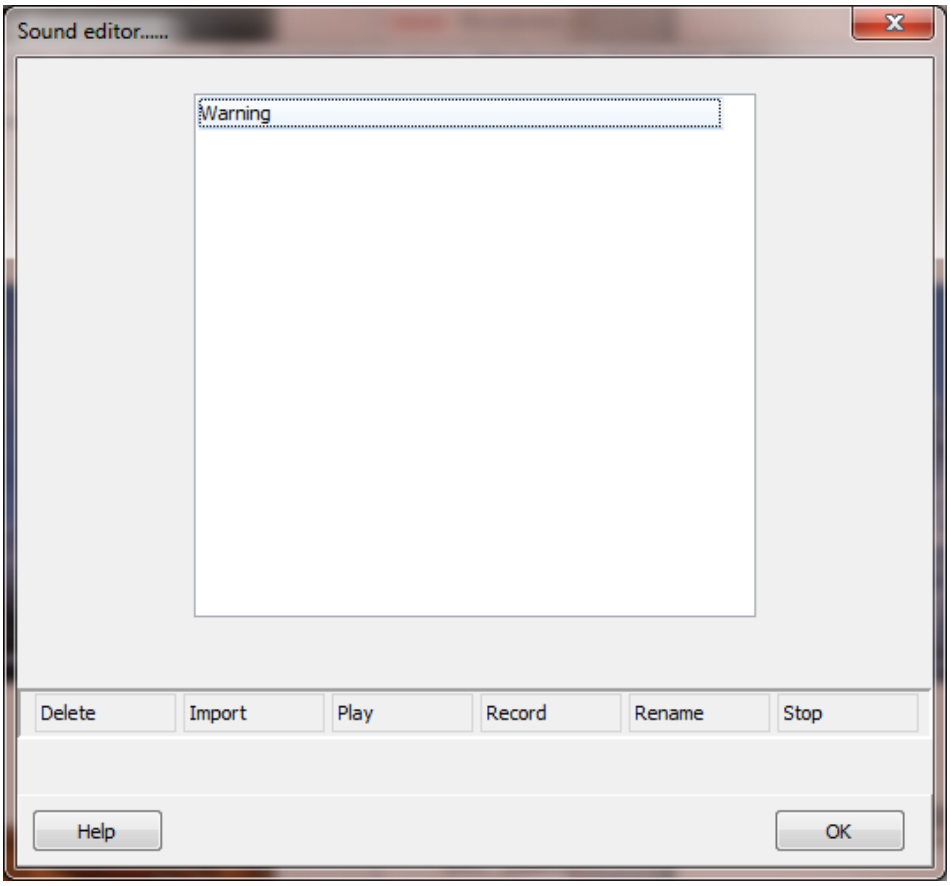
1) All port addresses, settings, reads, etc. are ignored during simulation.
2) All point configurations are considered valid. (address, type, read/write, etc.) i.e. the address could be invalid for actual use. The simulator does not validate port configuration settings.
3) Read/writes to external devices are not executed for ports in simulation.
4) Some ports that provide read/write counters, the counters are simulated.
5) Script command *OpenTagMonitorWindow("");* without a tagname specified, will open the "All tag monitor window." Left mouse clicking a point value column will open a dialog to change the point value. Normal script functions to read/write points can be used.
6) All points of simulated ports are set to read/write access rights for simulation.

Sounds

Sounds can be queued to play when an alarm transitions to true, by script command, etc..

Notes:

1) Files without an extension are assumed to be “WAV” sound files.
2) Due to changes in the OS sounds might not play, even if the dialog indicates the sound is playing, because the key combination “CTRL + Break/Pause” was previously pressed. With no sound playing press “CTRL + Break/Pause” and try the sound again. If assistance is needed contact support.



Delete

Delete the selected sound.

Import

This provides a method to import a "WAV" or “MP3” file.

Play

Select a sound or sounds and select the “Play” button. The selected sounds will play.

Record

After selecting a request for a sound name dialog will appear. Duplicate sound names are not allowed. After the "OK" button is selected and the name is allowed recording will begin. Recording will not stop until the "Stop" button is selected.
Recorded sounds are “WAV” sound files.

Rename

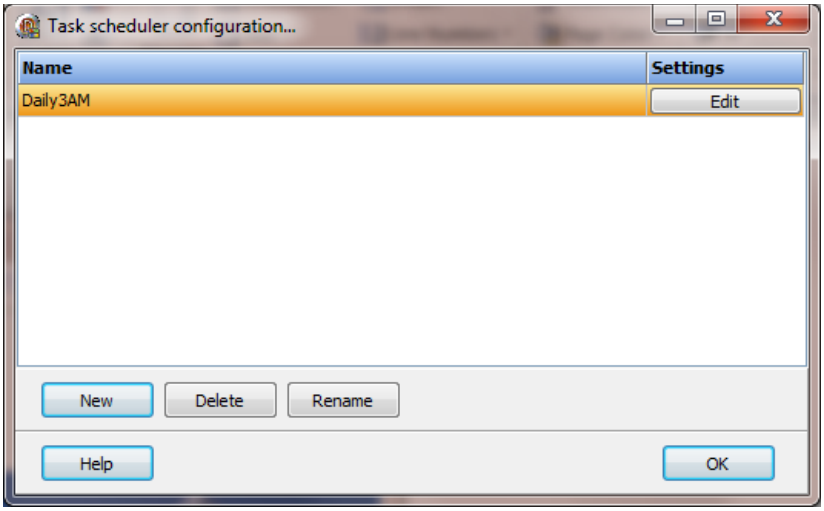
Select this to rename the selected file. **Note:** If a file extension is not provided the extension will be set to “WAV”. Verify this is the correct extension by using the “Play” button to test the sound.

Stop

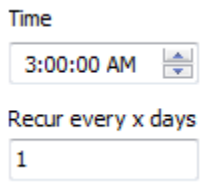
Select this to stop recording. Also, if several sounds have been selected for playing, selecting this button will stop playing the sounds after the current sound is finished.

TASK SCHEDULER

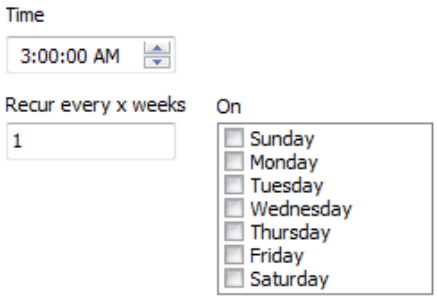
Each task scheduler object is listed in the window. A task is a command to the runtime application to take some action.



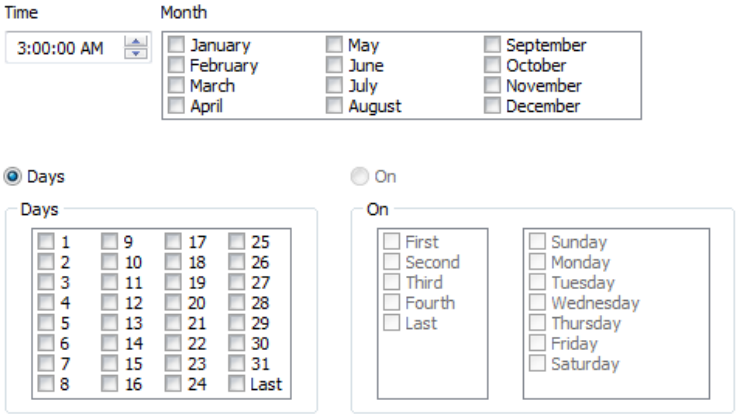
Daily



Weekly

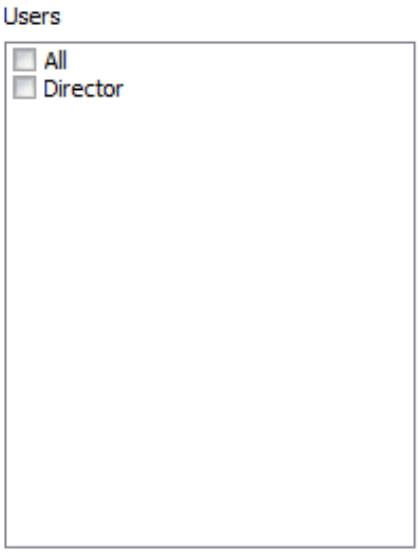


Monthly

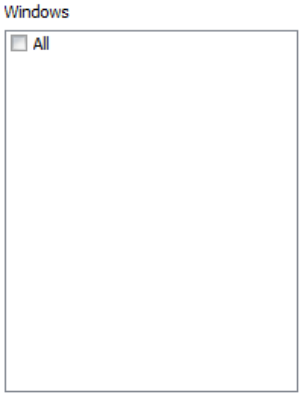


When runtime starts

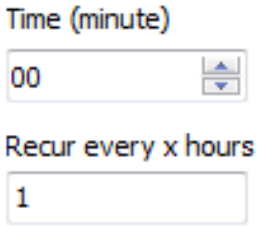
When a user logs in or out



When a window is opened or closed



Hourly



Trigger

This is the event that is used to trigger the actions.

Enabled

If enabled the task will be active at runtime. This setting is also modifiable via scripting. An object can be enabled but never execute due to configuration settings. For example, a monthly trigger could configured with the "Days" set to "31". Only those months with 31 days would trigger a task on the 31st day of the month. Or, a "Daily" trigger is set and runtime starts after the time setting.

Show

When the task is time based selecting the show button will display the next time the task would run in runtime.

Time

Some tasks are time based. This is the time to execute the actions configured. The seconds field is ignored.

Recur every X days

The actions may execute again. 1 would be every day, 2 would be every two days, etc.
0 = The actions will execute once on the day runtime starts.

-1 = If the design is to only execute at X time and runtime starts after the time settings, a value of -1 instructs the program to execute the actions now. "When runtime starts" might be a better trigger.

-2 = If the design is to only execute at X time and runtime starts after the time settings, a value of -2 instructs the program to execute the actions the next day at the X time setting. "When runtime starts" might be a better trigger.
Note: If runtime starts, stops and starts again the task might trigger. If that causes a problem shift to scripting. If assistance is needed, contact technical support.

Recur every X weeks

The actions may execute again. 1 would be every week, 2 would be every two weeks, etc.

Recur every X hours

The actions may execute again. 1 would be every hour, 2 would be every two hours, etc. If the minutes of the hour have passed when runtime starts the task will execute at the ”recur” hour.

Weekly – On

The day or days of the week the task is to execute.

Month

The month or months the task is to execute.

Monthly - Days

The day or days of the week for the task to execute.
Last - If selected the task will execute on the last day of the selected months regardless of the number of days in the month.

Monthly - On

On - The week or weeks the task is to execute and the day of the week.

Last - If selected the task will execute on the last day of the days selected.

When runtime starts

The task will execute each time the runtime program begins operations**.**

When a user logs in / logs out

The task will execute each time the named user logs in/out.
All - The task will execute each time any user logs in/out.

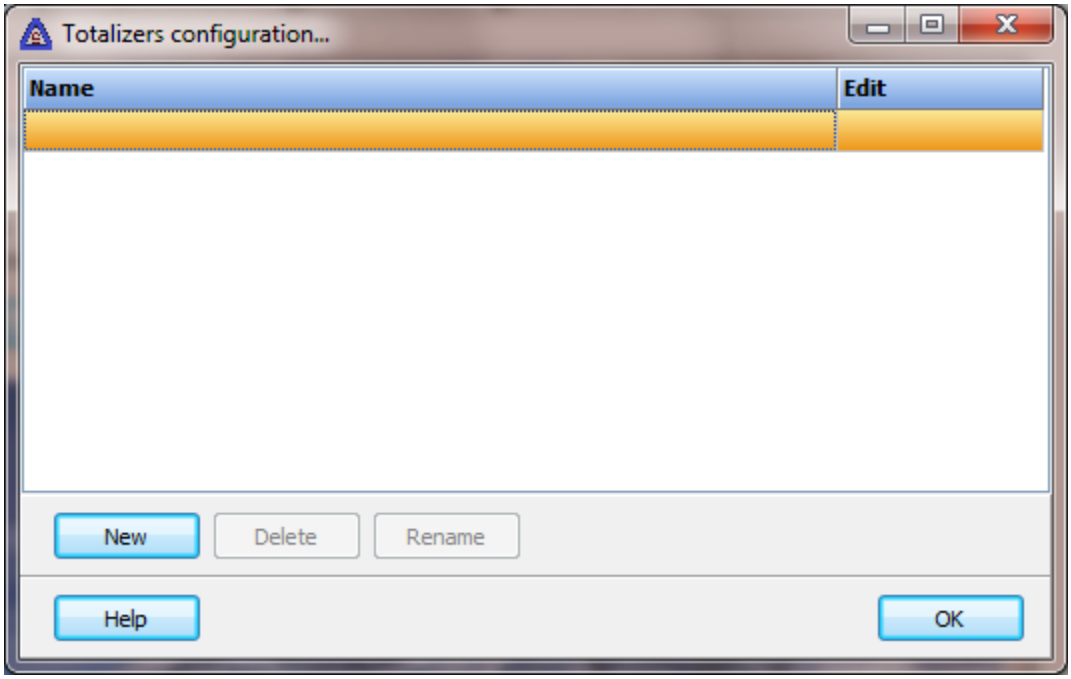
When a window is opened / closed

The task will execute each time the named window is opened/closed.
All - The task will execute each time any user window is opened/closed.

Actions

The actions to perform when the task is executed. See the mouse commands for a list of possible actions.

TOTALIZERS



Totalizers are used to add the input value at a fixed frequency to the current total. This allows the "totalizing" of the input value.

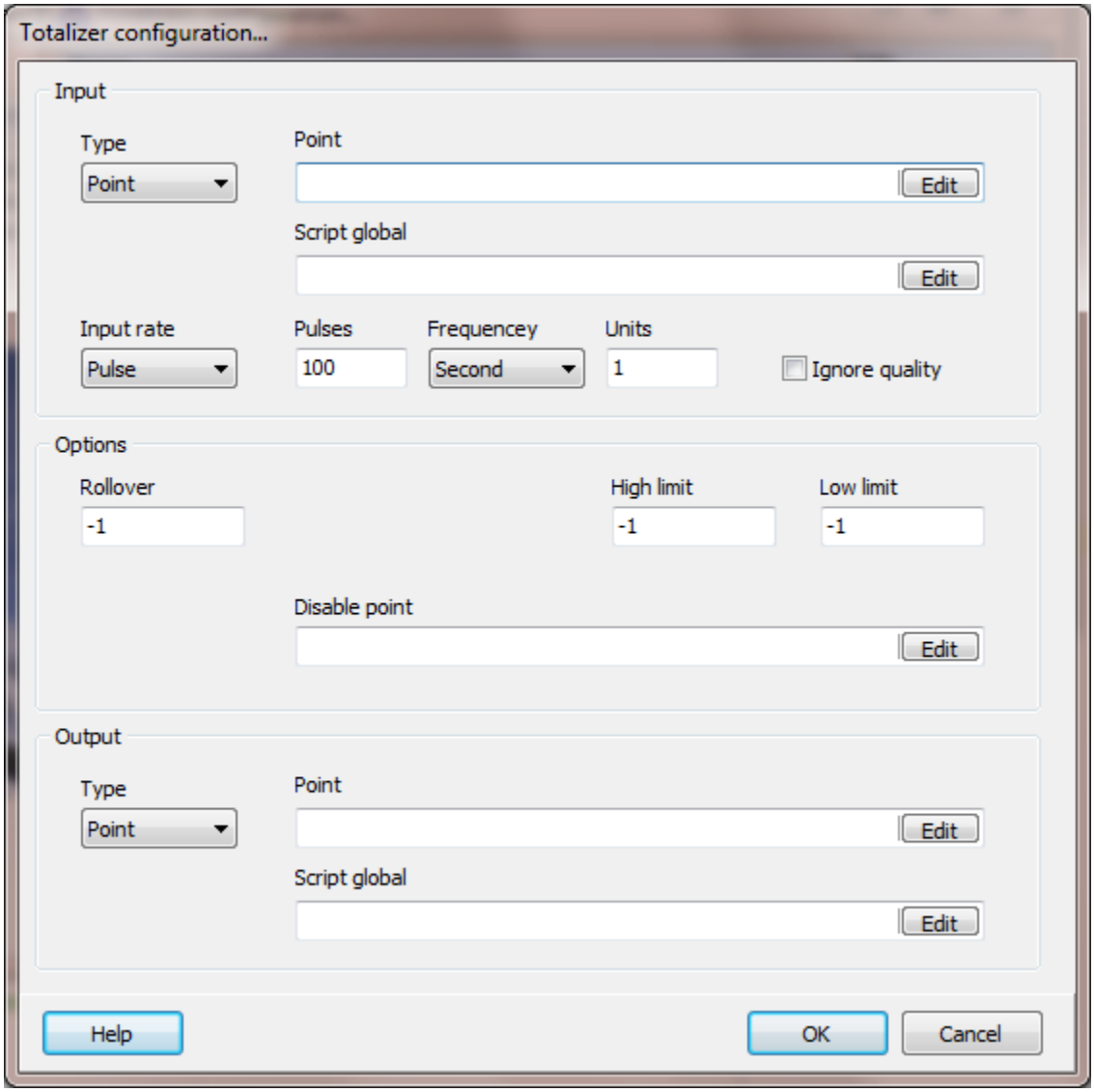
The totalizer begins when runtime starts and executes until runtime ends. Each total is calculated for twenty four (24) hours beginning at 00:00:00 (midnight).

If runtime ends via a normal shutdown and runtime later resumes the same day (the same 24 hour period as it began) the totalizing is resumed.

The totalizer provides data storage for 31 days. The first of the month is the first location, the second of the month is the next location and so on.

Each totalizer must have a unique name and is case sensitive.

TOTALIZER CONFIGURATION



Input

Type The input type can be an analog point or a script global. If a script global is used the value must be numeric.

Input rate The value can represent a value that is second, minute, hour or pulse timed. e.g. gallons per second, gallons per minute gallons per hour. A pulse input is counts over a time period e.g. 100 pulses per second = 1 gallon. The "Pulses" and "Units" must be defined if the input rate is "Pulse".

Ignore quality If this property is enabled and the input point quality is bad the point value will be used to calculate the total. This does not apply if the input is a script global.

Options

Rollover This option provides for the totalized value to be reset to zero (0) when the totalized value exceeds this value. Set this property to minus one (-1) to disable this feature. At midnight the value is reset to zero (0) regardless of the setting. Use scripting to query the totalized value for the current day, a specific day or a range of days.

High/Low limit This option provides a method to exclude an input value that is outside a specific range. i.e. some devices will output a negative or very high value when the flow has stopped, the measured substance is too cold, too viscous, etc.. Set both properties to minus one (-1) to disable this feature.

Disable point This option provides a method to stop totalizing the value based on another input. e.g. the pump is stopped, the tank is empty, etc..
If the input is true the totalizer will not add the input value to the total until the input is false.

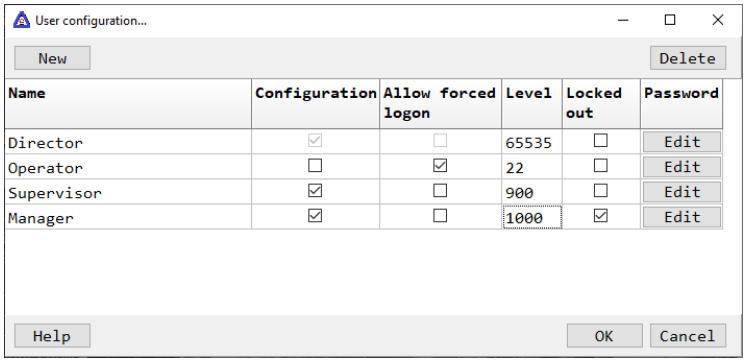
Output (optional)

Note: The output is optional. Scripting can be used to read the totalized values.

Type The output type can be an analog point or a script global. **Note:** Use caution when selecting the output. If the output is a point and the point is addressed to an external device, a write command will be issued to the device every second. This may overload the device or cause communication issues. It might be better to write to an analog host tag/script global and then use another method to periodically write to the external device at a lower frequency.

Users

The "Director" user is always present. The level is fixed at 65535 and the "Configuration" is always enabled. For new projects the password is blank (no password assigned).



Name

The name of this user.

Configuration

Enable this checkbox to allow the user to configure the selected project.

Allow forced logon

Enable this checkbox to allow this user to be logged on without the user entering a password via a mouse command or script command.

Level

During runtime monitoring the user created buttons, and most other buttons, are assigned a user level to allow the logged on user with a certain level to activate the button or not activate the button. The levels are 0 - 65335. A user at "X" level can activate all buttons at "X" level and all lesser levels.

Example:
Bob is logged on. Bob's user level is 100.

The button user level is 0. Bob **can** activate the button.
The button user level is 100. Bob **can** activate the button.

The button user level is 101. Bob **cannot** activate the button.
The button user level is 65535. Bob **cannot** activate the button.

Locked out

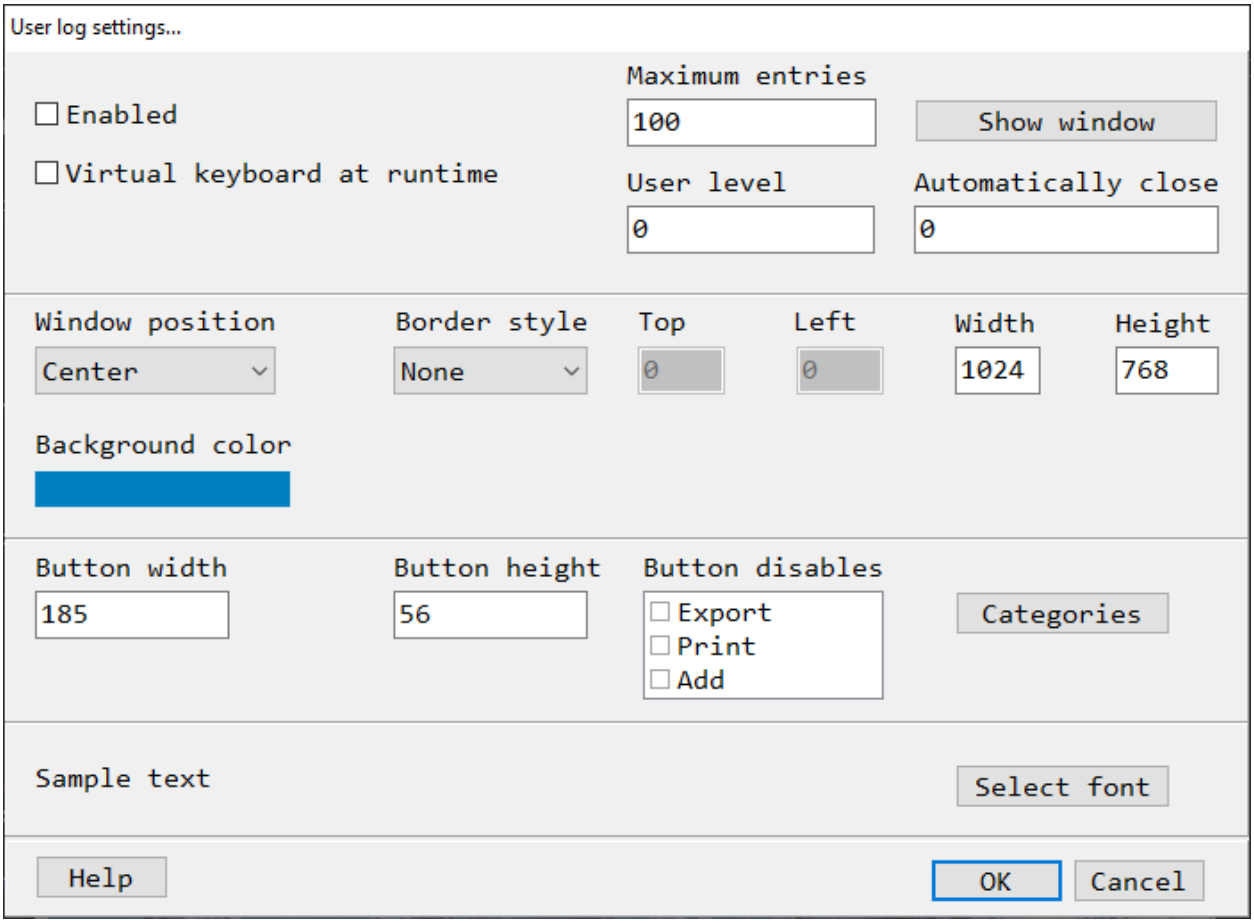
The user can be "locked out" via setting this attribute or failing to enter the correct password after <count> times.
See “Failed lockout count” in “Configuration/settings”

Password

Enter the password for the user. The password can be blank. A unique password is not required for each user.

User logs

The user logs provide a method for online operations to enter data into a log.



Enabled

This enables or disables runtime user logging.

Border Style

Microsoft Windows provides for several window border styles. None, single, sizable and dialog. These styles can be viewed by selecting the "Show Window" button in the middle of the screen.

Virtual keyboard at runtime

The virtual keyboard will be displayed below the text field when adding a log entry. Window Position

Select the position to open the window. "As designed" is the same as "Top Center".

Window width/height

This size of the window, including any border and title bar.

Show Window

View the window with the selected settings. When the window is visible, click the mouse button to close the window. **Note:** The size of the window is limited to the monitor size.

User Level

At runtime the logged on user must have at least the level entered to make additions to the user log. Any logged on user can view the user logs.

Automatically close

The number of seconds the window will be open. If the value is 0 the window will not automatically close. **Note:** This is for the log viewing window only.

Button width/height

This is the button on the right side of the window.

Button disables

Each button can be disabled and will not be displayed in the button area. Note: The 'Add' button might be visible but disabled based on the 'User level' setting above.

Categories (optional)

Each log entry can be assigned to a category. If no categories are defined the 'Categories' button will not be visible. When categories are used, only those log entries for the selected category are displayed. Use the 'Categories' button to select the category.

Select font

The font to use in the log text field.

Background color

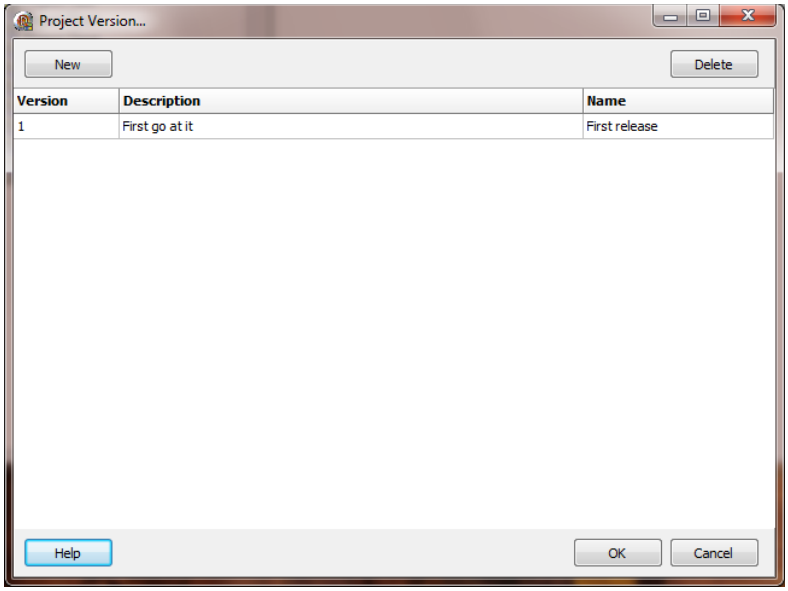
The background color of the log text field.

Maximum entries

The maximum number of entries, total for all categories. When the limit is reached and a new entry is added, the oldest entry will be deleted. The maximum limit is 1000.

VERSION

This provides a method to assign a version value, name and description to the project. The fields are strings and any data, limited to 255 characters, is permissible. The value in the "Version" column of the last line is shown on the main window at the "Version:" location.



Version

The version of the project.

Description

Normally, a brief description of the version. e.g. As Released, As Built, As Commissioned, Added point EG-101, etc..

Name

Normally the name of the user creating the version.

VIDEO

The HMI provides three methods to connect to video cameras or video encoders to display the image in a window, detection motion (IP cameras), record the video and capture snapshots of the image.

Warnings: Recording video can consume large amounts of disk space in a short amount of time.
Some VFW (video for windows) drivers are known to not function correctly with the Windows OS screen saver. If VFW is used verify the Windows OS screen saver is disabled.

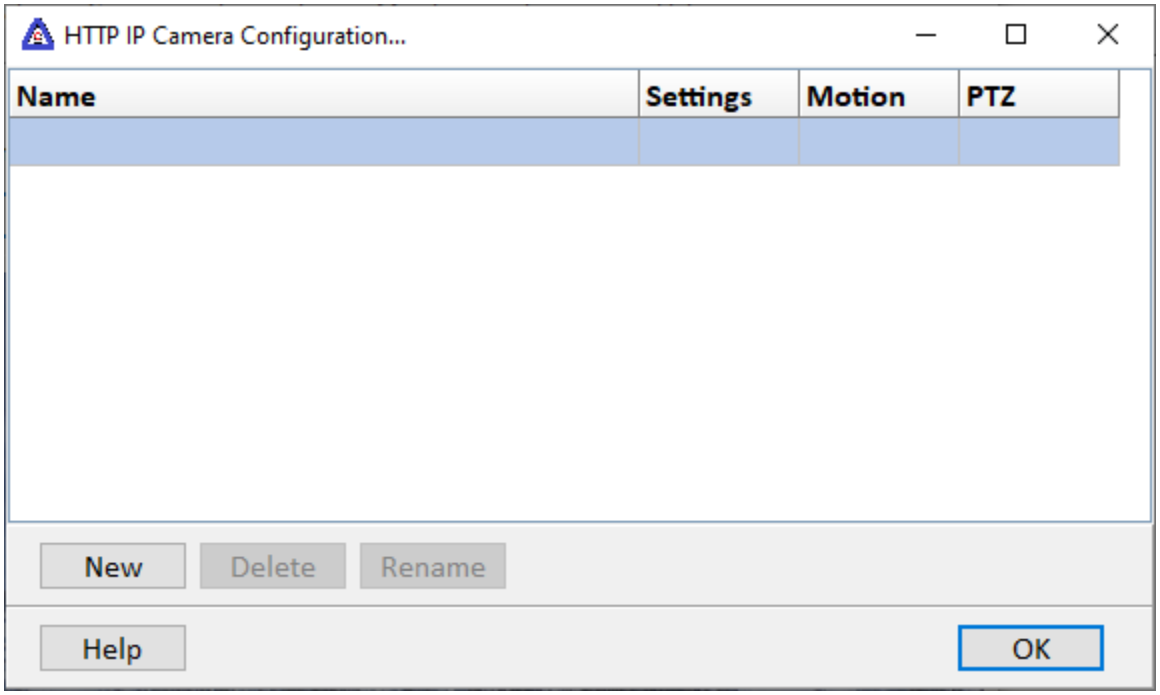
Each video must have a unique tagname assigned.

VFW: The same camera can be referenced more than once but live images can only be accessed in one graphic element at a time. If the same camera is referenced in an open window or windows more than once an error will occur requiring restart of the computer.

HTTP/RTSP IP Video: There is not a restriction on the display of images in windows. Displaying the camera image in more than one window will not cause an error.

HTTP Video

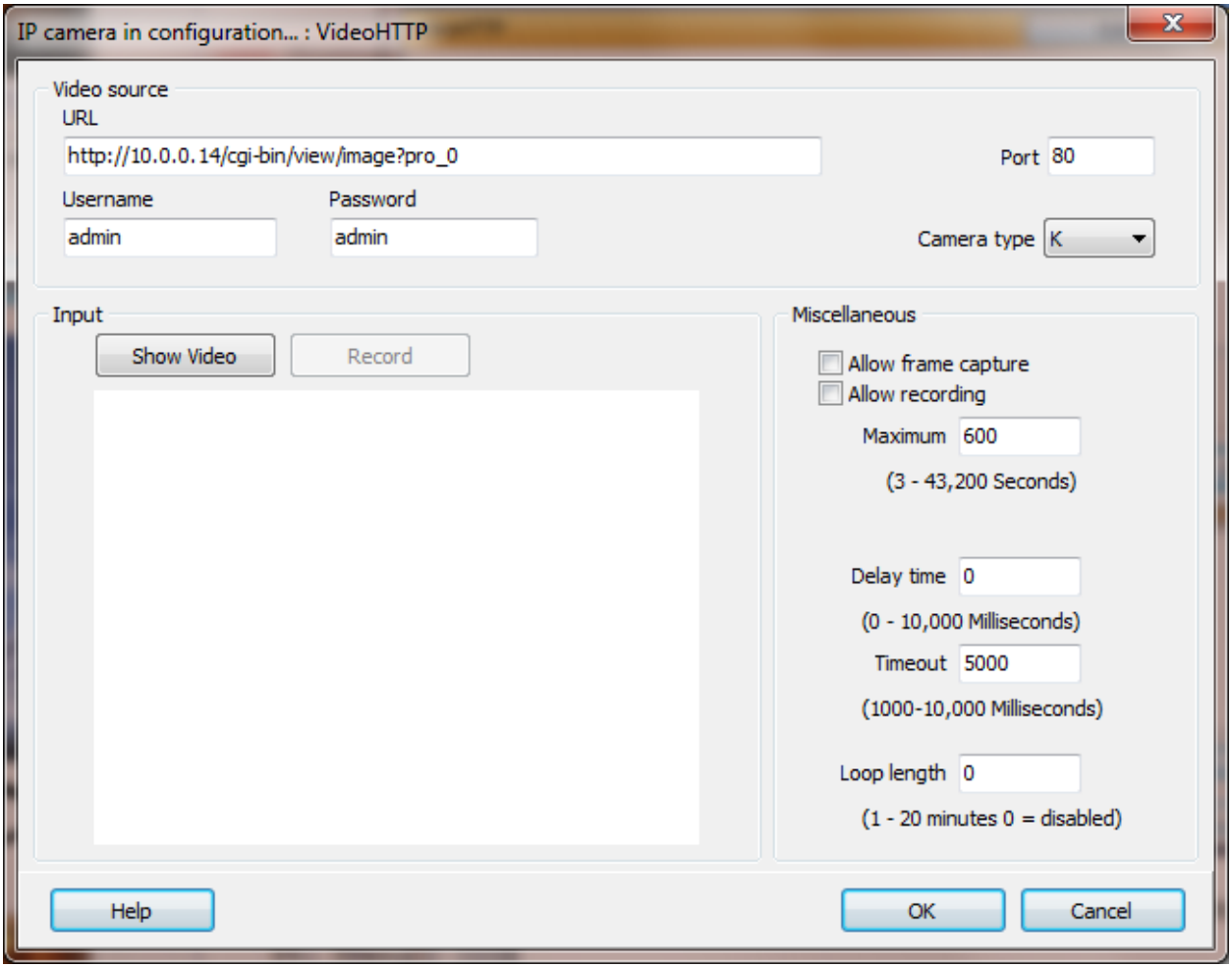
The HTTP protocol is used to access the camera video feed.



Multiple cameras/video encoders can be accessed concurrently.

Motion
PTZ (pan, tilt, zoom)

SETTINGS



Video Source

URL

The complete path to the camera/encoder including the HTTP "Get" text or other text.

Port

The TCP/IP port number. HTTP default is 80. RTSP default is 554.

Username/Password

The user name and password for camera/encoder requiring authentication.

Type

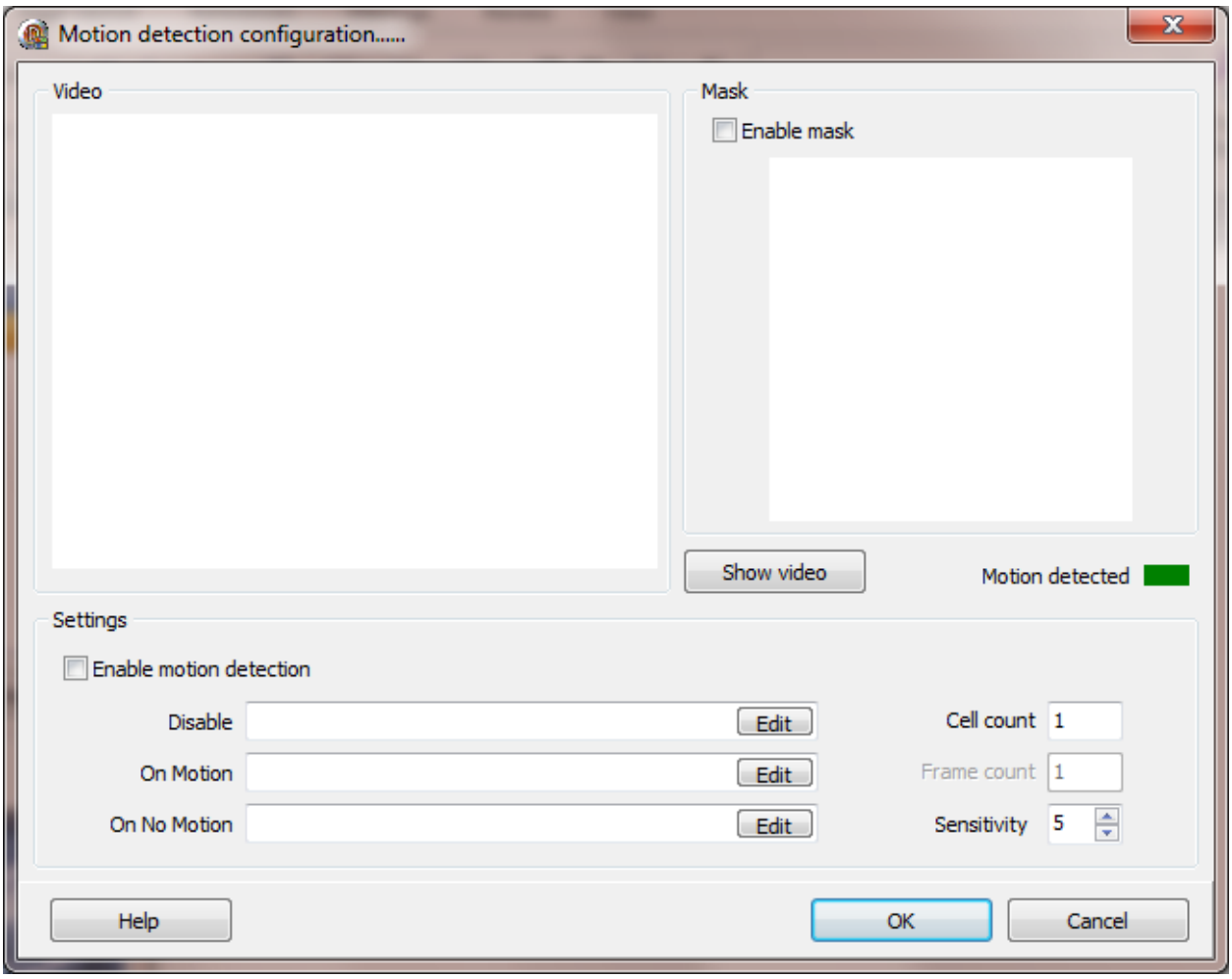
A: SkyIPCam 310 (GET)
B: Moxa VPort 351 Industrial Video Encoder (MJPEG)
C: Axis 241Q Video Server (GET) (Default user: root Default password: h)
D: SkyIPCam 250 (MJPEG)
E: Axis 241Q Video Server (MJPEG) (Default user: root Default password: h)
F: Moxa VPort 2141 Industrial Video Encoder (MJPEG) **Note:** It appears most of the possible settings are ignored when embedded in the request string. Be sure to set and save the settings using the Moxa interface. For example, the default frame rate is 30fps. This might be too high for the application. 10fps is normally a good frame rate.
G: Refers to camera in a remote HMI. The URL format is the name or IP address followed by the camera/port name. e.g. 192.168.1.1.Camera2
H: Moxa VPort 254 Industrial Video Encoder (MJPEG)
I: Trendnet TV-IP600 (MJPEG) (Default user: admin Default password: admin) This camera uses 'Basic' authentication.
J: Moxa VPort 364A Industrial Video Encoder (MJPEG)
K: Generic (MJPEG) This driver searches the stream for the JPEG image markers to determine a frame. Any content-length field is ignored. This is the **recommend driver** for all MJPEG applications. **Note:** While the HMI captures all transmitted images, the graphic window refresh rate is limited to maintain a responsive UI (user interface).

Example URL

1) Zavio camera using profile 3: http://<ip address>/stream?uri=video.pro3
2) Loftek CXS 2200 camera: http://<ip address>/videostream.cgi?rate=8
3) Axis M5054 camera http://<ip address>/axis-cgi/mjpg/video.cgi
   If user name and password required,
   http://<name>:<password>@<ip address>/axis-cgi/mjpg/video.cgi

The remaining settings are below under "Common Settings".

Motion



Notes:

1. Changing any setting while the video is displayed will not take effect until the video is stopped and restarted.
2. When the collection of images begins the motion detection code waits 1.5 seconds for the image to stabilize before motion detection logic executes. If RTSP is used the logic waits for the first frame to be received. Then the logic executes approximately every 100 milliseconds. If the watchdog timer triggers the detection code is reset and the initial wait logic executes.

Enable motion detection

If enabled the motion detection code will execute at runtime. The code will execute regardless of the display of the camera/encoder image.

Disable (optional)

A tagname/item id used to disable the motion detection OnMotion and OnNoMotion commands at runtime.

On Motion, On No Motion

Select a script to execute when motion and/or no motion is detected. These commands are edge triggered.

Cell count

The camera/encoder image is divided into a grid of 32 columns and 32 rows. This is the number of cells that must detect motion before the "On Motion" scripts executes.

Frame count

Not used at this time.

Sensitivity

This value determines the amount of change in a cell required for the cell "changed" state to be set. Higher values are more sensitive.

Mask

The mask is a grid used to disable a cell from inclusion in the "motion" calculation. The grid is 32 * 32 and the black cells are not examined in the motion calculation.

Enable mask

If enabled the mask is incorporated in the motion detection code.

Show video

Select this button to display the video image and to use the current settings to detect motion. The "Enable Motion Detection" checkbox must be enabled for the motion detection code to execute.

Motion detected

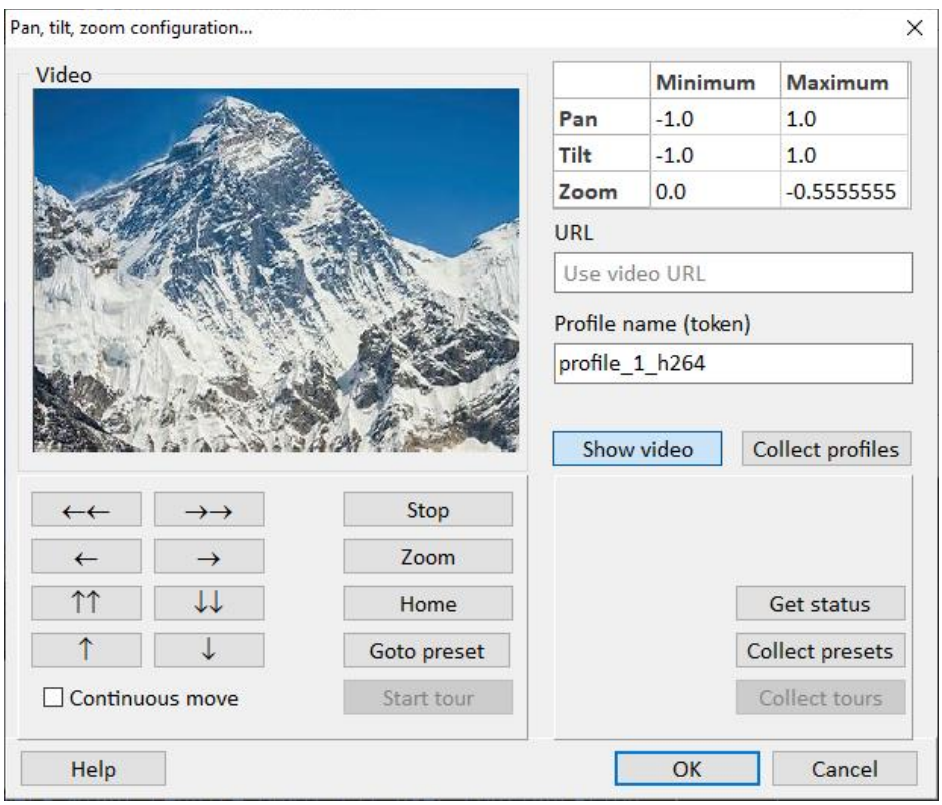
When video is displayed and the "Enable Motion Detection" checkbox is checked, the rectangle will be green when no motion is detected and red when motion is detected.

PTZ (Pan/Tilt/Zoom)

The program uses the ONVIF (Open Network Video Interface Forum) specification for PTZ control.

Notes:

1) Not all cameras are factory configured with ONVIF PTZ enabled. Generally, a user must be created in the ONVIF setup.
2) The clock on the PC and camera should be synchronized using the same time source. If not possible, manually set the time in the camera to the PC time. The HMI "attempts" to calculate any time difference when generating command messages. Some cameras are not time synchronization sensitive.
3) Not all cameras implement PTZ the same. For example, a camera may report it supports "zoom" and then does not have a zoom function. Or, camera A may stop a relative move at the end point and camera B will move to the other end point and offset. Or, camera A moving can be stopped with a "Stop" command and camera B only allows "Stop" for ending a preset pattern move.



Note: If the "**Profile name (token)**" is not blank, PTZ will be enabled, at runtime. The value is the "token" of the profile name. Use the "**Collect**" button to fetch the profile token(s) from the camera.

The **PTZ minimum/maximum** values specify the allowable ranges when sending a command to the camera. To disable the range limit set the minimum and maximum values to 0 (zero).

URL This can be a specific URL or the "root" of the video URL will be used.

For example, if the video URL is:
http://RED:DWARF@192.168.8.218/axis-cgi/mjpg/video.cgi and the PTZ URL is blank,
the program will use:
http://RED:DWARF@192.168.8.218 as the URL.
The HMI program will append the correct ONVIF sub path.

Note: If a port number is specified in the video URL settings and it is not 80, and PTZ will be used, the PTZ URL must be specified.

For **RTSP**, the above applies and the HMI program will change the RTSP to HTTP for PTZ.

The "**Show video**" button will attempt to show the camera feed.

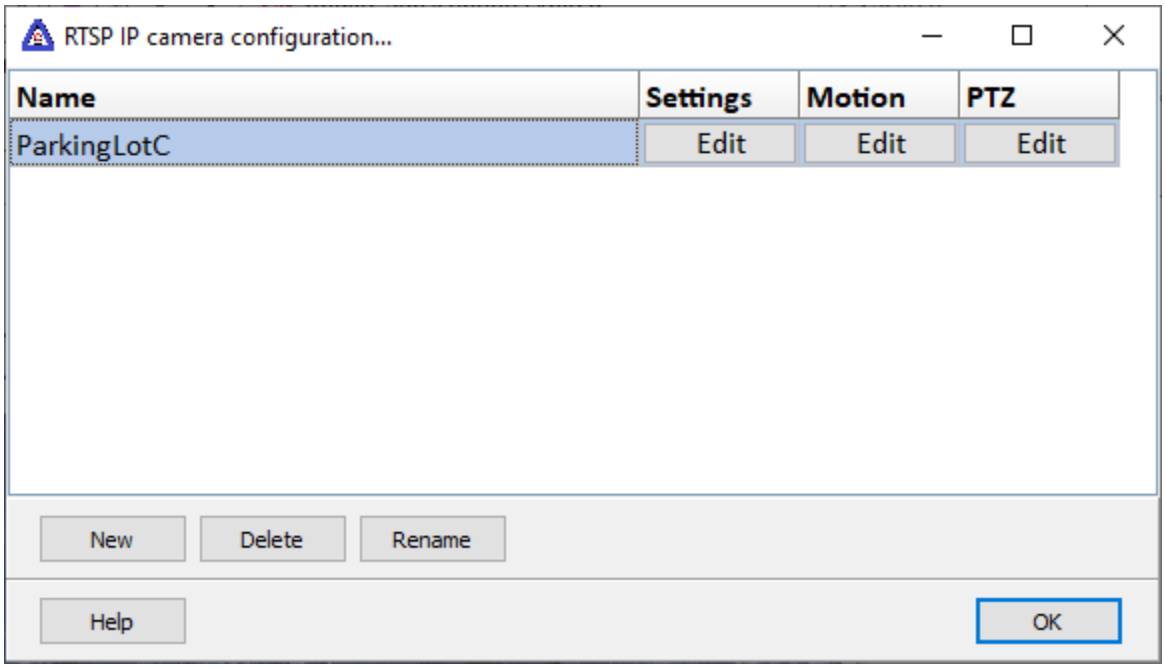
The **buttons** below the video image are to verify PTZ control testing and are enabled after successful ONVIF connection to the camera. Runtime provides finer PTZ control.

Get status Not all cameras support the ONVIF "GetStatus" command. If the "Fetch" command, in scripting, is to be used at runtime, test the "GetStatus" via the button to verify the camera supports the command. Using "Fetch" at runtime with a camera that does not support the operation can cause instability.

Continuous move A camera might respond with no error via "Absolute" and "Relative" commands and not move the camera. The camera might support "Continuous Move" and "Stop". Enable the check box and the double arrow buttons and "Zoom" button will use the "Continuous Move" command for testing. **Note:** The command includes a timeout value and not all cameras implement the timeout. Use the "Stop" button as needed.

RTSP Video

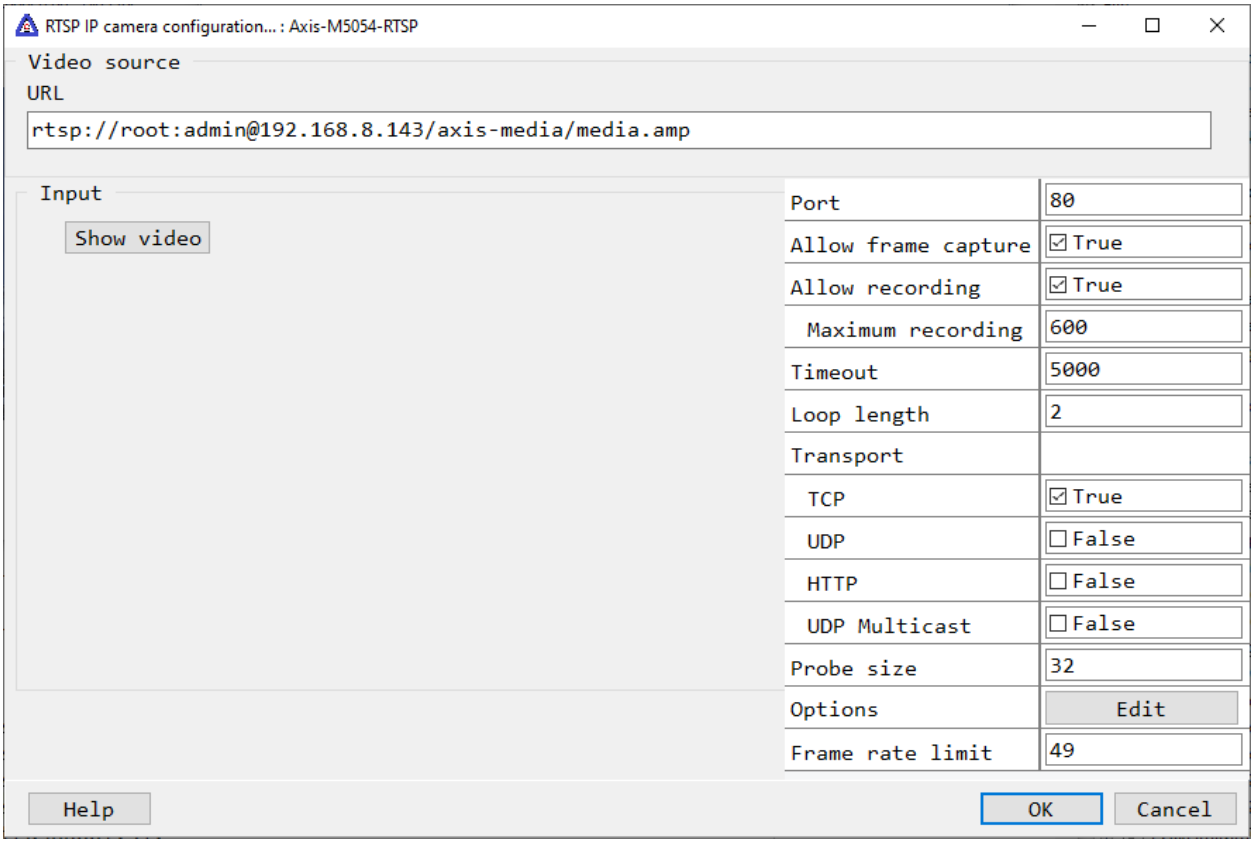
The RTSP protocol is used to access the camera/encoder video feed.



Multiple cameras/encoders can be accessed concurrently.

Note: If a camera image begins to lag "real-time", lower the FPS (frames per second) or resolution in the camera settings or increase the "Frame rate limit" in the HMI or a combination of both.

SETTINGS



Warnings:

1) Recording video can consume large amounts of disk space in a short amount of time.
2) Connecting to an RTSP camera/encoder across the internet can take many seconds, up to a minute.
3) The loss of a camera/encoder video feed, i.e. power failure, router failure, etc., can take several minutes to detect and can take several minutes to recover after the cause of the failure is corrected.

Each camera/encoder must have a unique tagname assigned.

Video Source

URL

The complete path to the camera/encoder including the RTSP command, port number and any other required text, e.g. user name, password, etc.. The order and format of the URL fields is device specific. Refer to the camera/encoder documentation. The examples have the "generally" accepted format and order.

Examples:

MOXA 364A video encoder.
Transport = UDP Multicast
Channel 1, port number 554
rtsp://10.0.0.11:554/multicastStream_ch1

IPC-HFW3200C video camera.
Transport = TCP
A required user name and password, channel 1, port number 554.
rtsp://<user name>:<password>@10.0.5.89:554/cam/realmonitor?channel=1&subtype=0

NCM-620W video camera.
Transport = N/A, TCP or UDP
A required user name and password, channel 11, port number 554. **admin:123456** are the factory default user and password
rtsp://admin:123456@10.0.0.100:554/11

Zavio D3100 video camera.
Transport = N/A, TCP or UDP
The user name and password are disabled, video profile 1, port number 554.
rtsp://10.0.0.14:554/video.pro1

Axis M5054 video camera
Transport = TCP
A required user name and password.
rtsp://root:admin@192.168.8.143/axis-media/media.amp

SCW Admiral video camera
Transport = TCP
A required user name and password.
rtsp://root:admin@192.168.8.143/media/video0 //the digit is the channel, normally 0 or 1

Transport

Select the required transport mode.

Mode
HTTP
TCP
UDP
UDP Multicast

Probe size	Select the probe size. Lower numbers can connect faster.
Options	Low level options using a name/value pair. Name=value Contact support for assistance.
Frame speed limit	This property is used to limit the frequency of drawing the image. High frame rate cameras coupled with large images can cause UI slow down or unresponsiveness. 100 (milliseconds), approximately 10 frames per second, during testing was a good balance between network/CPU/graphics/HMI/etc. loading.
Rotation	The video image can be rotated 0 (none), 90, 180, or 270 degrees.

The remaining settings are below under “Common Settings”.

Motion

The motion settings for the RTSP video feed are the same as the HTTP video feed discussed above.

Input

This provides for testing the camera/encoder settings.

Video for Windows only:
The video size is the size the camera is reporting. The image from the camera is scaled to show in the image area.

Show Video

Select this button to attempt a connection to the camera/encoder and display the camera/encoder image. Select this button to disconnect from the camera/encoder.

Warning: If the video input device is not connected and properly functioning selecting this button may cause the program to respond very slowly. If this occurs press and hold the left mouse button on the "show video" button until the program responds.

Miscellaneous

Via the camera/encoder graphic element configuration the user can be allowed to capture a frame or start/stop recording of the camera/encoder image.

Allow frame capture

This provides for a capture of a frame of the camera/encoder image.

Allow recording

This provides for a capturing of the camera/encoder video to an AVI file at approximately 10 frames per second.

Warning: Recording video can consume large amounts of disk space in a short amount of time.

Maximum

If recording is started the recording will stop:
A. If the window is closed and the recording is via Video for Windows
B. If the user selects to stop recording.
C. The maximum time (in seconds) is reached.

Note: When the size of the file is approximately 1.8 GB the file will be automatically closed and a new file created.

Delay time

This provides for a delay between "Get" request to the camera/encoder. 0 = issue the next "Get" command as soon as the last "Get" command is completed. This can be used to lessen the load on the camera/encoder or network. This attribute applies to camera/encoder types that use the "Get" method to collect images. For the type “K” camera this property is used to limit the frequency of drawing the image. High frame rate cameras coupled with large images can cause the UI slow down or to become unresponsive.

Timeout

This provides for a timer to issue a new collection/connect command after X time has elapsed without a response to a command.

Note: RTSP connections can take many seconds to establish. If watchdogs are occurring, increase this value. A value greater than 5000 milliseconds is recommended.

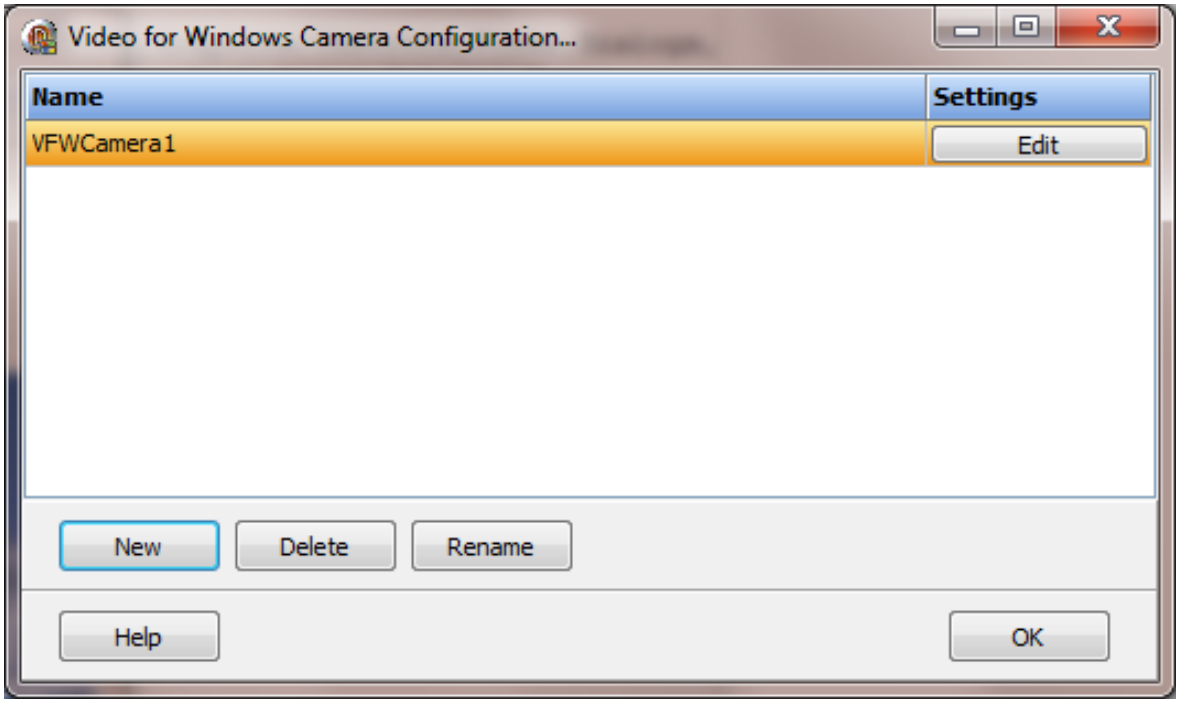
Loop length (IP camera/encoder only)

A zero (0) value disables this feature.

This provides for continuously recording the previous 1 to 20 minutes and via a command saving the loop to a file. This is accomplished by using two files. "A" file is the previous X minute clip and "B" is the current X minute clip. When the command to save the loop is executed the loops are saved to disk using the start time of the loop. The file name format is <port name-mmddyyyy-hhmmss>. Example file name, Camera_1-05312012-221435.avi

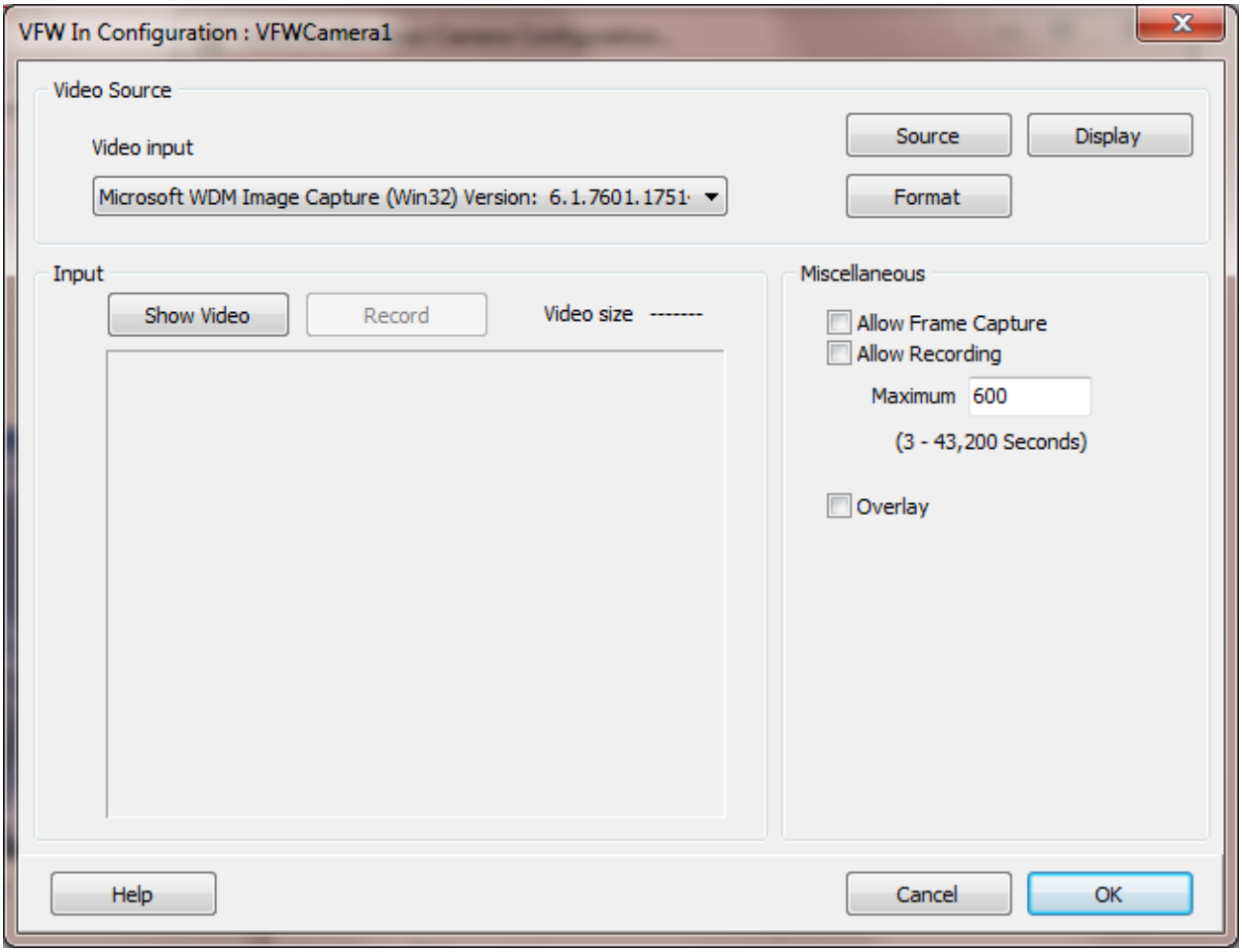
VFW Video

VFW (Video for Windows) is for cameras, generally, that are accessed via USB. While the HMI supports these type cameras they are not recommended. If using one is necessary be sure to read and heed the warnings about using a VFW camera.



The same camera can be referenced more than once but live images can only be accessed in one graphic element at a time. If the same camera is referenced in an open window or windows more than once an error will occur requiring restart of
the computer.

Settings



Video Input

Select the camera source. The selected name is the camera name and the version number.

Source, Display, Format

The camera driver may support additional settings for the camera. If the button is enabled selecting the button will attempt to display the driver dialog.

Overlay

In overlay mode, video is displayed using hardware overlay. When not in overlay, preview mode, frames are transferred from the capture hardware to system memory and then displayed in the capture window using GDI functions. The default and preferred is not in overlay.

The remaining settings are above under “Common Settings”.

Website

The website feature uses HTML5, JavaScript and CSS (cascading style sheets) complied from the user created graphic pages to render graphics in a web browser. A few built-in screens are present for use when needed. Most of the graphics/animations are not separately defined because they operate the same as the native HMI graphics. As is needed the manual will cover a website element in a section near the native element.

New

Directory

This creates a new directory in the project under the “Pages” path. The “Pages” path cannot be deleted or renamed.

Page

This creates a new page (graphic screen) and opens the new page in the graphic editor.

User

This creates a new user profile. The default “user” settings from the website “Settings” are applied when the new user profile is created. Users for the website are not associated with HMI users on the computer executing the HMI.

Settings

Enabled

This checkbox must be checked for the website feature to be active at runtime. The HTTP webserver cannot share the same port number.

Failed communications color

The web page in the browser communicates with the HMI for data values used to display the graphics. The communication to the HMI might be lost for various reasons (e.g. a router loses power). If the loss of communication time is greater than the watchdog timeout setting, the web page will fill with the color selected. The fill will be with 50% opacity. This affects all static graphic elements and some dynamic graphic elements.

Log “Log on” failures

If a “log on” fails, an entry will be added to the event log.

Log on required

If enabled, the user must log on, via the browser.

Maximum clients

This attribute limits the number of socket connections. It is not the number of connected users. A web browser may open several connections to load a page. Depending on the mix of graphic elements on the page, multiple concurrent connections could be required to read/write information from/to the HMI. To limit the connections with just one or two concurrent users set this attribute to 20 or so. 0 = unlimited connections.

Port number

This is the socket port number and is normally 80. Port number 80 is the assigned HTTP port. Port number 8080 is the assigned alternate HTTP port.

Bind IP address

For computers that have more than one NIC (Network Interface Card) the website server can be “bound” to “listen” using a specific NIC. Enter the IP address of the NIC to “bind” the website server. Leave this field blank to not bind the website server to a specific NIC. **Note:** Binding uses the IP address and port number (above).

Show title counter

This attribute is used to show that communication from the browser web page is occurring. An incrementing number 1 -10 will be displayed along with the “+” symbol. The number will be displayed when a read command is sent from the browser to the HMI and the “+” will be added when the HMI responds. This provides that the web page update logic is executing and the HMI is answering request.

Use password settings

This attribute is used to require the password complies with the settings defined in the settings.

Users

Default inactivity timeout (minutes)

This attribute defines how long the user can be inactive before the user is logged out. Default because it is applied when a new user account is created and can be changed for the user. Each user has this setting and it is not connected to other users.

Default refresh rate (milliseconds)

This attribute defines how frequently the browser will issue reads for dynamic data to the HMI. Default because it is applied when a new user account is created and can be changed for the user. Each user has this setting and it is not connected to other users. If "Log on required" is not enabled, this value is applied to all clients.

Default start page

This attribute defines the first page the user will access after log on. Default because it is applied when a new user is created and can be changed for the user. Each user has this setting and it is not connected to other users. If "Log on required" is not enabled, this value is applied to all clients.

Default watchdog timeout

See "Failed communication color" Default because it is applied when a new user is created and can be changed for the user. Each user has this setting and it is not connected to other users. If "Log on required" is not enabled, this value is applied to all clients.

Log user level rejections

Actions on the web pages, (i.e. user actions, press a button, move a slider) have a required user level. If the logged on user level is not high enough, the action will not execute and if this attribute is enabled, an entry will be added to the event log. If "Log on required" is not enabled, this attribute is not applied and user levels are ignored.

Alert user level rejections.

Actions on the web pages, (i.e. user actions, press a button, move a slider) have a required user level. If the logged on user level is not high enough, the action will not execute and if this attribute is not blank, the attribute text will be displayed to the user. If "Log on required" is not enabled, this attribute is not applied and user levels are ignored.

Default video refresh rate

This attribute defines the "poll" frequency for video images from a camera connected to the HMI. It is not the rate from the camera to the HMI. Default because it is applied when a new user is created and can be changed for the user. Each user has this setting and it is not connected to other users. Some user might require a higher value (lower frequency) because of poor connections. If log on is not required this is value is used for all clients. (Higher values equal more time between polls, slower update rate.)

Default video watchdog

Video elements are processed separate from the dynamic graphic data exchange between the browser and HMI. This value defines how long the browser logic is to wait for a video poll request to complete, before the logic considers the last poll a failure and issues a new poll. Default because it is applied when a new user is created and can be changed for the user. Each user has this setting and it is not connected to other users. If "Log on required" is not enabled, this value is applied to all clients.

Configuration engine

When configuring graphic elements that use a browser engine to render the element, Internet Explorer or Microsoft Edge (version 79 or greater) can be used. This property defines the default engine to use for rendering. If the default engine is not installed the program will attempt to use the other engine, if installed.

Note: Microsoft Edge "hardware acceleration", enabled, will cause video elements and perhaps other elements, to not properly render. Disable "hardware acceleration" if rendering issues are present.

JAVASCRIPT

The website feature builds web pages dynamically when the page is requested from a web browser. All web page graphic element names (identifiers), used for code to interact with the graphic elements, are dynamically created, and a "names" list is maintained in the HMI website server logic until the page is closed.

Adding Javascript code to the page is possible using the HTML (Advanced) dialog in the graphic editor. The code is not able, generally, to interact the user.

The Javascript "mouse command" provides a programming structure to allow user mouse clicks, on the graphic elements of the web page, to execute Javascript code. The dynamic nature of the web pages requires some "helper" functions to allow user code to access the graphic elements.

The web page has two element "classes".
Elements that are native to HTML (buttons, sliders, checkbox, etc.).
Elements that are not part of HTML and are rendered on a "canvas" (circles, rectangles, text, etc.).

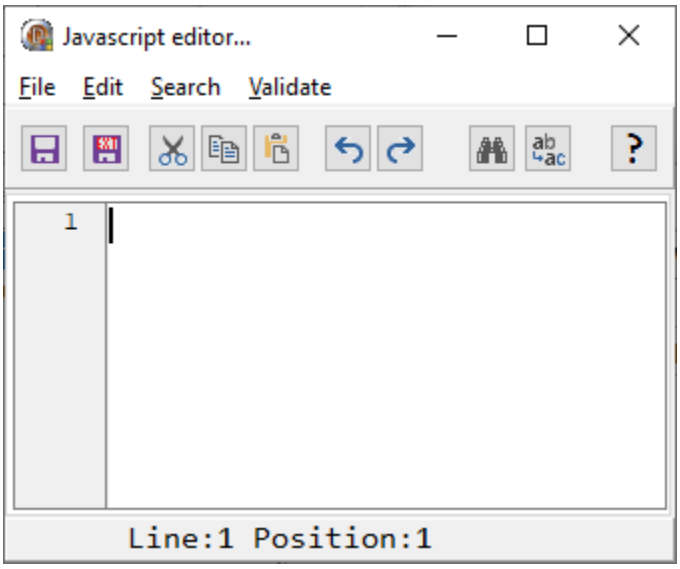
Both classes can respond to mouse clicks.

Each graphic element has a dynamically assigned name (ID-identifier).
To execute Javascript code, on a mouse click, two items must be defined.
First, a unique (ID-identifier) assigned by the user, the “ES_ID”. The name must be unique for the page and is assigned in the mouse commands dialog.

Parameter	Value
Code	
ES_ID	

The second item, some Javascript code. Selecting the button in the “Code/Value” field will launch the Javascript editor.

Parameter	Value
Code	...
ES_ID	



The format of the “OnClick” event is “<ES_ID>OnClick.”

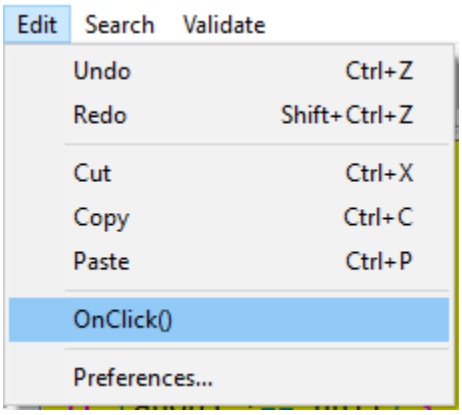
For example, an ES_ID:

ES_ID	HMI_Active_Btn

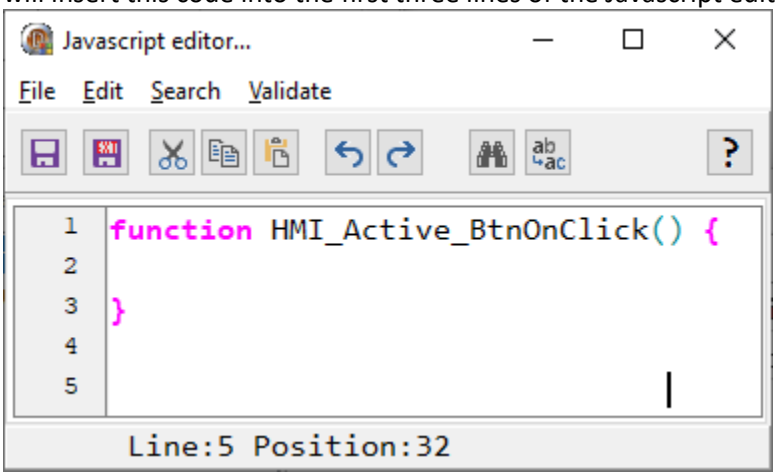
The “OnClick: event format would be:

```
function HMI_Active_BtnOnClick()
```

Selecting this menu item:



will insert this code into the first three lines of the Javascript editor:



Gaining access to an HTML element, example:

```
btnObj = ES_IDToHTMLObj("HMI_Active_Btn");//ES_ID
     //user ID -> object ID -> to HTML obj

if (btnObj!== null){
} //always check the object was found
```

Example: change the caption of the button when the mouse is clicked on the HTML button.

```
function HMI_Active_BtnOnClick() {
 btnObj =ES_IDToHTMLObj("HMI_Active_Btn");
 if (btnObj !== null) {
  if (btnObj.value == "Active"){
   btnObj.value = "Paused";}
  else {
   btnObj.value = "Active";}
  }
}
```

Gaining access to a canvas element, example:

```
anObj=ES_IDToCanvasObj("Cir_1"); //ES_ID
//user ID -> object ID -> to canvas obj

if (anObj !== null){
} //always check the object was found
```

Example: change the color of the circle when the mouse is clicked on the canvas graphic element.

```
function Cir_1OnClick() {
 anObj=ES_IDToCanvasObj("Cir_1");
 if (anObj !== null) {
  if (anObj.fillStyle == "#0000FF") { //blue
   anObj.fillStyle = "#006300"; }      //green
  else {
   anObj.fillStyle = "#0000FF"; }      //blue
  }
}
```

Note: The canvas refresh rate is set here. Altering a canvas graphic element, the visual change of the element, will not be seen until the next update call.

This line of code (if needed):

```
$("canvas.mainC").drawLayers();
```

can be added after the new element value(s)s has been set to force a canvas update.

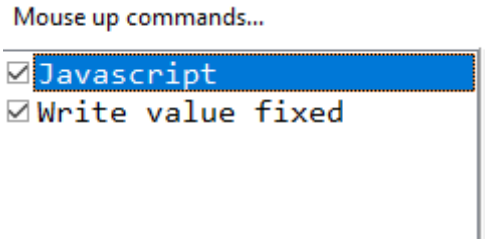
Example:

```
function Cir_1OnClick() {
 anObj=ES_IDToCanvasObj("Cir_1");
 if (anObj !== null) {
  if (anObj.fillStyle == "#0000FF") { //blue
   anObj.fillStyle = "#006300"; }      //green
  else {
   anObj.fillStyle = "#0000FF"; }      //blue
  }
 $("canvas.mainC").drawLayers();
}
```

Advanced chain button click calling.

Each graphic element, HTML and canvas, provides one mouse click event to be configured and called, when the mouse is clicked on the element.

Example:



The mouse up command is configured for Javascript code execution (regardless of command order in the list) and to write a “fixed value” to the website server. The graphic element’s “OnClick” event will call the Javascript code. To call all other configured commands from the Javascript code the graphic elements default “OnClick” event must be called.

Unless specified below, all “OnClick” functions use this format:

The format of the command is “pHMIg.onButtonClick(<anObj>);
The “pHMIg.onButtonClick” event handler requires an object with “id” field. The “id” field is the dynamic id for the graphic element.

The ES_ID is used to fetch the dynamic “id” using the “ES_IDToObjID(ES_ID)” function. The function returns “-1” if the ES_ID was not found.

If the graphic element object was not fetched earlier in the script, here is an example to fetch the id from the ES_ID and call the default mouse click handler.

Example:

```
 const anObj = {id:""};     //create the object
 anObj.id = ES_IDToObjID("HMI_Active_Btn");
 if (anObj.id !== "-1") { //successful?
  pHMIg.onButtonClick(anObj);
  } //end -1 check
```

Complete example:

```
function HMI_Active_BtnOnClick() {
 btnObj = ES_IDToHTMLObj("HMI_Active_Btn");
 if (btnObj !== null) {
  if (btnObj.value == "Active"){
   btnObj.value = "Paused";}
  else {
   btnObj.value = "Active";}

  const anObj = {id:""};   //create the object
  anObj.id = ES_IDToObjID("HMI_Active_Btn"); //assign
  if (anObj.id !== "-1") { //success?
   pHMIg.onButtonClick(anObj);
   }    //end of !== -1 check
 }      //end of null check
}
```

If the graphic element object was fetched earlier in the script, pass the object fetched earlier:

```
pHMIg.onButtonClick(btnObj);
```

Complete example:

```
function HMI_Active_BtnOnClick() {
 btnObj = ES_IDToHTMLObj("HMI_Active_Btn");
 if (btnObj !== null) {
  if (btnObj.value == "Active"){
   btnObj.value = "Paused";}
  else {
   btnObj.value = "Active";}

  pHMIg.onButtonClick(btnObj);
 }    //end of null check
}
```

Analog/digital grids

The format of the command is “pHMIg. digitalGridCellClick (<anObj>);

To pass the click to the default handler for the row mouse commands use:

```
//Note: inObj is used in this example and not the next example
function AN_1OnClick(inObj) {
 if (inObj!== null) {
  pHMIg.digitalGridCellClick(inObj);}
}
```

If the row value needs to be changed, for example to call a row’s mouse command(s), other than the row the mouse clicked occurred.

The “pHMIg. digitalGridCellClick” event handler requires an object with two properties/fields:

1. id
2. rowIndex

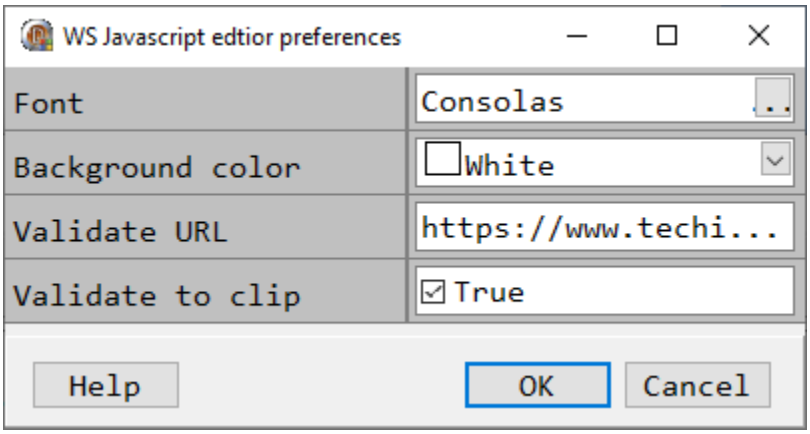
The “id” field is the dynamic id for the graphic element.
The “rowIndex” field is the row index of the field clicked.

```
const anObj = {id: "",
          parentNode:{rowIndex: "",
           parentNode:{
            parentNode: {id:""}}}};
```

This passes the grid ID and row ”0” to the server, via the client code.

```
function AN_1OnClick() {
  const anObj = {id: "",
    parentNode: {
      rowIndex: "",
      parentNode: {
        parentNode: {
          id: ""}}}};
  anObj.parentNode.parentNode.parentNode.id =
ES_IDToObjID("AN_1");
  if (anObj.parentNode.parentNode.parentNode.id !== "-1") {
    anObj.parentNode.rowIndex = 0; //change this value as needed
    pHMIg.digitalGridCellClick(anObj);
  }
}
```

JAVASCRIPT EDITOR PREFERENCES

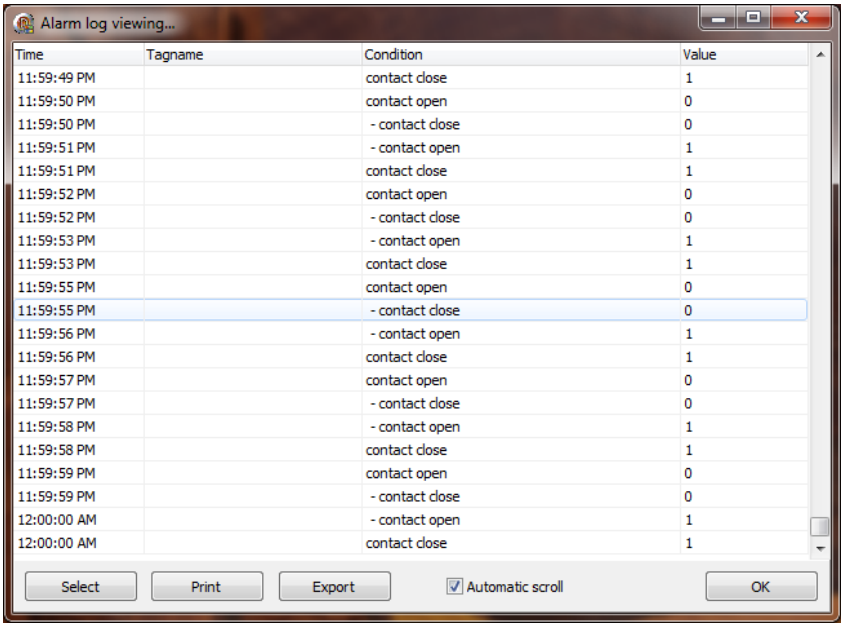


Font	Text appearance in the text memo field.
Background color	The background color of the text memo field.
Validate URL	 A method to validate the Javascript syntax. Using a web based validator, select the icon and the default browser will open the URL supplied.
Validate to clip	When the icon is selected, the contents of the editor are placed on the clipboard before the URL is launched in the default browser.

Logs

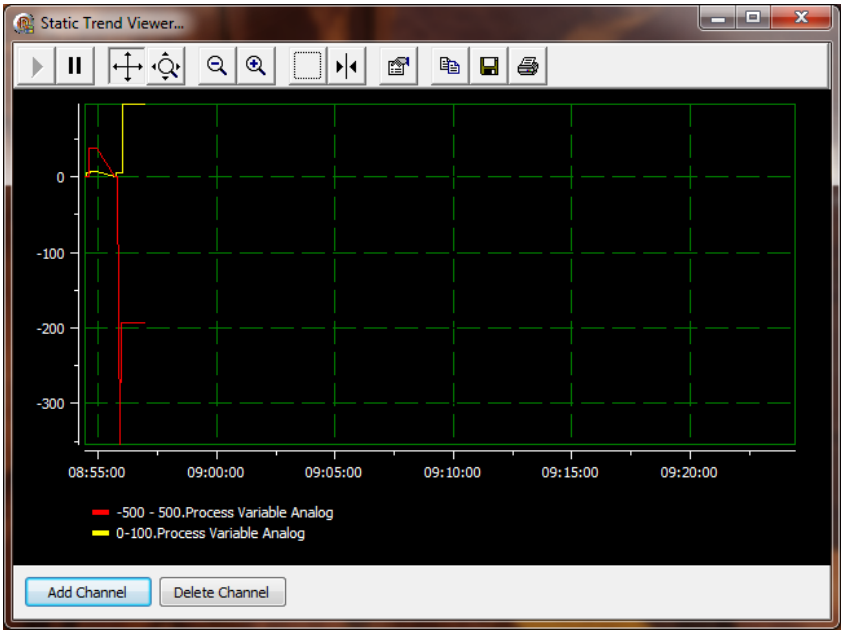
The “Logs” menu provides a method to view, print and export logs.

Alarm logs

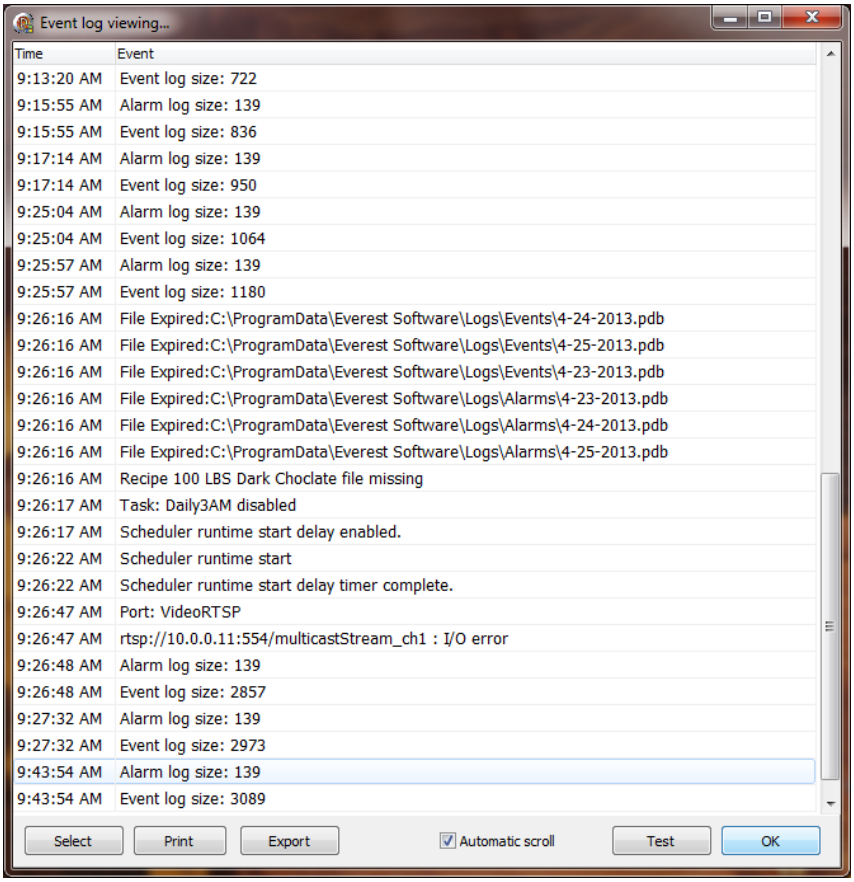


Time	Tagname	Condition	Value
11:59:49 PM		contact close	1
11:59:50 PM		contact open	0
11:59:50 PM		- contact close	0
11:59:51 PM		- contact open	1
11:59:51 PM		contact close	1
11:59:52 PM		contact open	0
11:59:52 PM		- contact close	0
11:59:53 PM		- contact open	1
11:59:53 PM		contact close	1
11:59:55 PM		contact open	0
11:59:55 PM		- contact close	0
11:59:56 PM		- contact open	1
11:59:56 PM		contact close	1
11:59:57 PM		contact open	0
11:59:57 PM		- contact close	0
11:59:58 PM		- contact open	1
11:59:58 PM		contact close	1
11:59:59 PM		contact open	0
11:59:59 PM		- contact close	0
12:00:00 AM		- contact open	1
12:00:00 AM		contact close	1

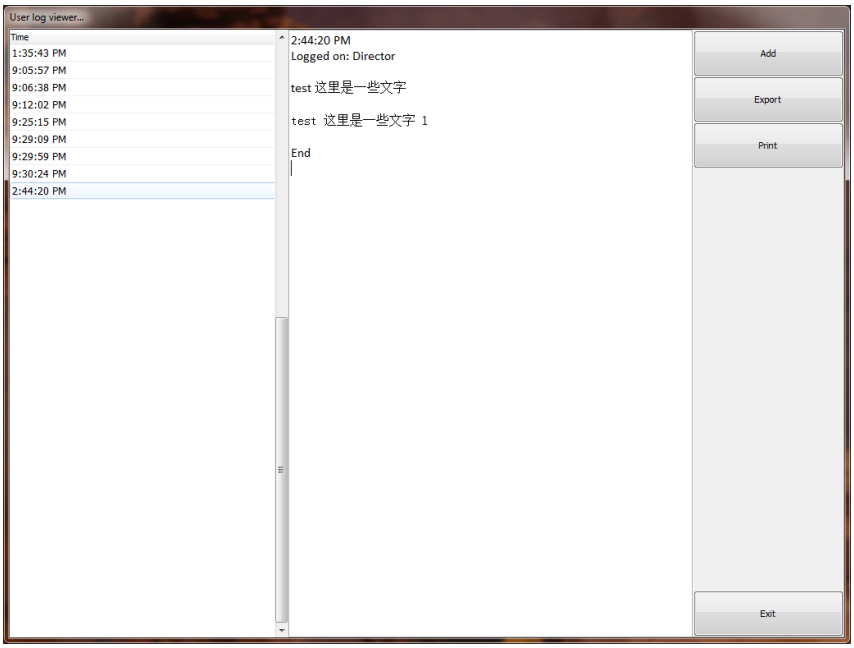
Data logs



EVENT LOGS

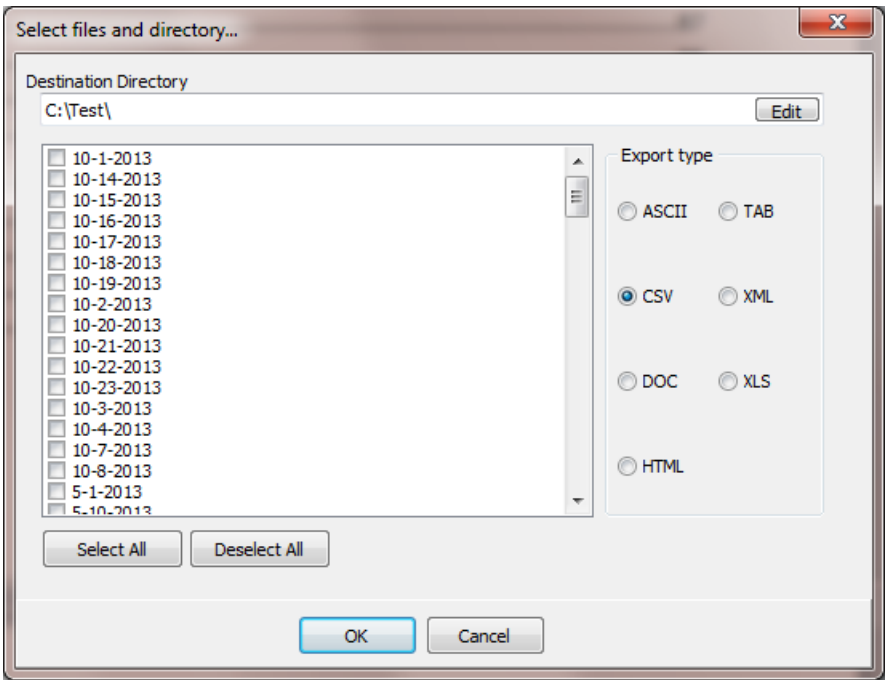


USER LOG



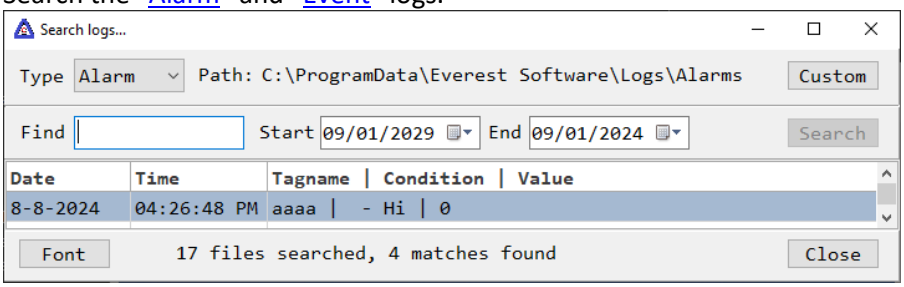
Export

All of the logs, except the user log, can be exported to ASCII, tab delimited, comma separated values, XML (Extensible Markup Language), DOC (MS Word), XLS (MS Excel) or HTML (Hypertext Markup Language). The user log can be exported to a text file.



Search

Search the "Alarm" and "Event" logs.



Type	The type of files in the path, "Alarm" or "Event" logs.
Path	The full path to the files to search. When the "Type" is selected, the path will be set to the configured path for the project.
Custom	Use this button to select a specific path.

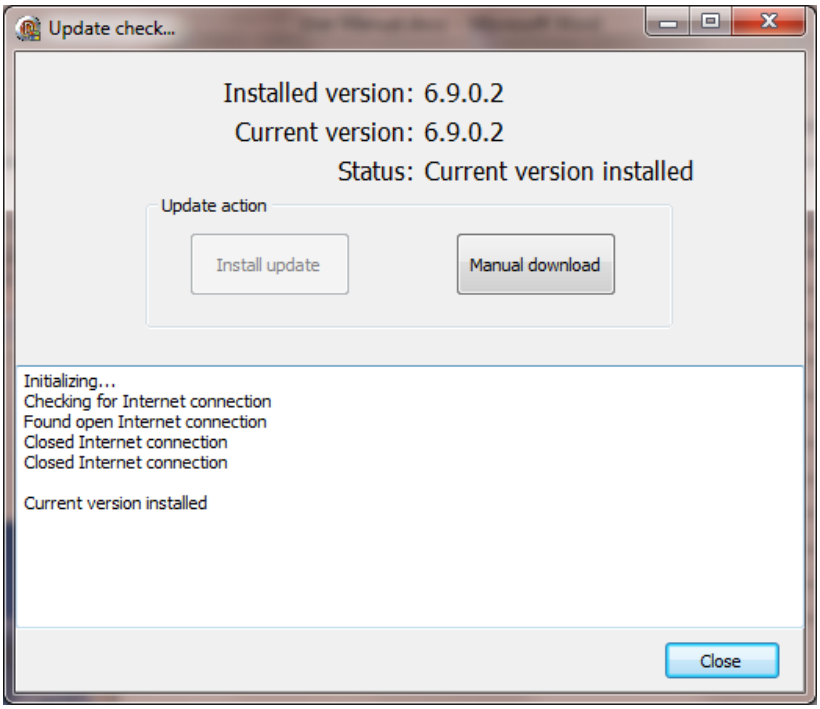
Find	The word(s) to find in the selected files. The file(s) is searched for each word, separated by a space character.
Start date End date	Used to limit the search to a range of days. **Note:** Set the "Start Date" **after** the "End Date" and all the files in the "Path" will be searched. Start date: 08/29/2099 End date: 08/29/2024
Font	Select the font settings for the search results grid.
Note:	Double click the left mouse button, on a line in the search results grid, and the alarm/event log file, for the date, on the line, will be opened in another window.

Help

The help menu contains several items.
F1 is the help command. The help system is this user manual and context sensitive is implemented. For example, the graphics help file is available when the graphic editor is open.

Check for update

A connection to the internet is required for this feature to function. The program will connect to the internet and attempt to fetch the version number of the latest HMI software release.



The installed software version and the current released version will be displayed. The “Install update” button will be enabled if a newer version is available. Selecting the button will download the version and via prompting, the latest version can be installed on the computer.

The “Manual download” button is used to manually download the installer to the computer. This would be used, for example, if the HMI software needed to be installed on another computer.

About

The about dialog will be displayed. It will display the software version, license date, licensee, key ID, type of license, website, support email address and copyright notice.

Web license

A web license can be moved from one computer to another computer. The program, "WLRegDereg.exe" located in "<installed path>\Tools\" can be used to deregister the license on the computer. Then run the normal installer on the target computer and follow the prompts. **Note:** There is a limit on the number of times the license can be transferred. Contact support if help is required.

Points

A point is a data object. Via configuration, it contains all the settings required to connect to a "port" (See the section on communications).

A point can be configured to provide alarming, publish the point's properties to the built in OPC server, scale values, etc..

The HMI has six point types.

Analog and Analog Host

Analog has a connection to an external device via one of the ports and Analog host is a point that is internal to the HMI.

Digital and Digital Host

Digital has a connection to an external device via one of the ports and Digital host is a point that is internal to the HMI.

Analog and Digital Host pointer

These two point types are provided to "point" at other points. It is provided to allow "indexing" or "arraying" of points.

The nature of two protocols requires a modifier to the six point types. The modifier is required to allow easier configuration of the source address (see source address below).

Generic

This is the default modifier applied to all points that are not OPC or Bacnet/IP.

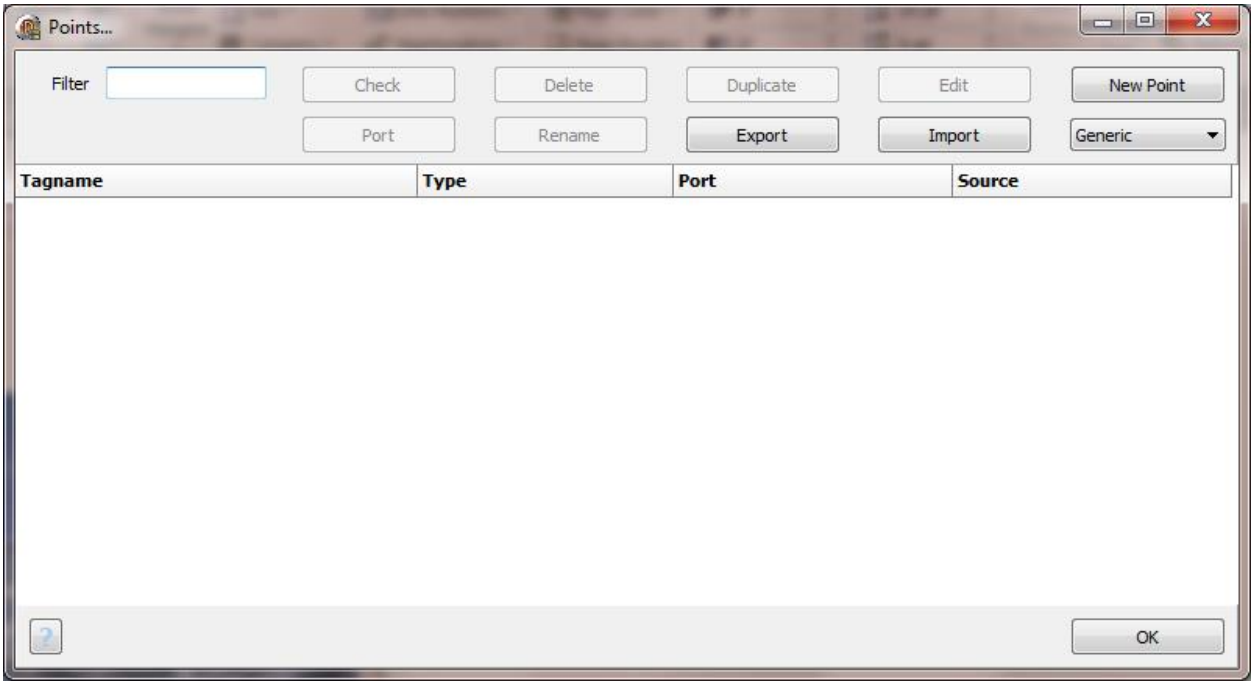
OPC

If the port type is "OPC client" the OPC modifier must be applied.

Bacnet/IP

If the port type is Bacnet/IP the "Bacnet" modifier must be applied.

CONFIGURATION



Filter — Enter characters in this field to show only those tagnames matching the filter string.

Delete — The selected point will be deleted.

Duplicate — Select a point(s) to duplicate. A window will appear to allow the duplicated point tagname and source to be modified. (See below)

Edit — The selected point edit mode will begin if selected.

Export — Export the point configuration to a file.

Import — Import the point configuration from a file.

New Point — Below the new point button is a drop down list. In the list are three point "modifiers".

- Generic — Most source port kinds are generic (MODBUS, DF1, etc.).
- OPC — Select this kind if the source port is an OPC server.
- Bacnet — Select this kind if the source port is a Bacnet/IP device.

Port	Use this command to change the port assignment for the selected port. This is provided to allow the change the port assignment for number of points with one action. (See below)
Rename	Select a point(s) to rename. A window will appear to allow renaming of the selected points. (See below)

POINT DEFINITIONS

The characteristic or attributes that are common to all point types are defined below.
Following that section are characteristics/attributes that are defined for at least two types of data points. Following that section are all the characteristics/attributes unique to a type of data point.

Common
Common 2
Analog
Digital
Analog host
Digital host
Analog host pointer
Digital host pointer
Alarm settings

COMMON

Tagname

All points must have a tagname that is unique for the project. Tagnames are case sensitive. Example: *AS-101-A* and *AS-101-a* are different and both are legal tagnames.

Tagnames can be up to 255 characters.

A tagname cannot begin with a <space> or <tab> character.

The following characters are not allowed in tagnames.

< > | "(double quote) \ / : * ? ~ ^

A tagname can contain a comma (,). If point configurations are exported or imported the comma may interfere with exporting and importing of points. Use a period (.) in place of a comma.

Notes:

1) If a point is published for the OPC server the "." (period) character is used by OPC devices to delimit items. While "." is a legal character in a tagname and can be used, as it is in some OPC client tagnames, be aware that it does, in some cases, have a special meaning.

2) Tagnames to be used to issue commands from a web browser cannot contain symbols that are reserved. Tagnames that are only monitored do not have this limitation. Only tagnames coming to the HTTP web server from the browser have this limitation.
Spaces are not allowed. The browser will convert a "space" to a "+".

In addition, special characters or 8-bit values are converted into their hexadecimal equivalents and prefaced with a percent sign (%).

For example"<" is converted to %3C,">" is converted to %3E.

Description

All points can have a description.

COMMON 2

Access rights

Read	Runtime will read the value from the source in the port configured.
Read/write	Same as read and writes are permitted. For writes, the internal value is changed followed by a write to the configured port.
Write	The point is "write" only. Commands to set the value will be sent to the source in the port configured.

Note:
AB Logix and AB Micro 8xx points do not support write only access. Points that need to allow write commands to the controller must have read/write access.

Port

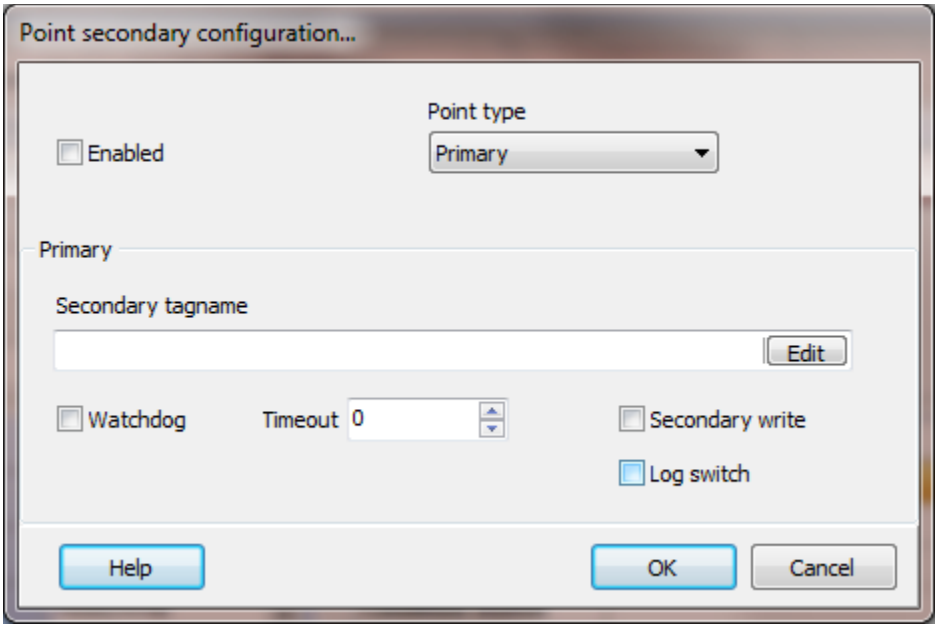
This will be one of the configured ports. E.g. MODBUS Master, OPC Client, AB Logix, GE SNP-X.

Source

This is the source address in the port. It is a reference to the data in the device defined in the port configuration. e.g. 400003, 400012/15, ES-100.PV, Q8

AB DF1 PCCC PLC5/SLC/Micrologix/SLC 5/05
AB Logix
AB Micro 8xx
Bacnet/IP
Delta Motion Control
DNP Serial
DNP TCP
Enron
Fatek
GE SNP-X
GE SRTP
HMI <-> HMI
IDEC (OpenNet)
Keyence
K-Sequence (DirectLogic)
Master-K
Mewtocol (Panasonic-Matsushita-Aromat-NAIS)
Mitsubishi FX 0/1, FX 2/3
Mitsubishi Q
MODBUS
MODBUS RS-485
MODBUS TCP SS
MODBUS TCP 2 SS
MODBUS Slave
Omni
Omron Sysmac
Omron FINS (RS-232)
Omron FINS TCP/UDP
Saia
Siemens S7-200
Siemens S7-300/400 Serial
Siemens S7-300/400/1200 TCP
SNMP
Toshiba
USB Digital I/O 15
WeMo

Secondary



A point can be configured for a secondary source of data.
Example:
A PLC has a MODBUS TCP (Ethernet) connection and also a MODBUS RTU (RS-232, serial) connection.
A primary port (A) is configured for the TCP connection and a secondary port (B) is configured for the RTU connection.
Points (PP) are created using the primary port (A) as a source.
Points (SP) are created using the secondary port (B) as a source.
All the primary points (PP) are configured to access the appropriate secondary point (SP).
Graphics, logging, alarms are all configured for the primary points (PP).
The secondary points (SP) do not process alarms and should not be configured for other access.
The secondary points (SP) are <u>only</u> used in the primary point (PP) configuration.
Should the primary port (A) experience a communication failure the data collected by the secondary points (SP) is used as the data for the primary points (PP) until the primary port (A) communication failure is corrected.

Enabled

Enables the secondary logic to function.

Point type

The point is a primary or secondary, if enabled.

Secondary tagname

The tagname of the secondary point for the primary point. Secondary points are only enabled. The primary points provide all configuration for the failover logic.

Timeout(seconds)

Some port types are not polled. The protocol uses exception reporting to update subscribers. If a port is polled, this property configures the logic to use the secondary source if the primary has a watchdog timeout (communication failure).

Secondary write

When the point is using the secondary point as a source of data and a write command is issued to the primary point, the write is redirected to the secondary point. The secondary source could be a read only connection. If writes to the secondary are not allowed, enable this property.

Log switch

Each time a primary point begins or ends using a secondary point as a data source, if this property is enabled, an event will be placed in the event log.

Engineering Units

Example: GPM, PSI

High EU (Engineering Units)

The highest value of the point.

Low EU (Engineering Units)

The lowest value of the point.

Normal Condition Status

If an alarm condition is enabled and configured to "log" and an alarm condition returns to a normal condition, this value is prefixed in the alarm log to the alarm condition status.

Initial value

The points “current value” is set to this attributes value at runtime start. For analog points, if the field is blank, the initial value is 0 (zero). This does not issue a write command to an external device.

OPC Published

All points are hidden from the OPC server. This instructs the OPC server to publish this point.

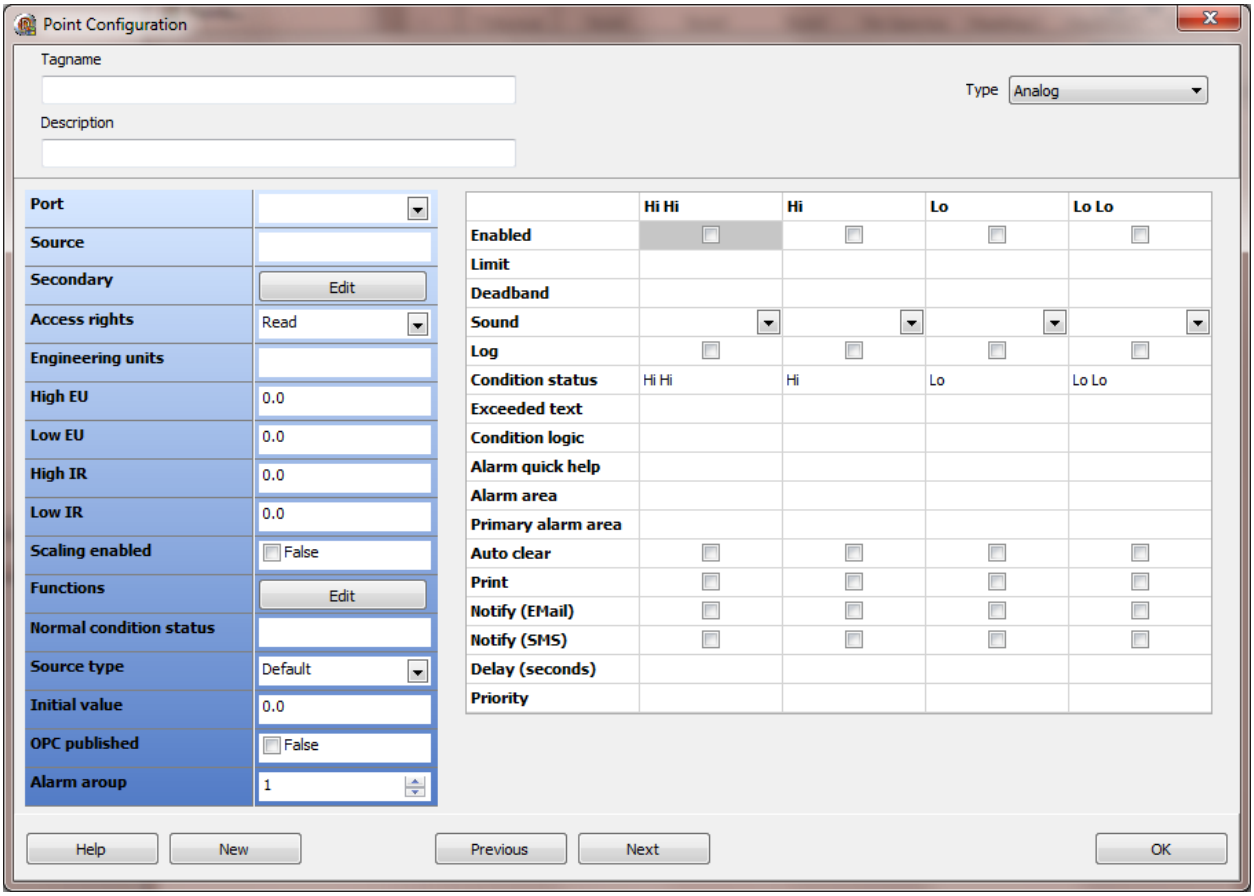
Script

The host analog and host digital points can contain script commands. These scripts are executed continuously.

Alarm Group

The alarms for this point are a member of this numbered alarm group.

ANALOG



The left panel is addressing, scaling, publishing, etc.. The right panel is the alarm panel. The alarm panel is described under "Alarm settings" below.

Note: Analog points only contain numbers. An analog point cannot contain or process a string. Use a script global for string processing.

High IR (Instrument Range)

The highest instrument value of the point. This is used if scaling is enabled. Example: The engineering range is 0-100 gallons. The instrument range is 0-4095. If scaling is enabled and the value from the source is 2048 the value of the point would be 50.

Count	Gallons
0	0
1024	25
2048	50
3072	75
4095	100

If scaling is not enabled, High and Low IR are ignored.

Low IR (Instrument Range)

The lowest instrument value of the point. See High IR.

Scaling enabled

Used to scale instrument value to engineering value. See High IR.

Functions

The "port" notifies the point when the value of the "source address" is returned from the external device. The point applies scaling, if enabled, to the raw (incoming) value and updates the point's current value. If a function is configured, the function is applied to the current value. The current value is used by the alarm logic and all other features of the HMI. See "Analog functions."

Source type

Unless specified by selecting one of the following options, the source data format is assumed to be a 16 bit integer. In most applications, it is correct. Not all port types support all data types.

AB LOGIX
AB Micro 8xx
BACNET/IP
BCD2
Delta Motion Control (DMCP)
Enron
MODBUS
ODBC In
Omni
Siemens S7-200/300/400/1200

MODBUS

By default, for MODBUS master ports, the source address for an analog point is a two byte register that is a word (0 .. 65,535).

If "Float" is selected the source address is considered the first register of the floating point data. Example: If the source address is 400100, the complete source is register 400100 and 400101. The float byte order is set in the port configuration.

If "Integer" is selected the source address is considered the first register of the integer point data and the range is -2,147,483,648 ... 2,147,483,647. Example: If the source address is 400100, the complete source is register 400100 and 400101. The integer byte order is set in the port configuration.

If "Small integer" is selected the range is -32,768 .. 32,767.

If "Unsigned integer" is selected the source address is considered the first register of the integer point data and the range is 0 .. 4,294,967,295. Example: If the source address is 400100, the complete source is registers 400100 and 400101. The integer byte order is set in the port configuration.

LOGIX, BACNET/IP

Select "Float" to allow for collection of floating point data from a controller/device for the point. If this attribute is not set to "Float" and the source address is a floating point reference, the data from the controller will not be valid.

If the source address referenced is a byte, small integer or integer, do not enable this attribute.

The 32-bit floating point values are assumed to be IEEE-754 standard format.

If "Signed integer 64" is selected the range: -2^{63} to 2^{63} -1

If "Unsigned integer 64" is selected the range: 0 to $2^{64} - 1$

Note: The “signed integer 64” and “unsigned integer 64” do not support scaling.

SIEMENS S7-200/300/400/1200

If the address is a floating point register or a double word register select the appropriate value. If the analog value is a byte or two byte word the program will select the correct data decoding based on the source address. For floating point references use the "D" address parameter and select "Float" Example: VD60 with "float" selected would instruct the program to read the four bytes as a floating point value. VD60 with "Integer" selected would instruct the program to read the four bytes as a double word value.

ODBC In

This does not apply to ODBC. The value is assumed to be a float.

BCD 2

For numeric use Automation Direct defaults to two byte BCD (Binary coded decimal).
K-Sequence, Omron and MODBUS are the only ports using this data type.
If the application requires BCD support, please contact support.
Note: We could not verify negative BCD operations.

Enron, Omni

Enron, Omni have 3 analog data types. 16 bit and 32 bit signed integers are supported by default; the type selection is not used. Float is the third date type and the type selection must be set to float for correct operation.

Delta Motion Control (DMCP)

DMCP has 3 analog data types.
1) 32 bit signed integers are supported by default; the type selection is not used.
2) 32 bit unsigned integers, select "Unsigned integer".
3) 32 bit float select "Float".

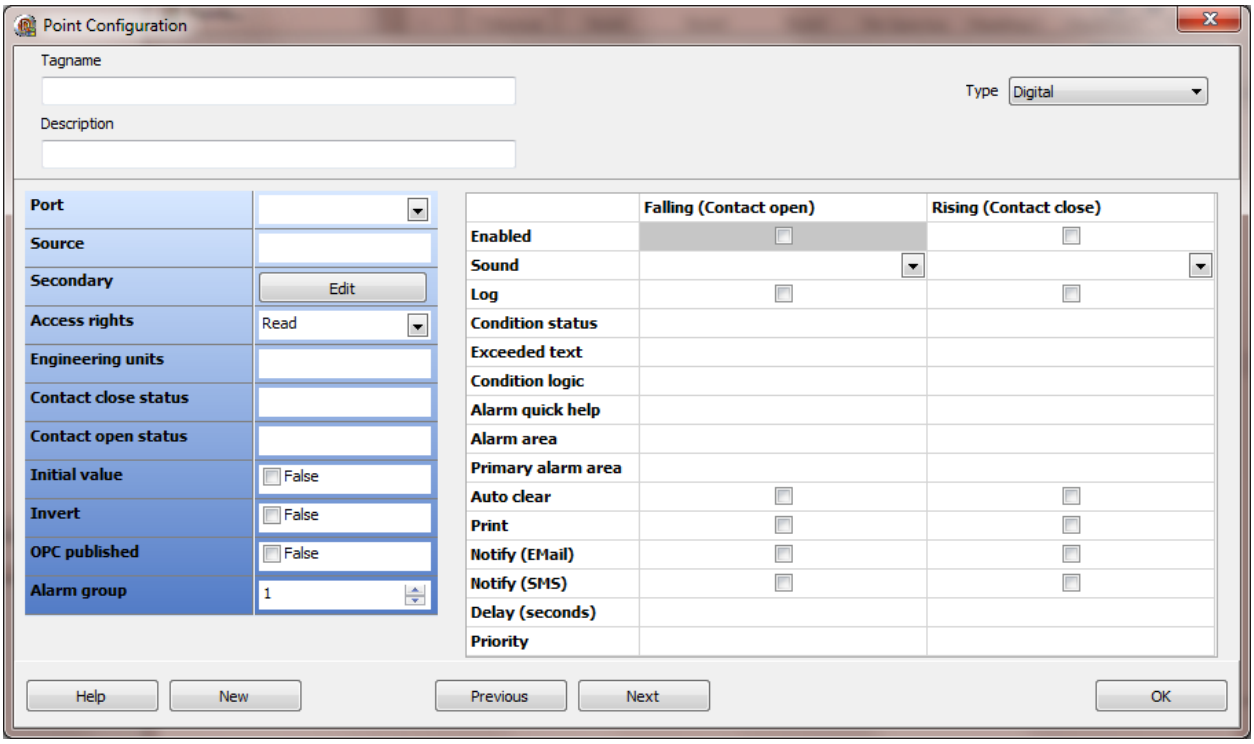
AB Micro 8xx

Micro 8xx type	Description	HMI source type	Range
SINT	Signed 8-bit value	Default	-128 to +127
INT	Signed 16-bit value	Small integer	-32768 to 32767
DINT	Signed 32-bit value	Integer	-2147483648 to +2147483647
LINT	Signed 64-bit value *1	Signed integer 64	-9223372036854775808 to 9223372036854775807
USINT/BYTE	Unsigned 8-bit value	Default	0-255
UINT/WORD	Unsigned 16-bit value	Default	0-65535
UDINT/DWORD	Unsigned 32-bit value	Unsigned integer	0 to 4294967295
ULINT/LWORD	Unsigned 64-bit integer value *1	Unsigned integer 64	0 to 18446744073709551615
REAL	32-bit IEEE Floating point	Float	-3.40282E+38 to 3.40282E+38 *2
LREAL	64-bit IEEE Floating point	Float	-1.7976931348623E+308 to 1.7976931348623E+308 *2,3
String	Character string (1 byte per character)	Support at port level	Maximum 252 characters

Notes:

1) The “signed integer 64” and “unsigned integer 64” do not support scaling.
2) Refer to the device documentation for actual online/offline ranges.
3) Read only.

DIGITAL



The left panel is for addressing, publishing, etc. The right panel is the alarm panel. The alarm panel is described under "Alarm settings" below.

Contact Close Status

Text to describe the meaning of the input when it is closed (true).

Contact Open Status

Text to describe the meaning of the input when it is open (false).

Initial Value

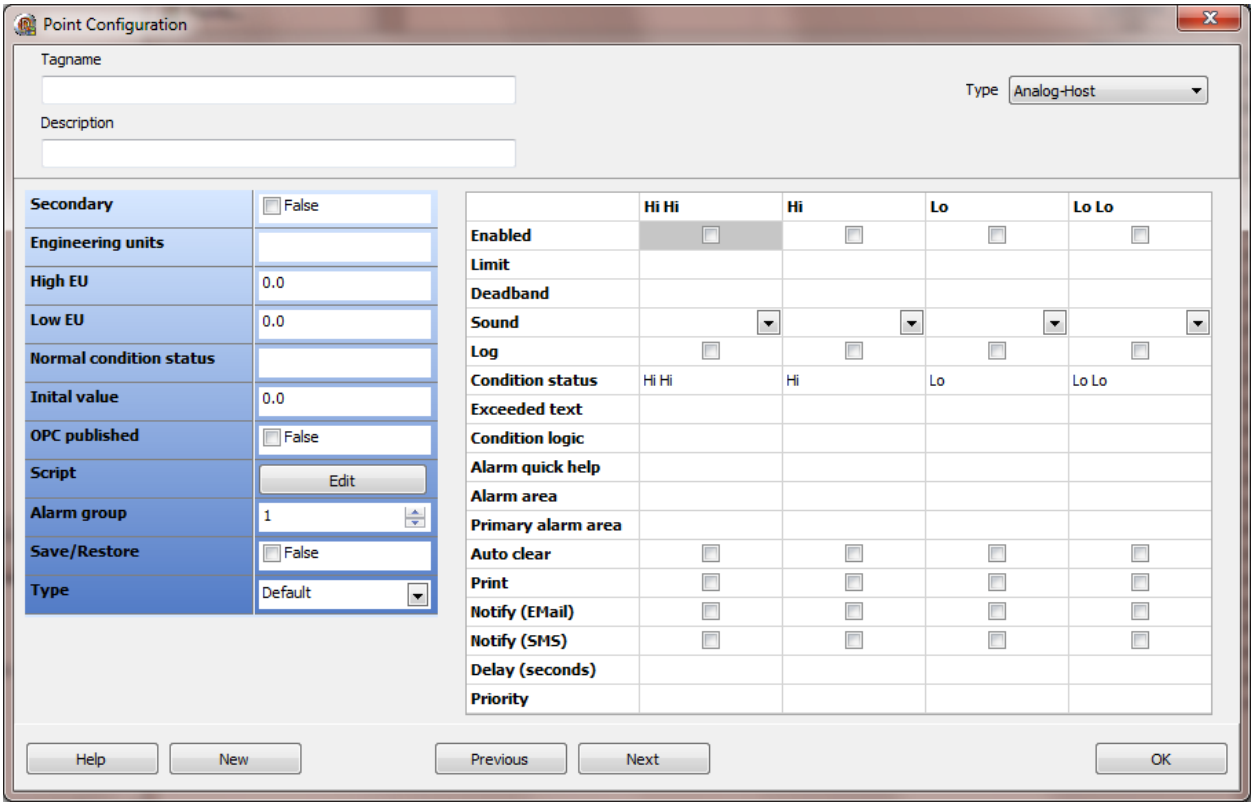
The alarms are transition triggered. i.e. The value was true and is now false or the value was false and is now true. This provides for the value, at runtime start, to be set true so it is not required for the value to transition true and then transition false to activate an alarm on a falling trigger, if enabled. If enabled and when the first data for the point is received if the new value is false the falling trigger alarm will be activated.

Invert

When the data value is collected from the external device the input is inverted.
If the tag is "write only" the data value is transmitted to the external device without change.
If the tag is "read/write" the data value is inverted and transmitted to the external device.

Analog Host

This point is similar to the analog point. It is internal to the HMI program. The type is “Float”. The main difference is the “Analog Host” point can contain a script. See the section on scripting for more information on point scripts.



Secondary

An analog host point can be a secondary for an analog point.

Save/Restore

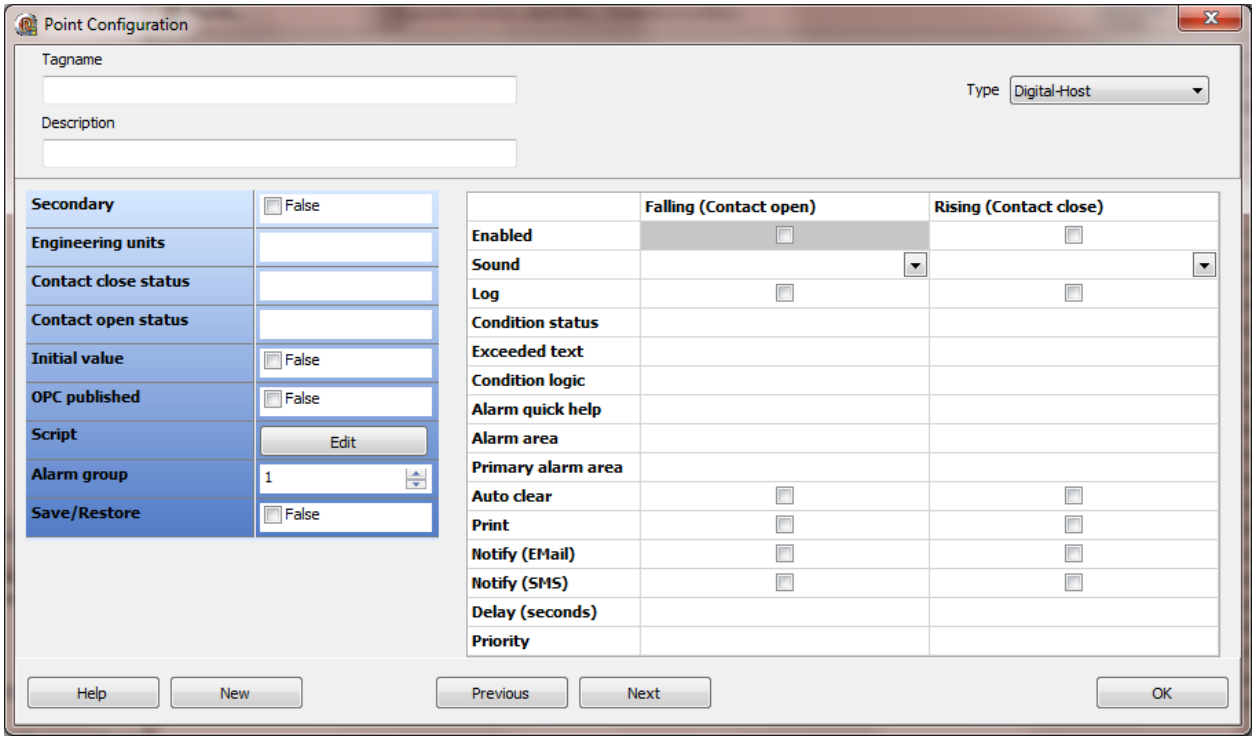
If enabled, when runtime monitoring is stopped (via a normal shutdown) the current value will be saved to disk and when runtime monitoring is restarted the value will be read from disk and written to the current value. This value is applied after the initial value attribute, the saved value will overwrite any initial value and before any override file setting.

Type

In most uses the default type will be correct. Signed/unsigned integers 64 are really the only other type that needs to be specified. BCD is not support for this point type.

Digital Host

This point is similar to the digital point. It is internal to the HMI program. The main difference is the "Digital Host" point can contain a script. See the section on scripting for more information on point scripts.



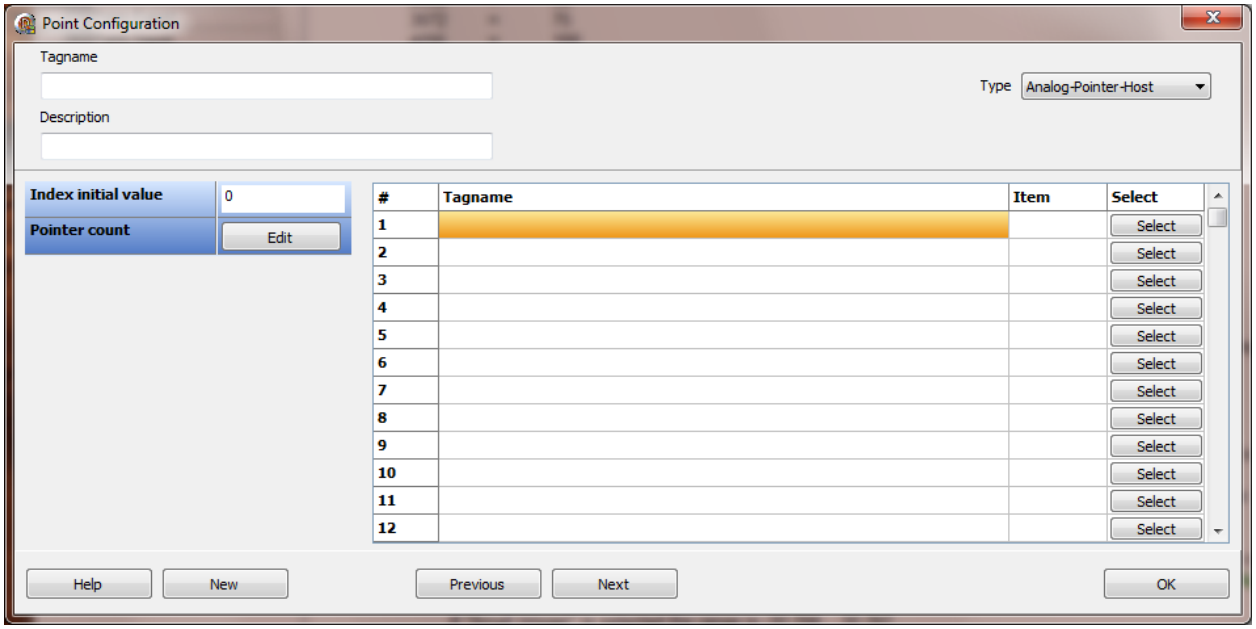
Secondary

A digital host point can be a secondary for a digital point.

Save/Restore

If enabled, when runtime monitoring is stopped (via a normal shutdown) the current value will be saved to disk and when runtime monitoring is restarted the value will be read from disk and written to the current value. This value is applied after the initial value attribute, the saved value will overwrite any initial value and before any override file setting.

Analog Host Pointer



This is a “pointer” type point. It is used to have a single tagname "point" to other tagnames based up the value of the "index".

For example: A system has 5 water pumps. All pumps have the same type of analog pressure input, flow input, lube oil pressure, etc..

The desired design is for the user to view each pump, one at a time, on a screen.

The pump pressure is to be shown as a "read out" on the screen. Place the first pump pressure tagname/item in the first row of the grid. The second in the second and so on. When the graphic is created tag the pressure read out to the "Analog Host Pointer" tagname/item.

When the screen is open, in a script, set the index value of the "Analog Host Pointer" to the number of the pump to be viewed.

One screen can be used to view many different systems of the same type.

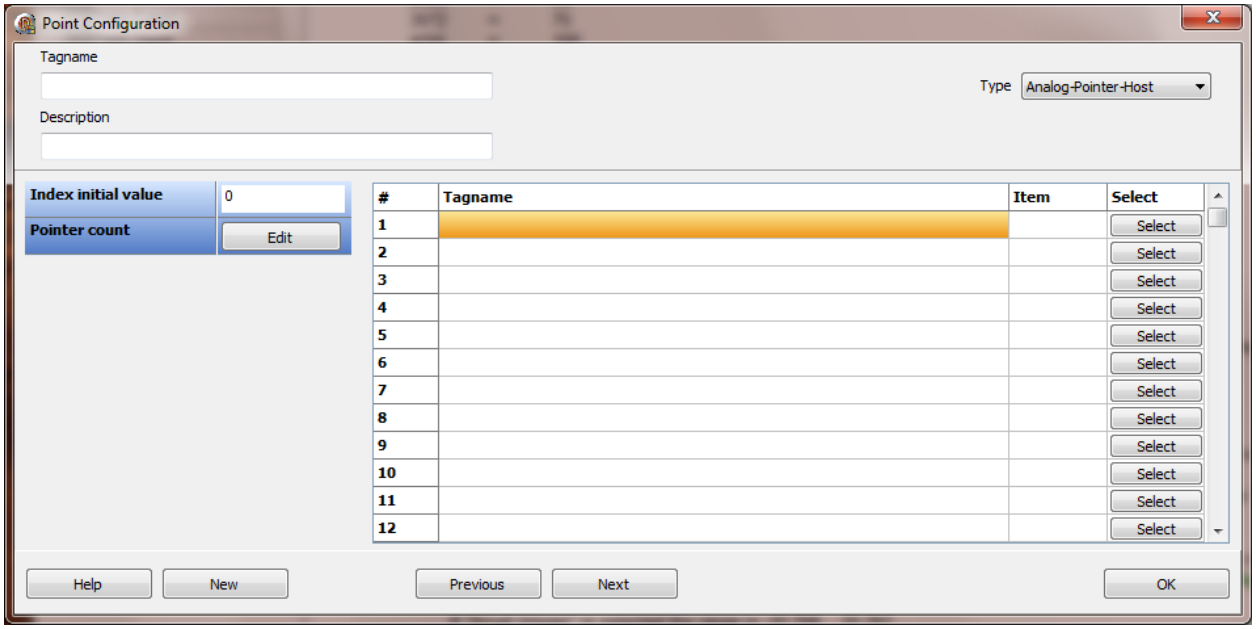
Index Initial Value

At runtime start, the value to set the index.

Pointer Count

The number of pointers this point supports. Set the value to the number of pointers this point requires. Setting this value greater than the number really needed only slows down processing in those areas utilizing this point type.

Digital Host Pointer

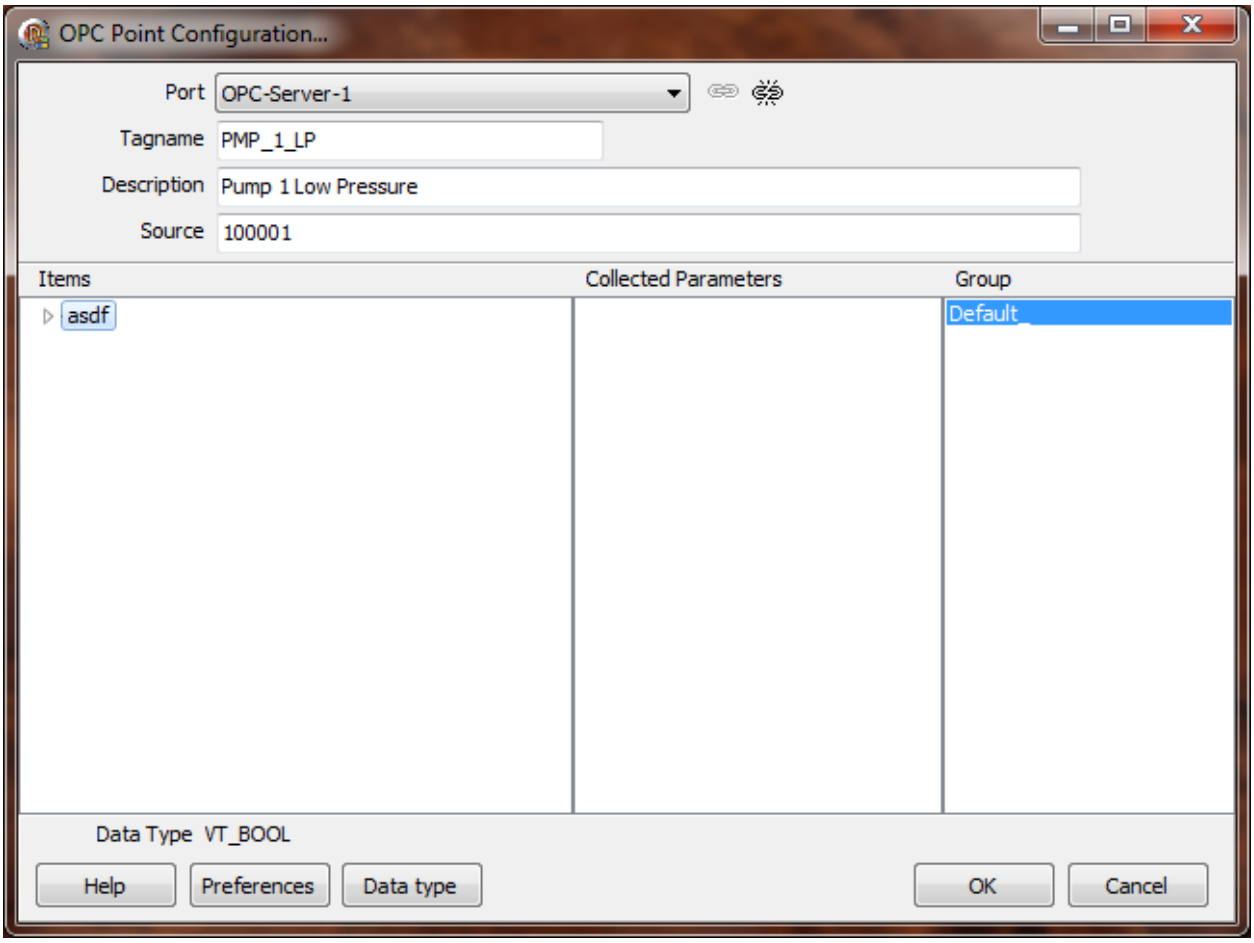


See the “Analog Host Pointer" above.

Source Address

OPC

If the point type modifier is OPC and the source address button is selected, the OPC Point configuration dialog is displayed.



Port

This is the OPC port assigned to the point.

Next to the "Port" combo box are two small icons. The first one is to connect to the server selected in the combo box and the second is to disconnect from the server if connected. Depending on the preferences connection to the server may occur when the window is opened and disconnected when the window is closed.

Tagname

The point tagname. The tagname might be generated if the server is browsed for items and the preferences are configured to generate the tagname. OPC Client Configuration Preferences.

Description

The point description.

Source

The item ID in the server. This is the full item ID used to identify what item in the server provides the data.

Unified Architecture: The namespace and item id.
<namespace> ?= <item id>
Example: http://www.someUAserver.com/OPCUA/StaticNodes?=GUIDDataItem

Items

When a connection to the server is established a command to browse all items in the server will be executed. The result will be displayed in the items tree view.

Collected Parameters

When the mouse button is clicked on one of the items in the items tree view and the item is a valid item, not a branch, a command to collect other item properties of the item is executed. The result is shown in the collected parameters checkbox list. Enable the checkbox for the values to be imported into the points fields. This operation is only performed at design time. The is provided so the values the server has configured for a point can be viewed, verified and used if needed for each point created. Those collected parameter items that are not checked are ignored. To disable the collection of other item properties. OPC Client Configuration Preferences.

Group

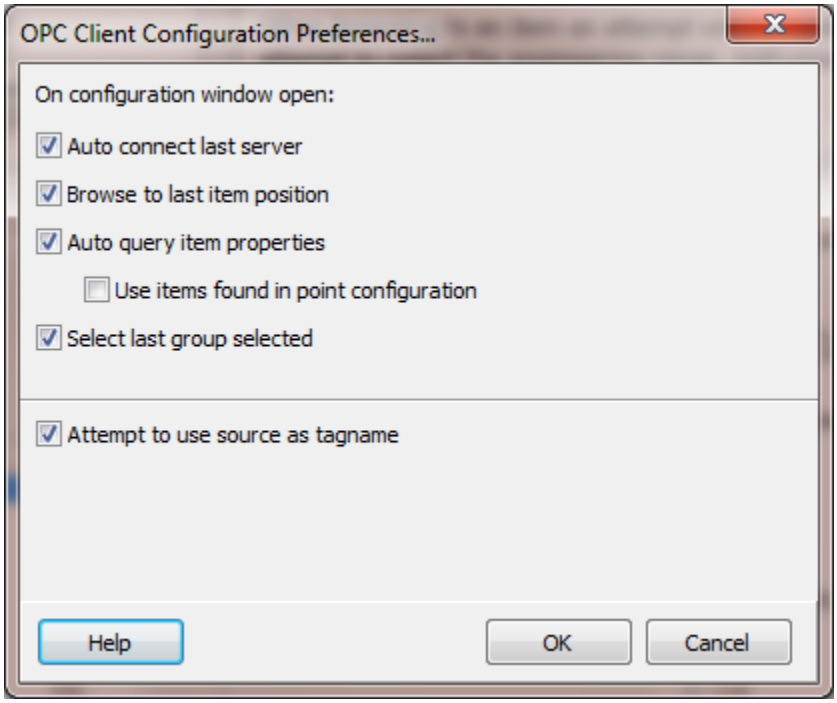
Each OPC point must belong to a group. This is a group in the server. Select which group this point is a member.

Data type field and button

This displays the data type of the item selected when the mouse button is clicked on an item in the items tree view. If possible the data type is collected from the OPC server. If a connection to the OPC server is not possible a data type must be selected.

OPC Client Configuration Preferences

Selecting the “Preference” button displays the “OPC Client Configuration Preferences” dialog.



Auto connect last server

When the window is opened an attempt to connect to the last server will be made.

Browse to last item position

If a connection to the server on window open is successful and this preference is enabled an attempt to browse the server to the last item selected will occur.

Auto query item properties

When browsing to an item an attempt will be made to query additional items connected to this "source item". For example, if the item is an analog type, an attempt to collect the engineering range, instrument range, etc. will be made. (This does not apply to OPC strings.)

Use items found in point configuration

If items are collected the point configuration values will be set to the server values.

Attempt to use source as tagname

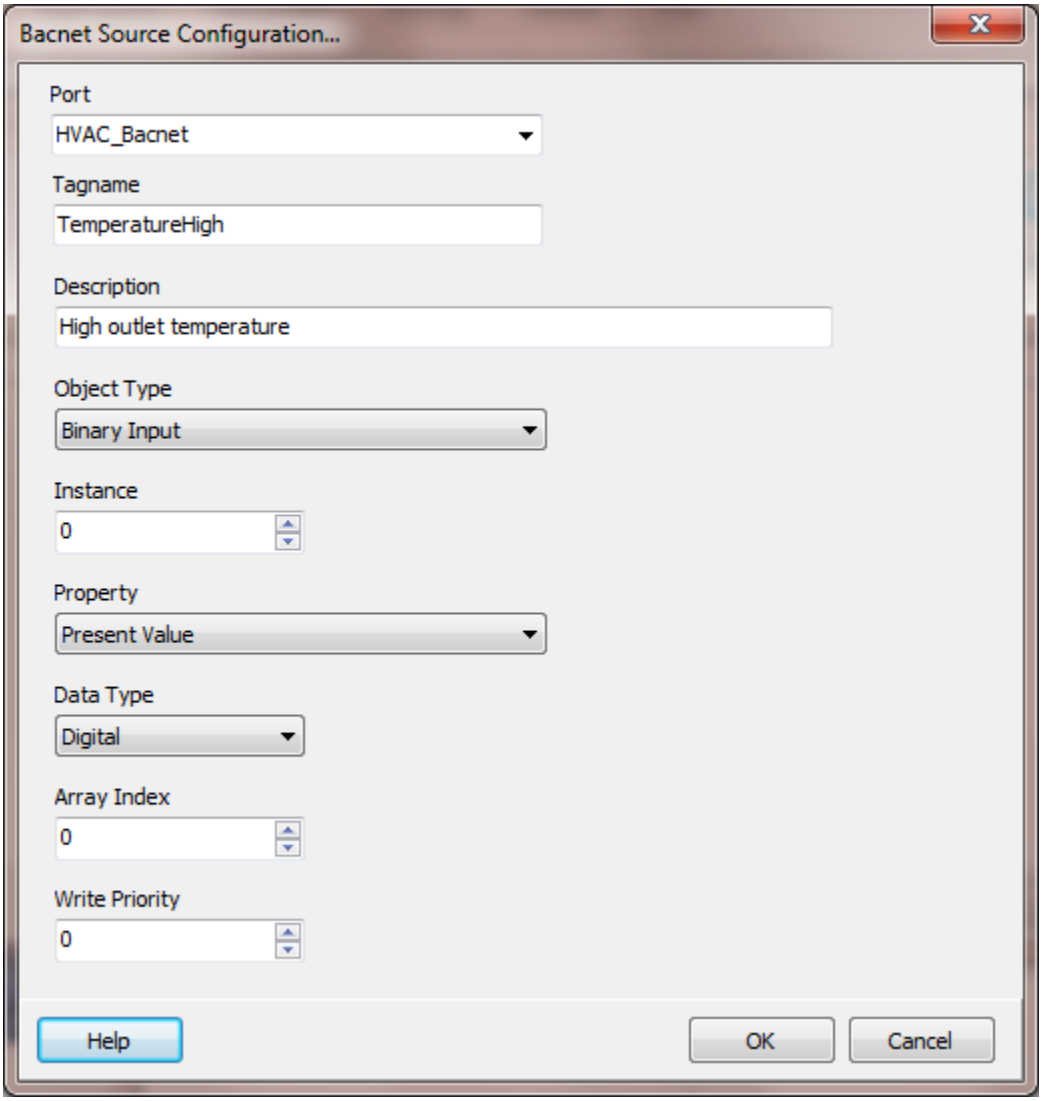
On a new item, when an item is selected the tagname field will be filled in with the source item name. (This does not apply to OPC strings.)

Select last group selected

On a new item, when the window is open the last group selected will be selected again.

BACNET/IP

If the point type modifier is Bacnet and the source address button is selected, the Bacnet source configuration dialog is displayed.



Addressing

A Bacnet/IP source address consists of three required parts and one optional part. The parts are delimited by a period (".") character.

1. Object type
2. Object instance
3. Property identifier

4. Array index (optional)

<object type>.<object instance>.<property identifier>.<array index>

Examples:

Analog Value.1.Present Value
Analog Value.1567.Present Value
Binary Input.33.Present Value
Binary Output.78.Present Value
Analog Output.35.Present Value.45

Object Type

Select the object type in the device.

Instance

Select the object instance in the device. The range is 0 - 4194303. Note: The device will define which instances are supported.

Property

All of the Bacnet properties are listed but not all data types are supported. For example: string, data/time, etc. data types are not supported. Verify the selected property type is analog or digital. Real and Boolean are supported.

Data Type

Select analog or digital.

Array Index

This field is optional. 0 indicates the value will be ignored. Array indexes begin at 1 and the upper limit is determined by the device.

Write Priority

Bacnet has 16 possible write priorities. 1 is the highest and 16 is the lowest. 0 is the default. If 0 is selected the write priority is ignored. If this point is not configured as read/write or write the write priority is ignored.

ALL OTHERS

The source address structure for all other port types is defined in the communications section for the port type. Examples are provided. Contact support if assistance is needed with source addressing.

Alarm Settings

These are all the alarm setting definitions for all points.

Alarm Area

Text that can provide the location of the alarm. (Optional)

Alarm Quick Help

Text that can provide additional information to the user about the alarm. (Optional)

Auto Clear

When the conditions are present to permit the alarm condition to return to the normal condition (reset, cleared) it will without any user action. Otherwise, the alarm must have been acknowledged.

Condition Logic

Additional text to describe the logic used in the calculation. (Optional)

Condition Status

The alarm text. (Optional)

Enabled

Used to enable the alarm.

Exceeded Text

Additional text to describe the alarm. (Optional)

Deadband

This is the percent of full scale the point current value must be above or below the limit to allow the alarm condition to be set to false (normal condition).

Example:
Range: 100, Hi Limit: 50, Deadband: 3

The alarm condition will become active when the point value is greater than or equal to 50. The alarm condition will be able to be cleared (reset, return to normal) at 47.

Limit

The alarm value threshold the point value must cross or be equal to, to set the alarm condition to true.

"Hi Hi" and "Hi" are rising alarms. The alarm condition is set to true when the point current value is equal or greater than the limit (current value >= alarm limit).

"Lo Lo" and "Lo" are falling alarms. The alarm condition is set to true when the point current value is equal or less than the limit (current value <= alarm limit).

Log

If enabled, when the alarm condition transitions to true, an entry will be entered in the daily alarm log. When the alarm condition transitions to false a "return to normal" entry will be entered in the daily alarm log. (Optional)

Primary Alarm Area

Text that can provide the location of the alarm. (Optional)

Sound

If a sound name is entered the sound will play when the alarm condition transitions to true. The sound will play until the "Silence" command is activated. (Optional)

Print

If enabled, when the alarm condition transitions to true, an entry will be output to the configured printer. When the alarm condition transitions to false a "return to normal" entry will be output to the configured printer. (Optional)

Notify Email/SMS

If enabled, when the alarm condition transitions to true the Notifications - Email and/or Notifications - SMS logic will process the alarm. (Optional)

Delay (Seconds)

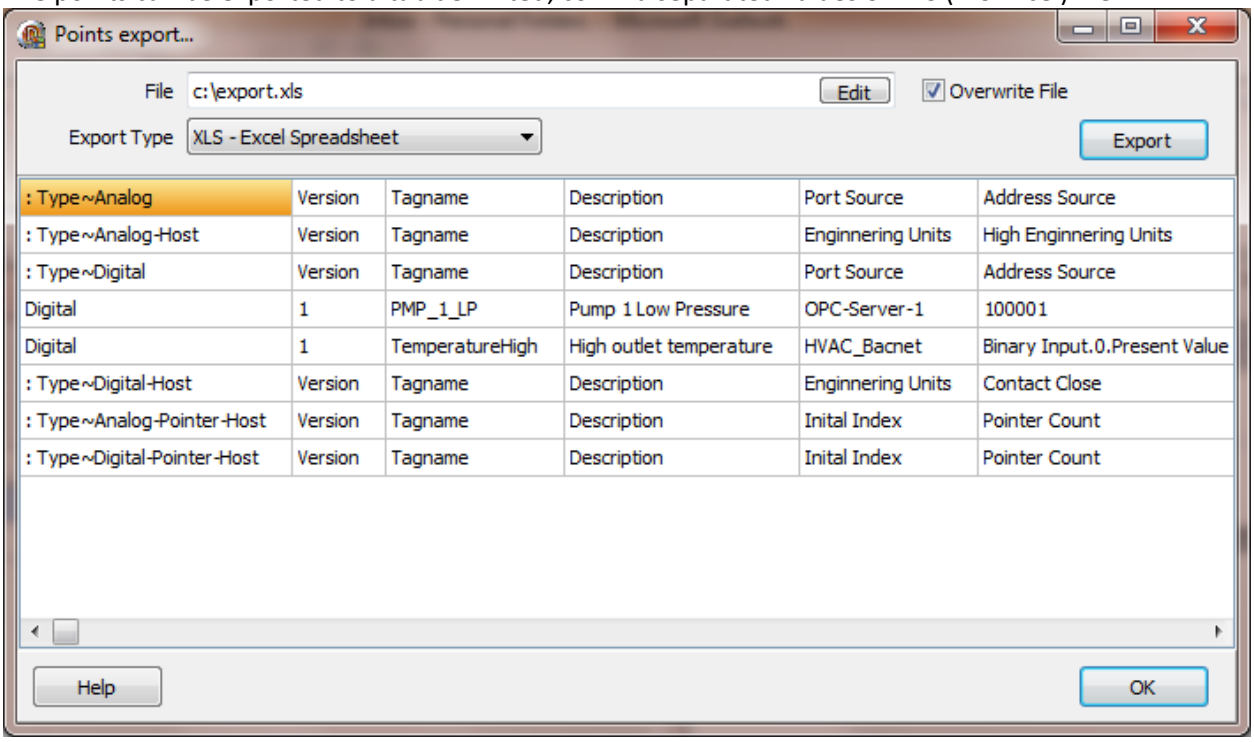
This is the number of seconds to delay setting the alarm active after the alarm condition logic becomes true. (Optional)

Priority

A value (0 - 65535) assigned to provide a method to group alarms based on a level of severity. If the value is out of range or blank 0 is used.

EXPORT

The points can be exported to a tab delimited, comma separated values or XLS (MS Excel) file.



: Type~Analog	Version	Tagname	Description	Port Source	Address Source
: Type~Analog-Host	Version	Tagname	Description	Enginnering Units	High Enginnering Units
: Type~Digital	Version	Tagname	Description	Port Source	Address Source
Digital	1	PMP_1_LP	Pump 1 Low Pressure	OPC-Server-1	100001
Digital	1	TemperatureHigh	High outlet temperature	HVAC_Bacnet	Binary Input.0.Present Value
: Type~Digital-Host	Version	Tagname	Description	Enginnering Units	Contact Close
: Type~Analog-Pointer-Host	Version	Tagname	Description	Inital Index	Pointer Count
: Type~Digital-Pointer-Host	Version	Tagname	Description	Inital Index	Pointer Count

File

Enter the destination file name. Use the edit button as needed to select a file and path. The file extension will automatically be added when the export button is selected.

CSV - Comma Separated Values
XLS - Excel Spreadsheet
TXT - Tab Separated Values

Export Type

Select how the data is to be exported. (Excel does not need to be installed to export to an Excel spreadsheet.)

Overwrite File

If a file with the same name as entered is detected it will be deleted before the data is exported if this checkbox is checked. If it is not checked a dialog will be displayed asking if the file should be overwritten. If "No" is selected the export operation will not proceed.

Export

Select the export button to export the data in the grid.

Notes:
Each point type (analog, digital, analog host, etc.) has a header row that defines the columns for the point type.

The points are grouped after the header row for that point type.

The header rows are always exported.

Some point types may be configured with an embedded script. The script may contain characters that interfere with parsing of the exported file. The scripts are exported as HEX codes in ASCII format. Example: 36C6F7365643A3D5265616456616C75652

Do not change any of the characters in this field.

Excel does not need to be installed to export to XLS.

The pointer type points can contain up to 5000 references. This exceeds the column limit of some spreadsheet programs. The pointer tagnames and pointer item numbers are grouped into two cells per pointer type point

The pointer type points contain one cell per row to hold all the pointer tagnames and one cell to contain all pointer item numbers. Both cells must begin and end with a quote (") character. The pointers are separated by a comma (,).

The format of the tagname cell is:

"dfPointer1tagname=<tagname name 1>,dfPointer2tagname=<tagname name 2>,dfPointerNtagname=<tagname name n>"

Each tagname must have the "dfPointerNtagname=" before the tagname. The number is the tagname index in the point.

The format of the item number cell is:

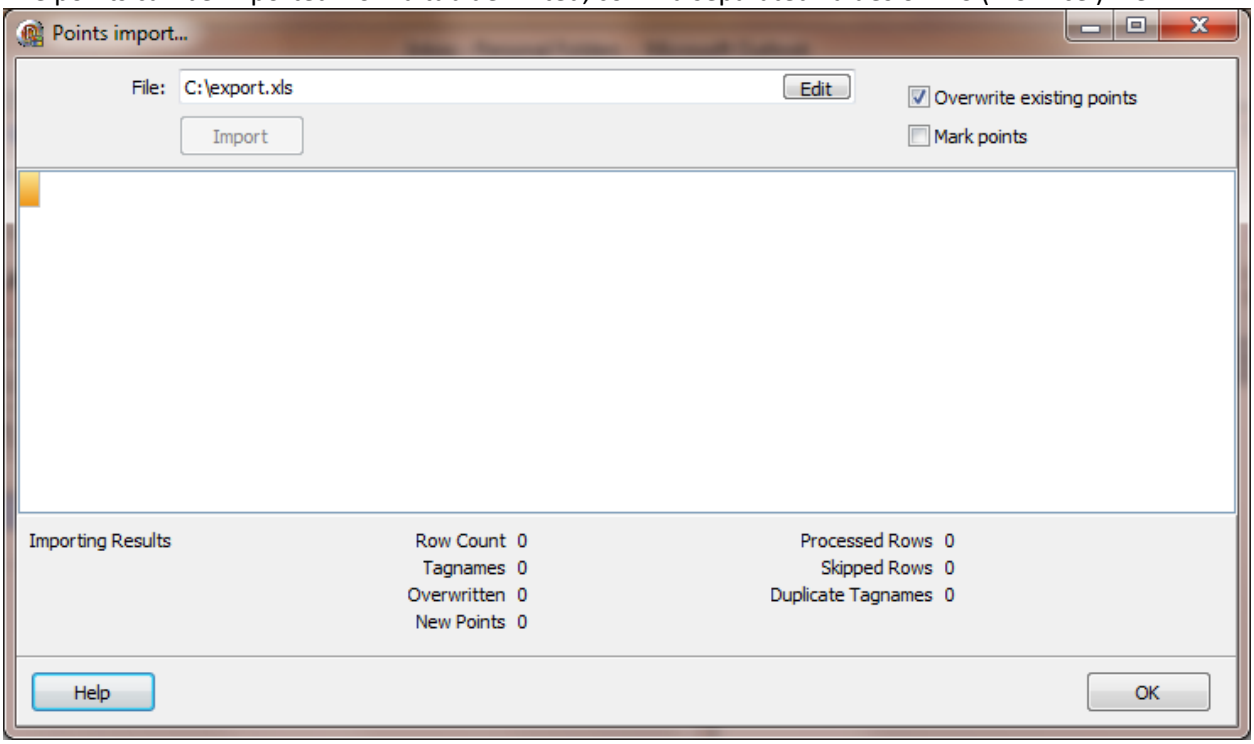
"dfPointer1item=<item 1>,dfPointer2item=<item 2>,dfPointerNitem=<item n>"

Each item must have the "dfPointerNitem=" before the item. The number is the item index in the point.

Only those indexes configured are exported.

IMPORT

The points can be imported from a tab delimited, comma separated values or XLS (MS Excel) file.



When the "Import" button is selected an attempt to import the last file to the grid is performed. If the file load failed the grid will be empty and the "Import" button on this window will be disabled.

File

Enter the input file name. Use the edit button as needed to select the file and path. The file extension will automatically determine the format of the file. Once the file is selected an attempt to import the contents to the grid will be performed. If the gird has data select the import button to attempt to import the data to the project database.
CSV - Comma Separated Values
XLS - Excel Spreadsheet
TXT - Tab Separated Values

Overwrite existing points

When importing the rows of the file if a point tagname is imported and the point tagname is in the database the point in the database will be deleted and the imported point will be added if this checkbox is enabled. If this checkbox is not enabled the imported point will be discarded and the point in the database will not be changed.

Mark points

The imported points will be marked for display. When the point is viewed in the "Points configuration" window the point text color will be "red" and "bold" until the point is displayed in the "Point configuration" window. If the points exist and are overwritten the points are not "marked."

Import Button

When the grid has been filed select the import button and an attempt to import the points will be performed. If errors are detected a window will be displayed indicating the row number, tagname and error message.

Importing Results

Row Count

The number of rows imported to the grid.

Tagnames

The number of points successfully imported.

Overwritten

If overwrite is enabled, the number if points overwritten with data from the import file.

New Points

The number of new points created from tagnames not present in the project tagname data base.

Processed Rows

The number of grid rows processed.

Skipped Rows

The number of gird rows skipped because the first character was a colon ':' or the first column of the row as blank.

Duplicate Tagnames

If overwrite is not enabled, the number of rows that contained a tagname that is present in the project database and the row was discarded.

Notes:
A row that begins with a ":" character or a blank cell is ignored.

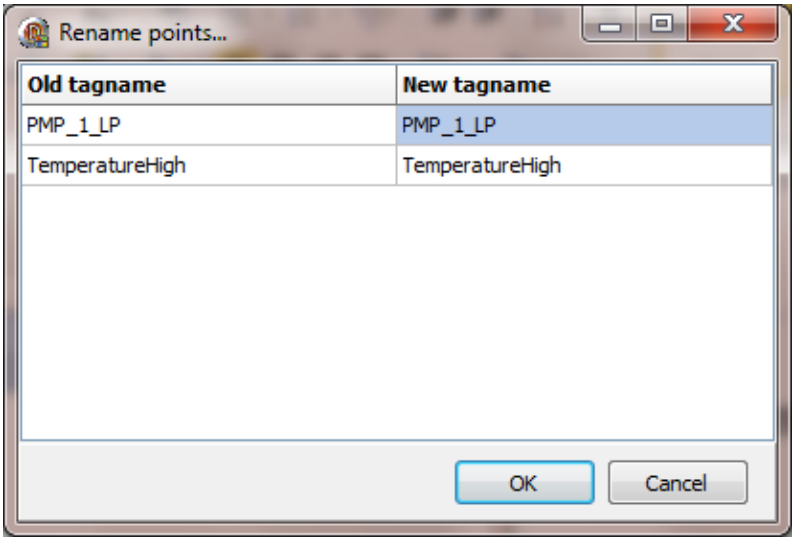
Excel does not need to be installed to import from an XLS file. Excel 2007 file format is not supported at present.

The importing of points does not perform extensive error checking. It is much like the points editor. For example in the point editor a "port" source can be entered that does not exist. This is permitted because the port may be created at a later time and the runtime engine would disregard any points that are not valid. Another example is the address source field. The address is not validated until runtime. This permits the creation of points that are known to be needed but the address is not defined when the point is created.

Do not alter the column order from the exported column order.
The column order on import **must be** the same order as exported.

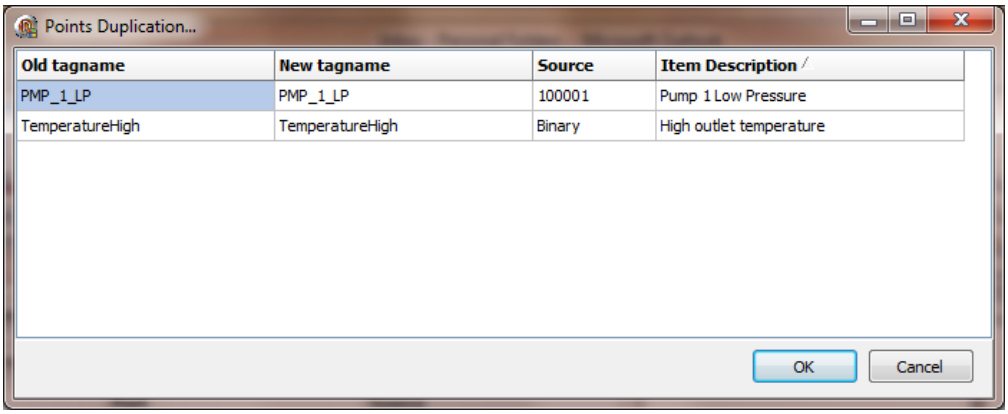
Rename Button

The rename dialog can be a fast method to rename one or more points without opening the point editor.



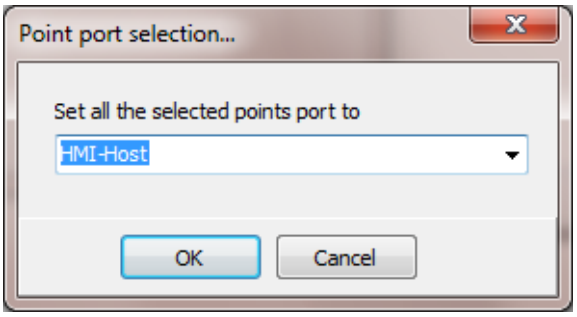
Point Duplicate Button

A point or many points can be duplicated with the "Duplicate" button. The tagname must be changed and the source address can be changed.



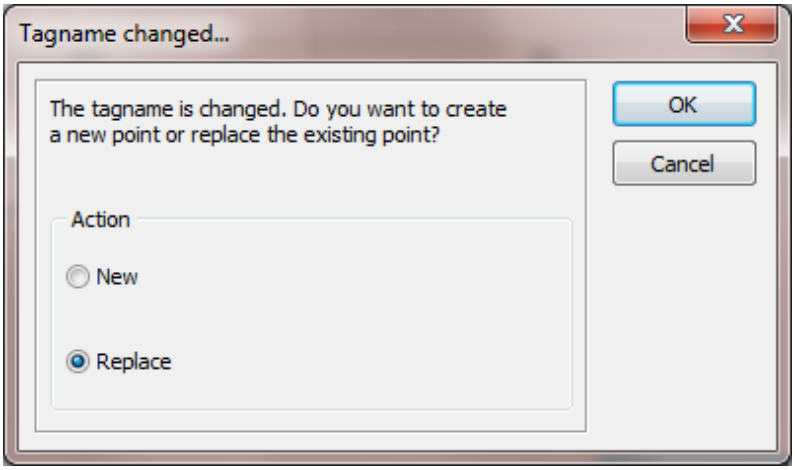
Port button

The "Port" button is a fast method to change the assigned port for one or many points.



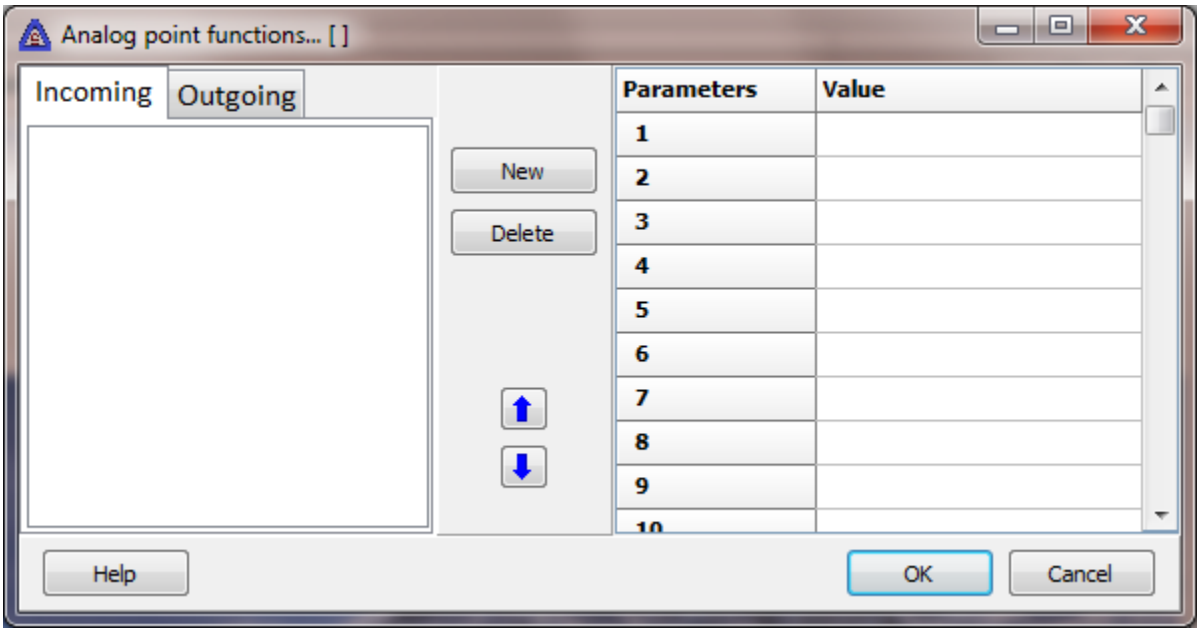
TAGNAME CHANGE

When the “OK” button is selected in the point editor and the tagname field does not match the tagname field when the dialog was opened the “Tagname changed” dialog is displayed.



ANALOG FUNCTIONS

Each point can be configured to apply a function or functions to the incoming value, after scaling is applied (if enabled) before the alarm logic is executed and before the other modules of the HMI can access the current value of the point.



The incoming "raw" value, from the port, is passed to the first function and the result of the first function is passed to the next function in the list and so on. Use the blue arrows to reorder the function list. The result of the last function is the "current value" of the point for all other operations.

The first parameter (parameter 0) passed to the function is the raw value or the result of the previous function and does not require configuration.

Example: The "afDivide" function requires two parameters, the incoming value and the divisor.

```
function afDivide(inValue,divisor);
begin
 if (inValue <> 0) and (divisor <> 0) then
  result:=(inValue / divisor)
 else
  result:=0;
end;
```

The incoming value (inValue) is automatically passed to the function and the divisor must be configured via parameter 1.

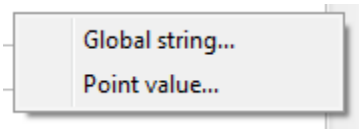
Parameters

A parameter can be a value, the value of another point or the value of a script global. The script global must be a number or an error will occur.

Parameter type	Cell contents example
Value	25, 35.2, -89
Point value	=PT(Tank.Process Variable Analog)
Script global value	=SG(Tank.Pressure)

Note: The current value is not updated if a point value or script global value changes. The functions are executed when the configured port sends notification to update the current value.

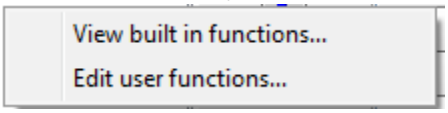
Right click on a cell in the parameters list and helper windows will be displayed.



Built in and user functions

The built in functions are defined in a file located in the HMI program path and cannot be altered. If functions are required, that are not supplied, user functions can be created. Contact support and the function could be added to the built in functions or if assistance is needed with creating a function. User created functions must be in a file named "analogFunctionsCustom.psc" and the file must be in the project path.

Right click on the incoming or outgoing list box and a popup menu will appear to allow viewing the built in functions or view/edit the user functions file.



Note: Be sure to select the "compile" button before saving the user functions file, if changes are made.

Built in functions

Function name	Parameter 1	Description
afAbs		Return the absolute value
afAdd	Addend	Adds the incoming and parameter 1
afCeil		Returns lowest integer value greater than or equal to incoming value.
afDec	Amount	Subtracts the amount from incoming value
afDivide	Divisor	Floating point division *1
afDivideInteger	Divisor	Integer division *1
afExp		Raises *e* (2.71828) to the power of the incoming value
afFloor		Returns highest integer value less than or equal to incoming value
afFrac		Returns the fractional part
afInc	Amount	Adds the amount from incoming value

afMax	CompareValue	Returns the greater of the numeric values
afMin	CompareValue	Returns the lesser of the numeric values
afMod	Divisor	Returns the remainder of incoming value divided by the divisor
afMultiple	Multiplier	Multiplies the incoming by the multiplier
afPower	Exponent	Raises base to the power of exponent
afRound		Uses "Bankers" round to round to an integer
afSquare		Returns the square of the input (input * input)
afSquareRoot		Returns the square root of the input
afSubtract	subtrahend	Returns the input minus the subtrahend (I – input)
afTrunc		Truncates a real number to an integer

Note:

If the divisor is 0 (zero) an error will not occur and the function result will be the incoming value (inValue).

Outgoing

The "outgoing" functions are applied when a write command is issued to the point. All configured outgoing analog functions are applied to the outgoing value, scaling (if enabled) is applied and the outgoing value is queued for transmission.

Notes:

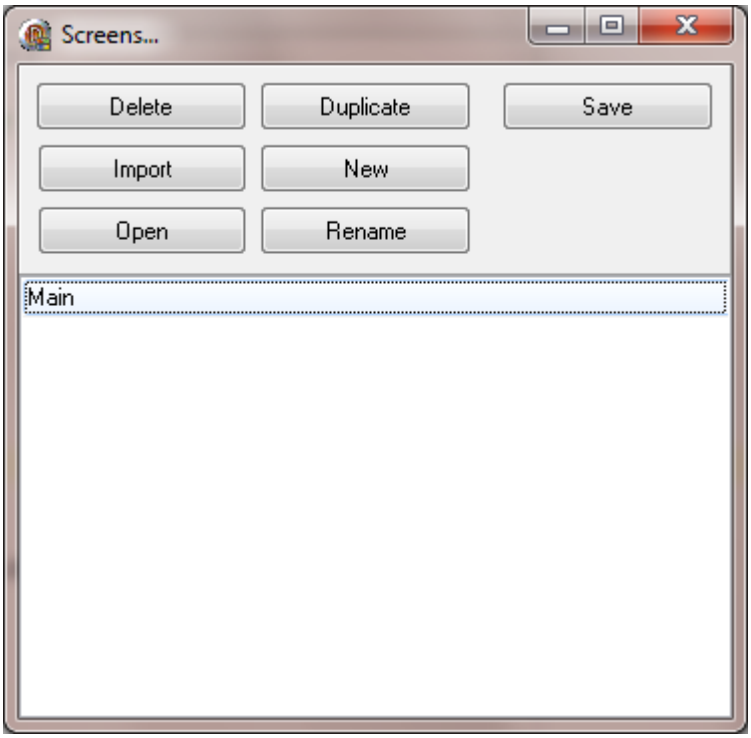
1) A runtime error in any function will halt function processing, set a point's bad quality property to 0 (Zero) (bad quality), the current value will be unreliable and the "Analog function fault" property (5128) will be set to the function index that produced the failure. A value of "-1" for the property indicates no error. Configured alarms will not be processed.
2) Runtime errors for outgoing functions, will halt write command processing, place an error message in the event log and the write command to the external device will not be executed.
3) User functions should be very simple, like the built in functions and not attempt to write to other points, log events, open windows, etc.. User functions not designed to be simple and fast could produce point processing bottlenecks and degraded HMI performance.
4) If a user function needs one (1) or more parameters, the configuration must use the parameters from the first parameter (#1) to the count required without skipping any parameter locations.
5) Analog functions are not applied to int64 or uint64 properties. The point current value "Process variable analog" (5000) and is the only property altered by analog functions.
6) The analog functions will not be applied to the current value if a runtime override value is configured or the initial value property.

Port setting (see communication port settings)
Some communication ports use a “polling” protocol and compare incoming data against the previous data. If the data has not changed, the point data processing is not called. If an analog function uses other point data as a parameter, enabling the “AP functions” property may allow the point to update when the data from the communication port has not changed. Enabling this property may impact performance.

Graphics

Graphics can be simple or complex. There are many built in complex objects. The simple drawing elements, i.e. lines, circles, rectangles, are all supported with fill colors, gradients, rotation, etc.

Each screen must have a unique name.



Delete

This deletes the selected screens.

Duplicate

This duplicates the selected screen. A prompt for a screen name will be displayed.

Import

This imports an HMI graphic screen; for example, from another project.

New

This creates and opens a new screen in the screen editor after the name prompt is displayed. Each screen must have a unique name.

Open

This opens the selected screen or screens in the editor. Double clicking the screen name will also open the screen editor

Rename

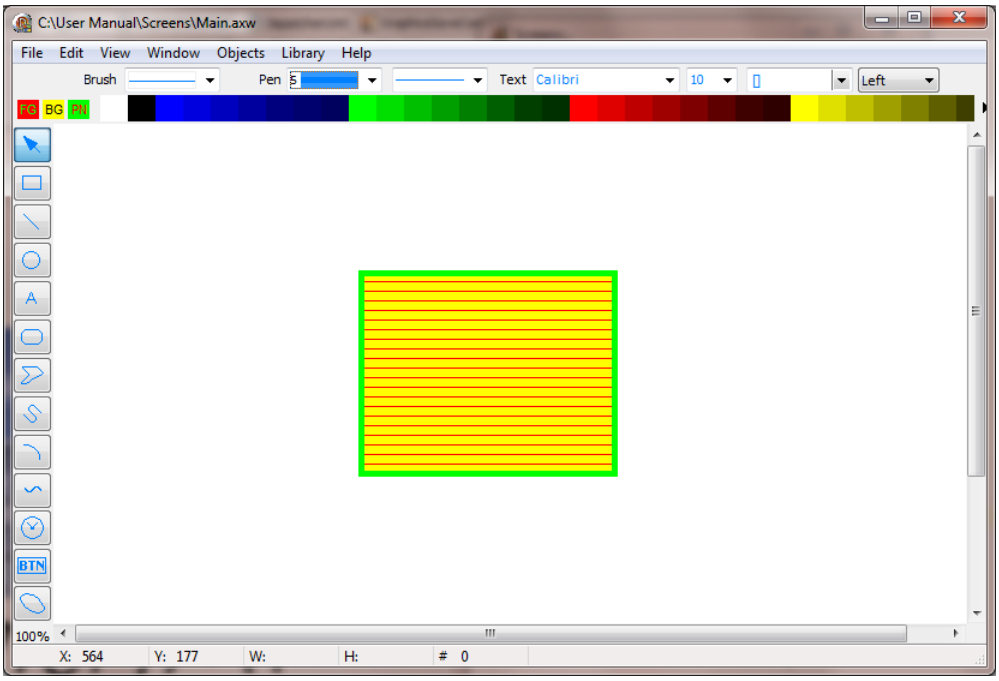
This is used to “rename” a screen.

Save

This is used to “save” all open screens and other shared screen resources.

EDITOR

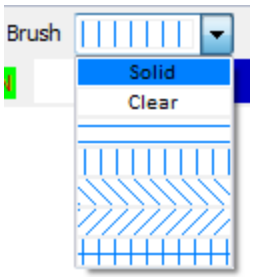
This is the graphics editor window. First all the controls of the window will be covered, the drawing tools, the menu items, and animations.



When discussing the brush style, pen width, foreground color, background color, pen color and pen style we will refer to the rectangle in the image above.

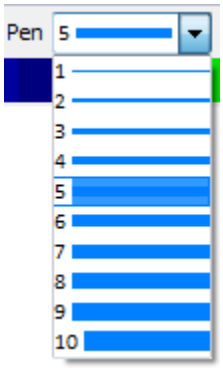
Foreground color	Red
Background color	Yellow
Pen color	Green
Pen style	Solid
Brush style	Horizontal
Pen width	5

Brush



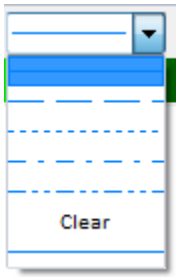
A brush is a pattern applied in the interior if an object. When a brush is applied to the graphic element the foreground and background colors are applicable.
If the brush style is “Solid”, only the foreground color is applied.
If the brush style is “Clear”, neither color is applicable.
All other brush styles; the foreground color is applied to the lines or pattern and the background color is the filler.

Pen width



The pen width is set to 5.

Pen style



The pen style is solid. The pen style can only be applied when the pen width is one (1).

COLOR BAR

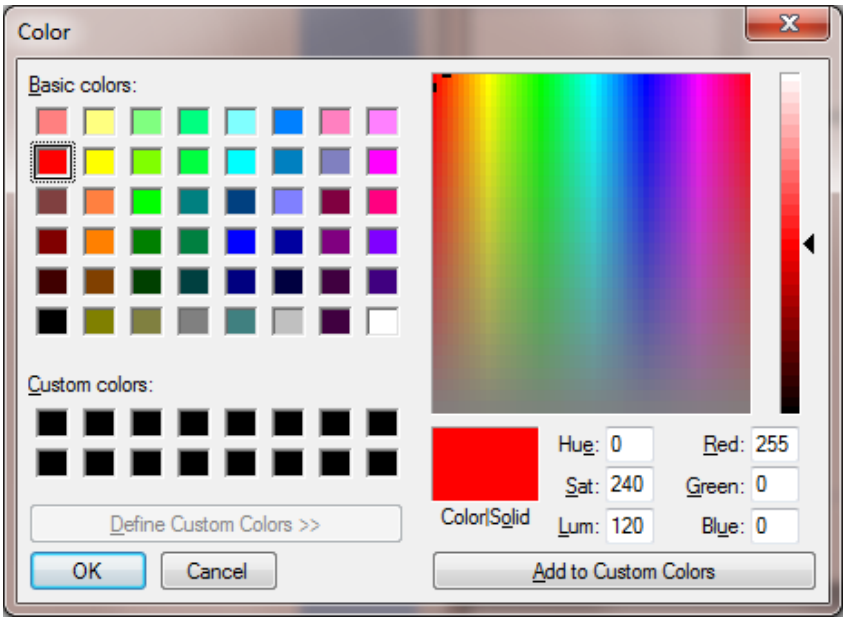
The above is the color bar.

The three colored rectangles on the left serve two purposes.

FG	Foreground color
BG	Background color
PN	Pen color

1) When a graphic element is selected the colors of the rectangles will be the colors of the graphic element.
2) A left click in a rectangle will display the custom color dialog, changing the color of all selected graphic elements.

   Custom color dialog



The color bar responds to mouse clicks.
When a graphic element is selected:

Left click	Foreground color
Right click	Background color
Middle click	Pen color

Holding down the “CTRL” key and left clicking a color in the color bar will display the custom color dialog allowing the color in the selected position to be set.

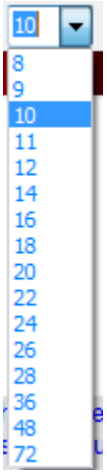
TEXT SETTINGS

This is the text used for the user manual.

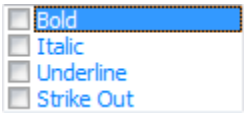
The brush style, foreground color and background color also apply to text elements. The foreground color is blue, the brush style is “Solid” and the background color is grey.

Text font

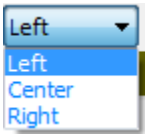
Text size



Text style



Text alignment



Tool bar

Selection tool

Use the arrow tool to select, resize, move, etc. graphic elements. Double clicking an object will display the animation dialog.
Holding down the "SHIFT" key while clicking the left mouse button on a graphic element will toggle the elements "selected" state.
Dragging a selection box around elements will add the elements to the selected list. If the "SHIFT" key is down it will toggle the elements "selected" state.

Rectangle tool

This tool creates a rectangle. Press the left mouse button while the arrow is in the window drawing area and move the mouse, release the mouse button. A rectangle will be created and the size will be determined when the mouse button is released. Hold down the "SHIFT" key to force the sides to be equal length.

Line tool

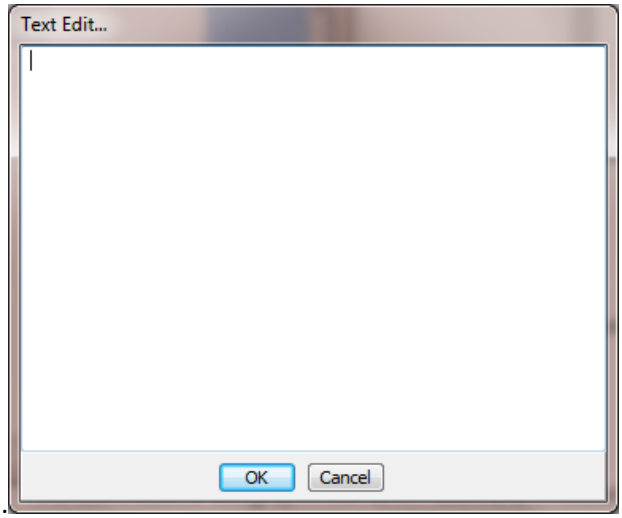
This tool creates a line. Press the left mouse button while the arrow is in the window drawing area and move the mouse, release the mouse button. A line will be created and the length will be determined when the mouse button is released. Hold down the "SHIFT" key to force the line to draw at "45" degree angles. Select the 'Edit' menu item to configure the arrow settings.

Circle tool

This tool creates a circle. Press the left mouse button while the arrow is in the window drawing area and move the mouse, release the mouse button. A circle will be created and the size will be determined when the mouse button is released. Hold down the "SHIFT" key to force the radius to be the same at all points.

TEXT TOOL

This tool creates a text graphic element. Press the left mouse button while the arrow is in the window drawing area and move the mouse, release the mouse button. A rectangle will be created and the size will be determined when the mouse button is released. This is the bounding rectangle for the text. When the mouse button is released a dialog will appear to allow entry of the text. **Note:** Not all fonts can be rotated.



ROUND RECTANGLE TOOL

This tool creates a round rectangle. Press the left mouse button while the arrow is in the window drawing area and move the mouse, release the mouse button. A rectangle with rounded corners will be created and the size will be determined when the mouse button is released. Hold down the "SHIFT" key to force the sides to be equal length.

POLYLINE TOOL

This tool creates a round polyline. A polyline is a one or more straight lines drawn as a single object. A line has two end points. Note: Polylines might not appear the same in the graphic editor and when rendered on a web page.

Press and release the left mouse button while the arrow is in the window drawing area. A line will be created with one end point where the mouse was clicked and the other line end point will be at the location of the mouse pointer. Continue to move the mouse and the line end point will follow. Press and release the left mouse button and the line end point will be set. This point now acts as the first line end point for the next line. Repeat until all the desired lines have been drawn. To end double click the left mouse button or press the "ESC" key.

Hold down the "SHIFT" key to force the line to draw at "45" degree angles.

Polyline, freehand tool

A freehand polyline is a group of lines used to draw irregular shapes.
Press and hold the left mouse button while moving the mouse in the drawing area. When the left mouse button is released the object creation is complete. Note: Polylines might not appear the same in the graphic editor and when rendered on a web page.

Arc/wedge tool

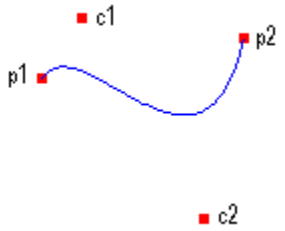
Press the left mouse button while the arrow is in the window drawing area and move the mouse, release the mouse button. An arc will be created and the size will be determined when the mouse button is released. Hold down the "SHIFT" key to force the sides to be equal length. The default arc starts at 0° and ends at 90°. It can be adjusted via the menu Objects/Arc menu.

Bezier tool

Press and release the left mouse button while the arrow is in the window drawing area. A line will be created with one end point where the mouse was clicked and the other line end point will be at the location of the mouse pointer. Continue to move the mouse and the line end point will follow. Press and release the left mouse button and the line end point will be set. This point now acts as the first line end point for the next line. Repeat until at least three line segments have been drawn. To end double click the left mouse button or press the "ESC" key.

Hold down the "SHIFT" key to force the line to draw at "45" degree angles.

A Bezier spline is a curve specified by four points: two end points (p1 and p2) and two control points (c1 and c2). The curve begins at p1 and ends at p2. The curve doesn't pass through the control points, but the control points act as magnets, pulling the curve in certain directions and influencing the way the curve bends. The following illustration shows a Bezier curve along with its endpoints and control points.



Note that the curve starts at p1 and moves toward the control point c1. The tangent line to the curve at p1 is the line drawn from p1 to c1. Also note that the tangent line at the endpoint p2 is the line drawn from c2 to p2.
The line can contain multiple Bezier line segments. The last point of each line serves as the first point of the next line. The first segment would have 4 points and all subsequent segments contain 3 points.

Complex object tool

Press and release the left mouse button in the drawing area and the complex object selection dialog box will be displayed. Complex objects are covered later in the manual.

Button tool

Press and release the left mouse button in the drawing area and the button object selection dialog box will be displayed. Button objects are covered later in the manual.

Bitmap tool

Press the left mouse button while the arrow is in the window drawing area and move the mouse, release the mouse button. This tool creates a bit map and opens the bit map editor. "Drag & Drop" can also be used to add an existing bitmap file to the graphic window.

Database tool

Press and release the left mouse button in the drawing area and the database object selection dialog box will be displayed. The database objects are covered later in the manual.

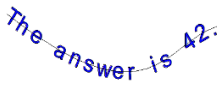
Ellipse tool

This tool creates an ellipse circle. Press the left mouse button while the arrow is in the window drawing area and move the mouse, release the mouse button. An ellipse will be created and the size will be determined when the mouse button is released.

Pipe tool

This tool creates a gradient filled pipe segment. Press the left mouse button while the arrow is in the window drawing area and move the mouse, release the mouse button. A pipe segment will be created and the length will be determined when the mouse button is released. See pipe graphic element settings. Pipe segments are limited to vertical, horizontal and forty five degree offsets.

Bezier text tool



See Bezier tool for steps to draw the Bezier. Bezier text objects are covered later in the manual.

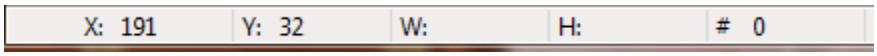
RTF tool

This tool creates a text field that supports "Rich Text Format" (RTF). Press the left mouse button while the arrow is in the window drawing area and move the mouse, release the mouse button. This is the bounding rectangle for the text. When the mouse button is released a dialog will appear to allow entry of the text.

100% ZOOM/SCALING

The drawing area can be zoomed and this displays the amount of zoom.
100% = no zoom/ 0 scaling.
Using the left mouse button, click in the area and the drawing area will zoom in, area larger.
Using the right mouse button, click in the area and the drawing area will zoom out, area smaller.

Status bar



The status bar is at the bottom of the window. It can be hidden in the "Settings/Window settings" dialog.

<u>When no object is selected;</u>
The X value is the mouse horizontal position, 0 (zero) is the left side.
The Y value is the vertical position, 0 (zero) is the top.
<u>When one object is selected:</u>
The X is the left position of the object.
The Y is the top position of the object.
The W and H are the width and height of the object.
<u>When more than one object is selected:</u>
The X is the leftmost position of the objects.
The Y is the topmost position of the objects.
The W and H are the total width and height of the objects. All these add up to the "bounding" rectangle.

The "#" displays how many (count) objects are selected.

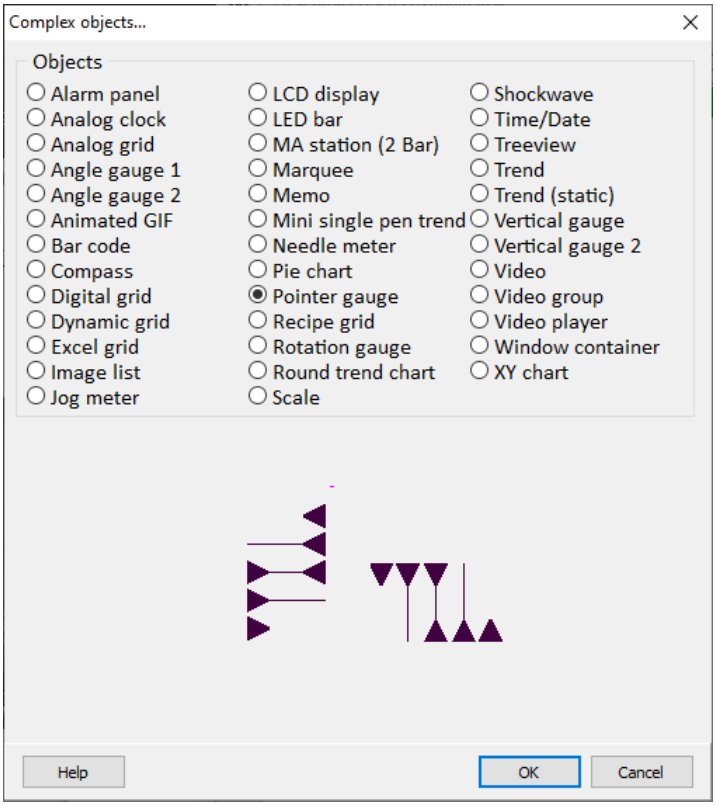
DRAG & DROP

The graphic editor supports drag and drop for external files. The file types supported are:

PNG	Portable Network Graphics
GIF	Graphics Interchange Format
JPG/JPEG	Joint Photographic Experts Group
WMF/EMF	Windows Meta File/Enhanced Metafiles
BMP	Bitmap Image File
ICO	Icon
SVG	Scaled Vector Graphic (static)
RTF	Rich Text Format

The file name extension determines the file type. The clipboard can continue to be used to import graphics. The nature of WMF/EMF files might require a particular file to be imported via the clipboard.

COMPLEX OBJECTS 



The complex objects in alphabetical order.

Alarm panel
Analog clock
Analog grid
Angle gauge 1
Angle gauge 2
Animated GIF
Barcode
Compass
Digital grid
Dynamic grid
Excel grid
Image list
Jog meter
LCD display
LED bar
MA station (2 bar)
Marquee
Memo
Mini single pen trend
Needle meter
Pie chart
Pointer gauge
Recipe grid
Rotation gauge
Round trend chart
Scale
Shockwave
Time/Date
Treeview
Trend
Trend (static)
Vertical gauge
Vertical gauge 2
Video
Video group
Video player
Window container
X/Y chart

ALARM PANEL

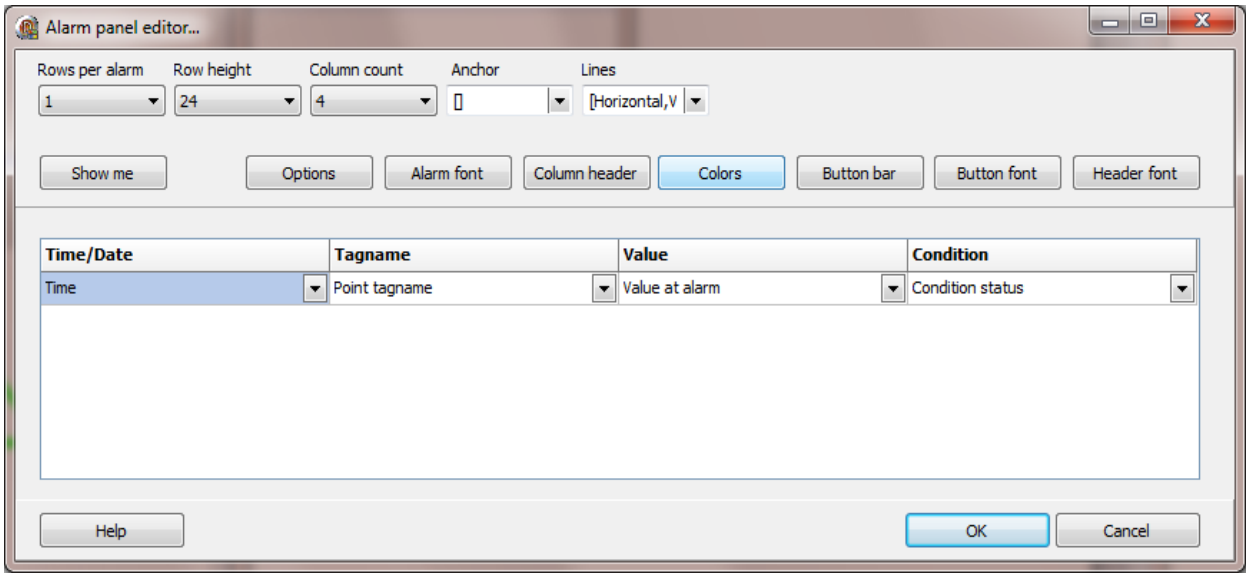
This graphic element is used to create a custom active alarms list if the built-in active alarms window is not suitable. The animation System variable can also be used to display selected alarm information.

Note: The 'Show me' button increases the window width to the sum of the column widths if the window width will not display all the columns. The graphic element does not automatically size. Set the size of the graphic element to the desired width.

Any unused area of the panel will be the foreground color.

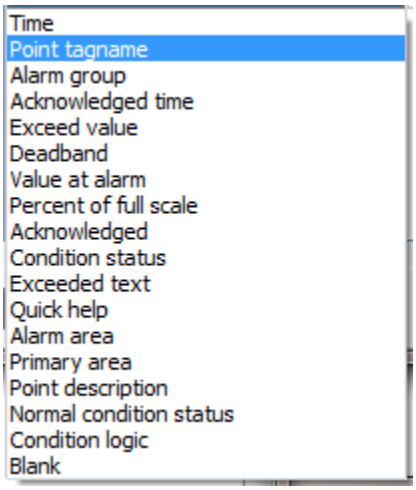
The alarm panel editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.

Alarm panel editor



The 'Show me' button displays some of the fields of an alarm. Its purpose is to show an example of how the panel will appear at runtime.

Each column can display the property of a point.



Rows per alarm

Each alarm has many fields and the number of fields to display might not fit in the space allocated. Each active alarm can use 1 to 4 rows to display the alarm fields.

Row height

This is the height of each row.

Column count

This is the number of columns per row.

Anchor

The alarm panel element can be anchored (locked) to the any or all four sides of the window. See margins.

Lines

Border	around the panel
Horizontal	between each column
Vertical	between each row

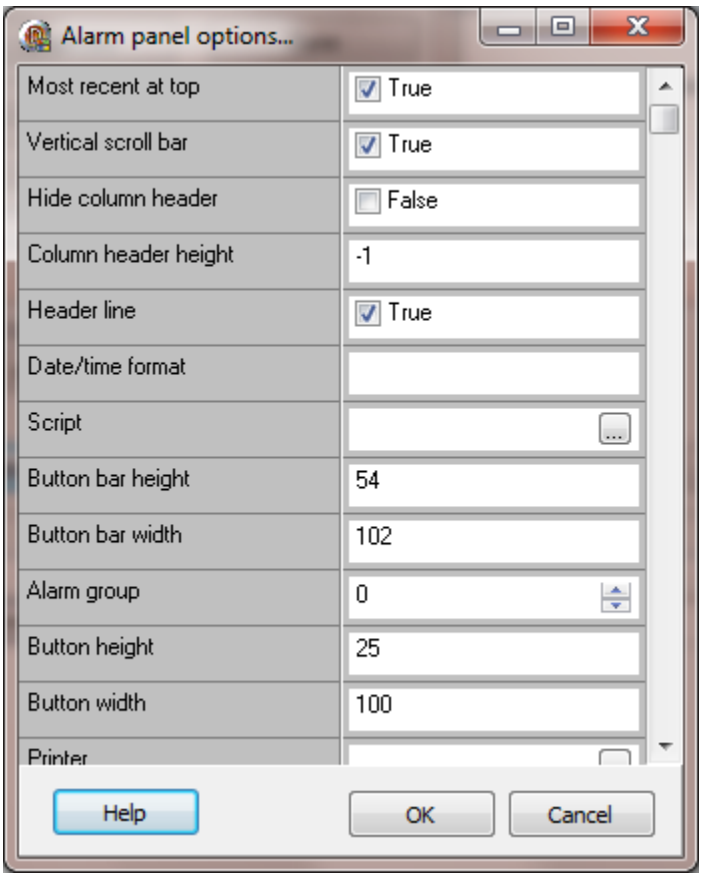
Fixed (website only)

The alarm panel will not scroll if the browser window is scrolled.

Alarm count

This property is for website pages. Web page table management is different than regular HMI graphic windows. This property defines the maximum active alarms displayed (1-512). row count = (Alarm count * rows per alarm) + column header.

Options



Most recent at top

The alarms will be listed with the most recent alarm at the top of the panel.

Vertical scroll bar

If there are more alarms than can be displayed in the panel area, a scroll bar will be displayed.

Hide column header

If enabled the header column will not be displayed.

Column header height

If the column header is not hidden, this is the height of the column header. If less than 1 the header will be the same height as the row height.

Header line

If enabled a line will be drawn between the header and the first row.

Date/time format

The format of the date/time field. A blank field uses the operation system configuration.

Script

If the button bar is enabled and the 'Execute script' button is enabled or the "Enable mouse up" property is enabled, this is the script that will be executed.

Button bar height

The height of the button bar if the 'bar position' is top or bottom.

Button bar width

The width of the button bar if the 'bar position' is left or right.

Alarm group(s)

Select the alarm groups to display. If this field is '0' (zero), 'all' alarm groups will be displayed. Also see "Priority filtering"

Button height

The height of each button. Default = 25

Button width

The width of each button. Default = 75

Printer

The printer to print the alarms if one of the below print buttons is selected.

Save path

The path to save the file if the 'Save' button is selected.

Save name

The script global containing the file name if the 'Save' button is selected. The file format is RTF (Rich Text Format).

Notes:
1) If the 'Save path' is blank the 'Save name' must be a fully qualified path.
2) If the script global does not exist or the file name is invalid, the save command will fail and an entry will be placed in the event log.
3) If a file exists at the path with the same name it will be overwritten.

Log all button actions

If enabled and one of the buttons is selected an entry will be placed in the event log.

Alternate shading

If enabled the background of each alarm section will be lighten or darkened.

Use alarm colors

If enabled the background of each alarm section will be the color selected for the alarm type. Other background color settings are ignored.

User level

The user level required to initiate button actions.

Blank acknowledge time

Unacknowledged alarms have a year of 1899 in the 'Acknowledge time' field. If enabled, the field will be blank if the alarm has not been acknowledged.

Left, top, right, bottom margin

When an anchor is enabled it 'locks' to a window side (top, left, bottom, right). If the margin value for the side is greater than 0 (> 0), the alarm panel side is locked to the side plus or minus the margin value.

For example, if all the anchors are set and the top margin is 20 the alarm panel will be flush will all the sides except the top. The top of the alarm panel will be 20 pixels from the top of the window.

Digital value display

The value of a digital point is 0 or 1 (true/false) when the alarm condition (falling/rising) is satisfied. By default, the "value at alarm" is displayed as a 0 or 1. This field allows for one of the digital alarm fields to be selected for display.

Alarm tagname (script global)

If a script global is specified, when the user left clicks the mouse button in the alarm panel, the tagname of the alarm will be placed in the script global.

Alarm type (script global)

If a script global is specified, when the user left clicks the mouse button in the alarm panel, the alarm type will be placed in the script global.

Value	Type
0	Lo Lo or Falling
1	Lo or Rising
2	Hi
3	Hi Hi

Alarm mask

If no mask flags are selected, all alarms are listed. The mask becomes "AND" logic if any flag is enabled.

Examples:
If "Unacknowledged' is enabled, only alarms, that are unacknowledged will be listed.
If "Unacknowledged' and "Analog points" is enabled, only analog point alarms that are unacknowledged will be listed.
If "Unacknowledged', "Analog points" and "Hi Hi" is enabled, only analog point Hi Hi alarms that are unacknowledged will be listed.

Priority

This can be used to filter alarms based on priority. For example, “Priority A” is set to 100. If a point alarm has a priority value of 22 and this setting is “>=” (greater than or equal), the alarm will not be displayed.

Value	Type
N/A	Not applicable, no filtering
>=	Alarm priority greater than or equal to Priority A
=	Alarm priority equal to Priority A
<=	Alarm priority less than or equal to Priority A
Between	Alarm priority >= Priority A and <= Priority B

Priority A/B

The value used for the priority filtering above.

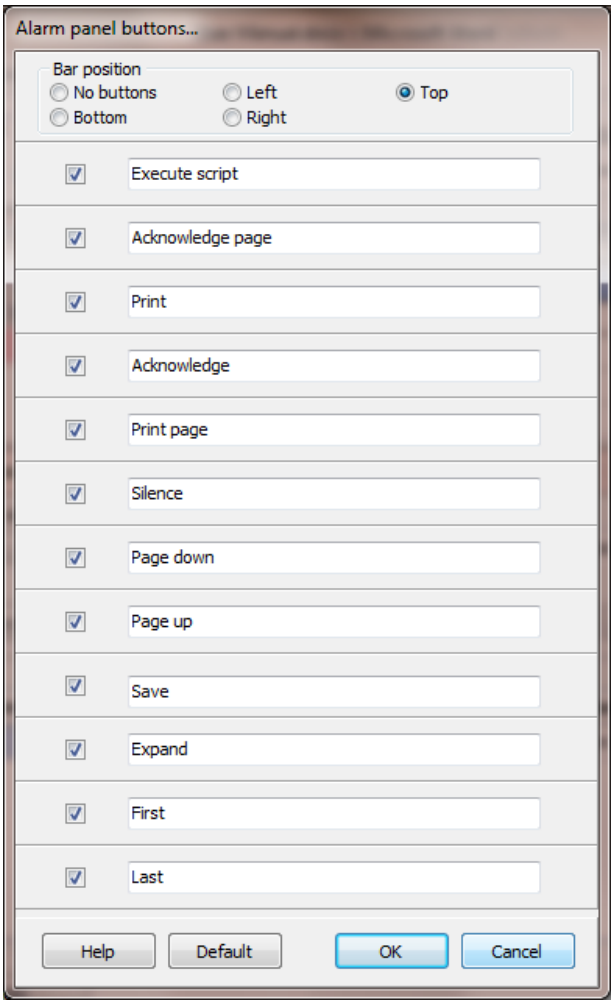
Enable mouse up

If enabled, when the left mouse button is pressed and released in the alarm panel, and the script property is assigned, the script will be queued for execution.

Cell draw script override

Used to override some properties of cell drawing. Use with caution. A long script or runtime errors will cause failure. The function should be in a "standalone" script for faster processing. See script function: OnAlarmPanelCellDraw for more information.

Alarm panel buttons



Bar position

The position of the button bar in the alarm panel. The bar will be placed on the top, bottom, right, left or not displayed.

Acknowledge page

This will set the alarm 'Acknowledge state' for the points currently displayed to 'Acknowledged'. **Note:** This will reset the 'Alarm pulse'.

Acknowledge

This is global 'Acknowledge' command.

Silence

This is the 'Silence' command.

Page up

If the alarm list has 'paged down' or ‘scrolled down' this will page up one page.

Page down

If the alarm list is not all the bottom, this will 'page down' one page or to the bottom.

Print page

This will send the current page to the printer selected in the options above.

Print

This will send all the alarms to the printer selected in the options above.

Save

This will save the alarms to the file specified in the options above.

Execute script

The script, selected above, will be queued for execution.

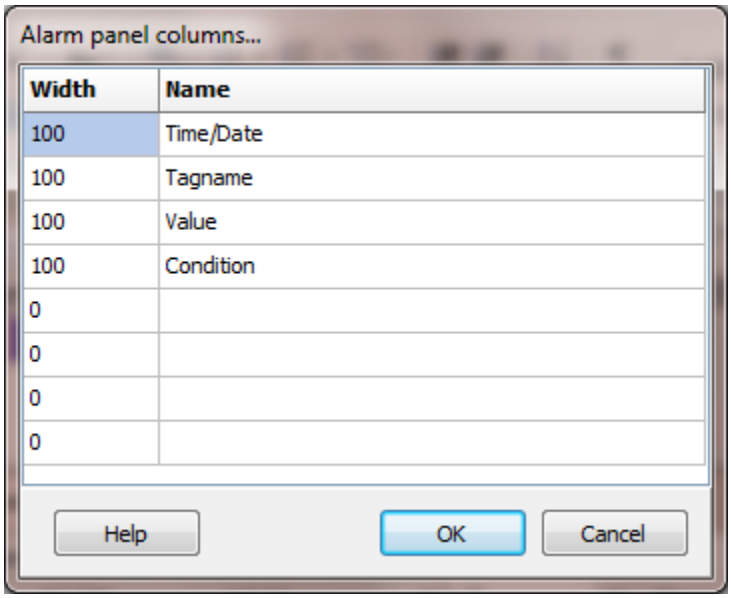
Expand

A dialog is displayed, for the selected alarm, with all the alarm fields for the point.

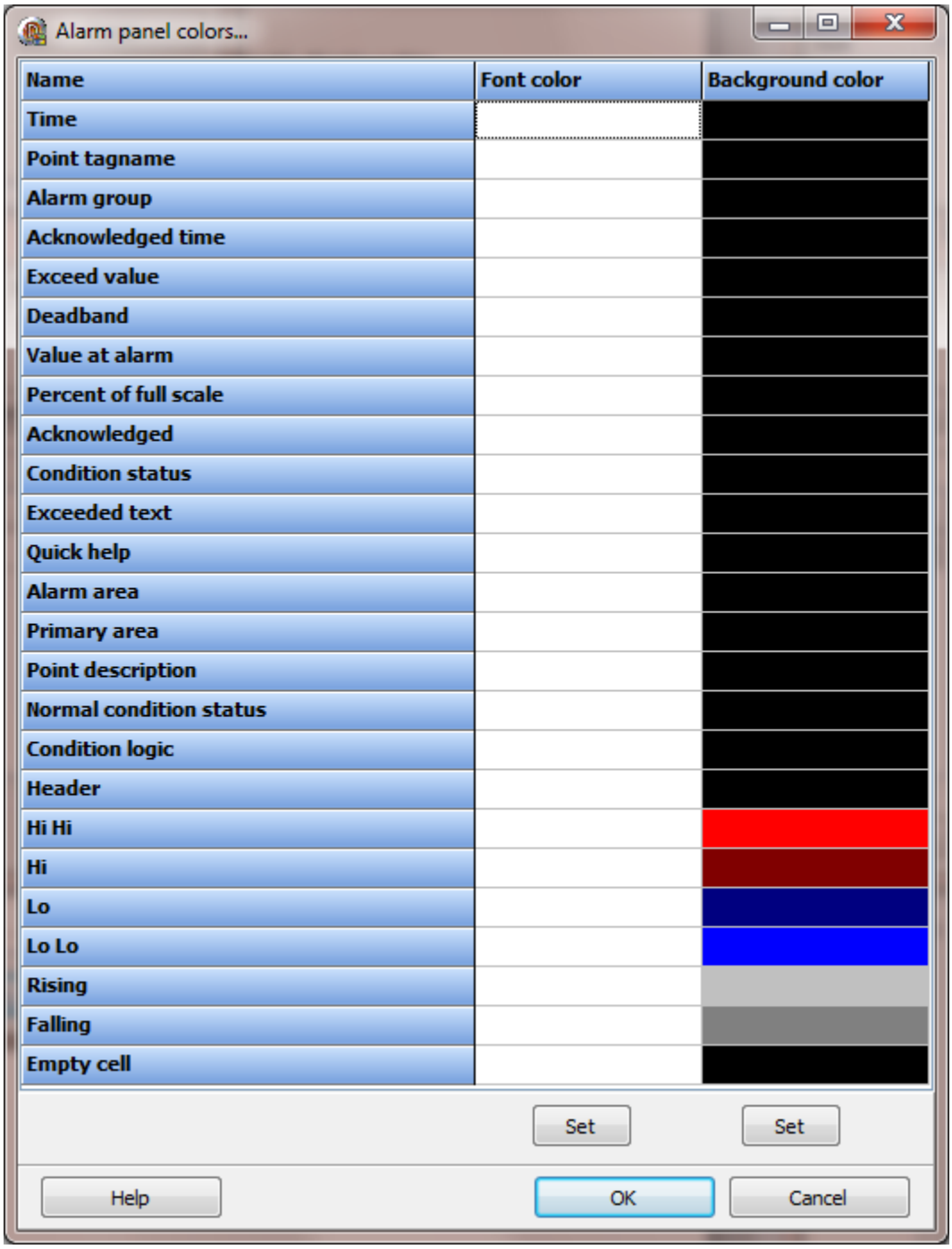
First/Last

The list will move to the first alarm in the list or last alarm in the list. (The list moves to the beginning or end.)

Column header editor

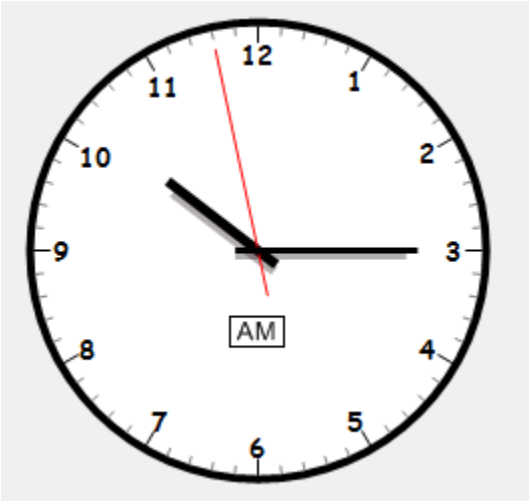


Colors editor

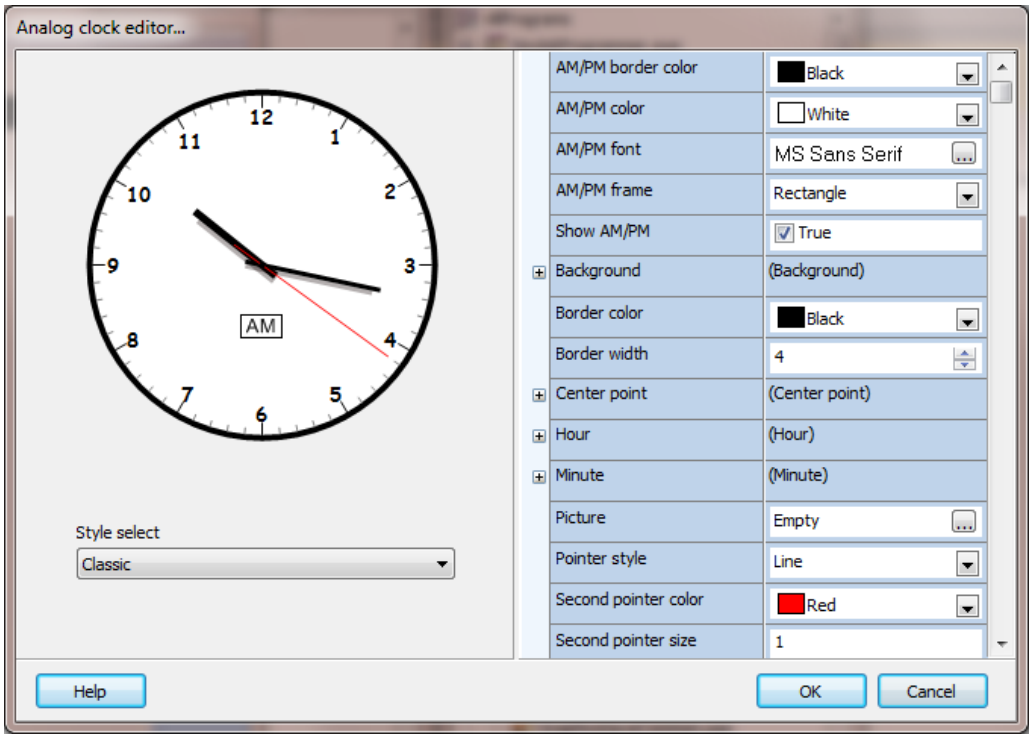


Back to list

ANALOG CLOCK



The analog clock editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Notes:

1) There are many color selections for this graphic element. Many of the colors operate in pairs. i.e. "background color" and "background color to". Not all color properties are listed in this help file. Use the configuration editor to change colors and view the effect of the color.

2) When changing values in the configuration editor, the change will not be reflected in the example clock until the left mouse button is clicked outside the property editor/item.

AM/PM border color

The color of the border for the AM/PM field.

AM/PM color — The background color of the AM/PM field.

AM/PM font — The font for the AM/PM field.

AM/PM frame — The frame shape for the AM/PM field.

Show AM/PM — If enabled the AM/PM field will be visible.

Background

- Color — The color of the background.
- Color to — The color “to” when used with a gradient fill.
- Gradient — The gradient style. If a picture is assigned, the gradient will not be applied.

Border color — The color of the border.

Border width — The width of the border. To not display a border set the width to 0.

Center point

- Border color — The color of the border.
- Color — The color center point.
- Outer border color

  The color of the outer border.
- Size — The size of the center point.

Hour

Font	The font used to render the hour numbers.
Mark color	The color of the hour tick marks.
Mark length	The length of the hour tick marks. If the value is 0 (zero), the length is determined by the size of the clock.
Mark width	The width of the hour tick marks.
Mark style	The hour tick mark style.
Pointer color	The color of the hour pointer.
Pointer shadow color	The shadow color of the hour pointer.
Pointer size	The size of the hour pointer.

Minute

Mark color	The color of the minute tick marks.
Mark length	The length of the minute tick marks. If the value is 0 (zero), the length is determined by the size of the clock.
Mark width	The width of the minute tick marks.
Pointer color	The color of the minute pointer.
Pointer shadow color	The shadow color of the minute pointer.
Pointer size	The size of the minute pointer.

Picture	If selected, a picture that is scaled to fill the background rectangle of the clock.
Pointer style	The style of the hour and minute pointer.
Second pointer color	The color of the "seconds" pointer.

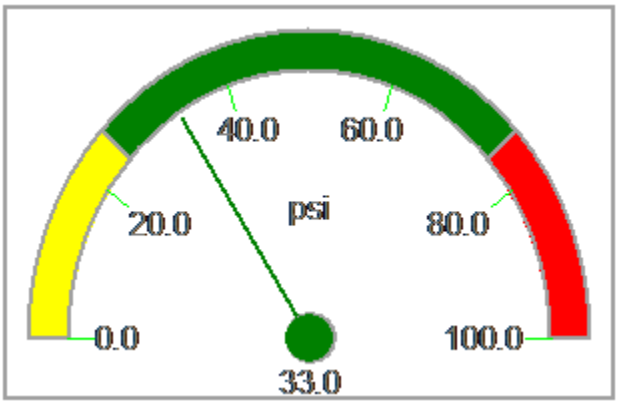
Second pointer size	
	The size of the "seconds" pointer.
Shape	This shape of the clock.
Size	This size of the clock. If the value is 0 (zero), the size of the clock is determined by the size of the bounding rectangle.
Tick marks	The type of tick mark style.
Show seconds	If enabled the "second" pointer will be displayed.
Show numbers	If enabled the “hour” numbers will be displayed.

Back to list

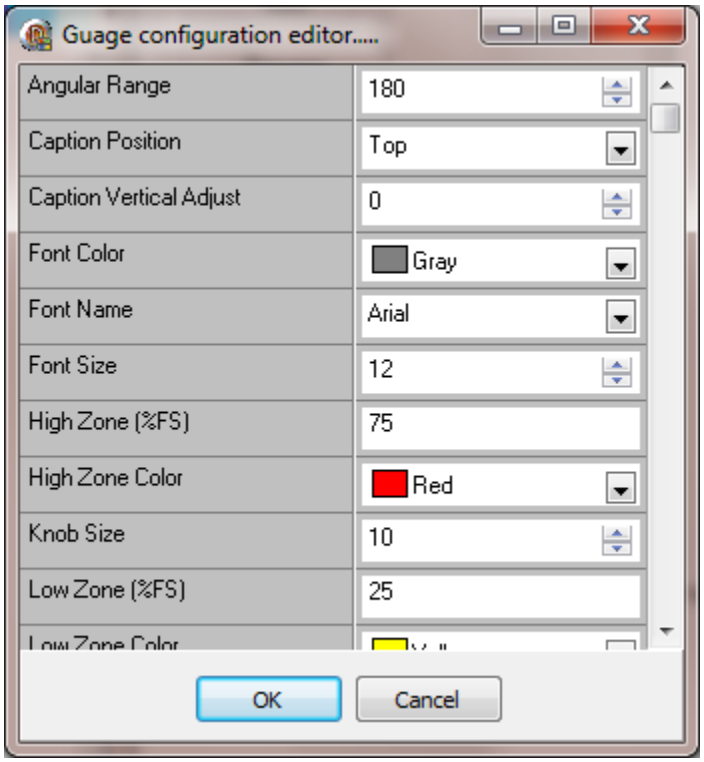
Analog grid

All of the analog grid settings are accessed via the analog grid animation dialog.

Angle gauge 1



The angle gauge 1 editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Angular range

The arc range. The value is from 10 to 360

Caption position

The position of the “units” text.

Caption vertical adjust

An offset to the vertical position of the “units” text.

Font color

The font color for the element.

Font name

The font type for the element.

Font Size

The size of the font for the element.

High, Low, OK Zone & Color

The color of the arc on the scale indicating the low, OK, and high range and the color. Set the low and high value and the range between the low and the high is the 'OK' range. Use the color setting to indicate the range.

Knob size

The size of the circle in the middle of the gauge.

Needle width

The width of the needle.

Show frame

A frame is displayed around the gauge.

Show tick scale

Display the values for the tick marks.

Show ticks

Display the tick marks.

Show value

Display the value of the needle.

Show zones

The arc is filled with the color selected for the three zones. Also see “Transparent zones”.

Tick count

The number of scale ticks. 3 would put a tick at the start, the middle and the end.

Tick digits

The number of digits to display for the tick values.

Tick precision

The precision of the tick values displayed.

Transparent zones

If enabled the zones are not drawn with the zone color. The arc is drawn and filled with the background color.

Units

This is the string for the caption.

Use inherited units

This overrides the "Units" above and uses the units from the point assigned via animation.

Use inherited limits

This overrides the "Value Maximum" and "Value Minimum" below and uses the engineering min/max from the point assigned via animation.

Value

The current value of the input.

Value digits

The number of digits for the value field.

Value maximum

The upper range of the monitored value. Used to display the scale and needle.

Value minimum

The lower range of the monitored value. Used to display the scale and needle.

Value precision

The precision of the value text string.

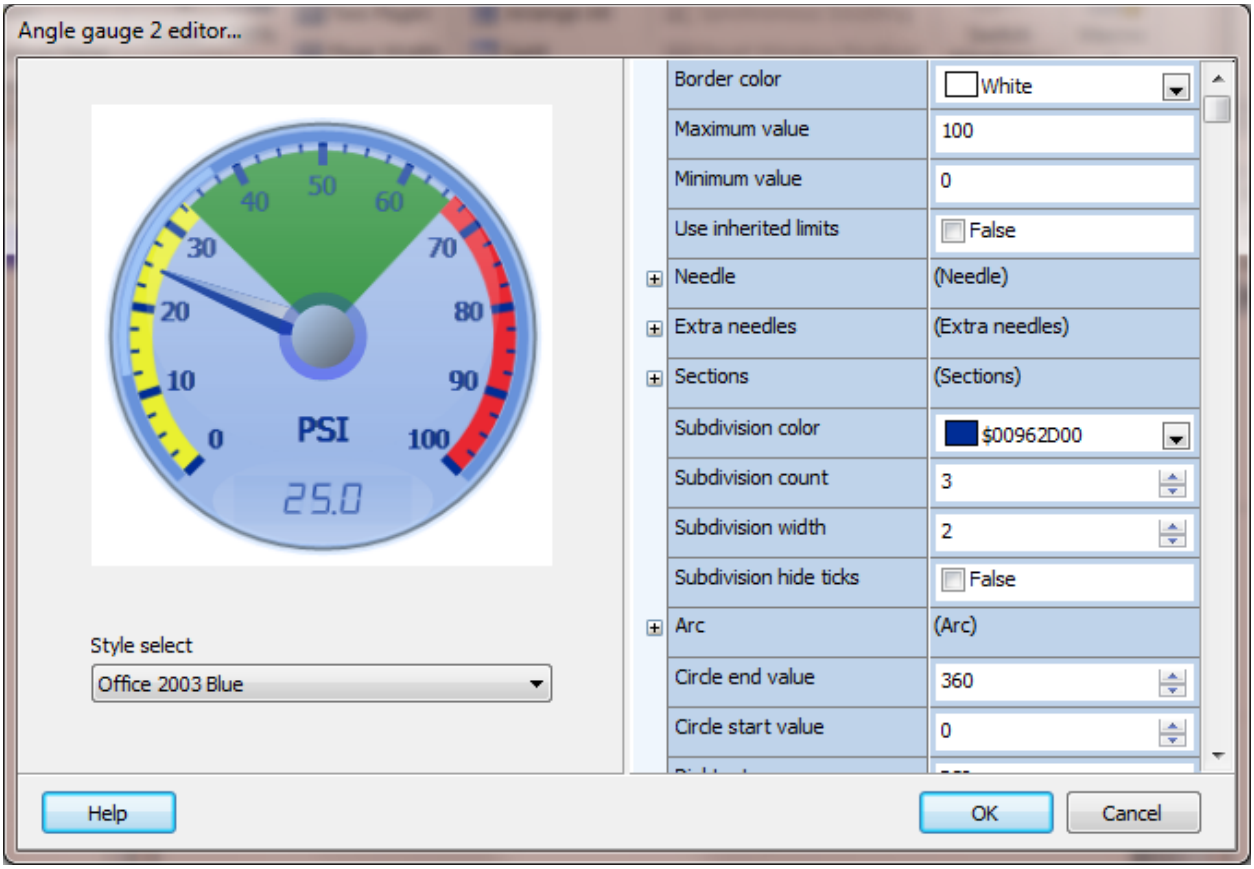
Value vertical adjust

An offset to the vertical position of value text.

Back to list

ANGLE GAUGE 2

The angle gauge 2 editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”. Website angle gauge 2 is here.



Notes:
1) There are many color selections for this graphic element. Many of the colors operate in pairs. i.e. "Inner center color" and "Inner center color to". Not all color properties are listed in this help file. Use the configuration editor to change colors and view the effect of the color.
2) Opacity is the "blending" or "transparency" of a color/object and what is below the object. A value of 0 is transparent (the below colors will be fully visible) and a value of 255 is opaque (the below colors will be not be seen). Not all "opacity" properties are listed in this help file. Use the configuration editor to change the opacity level and view the effect.
3) When changing values in the configuration editor the change will not be reflected in the example gauge until the left mouse button is clicked outside the item editor.

Border color

The color outside the circle of the gauge.

Maximum value

The upper range of the monitored value. Used to display the scale and needle.

Minimum value

The lower range of the monitored value. Used to display the scale and needle.

Use inherited limits

This overrides the "Maximum value" and "Minimum value" above and uses the engineering min/max from the point assigned via animation. If enabled the limits are collected from the source for needle 1.

Needle

This gauge can display 1 to 3 needles. Needle 1 cannot be removed. Needles 2 & 3 are optional.

The needle must be enabled to display. To see the needle position for needle 1 at design-time, change the ''Value" field below.

Needle one also contains the settings for the center to circle.

If the value text is enabled, it always displays the value of needle 1.

Extra needles

This gauge can display 2 additional needles.

Sections

This gauge can display 3 sections. Sections are used to show a range in the gauge. For example low, normal and high.

Subdivisions/divisions

These are the marks along the outer edge of the gauge.

Font

The font settings for the “Dial text”.

Value font

The font settings for the numbers at the division tick marks.

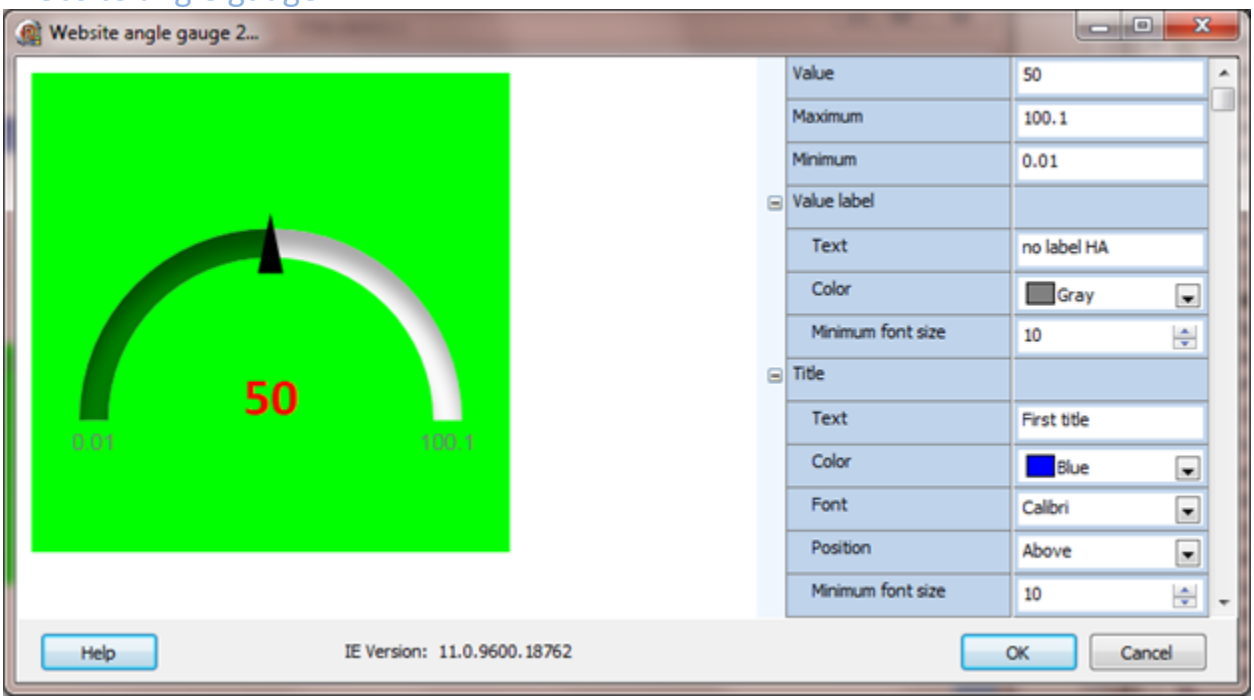
Select style

This sets some colors of the gauge to the style selected, when selected. All the colors of the gauge can continue to be changed regardless of selecting a style.

Note: When a selection is made and the colors are set in the gauge. The colors of the gauge are not compared to determine the style. The combo is used for selection, it is not an indicator. 'Custom' has no color settings.

Back to list

Website angle gauge 2



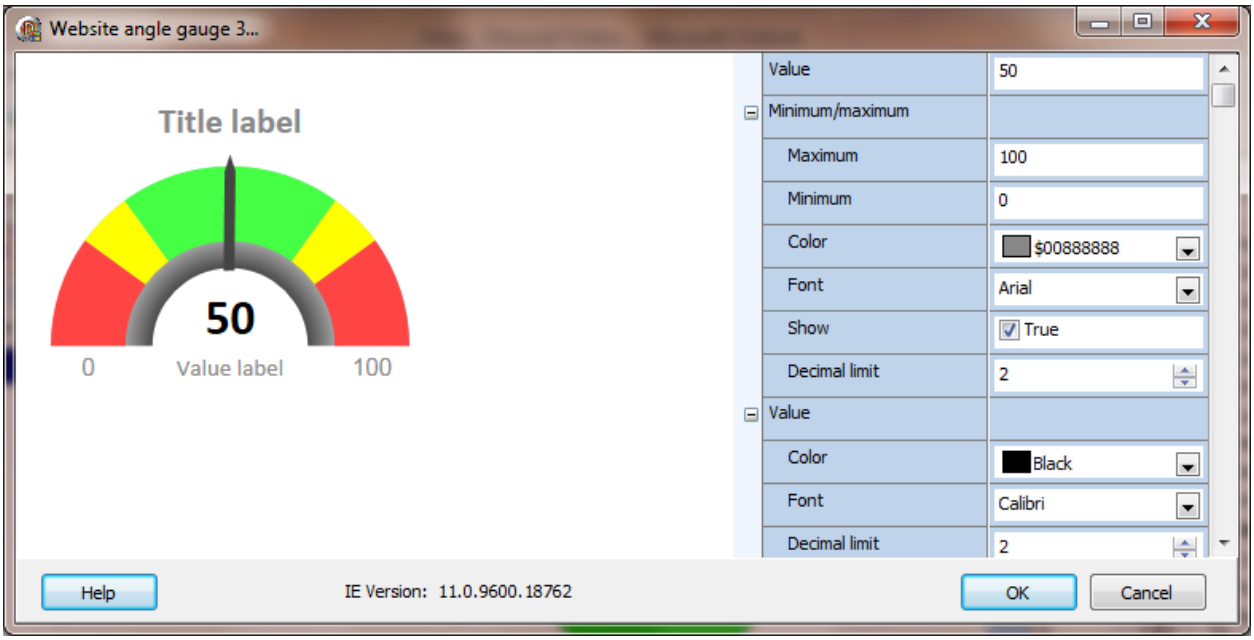
Note:

1. The sample gauge requires Microsoft Internet Explorer (IE) . If IE is not installed the sample gauge will not display the settings. The settings can be altered and saved. The gauge will display in a browser at runtime.
2. When changing values in the configuration editor the change will not be reflected in the example gauge until the left mouse button is clicked outside the item editor.

Value	The gauge value is used for viewing the gauge in the sample panel and the initial value when the gauge is created.
Maximum	The maximum range of the gauge.
Minimum	The minimum range of the gauge.
Value label	
Text	The text to appear below the value. The property can be empty. The font is Arial.
Color	The color of the value label text.
Minimum font size	The gauge adjust the size of the text as the gauge size changes. This defines the minimum font size.
Title	
Text	The text to appear for the title. The property can be empty.
Color	The color of the title text.
Font	The title font name.
Position	The title can be rendered above or below the gauge.
Minimum font size	The gauge adjust the size of the text as the gauge size changes. This defines the minimum font size.
Value	

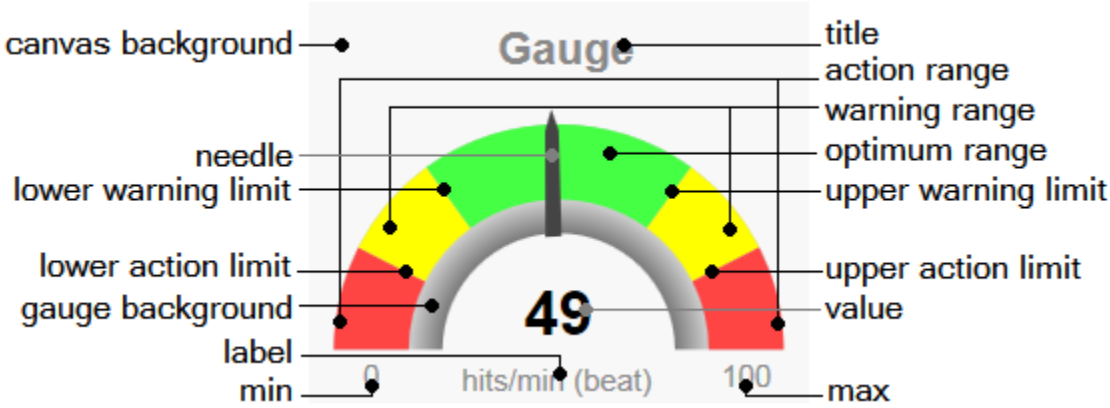
Decimal count	The number of decimals for the value text.
Color	The color of the value text.
Font	The value text font name.
Minimum font size	The gauge adjust the size of the text as the gauge size changes. This defines the minimum font size.
Hide	If enabled the value text will not be rendered.
Symbol	The text to appear after the value text. This property can be empty.
Min/Max	
Hide	If enabled the minimum and maximum text will not be rendered.
Max/Min minimum font size	The gauge adjust the size of the text as the gauge size changes. This defines the minimum font size.
Human friendly	The value label can render the value text in “friendlier” terms. For example, 12345 would be rendered as 12.3K.
Decimal count	The number of decimals for the value text when “human friendly” is enabled.
Level colors	The gauge can alter the arc color based on the gauge value and the level color. To set the gauge arc to one color, set the level 1 color to the desired color and all other levels to “None”. The levels are defined as the percent of full scale.
Additional	
Reverse	The default arc is left to right. Enable this property for right to left.
No gradient	The arc default is rendered with a gradient fill. Enable this property to render as a solid color.
Hide inner shadow	If this property is enabled the arc inner shadow will not be rendered.
Donut	The gauge can be rendered as a complete arc. See start angle.
Start angle	When the “donut” property is enabled this property defines the start of the arc. 0° is 9 o’clock.
Background color	This property defines the background color of the gauge.
Width scale	This property defines the width of the arc. (0.0 – 1.0)
Shadow opacity	This property defines the opacity of the shadow, if enabled. (0.0 – 1.0)
Shadow size	This property defines the shadow width.
Shadow vertical offset	This property defines the shadow offset. A value of 0 would be no offset, no shadow.
Fixed	This property defines if the gauge has a fixed location on the page or will move on scrolling.
Pointer	
Enable	If enabled the pointer will be rendered using the settings below.
Top length	The top of the pointer from the top of the arc. The value can be negative.
Bottom length	The bottom of the pointer from the bottom of the arc. The value can be negative.
Bottom width	The width of the point base.
Fill color	The color of the pointer.
Stroke color	The color of the pointer pen.
Stroke width	The width of the pointer pen.
Stop cap	The end cap of the pointer lines.

Website angle gauge 3



Note:

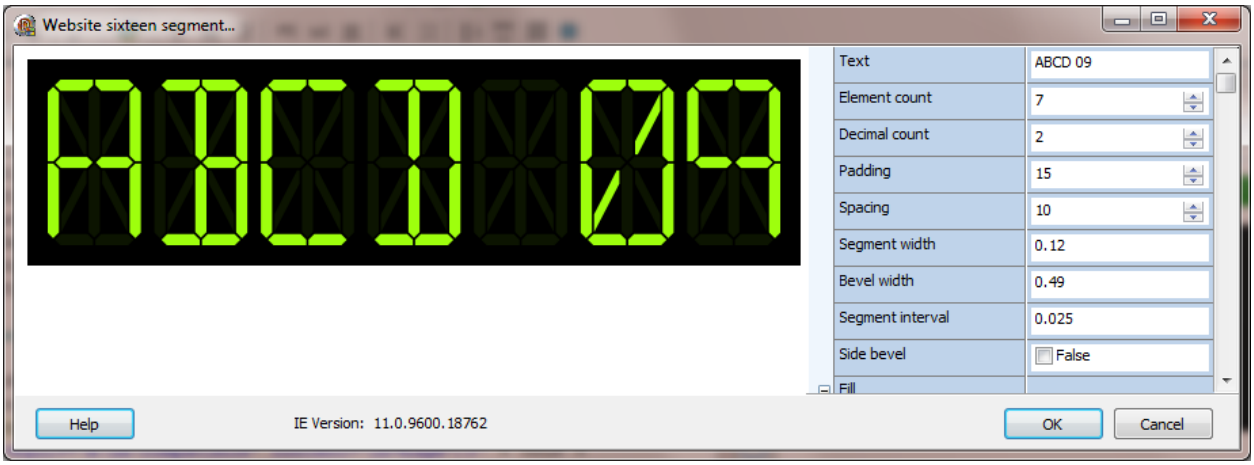
1. The sample gauge requires Microsoft Internet Explorer (IE) . If IE is not installed the sample gauge will not display the settings. The settings can be altered and saved. The gauge will display in a browser at runtime.
2. When changing values in the configuration editor the change will not be reflected in the example gauge until the left mouse button is clicked outside the item editor.



Value	The gauge value is used for viewing the gauge in the sample panel and the initial value when the gauge is created.
Maximum/Minimum	
Maximum	The maximum range of the gauge.
Minimum	The minimum range of the gauge.
Color	The color of the maximum/minimum label text.

Font	The maximum/minimum font name.
Show	Render the maximum/minimum values as text
Decimal limit	The maximum count of decimal digits.
Value	
Color	The color of the value text.
Font	The value font name.
Decimal limit	The maximum count of decimal digits.
Value label	
Text	The text to appear for the value label. The property can be empty.
Color	The color of the value label text.
Font	The value label font name.
Title	
Text	The text to appear for the title. The property can be empty.
Color	The color of the title text.
Font	The title font name.
Limits	
Lower action limit	Arc color change value for lower action level.
Lower warning limit	Arc color change value for lower warning level.
Upper warning limit	Arc color change value for upper warning level.
Upper action limit	Arc color change value for upper action level.
Action range color	The arc color of the action range.
Optimum range color	The arc color of the optimum range.
Warning range color	The arc color of the warning range.
Needle color	The needle color.
Gauge width scale	Larger values create a wider gauge arc.
Gauge border width	The width of the pen around the gauge arc.
Gauge shadow scale	Larger values create wider shadow.
Show gauge shadow	If enabled the arc shadow will be visible.
Gauge border color	The color on the border of the gauge arc.
Gauge back color	The lower arc color.
Gauge shadow color	The shadow color, if visible.
Fixed	This property defines if the gauge has a fixed location on the page or will move on scrolling.

Sixteen segment (website)

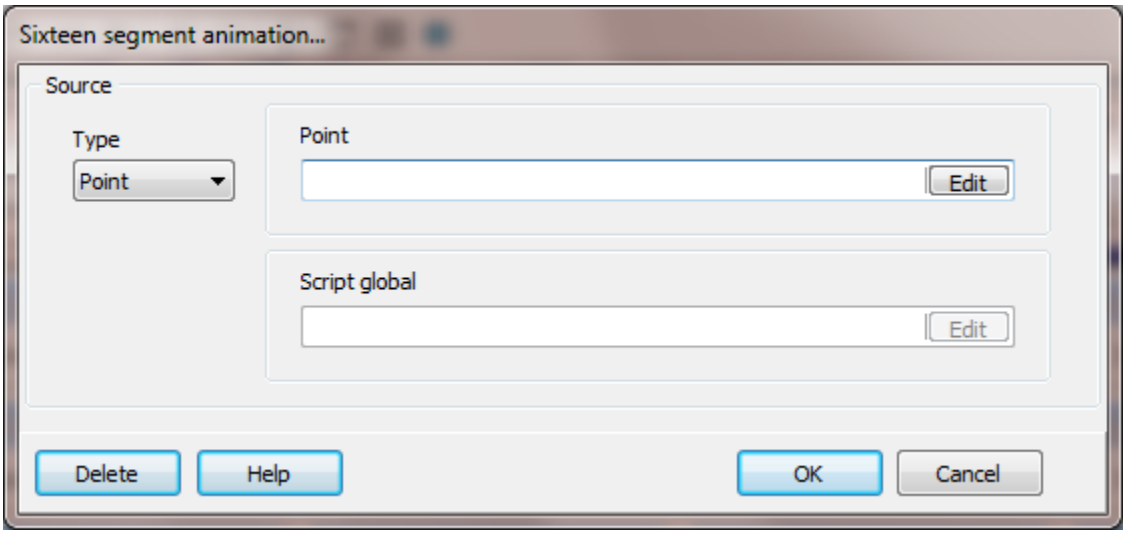


Note:

1. The sample gauge requires Microsoft Internet Explorer (IE) . If IE is not installed the sample gauge will not display the settings. The settings can be altered and saved. The gauge will display in a browser at runtime.
2. When changing values in the configuration editor the change will not be reflected in the example gauge until the left mouse button is clicked outside the item editor.
3. The gauge can display numbers and text (A..Z).

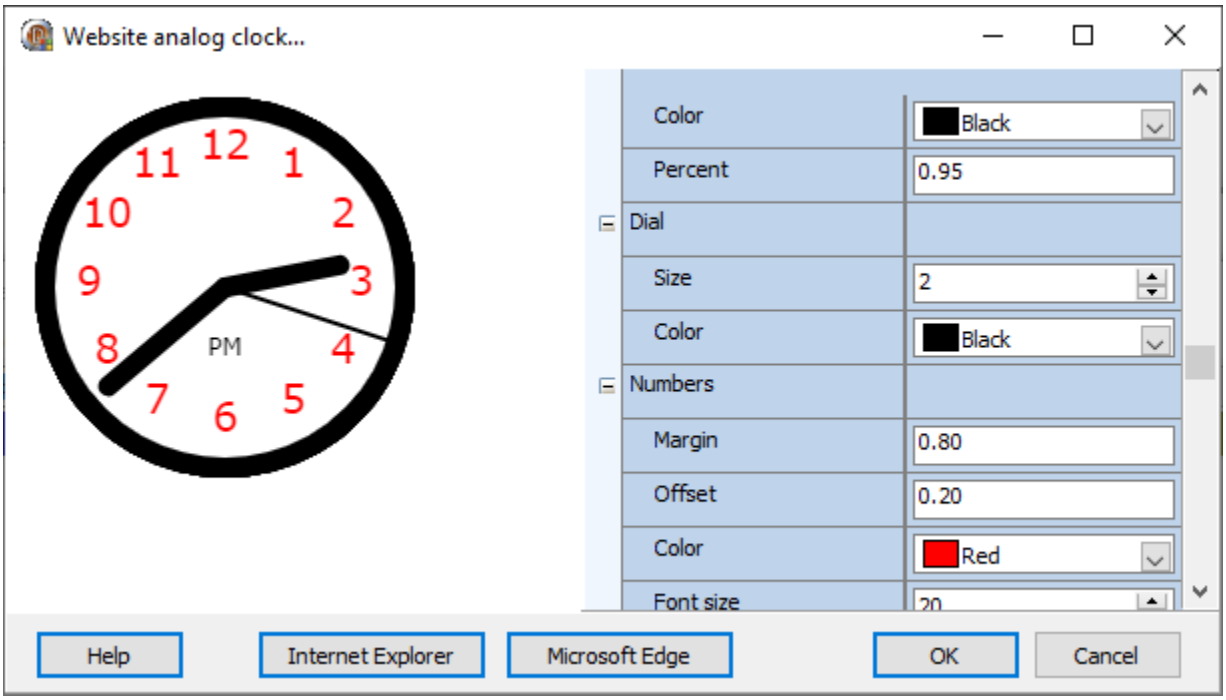
Text	The “text” is used for viewing the gauge in the sample panel and the initial value when the gauge is created.
Element count	The number of character positions.
Decimal count	The number of decimals when the source type is “point”.
Padding	The padding around the display.
Spacing	The spacing between the elements.
Segment width	The width of the segments. (% of element width)
Bevel width	The size of the corner bevel. (% of element width)
Segment interval	The space between the elements. (% of element width)
Side bevel	If true, the sides will be beveled.
Fixed	This property defines if the gauge has a fixed location on the page or will move on scrolling.
Fill	
Light	The color of an “on” segment.
Dark	The color of an “off” segment.
Stroke	
Light	The color of an “on” segment outline.
Dark	The color of an “off” segment outline.
Width	The outline width.
Size	
Height	The graphic element height.
Width	The graphic element width.

Sixteen segment animation (website)



Type	The display value source can be a point or script global.
Point	Select the point source.
Script global	Select the script global source.

Website analog clock



The website analog clock editor is accessed via the “Object/Edit” menu Item.

Notes:

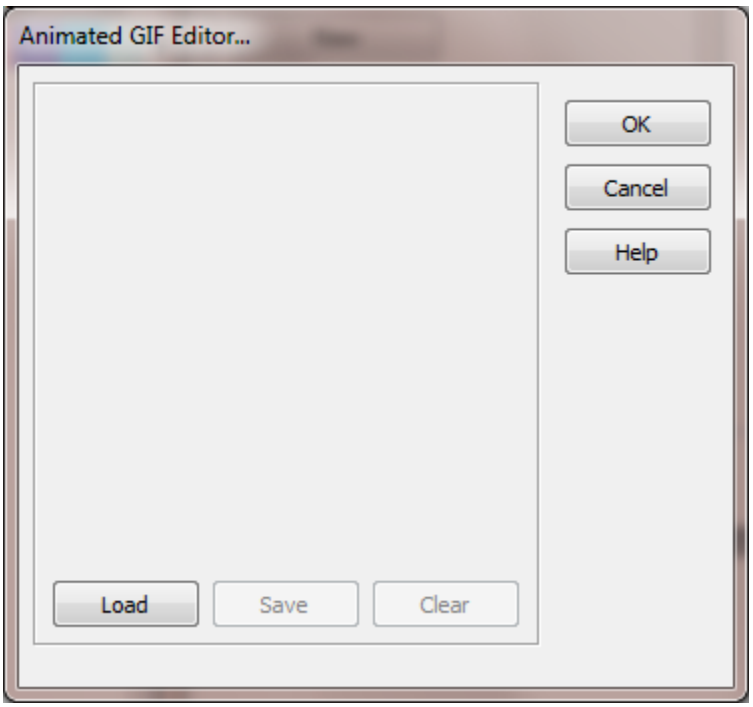
1. The sample clock requires Microsoft Internet Explorer (IE) or Microsoft Edge (version 79 or greater). If IE/ME is not installed the sample clock will not display the settings. The settings can be altered and saved. The clock will display in a browser at runtime.
2. When changing values in the configuration editor the change will not be reflected in the example clock until the left mouse button is clicked outside the item editor.
3. For the best results the clock width and height should be equal, a square bounding rectangle.

Background color	This property defines the background color.
Border color	This property defines the border color.
Border margin	This clock size is defined by a rectangle. This property defines the clock edge to the rectangle edge. A value of zero might cause a portion of the clock edge to not be visible.
Border width	This property defines the border width.
Hand end	This “ends” of the hour, minute and second hand can be squared or rounded.

Fixed	The clock position will not scroll if the browser window is scrolled. (Website only)
Hour, minute, second	
Width	This property defines the width of the hand (in pixels).
Color	This property defines the color of the hand.
Percent	This property defines the length of the hand in percent of radius of the clock.
Dial size	This defines a circle that covers the hands in the middle of the clock. A value of 0 (zero) and the dial will not be visible.
Dial color	This property defines the dial color, if visible.
Numbers margin	This property, in percent of clock diameter, defines the position of the numbers to the clock border.
Numbers offset	Due to differences in font sizing and clock diameter the vertical alignment of the numbers might need to be shifted down. This property, in percent of clock diameter, moves the numbers down. A good judge of correct alignment is when the second hand is on 3 (15 seconds) and the second hand is in the middle of the 3 character.
Number color	This property defines the number color.
Number font size	This property defines the font size of the numbers.
Number font name	This property defines the font family of the numbers.
Image	This property defines an optional image to use for the clock background. If an image is defined only the image, the clock hands and the AM/PM (if visible) will be rendered.
AM/PM visible	This property defines if the AM/PM indicator is rendered.
AM/PM offset	This property defines AM/PM indicator vertical offset from center of the clock. The value is in percent.
AM/PM frame	This property defines if a graphic element is rendered around the AM/PM text. The options are “None”, “Rectangle” and “Round Rectangle”.
AM/PM frame color	This property defines the color of the AM/PM frame.

AM/PM text color	This property defines the text color of the AM/PM indicator.
AM/PM text	This property defines text from AM (midnight to noon) and PM (noon to midnight).
AM/PM font	This property defines the font size of the AM/PM text.
AM/PM font name	This property defines the font family of the AM/PM text.

Animated GIF



The animated gif editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.

Note:

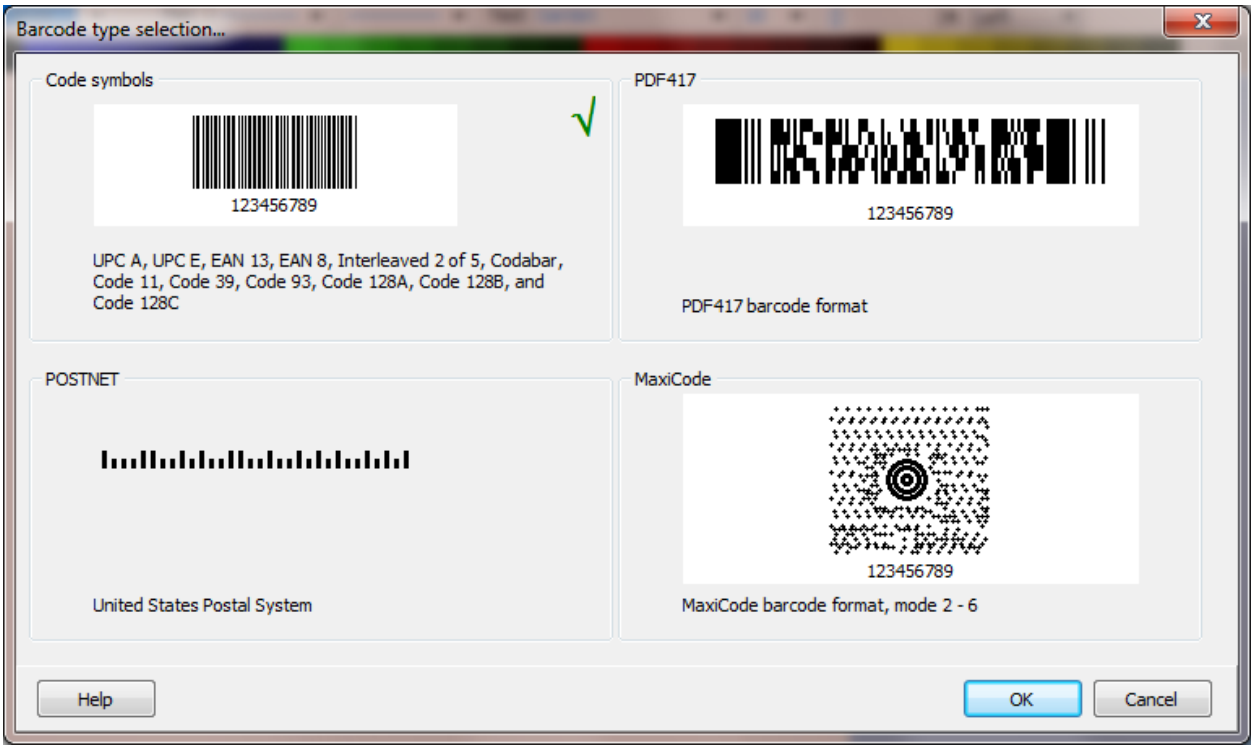
The animated GIF, at design time, has a transparent background. The animated GIF at runtime does not have a transparent background.

In configuration, the GIF will play when the element is selected. When moving the element in the window it may not display correctly until the element is unselected.

Load	Select the load button to import a graphic file.
Save	Select the save button to export the GIF.
Clear	Select the clear button to erase the image.
OK	Select the OK button to accept the settings. When the OK button is selected the element will be sized to the dimensions of the GIF. The element can then be sized as needed.
Cancel	Select the cancel button to not make any changes to the element.

Back to list

BARCODE



The HMI supports four types of barcodes.

Barcode symbols: UPC A, UPC E, EAN 13, EAN 8, Interleaved 2 of 5, Codabar, Code 11, Code 39, Code 93, Code 128A, Code 128B, and Code 128C

POSTNET: United States Postal System for routing mail

PDF417 barcode format

MaxiCode: mode 2 – 6

If QR (Quick Response Code) barcodes are needed, contact support for assistance.

Settings

The barcode types have some attributes in common and some attributes are unique for the type of barcode selected. To provide a central location to manage bar code attributes and apply the attributes when drawing and printing, a script global is used for attribute storage. The script global can be used for one or many barcodes.

A matrix of attributes and which barcode type uses the attribute is below. A description of each attribute is after the table.

	Barcode symbols	POSTNET	PDF417	MaxiCode
AddCheckChar	•			
Alignment			•	•
AutoScale				•
BackgroundColor[1]			•	•
BarCodeType	•			
BarColor[1]	•		•	
BarHeight			•	•
BarHeightToWidth			•	•
BarToSpaceRatio	•			
BarWidth	•		•	•
BearerBars	•			
BitmapHeight	•	•	•	•
BitmapWidth	•	•	•	•
Caption			•	•
CaptionLayout			•	•
CarrierCountryCode				•
CarrierPostalCode				•
CarrierServiceClass				•
Code[2]	•	•	•	•
Code128Subset	•			
ECCLevel			•	•
ExtendedSyntax	•			
Filename	•	•	•	•
Fixed[3]	•	•	•	•
HorPixelsPerMM				•
Mode				•
NumColumns			•	
NumRows			•	
Printer	•	•	•	•
PrintHeight	•	•	•	•
PrintOnClick	•	•	•	•
PrintOnClickPrompt	•	•	•	•
PrintXOffset	•	•	•	•
PrintYOffset	•	•	•	•
QuietZone			•	•
RelativeBarHeight			•	•
ShowCode	•			
ShowGuardChars	•			

SupplementalCode	•			
TallGuardBars	•			
Truncated			•	
VerPixelsPerMM				•

1. The “BarColor” and “BackgroundColor” are set with the graphic element foreground and background color selection when the barcode is a graphic element in a window. The script global attribute is used when the barcode saves to a picture.

2. The “code” is only used when the barcode saves to a picture. To set the code via scripting, set the “text” property of the graphic element. Otherwise, the code is obtained from the source property.

3. Website only.

These are the attribute definitions for the barcodes. The default value follows the name.

NOTE: All attribute names and value names ARE case sensitive.

AddCheckChar (false)

If AddCheckChar is “True”, a check character is computed and added to the displayed symbol. This property applies to BarCodeType values of bcInterleaved2of5, bcCode11, bcCode93, and bcCode128. It is ignored by all others.

Alignment (taCenter)

This attribute specifies the alignment of the barcode and the caption. The alignment property controls the horizontal placement of the barcode and the caption within the graphic element rectangle.

Value	Meaning
taLeftJustify	Text is left-justified: Lines all begin at the left edge of the control.
taCenter	Text is centered in the control.
taRightJustify	Text is right-justified: Lines all end at the right edge of the control.

Autoscale (True)

This attribute controls if the barcode is automatically sized. When this property is “True”, the barcode will calculate the size of the barcode based on the pixels per millimeter of the canvas that the barcode is painted on. The normal size of a MaxiCode barcode is 25.50mm by 24.37mm, about one inch square. If the scaling of the barcode needs to be done manually, setting AutoScale to “False” and either specifying BarHeight and BarWidth or HorPixelsPerMM and VerPixelsPerMM will handle it. BarHeight and BarWidth are used to specify the size of a single hexagonal element in pixels. HorPixelsPerMM and VerPixelsPerMM are used to manually specify the number of pixels per millimeter. When the barcode is manually scaled, the values for BarHeight and BarWidth are checked first. If these are non-zero, they will be used to calculate the size of an element. If they are zero, HorPixelsPerMM and VerPixelsPerMM will be used. The checks are done independently. For example, BarHeight can be used with HorPixelsPerMM.

BackgroundColor (clWhite)

Sets the background color for the barcode. See note 1 above.

BarCodeType (bcUPC_A)

bcUPC_A, bcUPC_E, bcEAN_8, bcEAN_13, bcInterleaved2of5, bcCodabar, bcCode11, bcCode39, bcCode93, bcCode128

BarColor (clBlack)

Sets the color of the bars in the barcode. See note 1 above.

BarHeight (0)

Specifies the height of a single element. In PDF417, if RelativeBarHeight is False and BarHeightToWidth is zero, this value will be used to determine the bar height in pixels. In MaxiCode, this will value will be used to calculate the size of a hexagon when AutoScale is False.

BarHeightToWidth (3)

Sets the aspect ratio of the bars in stacked barcodes. In stacked bar symbologies like PDF417, BarHeightToWidth sets the height of a single row in relation to the width of the narrowest bar. For example, the default value of 3 will set the height of a row to 3 times the width of the narrowest bar. In most cases, a value of 3 or 4 is recommended. If RelativeBarHeight is “True”, this property will be ignored. Instead, the height of the bars in the symbol is calculated based on the size of the control.

BarToSpaceRatio (1)

Determines the ratio of bar and space width. This option is forced to 1 for BarCodeType values of bcInterleaved2of5, bcCode11, bcCode39, bcCode93, and bcCode128 since these symbologies require a 1:1 bar to space ratio.

BarWidth (12)

Sets the size of the smallest element in the barcode. For stacked bar symbologies like PDF417, BarWidth is used to indicate the width of the narrowest bar in the code. For MaxiCode, this value will be used to calculate the size of a hexagon when AutoScale is False.

Determines the width of the narrow bar in a symbology. BarWidth is given in mils (thousandths of an inch).

Note: For PDF417 the bar width default is 2.

BearerBars (False)

Determines if a top and bottom bar is drawn with the bar code symbol. This property is ignored unless the BarCodeType is bcInterleaved2of5.

BitmapHeight (75)

This attribute is the height of the saved bitmap when the barcode is saved to a file via the BarCode script command.

BitmapWidth (280)

This attribute is the width of the saved bitmap when the barcode is saved to a file via the BarCode script command.

Caption (No caption (blank))

Specifies a caption to be displayed with the barcode. Since two-dimensional barcodes can hold large amounts of data, setting the caption to be the same as the encoded data may not be practical. Descriptive text, as opposed to the literal data, should be used. Note: For PDF417 and MaxiCode, if the caption value is 0 the caption will not be drawn.

CaptionLayout (tlBottom)

Positions the caption in relation to the barcode. The CaptionLayout property indicates where the caption will be position in relation to the barcode. Caution: A CaptionLayout of “tlCenter” will center the caption in the barcode. This may lead to an unusable barcode.

Value	Meaning
tlTop	The text appears at the top of the control.
tlCenter	The text is vertically centered in the control.
tlBottom	The text appears along the bottom of the control.

CarrierCountryCode (0)

Specifies the country code to be used in mode 2 and mode 3 barcodes. Country codes are three digit numeric values in accordance with ISO 3166.

CarrierPostalCode (No value (blank))

Specifies the postal code to be used in mode 2 (numeric postal code) and mode 3 (alphanumeric postal codes) barcodes. Postal codes can either be nine numeric digits or six alphanumeric characters. Alphanumeric postal codes longer than six digits will be truncated. If the postal code is numeric and the barcode's mode is mode 3 (alphanumeric) the barcode's mode will be changed automatically to numeric. If the postal code is alphanumeric and the barcode's mode is 2 (numeric), the mode will be changed automatically to alphanumeric.

CarrierServiceClass (0)

Specifies the service class used in mode 2 and mode 3 barcodes. The service class is a three digit numeric value that is determined by the carrier.

Code (No value (blank))

Determines the data that is displayed as the bar code symbol. The “code” is only used when the barcode is printed/saved via the BarCode script command. To set the code via graphic scripting, set the “text” property of the graphic element. Otherwise, the code is obtained from the source property. Note: For POSTNET, the length of the code must be 5, 9 or 11 characters.

Code128Subset (csCodeA)

csCodeA, csCodeB, csCodeC The property is ignored unless the BarCodeType is bcCode128.

ECCLevel (ecAuto/0)

Specifies the Reed-Solomon error correction level to use. ecLevel0 denotes the least amount of error correction, with ecLevel8 being the maximum. ecAuto will enable the bar code engine to use the optimal error correction for the amount of data. For the numbers of errors that can be detected and fixed, please see the PDF417 standards.

ecAuto, ecLevel0, ecLevel1, ecLevel2, ecLevel3, ecLevel4, ecLevel5, ecLevel6, ecLevel7, ecLevel8

MaxiCode: The value is an integer and defaults to 0.

ExtendedSyntax (False)

Sets capability to change Code 128 symbology. Setting ExtendedSyntax to “True” provides the ability to easily change the Code Set in the middle of the bar code. This capability is only available for bar codes with a BarCodeType of bcCode128; it is ignored for all others. This capability is provided by an escape character, the backslash '\'. Preceding particular characters in the Code property with the backslash tells the bar code engine to switch Code Sets before continuing with encoding. \A will switch to Code Set A, \B to Code Set B, and \C to Code Set C. The double backslash, \\, will encode a real backslash in the bar code. If the bar code engine finds a backslash that is not followed by one of these four characters, it will be suppressed. Escaped codes are case insensitive. For example, \a and \A will both cause a switch to Code Set A in the middle of a Code string.

Filename (No value (blank))

The file name and complete path to save the barcode image when using a script command to create a barcode image. Because barcodes are precise, only bitmap files are created (no loss). Note: If the file exists, it will be overwritten.

Fixed (false)

The barcode position is “fixed” on the web page. The default is false. The barcode will move when the page is scrolled. Note: Website only.

HorPixelsPerMM (4)

Manually controls the width of the barcode by specifying the horizontal pixels per millimeter. If AutoScale is False (implying manual sizing of the barcode), the BarWidth property is checked. If BarWidth is non-zero, its value will be used to specify the width of a single hexagonal element in pixels. If BarWidth is zero, HorPixelsPerMM will be used to determine the size of the barcode.

Mode (cmMode4)

Sets the error correction level and type of data encoded in the barcode. The mode property controls whether or not the barcode is encoding carrier data and the level of error correction used (see the following table).

Mode	Description
cmMode2	Structured carrier message with numeric postal code.
cmMode3	Structured carrier message with alphanumeric postal code.
cmMode4	Standard symbol with standard error correction.
cmMode5	Standard symbol with enhanced error correction.
cmMode6	Reader programming.

Note: Mode 0 and mode 1 are obsolete.

Modes 2 and 3 are used for package transport. The values of CarrierCountryCode, CarrierPostalCode and CarrierServiceClass are encoded in the primary message. The secondary message will contain the data specified in the Code property.

Modes 4 and 5 use the full symbol to code the data in the Code parameter. The Carrier fields are not used.

Mode 5 is the only mode that uses enhanced error correction on the secondary message. Enhanced error correction is always used on the primary message. In mode 5, only 77 codewords are available for data encodation. In the other modes, 93 codewords are available.

Mode 6 indicates that the symbol is used to program the reader system. No data is actually transmitted in mode 6.

NumColumns (0)

Specifies the number of columns in the barcode. If the number of columns is zero, the barcode will attempt to automatically size itself to best fit the data.

NumRows (0)

Specifies the number of rows in the barcode. If the number of rows is zero, the barcode will attempt to automatically size itself to best fit the data.

Printer (No value (blank))

This attribute specifies the name of the printer when printing from a script or “PrintOnClick”.

PrintHeight (0.5)

This attribute specifies the height of the barcode for printing/saving. The value is in inches.

PrintOnClick (True)

This attribute specifies if the barcode will be output to the printer when the left mouse button is clicked on the barcode.

PrintOnClickPrompt ((No value (blank))

If the “PrintOnClick” is enabled and this attribute is not blank, when the user clicks on the barcode, a dialog will appear with this attribute as the prompt. A “Yes” answer from the user will print the barcode. Any other answer will not print the barcode. If this attribute is “blank”, no value, the user will not be prompted and the barcode will print.

PrintXOffset/ PrintYOffset (0.0)

This attribute specifies the X (horizontal) and Y (vertical) offset from 0 (left/top) to print/save the barcode. The value is in inches.

QuietZone (8)

Sets the size of the quiet zone around the barcode. The quiet zone is an area around the barcode in which no data is written. This area allows for scanners to determine the size of the barcode. This is also referred to as a "Clear Area". For PDF417 barcodes, this size should be a minimum of 2 times the BarWidth. For MaxiCode barcodes, the quiet zone should be a minimum of either 1/33rd of the BarCodeHeight or 1/30th of the BarCodeWidth, whichever is larger.

RelativeBarHeight (False)

Adjusts bar heights to the size of the control. In stacked bar symbologies like PDF417, the height of a row is usually a multiple of the width of the narrowest bar (BarWidth). When this property is “True”, the bar height is calculated based on the available area in the control.

ShowCode (True)

Determines if the value of the code is displayed beneath the symbol.

ShowGuardChars (False)

Determines if guard characters are displayed. This property applies to the start and stop characters used by the Codabar and Code 39 symbologies. For others, it is ignored.

SupplementalCode (No value (blank))

Determines the value of the supplemental code. If this property is assigned a two or five digit value, it is displayed to the right of the normal bar code symbol. This property applies to the UPC and EAN symbologies only; for others, it is ignored.

TallGuardBars (True)

Determines if the guard bars are drawn above and below the normal symbol bars. If TallGuardBars is True, any guard bars in the symbol (not all symbols have guard bars) are drawn taller (above and below) than the normal bars. If False, the guard bars are drawn the same as the normal bars.

Truncated (False)

Controls if truncated PDF417 mode is used. Truncated PDF417 reduces the amount of space required for the barcode by removing the right hand indicators and by reducing the stop pattern to a single bar. Truncated PDF417 should only be used in situations where the label is unlikely to be damaged. Truncated PDF is fully compatible with PDF417.

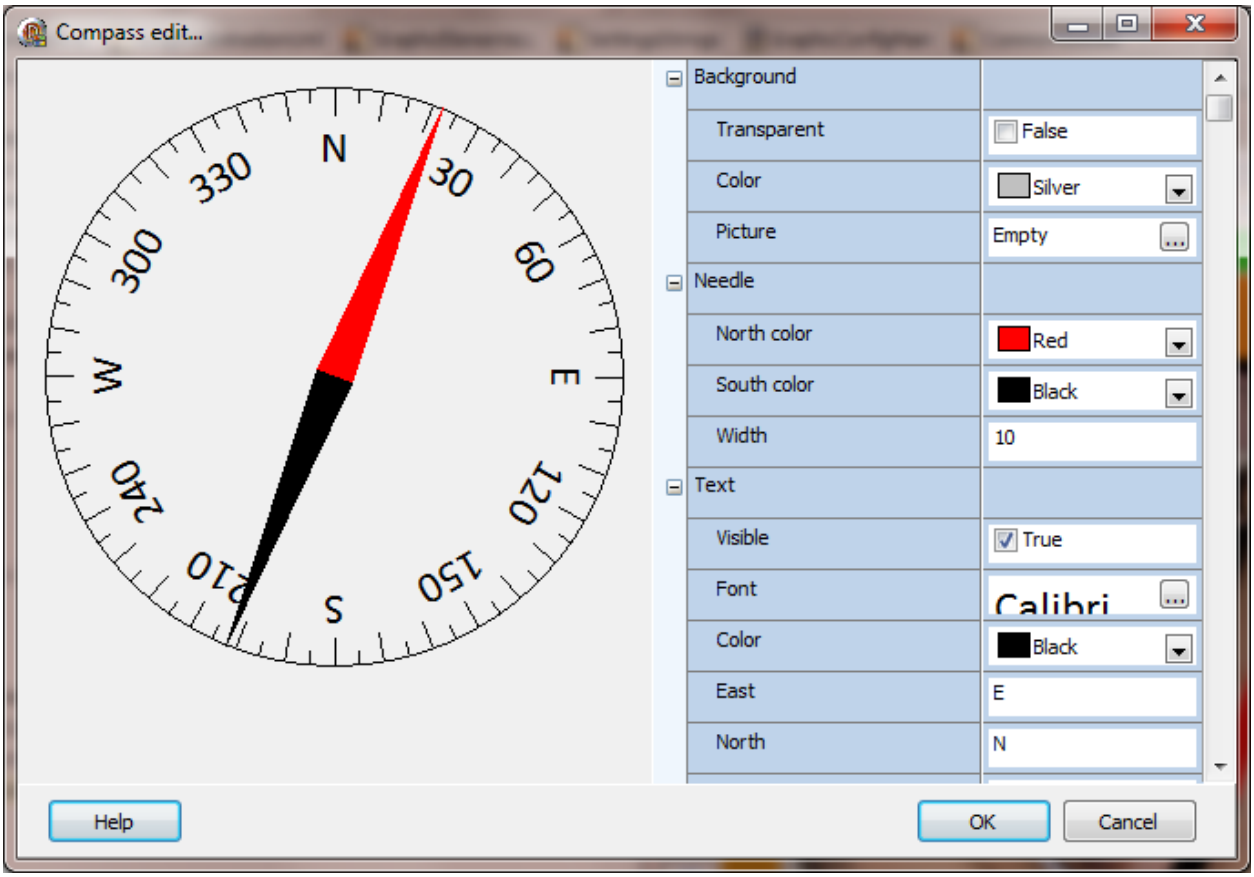
VerPixelsPerMM (4)

Manually controls the height of the barcode by specifying the vertical pixels per millimeter. If AutoScale is False (implying manual sizing of the barcode), the BarHeight property is checked. If BarHeight is non-zero, its value will be used to specify the height of a single hexagonal element in pixels. If BarHeight is zero, VerPixelsPerMM will be used to determine the size of the barcode.

Back to list

Compass

The “Compass” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Background

Transparent If enabled the background will not be filled with the background color.

Color The background color. The same value as the background color.

Picture If a picture is defined, the picture will scaled to fit in the background of the compass graphic element.

Needle	**North/south color**	These properties define the needle color.
	Width	This defines the width of the needle center. It is a ratio of the compass size.
Text	**Visible**	If enabled the text of the compass scale will be rendered.
	Font	This defines the text font name, size and style.
	Color	The text color. The same value as the foreground color.
	East/North/South/West	This defines the text for the four major compass points.
Scale	**Visible**	If enabled the compass scale will be rendered.
	Color	The scale color. The same value as the pen color.

Heading

The needle heading. This property is exposed to show how the needle will be rendered at the entered value. This property is controlled by the animation configuration at runtime.

Offset

This property adds an offset to the needle or scale based on the “Rotate” property value.

Rotate

This property defines if the scale is fixed and the needle rotates or if the needle is fixed and the scale rotates.

Fixed (website only)

The compass will not scroll if the browser window is scrolled.

Back to list

Digital grid

All of the digital grid settings are accessed via the digital grid animation dialog.

Dynamic grid

All of the dynamic grid settings are accessed via the dynamic grid animation dialog.

Excel grid

All of the Excel grid settings are accessed via the Excel animation dialog.

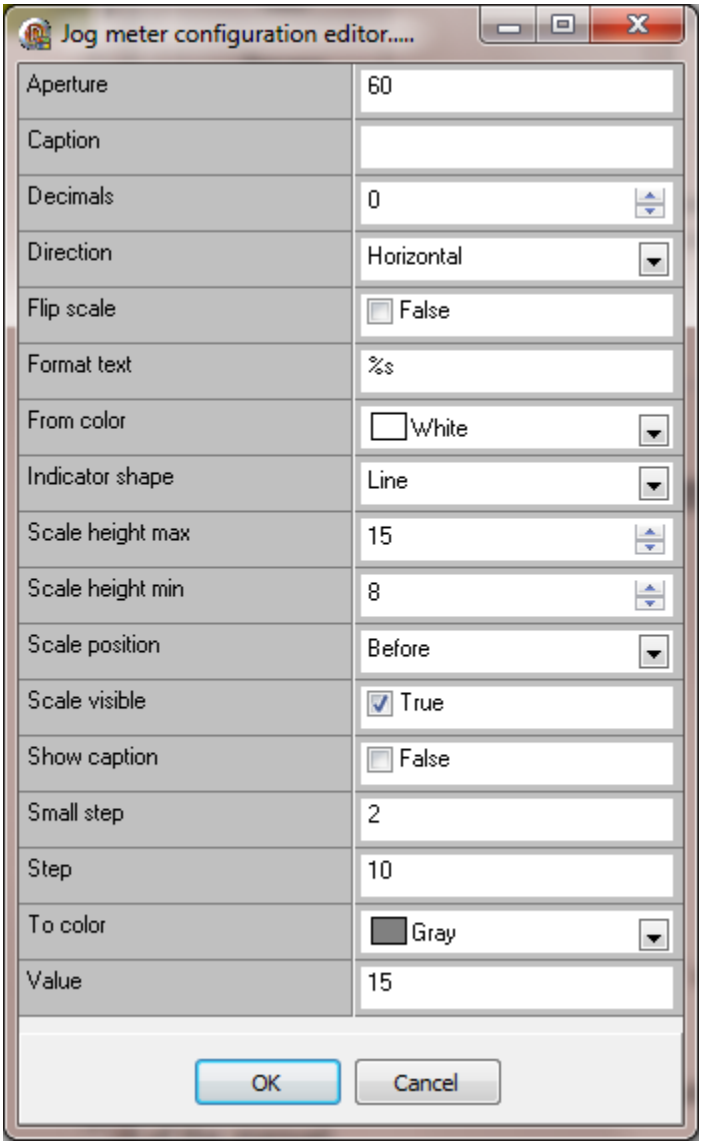
Image list

All of the "image list" settings are accessed via the image list animation dialog.

Back to list (complex objects)
Back to list (animations)

JOG METER

The “Jog” meter editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.

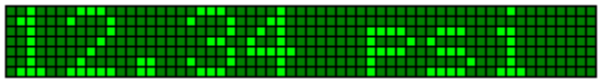


Aperture	This is the displayed range of the gauge. The gauge does not have a range. It shows the value by changing the displayed scale.
Caption	The text in the gauge
Decimals	The number of decimal places, if any, that are displayed on the scale.

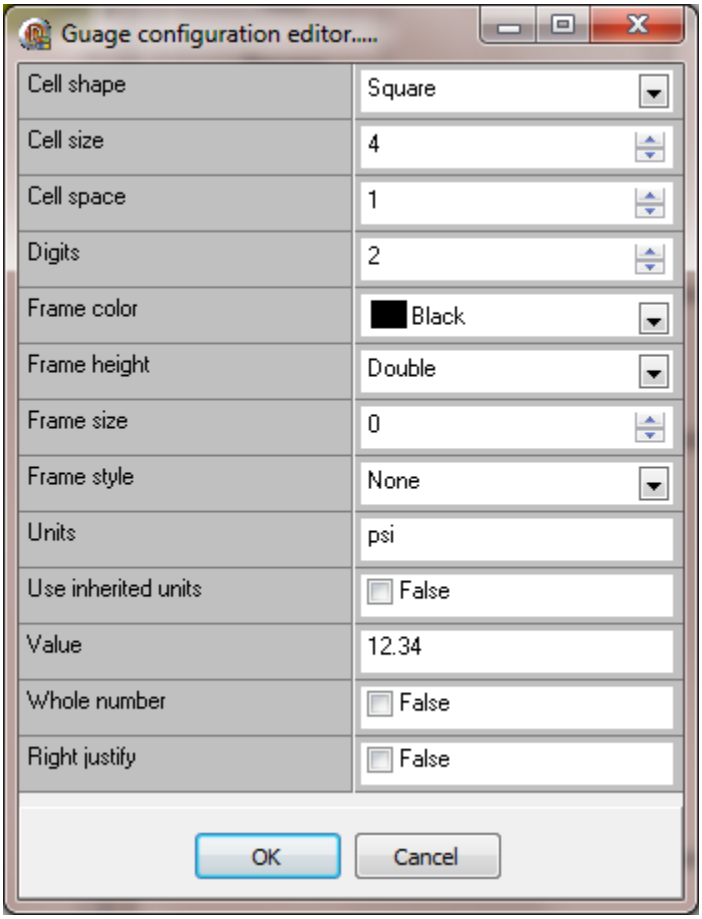
Direction	The gauge can be horizontal or vertical.
Flip scale	Positive on the top/left or bottom/right.
Format text	%s is the default. Any text before the '%s' would be displayed before the scale values and any text after the '%s' would be displayed after the scale values.
From/to color	The background gradient colors. For a single color set both colors to the same color.
Indicator shape	The indicator can be a triangle or a line.
Scale height max, scale height min	This is the line length for the major and minor tick marks/
Scale position	The scale can be on the left/top or right/bottom.
Scale visible	The scale can be displayed or hidden.
Show caption	The caption can be displayed or hidden.
Small step	The number or minor tick marks. Smaller numbers produces more tick marks.
Step	The number of major tick marks. Smaller numbers produces more tick marks.
Value	The value displayed at design time.

Back to list

LCD Display



The “LCD” display editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Colors The cell on color is the pen color.
The cell off color is the foreground color.
The board color is background color.

Cell shape The cell can be round or square.

Cell size The size of the grid cells.

Cell space The size of the space between the cells.

Digits The number of digits to display after the decimal. If the value is zero the decimal is not shown.

Frame color The color of the frame if displayed.

Frame size The size of the frame if displayed.

Frame style The frame style if displayed.

Units The units to display. To not display the units blank this field and disable the next attribute.

Use inherited units

This overrides the "Units" above and uses the units from the point assigned via animation.

Value The value of the display. (Provided to show how the control would appear at runtime with a value.)

Whole number

If enabled, the floating point number is truncated and displayed.

Right justify If enabled, the value will be aligned to the right side of the graphic element.

Leading characters

If this field is not blank the text entered will be prepended to the value.

Fixed width

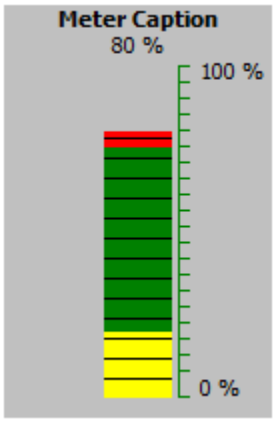
If the “leading characters” field is not blank, the leading characters will be prepending to the value. The displayed text might be longer than the fixed width if the leading characters is greater than one character.

Examples:
Value 123

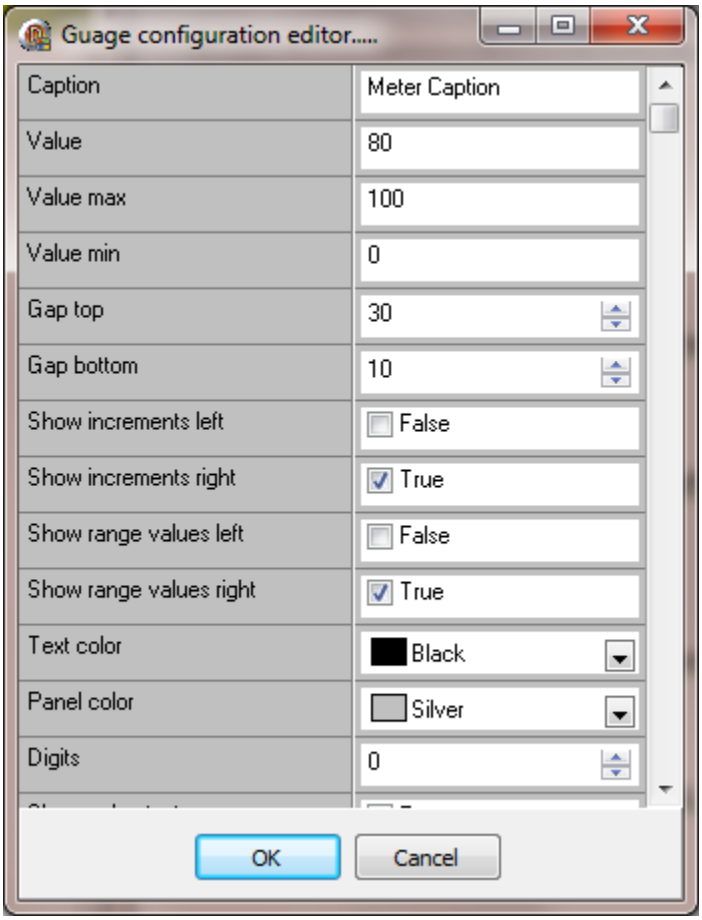
Leading characters	Fixed width	Displayed text
0	6	000123
0	2	123
AB	6	ABAB123
A	6	AAA123
EX:	0	EX:123

Back to list

LED Bar



The “LED Bar” display editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Caption

The text at the top of the element.

Value

The value of the led bar. (Provided to show how the control would appear at runtime with a value.)

Value max

The maximum value of the "value".

Value min

The minimum value of the "value".

Gap top

This is the gap between the top of the element and the indicator part of the control. The "show value text" and the caption value affect the "top" of the control from the indicator part.

Gap bottom

This is the gap between the bottom of the element and the indicator part of the control.

Show increments left

Show tick marks on the left side of the filled bar.

Show increments right

Show tick marks on the right side of the filled bar.

Show range values left

Show the maximum and minimum range values on the left side.

Show range values right

Show the maximum and minimum range values on the right side.

Text color

The color of all the text in the element.

Panel color

The background color of the panel. The panel can also be set to transparent via the “Objects” menu.

Digits

The number of digits to display after the decimal. If the value is zero the decimal is not shown.

Show value text

Enable this attribute to display the value at the top of the control below the caption.

Units

The engineering units, if any.

Used inherited units

This overrides the "Units" above and uses the units from the point assigned via animation.

Used inherited ranges

This overrides the "Value Min/Max" above and uses the engineering min/max from the point assigned via animation.

Font Name

The font type for the element

Font Size

The size of the font for the element.

High range text margin

The high range text vertical margin. A positive number positions the text lower, a negative number positions the text higher.

Low range text margin

The low range text vertical margin. A positive number positions the text lower, a negative number positions the text higher.

High value

The value the led color changes from normal to high. This value is PFS - percent of full scale.

High color

The color above the normal color.

Normal color

The color above the low color and below the high color.

Low value

The value the led color changes from normal to low. This value is PFS - percent of full scale.

Low color

The color below the normal color.

Block height

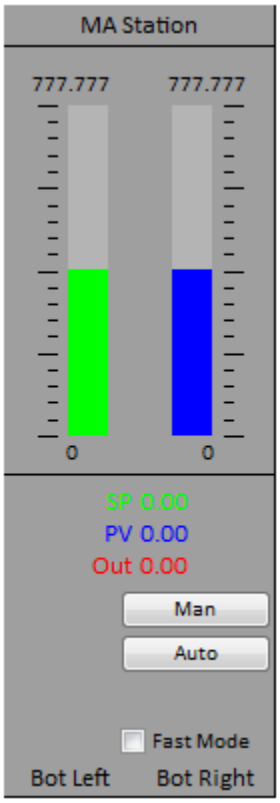
The height of each LED segment.

Side margin

The number of pixels on each side of the LED bar.

Back to list

MA Station (2 Bar)

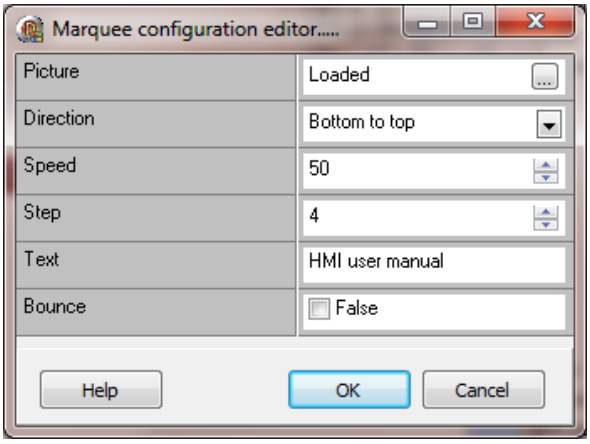


The MA station has many settings. The station is assigned a “configuration” name.
The Manual/Auto (MA) configuration is not contained in the graphic element to allow an MA to be used on a screen and the screen reused. The MA configuration used by an MA element can be set via scripting using the SetMAStationConfiguration command. The graphic element has an initial MA configuration selection.

The MA configuration is covered in the “Graphics Menu” section of this manual.

Back to list

The “Marquee” display editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Picture

Defines an optional picture for the background.

Direction

The text can scroll from right to left, left to right, up to down or down to up.

Speed

The speed the text scrolls. Smaller numbers are faster. (20 – 100,000 milliseconds)

Step

The number of pixels the text moves for each speed pulse.

Text

The text displayed at runtime. Any of the text animations can be used to change the text or the text can be set via the script animation. **Note:** Website pages only support script global animation.

Bounce

If enabled, when the text reaches the end point, the text will reverse direction.

Script global (website only)

This optional property defines a script global to use as the source of the marquee text. If this property is not configured, the “text” property, above, will be used.

For normal screen marquees, use the “Script global” animation, if required.

Fixed (website only)

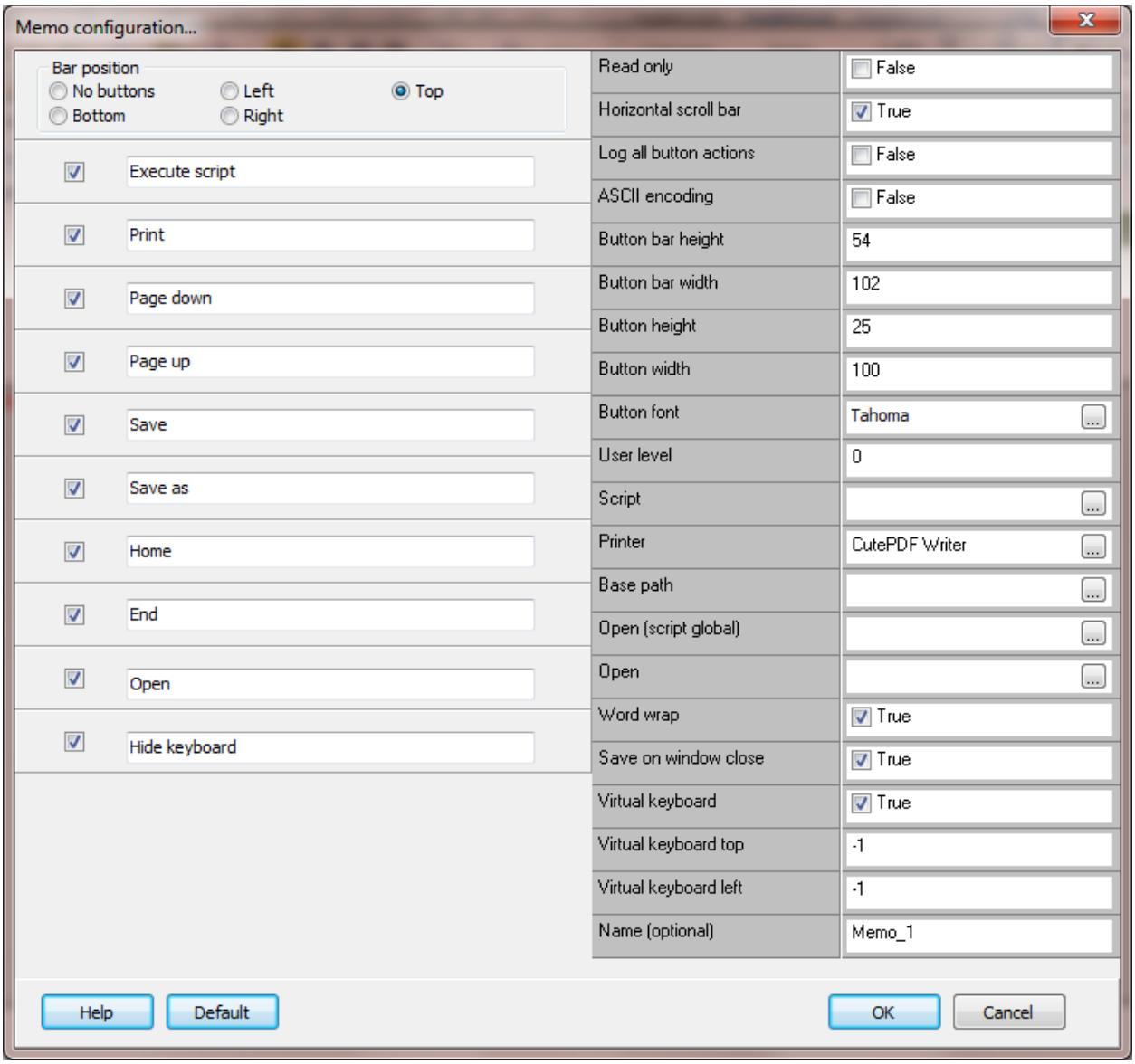
The marquee will not scroll if the browser window is scrolled.

Note: The website marquee enable is reversed from the regular window enable. Enabled = paused, disabled = run (marquee scrolls).

Back to list

MEMO

This graphic element is used to display and possibly edit a text file. All of the settings are accessed via the “Object/Edit” menu Item.



Note: While RTF is supported, the version of MS Windows and the creator of the RTF file may not produce identical results.

Buttons

Bar position

The position of the button bar. The bar will be placed on the top, bottom, right, left or not displayed.

Execute script

The script selected above will be queued for execution.

Print

This will send all the text to the printer selected in the options, below.

Page down

If the text is not all the bottom, this will 'page down' one page or to the bottom.

Page up

If the text has 'paged down' or scrolled 'down' this will page up one page.

Save

This will save the text to the file specified in the options, above.

Save as

This will allow the user to select the file name/path to save the text.

Home

The text will be scrolled to the top.

End

The text will be scrolled to the bottom.

Open

The user will be allowed to select a file to open.

Hide keyboard

If the "Virtual keyboard" option below is enabled and the keyboard is visible, this button will hide or show the keyboard. The keyboard is shown when the memo field has focus.

Options

Read only

If enabled the user will not be able to "edit", "save" or "save as" the text. The show/hide button will also not be visible

Horizontal scroll bar

If enabled the memo will display the horizontal scroll bar. If "Word wrap" is enabled the horizontal scroll bar will not appear.

Log all button actions

If enabled and one of the buttons is selected an entry will be placed in the event log.

ASCII encoding

The HMI uses Unicode text encoding by default. If enabled the text encoding will be set to ASCII when the save/save as command is executed.

Button bar height

The height of the button bar if the 'bar position' is top or bottom.

Button bar width

The width of the button bar if the 'bar position' is left or right.

Button height

The height of each button. Default = 25

Button width

The width of each button. Default = 75

Button font

The button font.

User level

The user level required to initiate button actions.

Script

If the button bar is enabled and the 'Execute script' button is enabled, this is the script that will be executed.

Printer

The printer to print the alarms if one of the below print buttons is selected.

Base path

If specified, it is the path to begin all user interactive "open" and "save as" commands.

Open (script global)

If this field is valid, when the memo is first displayed the file to open is specified in the script global. If the field is not valid, or if the file does not exist the open property is used.

Open

If this field is valid, when the memo is first displayed the file to open is specified.

Word wrap

If enabled the text will wrap at the bound of the graphic element.

Save on window close

If enabled and read only is not enabled, the memo text will be saved to the file that was last opened for display if the text has been modified.

Virtual keyboard

If enabled the virtual keyboard will be displayed. When the user clicks the mouse in the memo field the keyboard will be displayed. If the user clicks the mouse in another control the keyboard will be hidden.

Virtual keyboard top

The top position of the virtual keyboard.

Notes:

1) If the top or left is -1, the keyboard will be centered at the bottom of the main monitor (the monitor containing the main form).
2) If the OS keyboard is enabled, the top and left positions are ignored. The keyboard will display at the position the keyboard was at when it was last closed.

Virtual keyboard left

The left position of the virtual keyboard.

Name (optional)

This is used when a script "MemoCommand" is used.

Border visible

If enabled a border will be drawn around the memo.

Bevel inner/kind/outer

These properties define how the border will appear. **Note:** Depending on these property values, the design time and runtime appearance of the border may be slightly different.

Width

This property defines a space between the border and the text of the memo.

MINI SINGLE PEN TREND

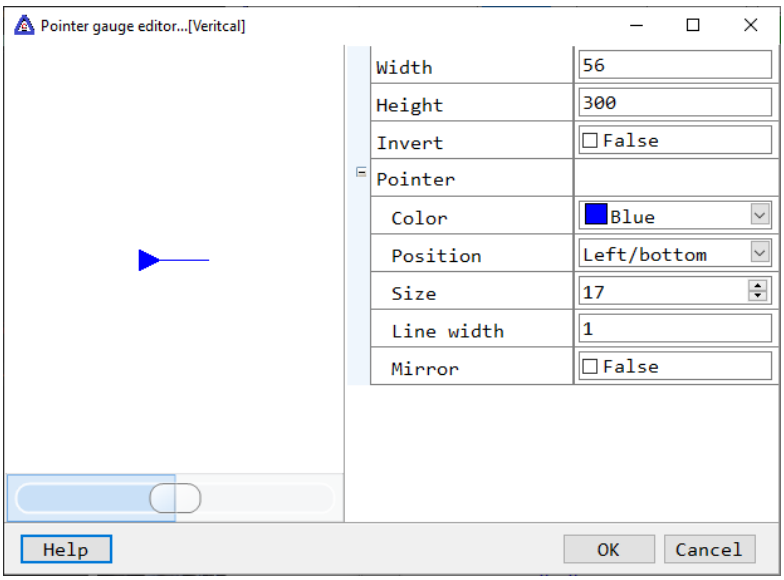
All of the "Mini single pen trend" settings are accessed via the mini trend animation dialog. See the "Animation" section of the manual. Under the "Settings/Miscellaneous" the "Mini single pen trend enable" checkbox must be checked for the graphic element to function at runtime.

PIE CHART

All of the "Pie chart" settings are accessed via the pie chart animation.

Back to animations list
Back to complex objects list

POINTER GAUGE



This gauge uses a 0 - 100 range. Use the point's "percent full scale" (PFS) property, of the selected analog point, for automatic scaling.

Notes:

1) If the selected point's range is not 0 – 100 use scaling. If for example, the point is 4-20 and the value "Process variable analog" (5000) is used and need to be 4-20, set the scaling the High/low IR to the same value as High/low EU.
2) Scripting can be used to set the "value" to 0-100.
3) The pointer will not move outside the bounds of the gauge. See pointer size, below.
4) When changing values in the configuration editor, the change will not be reflected in the example until the left mouse button is clicked outside the property editor/item.

Width/ height The orientation of the gauge is defined by the width and height. If the width is greater than the height the gauge will be horizontal.

Invert Left to right and bottom to top is the default, 0-100. Enable this property and left to right and bottom to top will be 100 – 0.

Fixed (Web site only) This property defines if the gauge has a fixed location on the page or will move on scrolling.

Color Same as the foreground color and can be altered via animations.

Position Set the pointer orientation. Left/bottom or right/top.

Size The size of the pointer. The value must be odd. The movement range is limited to the size of the graphic element and one half (1/2) the size of the pointer. For example: The pointer size is 5, the gauge is horizonal and 300 pixels wide (left to right). The pointer movement is 2.5 – 297.5 pixels.

Line width If the value is not zero a line will be rendered from the pointer tip to the bounds of the graphic element.

Mirror If the "position" is left/bottom and this property is enabled, the pointer will be mirrored on the opposite side.

Back to animations list
Back to complex objects list

Recipe grid

All of the "Recipe grid" settings are accessed via the recipe animation.

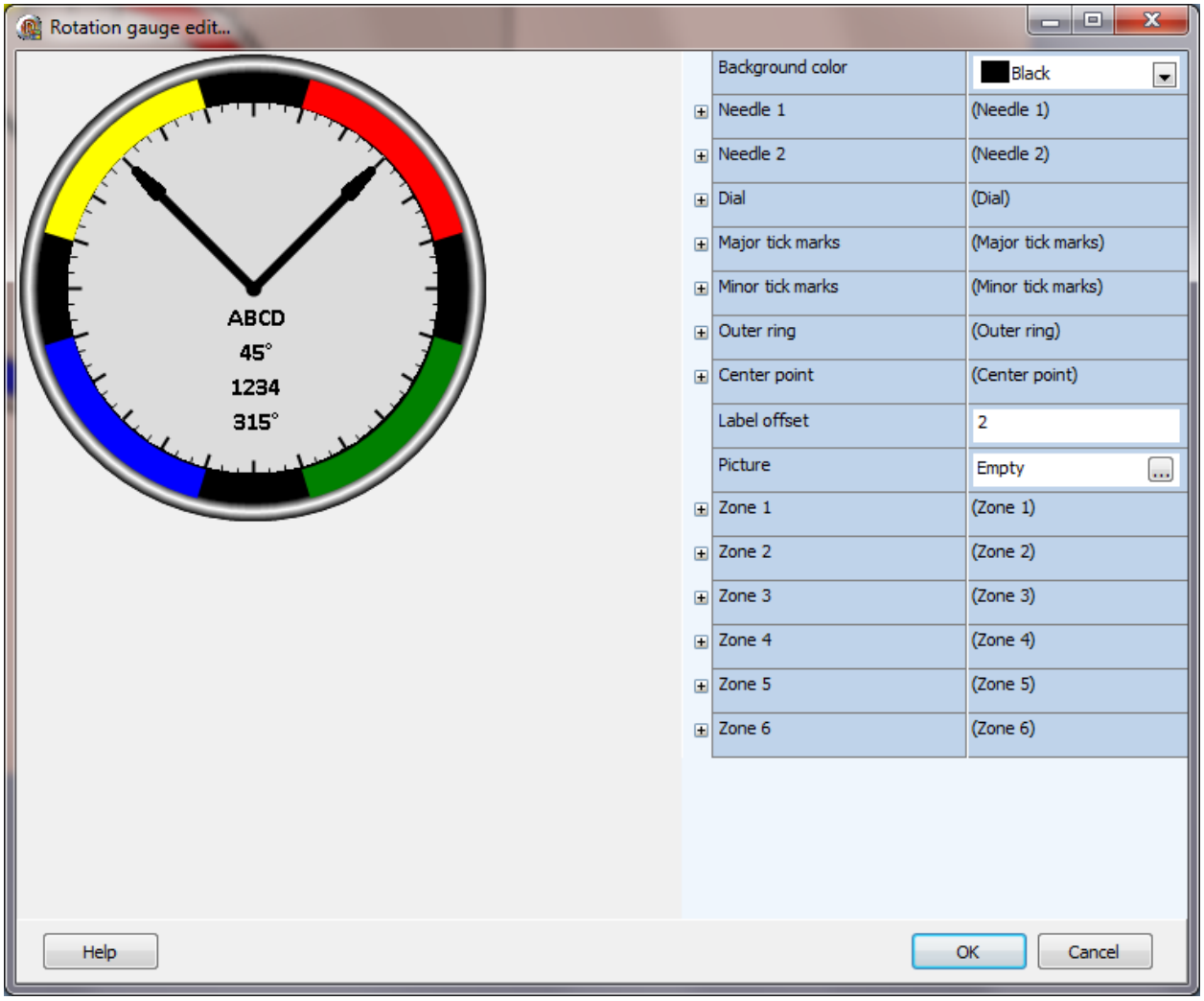
Back to animations list
Back to complex objects list

ROTATION GAUGE

This graphic element gauge has a range of 0 – 360, supports two (2) needles and six (6) range zones. The range zones are static or dynamic. This gauge is configured via the "Objects/edit" menu (or right click on the gauge and select the "Edit" menu) and via the gauge animation configuration editor. The static and dynamic property descriptions are below.

Static settings

Note: When using the editor and a change is made to a property, click on another property to end the edit mode for the changed property and to view the change in the example gauge.



Background color

This property defines the color between the outer ring and the dial. In the above picture, the black between the zone colors is the background color.

Note: A needle is enabled when a point is configured in the dynamic settings.

Needle 1/2

Length

This is the length of the needle from the center of the dial. The value is the percent of dial size.

Width

This is the width of the needle in pixels.

Color

This is the needle color.

Style

The needle can be a line or a line with an arrow head.

Value

The property is not saved and is provide to display the needle at various values.

Value visible

If this property is enabled the value of the needle will be displayed below the center of the dial.

Value suffix

If this property defines (is not empty) text that is appended to the value text displayed on the dial. See value visible property.

Font

This is the font configuration for the label and value text. The font color is ignored and the needle color is used as the font color.

Dial

Margin

This property defines the dial size. It is the percent of the graphic element width and is centered in the graphic element bounds.

Color/color to

The color of the dial. The color “to” when used with a gradient fill.

Gradient

The gradient style.

Steps

The number of gradient steps.

Border color

The dial can have a border. This property defines the color.

Border width

The dial can have a border. This property defines the width of the border. Setting the value to 0 (zero) disables the border.

Major tick marks

Count

This is the number of major tick marks. The actual count is one more than the entered value.

Length/width

This is the length and width of major tick marks.

Color

The color of the major tick marks.

Minor tick marks

Count

This is the number of minor tick marks.

Length/width

This is the length and width of minor tick marks.

Color

The color of the minor tick marks.

Outer ring

Color/color to

The color of the outer ring. If the “color to” is different from the “color” property the ring will be filled with a gradient.

Width

This is the width of the outer ring.

Center point

Color

The color of an ellipse drawn at the center of the dial.

Size

This is the size of the ellipse.

Label offset

This is the offset from the center towards the bottom of the dial to render the text. The label, if configured, is rendered, the next line contains the value, if enabled.

Picture

If this is configured, it is an image that is rendered in the graphic element bounds rectangle. Only the image portion outside the outer ring is visible.

Zones 1-6

These are the static settings for the zone. To disable a zone in the static settings, set the start and end angle to 0 (zero) and the zone will not be rendered.
The start and end angle values can be dynamic via the dynamic settings.
The zone can be enabled or disabled via the dynamic settings.

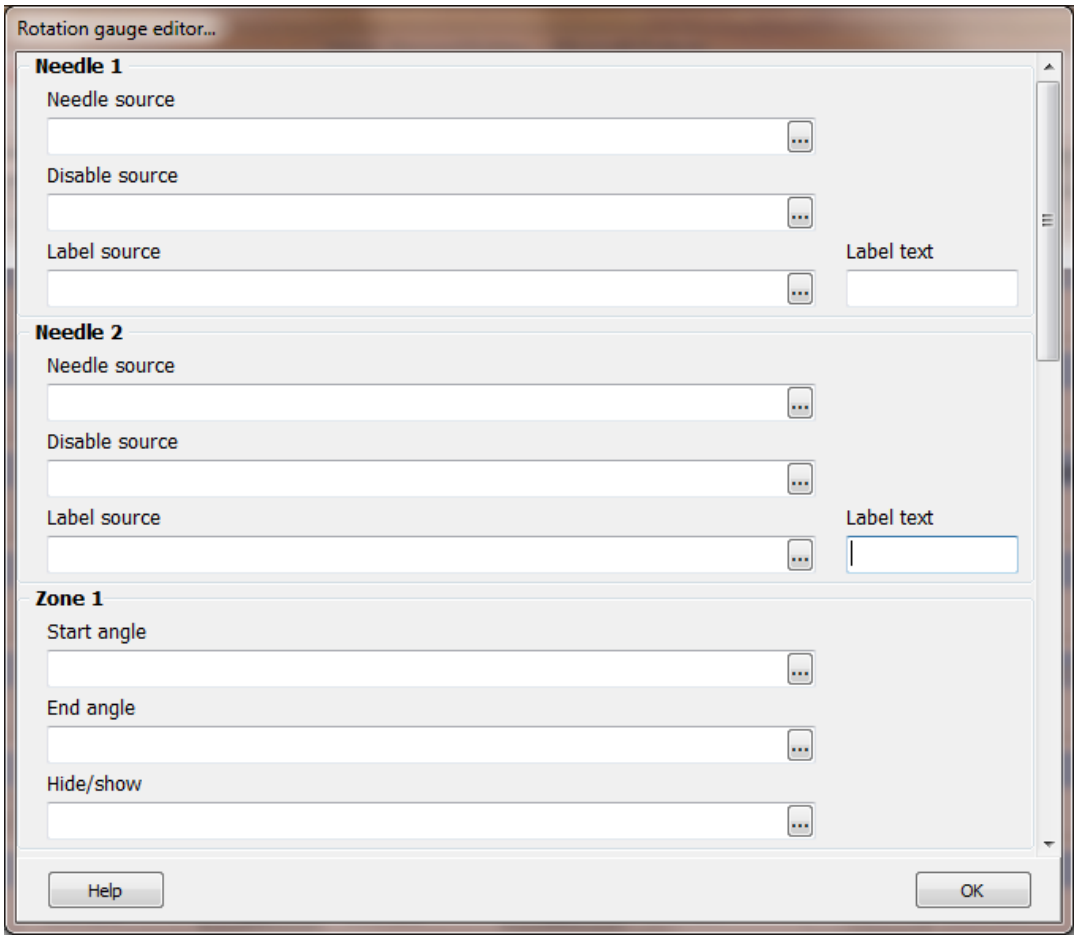
Start/end angle

This is start/end angle of the zone arc. 0 degrees is 12 o’clock.

Color

This is the zone color.

Dynamic settings



Needle 1/2

Needle source

This is the point for the needle. The value must be in the range of 0-360.

Disable source (optional)

This is the point to hide/show the needle. If the value is true the needle, label and value will not be rendered (will be hidden).

Label source (optional)

This is the point to fetch the label text. If configured the “Label text” field is ignored. If the “label source” is not configured and the “Label text” field is configured the label will be rendered.

Zone 1 - 6

Start/end angle

This is the point for the start/end angle for the zone. If the field is not configured, the static property value will be used.

Hide/Show

This is the point to hide/show the zone. If the point value is true, the zone is hidden.

Back to complex objects list
Back to animations list

ROUND TREND CHART

This graphic element is used to display data that has been collected over a period of time on a circular chart. Each chart can display one day. One to eight data point trends can be displayed.

The configuration screen is divided into several sections. The largest section is an example of the chart using the selected settings.

Instance configuration

Graphic height	560
Graphic width	544
Center horizontal	272
Center vertical	270
Inner diameter	0
Outer diameter	512
Theme	_Default
Pens	Select
Name (Optional)	
Title text (Optional)	
Center chart	Now

This section applies to a single graphic element instance. **Note:** When editing these fields, the editing must end before the change is visible in the chart. For example, when changing the inner diameter, change the value by entering a value or using the up/down buttons and then click the mouse in another field. This stops editing on the field and accepts the change.

Graphic height/width

The width and height of the graphic element. The graphic element size can be changed in this dialog up to the size of the window.

Center horizontal/vertical

The graphic element may or may not be square. Also the button bar and legend bar options move the center of the circular chart from the center of the graphic element. This provides for settings the center point for the chart in the graphic element.

Inner diameter

This defines the size of the inner most ring and as the “low range” position. The value can be zero but, it normally is not zero to allow lower values to be visible.

Outer diameter

This defines the diameter of the outer most ring of the chart and is the “high range” position.

Theme

The name of the theme used to render the chart. Themes are used to allow one group of settings to be applied to more than one instance of a chart. Theme selection is covered below.

Pens

This is used to select one to eight “pens”. The pens are for an chart instance and are not part of the theme.

Name (Optional)

The name of the instance of the graphic element. This field is optional because it is only needed if runtime scripting is used to change a charts operation from the configured settings.

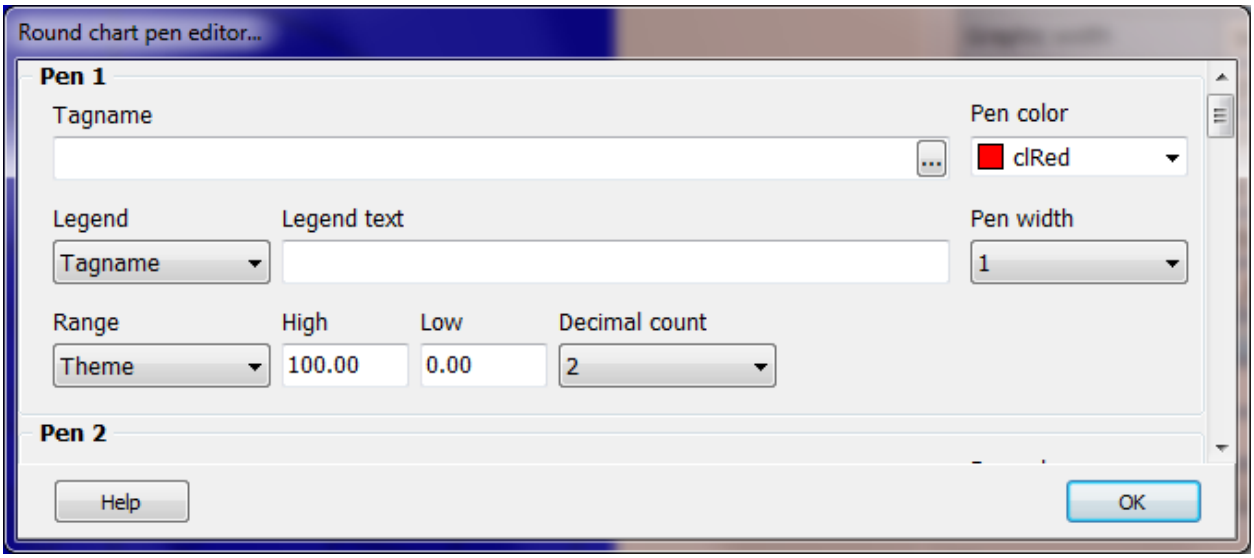
Title text (Optional)

If the theme title position attribute is configured to display the title and the “Use day” attribute is not set, this value will be used for the title. If this attribute is blank, the theme title text will be used.

Center chart

Selecting this button will center the chart in the middle of the chart area of the graphic element. The “center horizontal” and “center vertical” attributes will be modified to center the chart. Each round chart graphic element can have three areas, the button bar area, the chart area and the legend area. The chart area is always present and the other two areas are present based on configuration options. The chart area is defined by the size of the graphic element and the size of the other two areas.

Round chart pen editor



Tagname

The point tagname for the pen. The point.item must be configured for data logging.

Legend

The chart can be configured to display a legend and the legend can be configured to display a text field for the pen.

Tagname	= The point tagname will be used.
Point description	= The point description will be used.
Text	= The value entered in the “Legend text” field will be used.

Pen color/width

The color and pen width for the pen.

Range

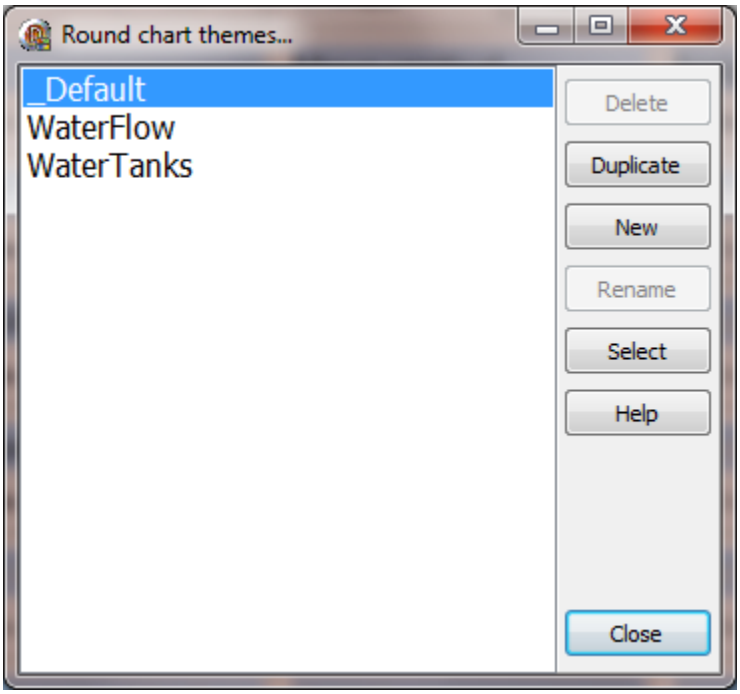
The range of the pen can be configured three ways.

Point	= The range will use the engineering range of the point.
Theme	= The range will use the theme range.
Values	= The values entered in the “High” and ”Low” fields will be used.

Decimal count

The number of digits after the decimal point.

Theme selection



The theme "_Default" cannot be deleted or the name changed. The configuration can be altered. Theme names must be unique.

Delete

This deletes the selected theme.

Duplicate

This duplicates the selected theme and saves it when the name entered.

New

This creates a new theme and saves it when the name entered.

Rename

This prompts for a new name for the selected theme. It does not change the theme name selected for any graphic element.

Select

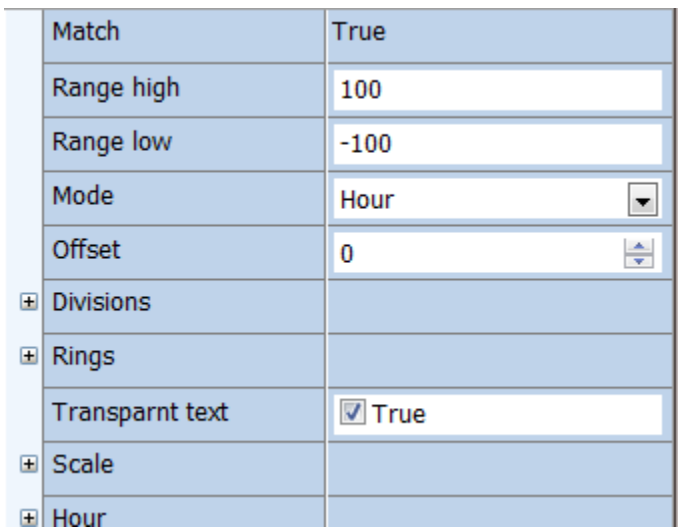
This selects the theme for the graphic element and loads the settings into the theme editor. Double clicking on a theme name will also select the theme.

Simulation area

This is used to draw a plot line on the chart using the pen settings selected via the “Pen” drop list. The slider is used to change the value of the “now” plot value. The “All pens” checkbox plots all eight pens. Pen one is dynamic and pen two though eight are a fixed value. **Note:** The range must be 1 or greater for the pen simulation to function. While the chart will function with a small range, e.g. 0 – 0.5 the pen slider will be disabled if the range is less than 1.

Theme settings

This area is for all the theme settings. **Note:** When editing these fields, the editing must end before the change is visible in the chart. For example, when changing the offset, change the value by entering a value or using the up/down buttons and then click the mouse in another field. This stops editing on the field and accepts the change.



Match

This field will be true or false. A true value means the distance between the rings is equal and false means the distance between the rings are not equal. This only applies if the ring count is greater than zero. The inner and outer rings are always required. Adjust the graphic element size, the inner margin, the outer margin or all three to get a match in this field. In the top left of the example area are the rings sizes.

Range high/low

This defines a range for the pens. It can apply to all pens or no pens. See the section on pens.

Mode

A theme can have one of two possible modes.
Pen = The chart rotates and the pen is fixed at zero degrees.
Hour = The pen moves around the chart. Also see offset.

Offset

This field applies when the mode is hour. It rotates the chart X degrees from 0. Positive values rotate the chart counterclockwise and negative values rotate the chart clockwise.

Divisions

This property is used to divide the chart into sections.

Pen color/width

The color and pen width for the division lines.

Rings

This property is used to draw rings at equal distances on the chart. Some ring counts calculate wrong because adding or subtracting one more pixel to each ring would make the rings to big/small.
For example: scale of 0 – 100. A ring count of 19 gives a ring every 5 divisions. A ring count of 17 or 18 is not correct. The same scale, a ring count of 9 gives a ring every 10 divisions and scales correctly.

Pen color/width

The color and pen width for the rings.

Transparent text

All the text rendering on the chart will have a transparent background.

Scale

Decimal count

The number of digits after the decimal point.

Font color/font

If the scale is rendered this is the font attribute selection and font color selection.

Hour

The chart is a 24 hour chart and the hour number can be rendered on the chart.

Font color/font

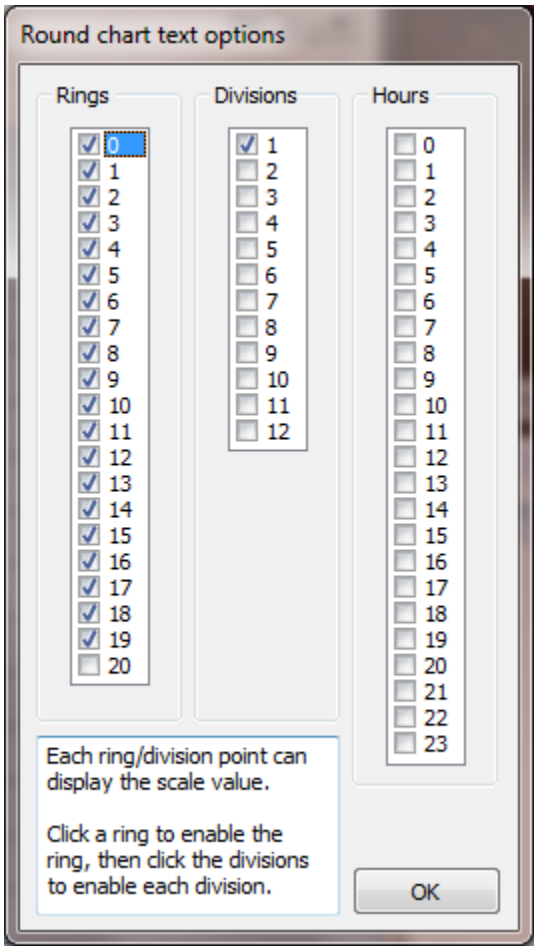
If the hour is rendered these are the font attribute selections and font color selection.

Margin

The amount of space between the text of the hour value and the outer ring of the chart.

Text

This provides selection for which text of the scaling and hours are to be rendered.



Each ring has twelve divisions. Enable a ring then select which divisions will display text at the intersection of the ring and division.

Select which hour positions are to be rendered.

Image

This provides for a custom made image to be used. If an image is selected the “built-in” drawing will be disabled. The zero ring and outer most ring will be visible during configuration. These two rings must align with the low and high range rings on the custom image. The mode and mode offset are ignored if an image is configured.

Button bar

The button bar can be enabled to display several buttons.

Position

The button bar can be visible on any one of the four sides of the chart.

First

The button bar will be first (above) the legend bar if the legend bar and the button bar are configured on the same side (top or bottom) of the chart.

Bar height

This is the height of the total button bar. This attribute applies when the button bar position is top or bottom.

Bar width

This is the width of the total button bar. This attribute applies when the button bar position is left or right.

Button height

This is the height of each button.

Button width

This is the width of each button.

Font

These are the font attributes used for the button.

Script

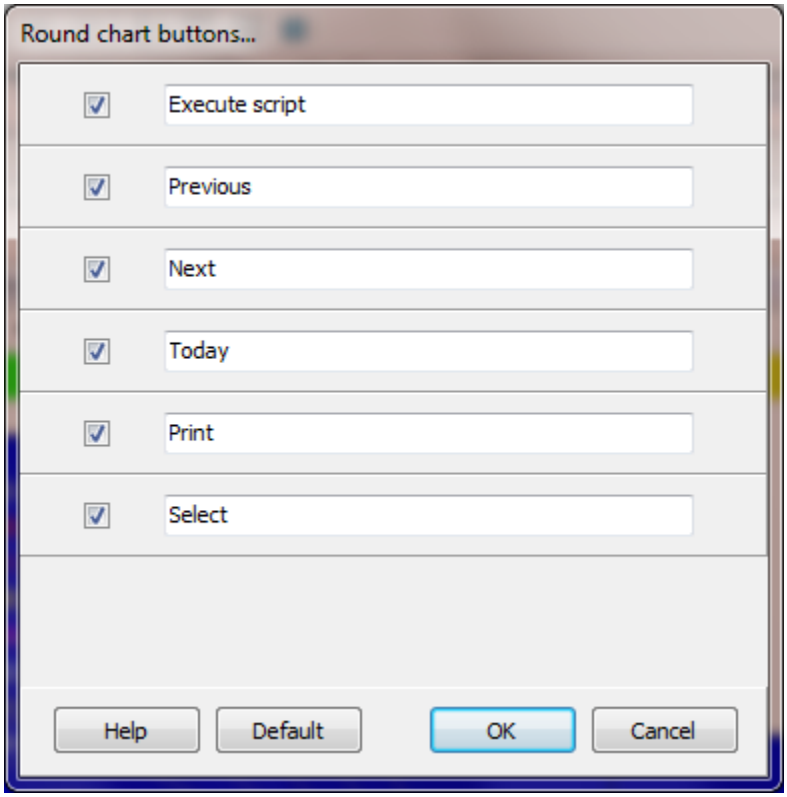
This is the script to execute when if the script button is selected.

Days back

This is used to limit the number of previous days the user can view via the "Previous" and "Select" buttons.

Buttons

This is the button selection. The name rendered in the button can be changed but, the action of the button does not change from the default action.



Script

The script configured above is executed.

Previous

The previous day's data log is rendered in the chart.

Next

The next day's data log is rendered in the chart.

Today

The current day's data log is rendered in the chart. This is a fast way to jump back to the current day if another day's data logs are currently rendered.

Print

This prints the chart to the default selected printer.

Select

This displays a calendar and allows the user to select which day's data log to render. The first pen point.item is used to parse which days to allow for selection.

Log all button actions

If enabled each button user selection will be logged to the event log.

User level

This is the minimum logged on user level required to select a button.

Legend

This is the legend configuration.

Position

The legend can be above or below the chart.

Format

The legend format can be Value – text, Text – value, Text only or Value only. The value is the current value of the pen's data point and text is the selection made in the pen configuration.

The Value – text and Text – value options use two columns and the Text only and Value only options use one column.

Sections

The legend selection can be in one section of one or two columns, based on the format selected, or two sections of one or two columns.

Height

This is the height of the total legend bar.

Value width/Text width

This is the width of the value and text column as a percentage of the total width. If the section count is two this value is divided by two.

Row height

This is the height of each row of the legend.

Use pen color

If this attribute is enabled the text and value for the pen will be rendered using the pen color. Otherwise the font color, below, will be used.

Font color/font

The font attributes to render the legend data. The color is used if the “Use pen color” attribute is not enabled.

Printing

This attributes are used when printing the chat via the print button.

Font color

The font color to use for all text when printing.

Background color

The background color selection to use when printing the chart.

Line color

The line color selection to use when printing the chart.

Use print colors

If enabled, the color selection above will be used when printing. If not enabled the same colors used to render the chart on the monitor will be used.

Print orientation

If enabled, when the print command is issued the trend will print with the orientation selected.

Include legend

If the legend is enabled and this attribute is enabled, the legend will be included in the print when the trend is printed via the print button.

Title

The title displayed on the chart.

Position

The title can be rendered along the top or bottom of the chart. The title can be aligned to the left, center or right.

Use day

The day of the displayed data points will be used as the title.

Date format

If “Use day” is selected this attribute sets the format for the day. Date formats

Text

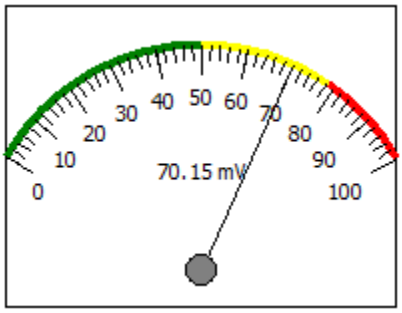
If “Use day” is not selected, this attribute will be used for the title.

Font/Font color

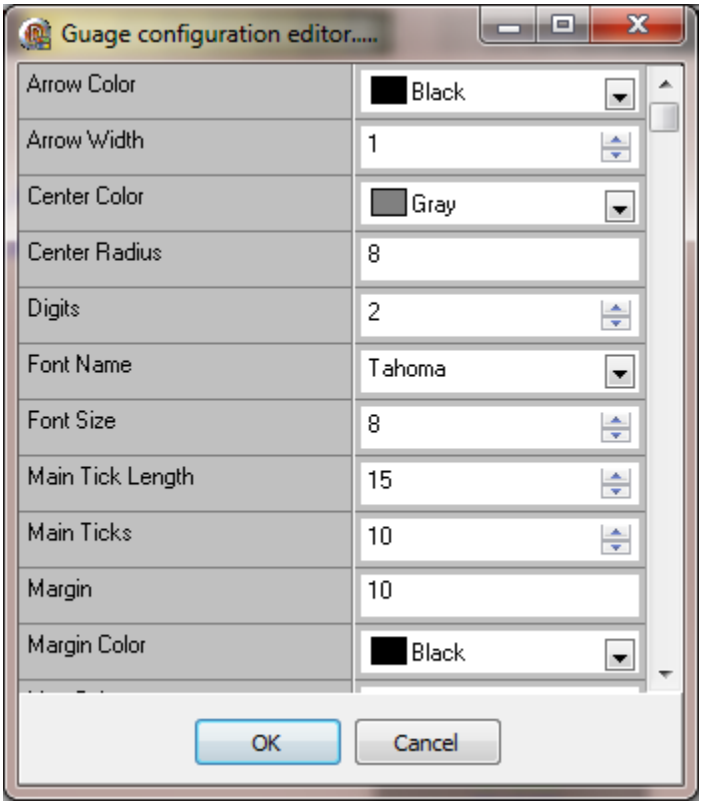
If the title is rendered, these are the font attribute selections and font color selection.

Back to list

NEEDLE METER



The “Needle Meter” display editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Arrow color

The color of the needle.

Arrow width

The width of the needled.

Center color

The color of the circle at the base of the needle.

Center radius

The radius of the circle at the base of the needle.

Digits

The number of digits to display after the decimal. If the value is zero the decimal is not shown.

Font name

The font type for the element.

Font size

The size of the font for the element.

Main tick length

The line length for the main tick marks of the scale.

Main ticks

The number of main tick marks to display.

Margin

The width in pixels around the edge of the meter.

Margin color

The color of the margin border.

Max color

The color of the arc on the scale indicating the upper range of the value. A color can be displayed to show to the operator that the value of the monitored variable might be in a certain range. For example, if the scale was 0-100. A good range might be 0-55. This could be indicated using a color setting in "Min Color". A caution range might be 56 to 79 and could be indicated using a color setting in "Mid Color". The trouble range could be > 79 and that could be indicated using this color selection. See maximum indicate value, mid color, min color and minimum indicate value.

Maximum indicate value

The lower bounds of the maximum indicate range. (Value is the percent of full scale)

Mid color

The color of the mid indicate arc on the scale.

Min color

The color of the minimum indicate arc on the scale

Minimum indicate value

The lower bounds of the minimum indicate range. (Value is the percent of full scale)

Scale angle

The length of the scale.

Show 3D

Draws the outer frame with a 3D appearance.

Show center

Display the circle at the base of the needle.

Show frame

Display the meter frame.

Show margin

Display the margin area.

Show main ticks

Display the main tick lines on the scale.

Show indicator max

Display the maximum indicator arc. See "Max Color"

Show indicator mid

Display the mid indicator arc. See "Max Color"

Show indicator min

Display the minimum indicator arc. See "Max Color"

Show sub ticks

Display the sub tick lines on the scale.

Show units

Display the units string.

Show values

Display the range values at the main tick marks.

Show value as text

Display the needle value as text.

Style

The gauge can be displayed with the needle anchor in the bottom left, center or bottom right of a rectangle.

Sub tick length

The line length for the sub tick marks of the scale.

Ticks color

The color of the tick marks.

Units

The engineering units, if any.

Units color

The color of the units text.

Use inherited units

This overrides the "Units" above and uses the units from the point assigned via animation.

Use inherited ranges

This overrides the "Value Max" and "Value Min" below and uses the engineering min/max from the point assigned via animation.

Value

The value of the needle. (Provided to show how the control would appear at runtime with a value)

Value color

The color of scale text.

Value max

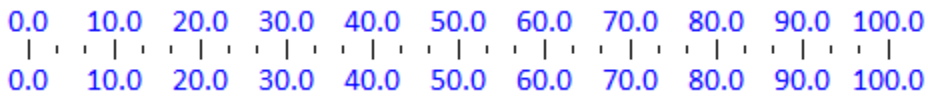
The maximum value of the "value".

Value min

The minimum value of the "value".

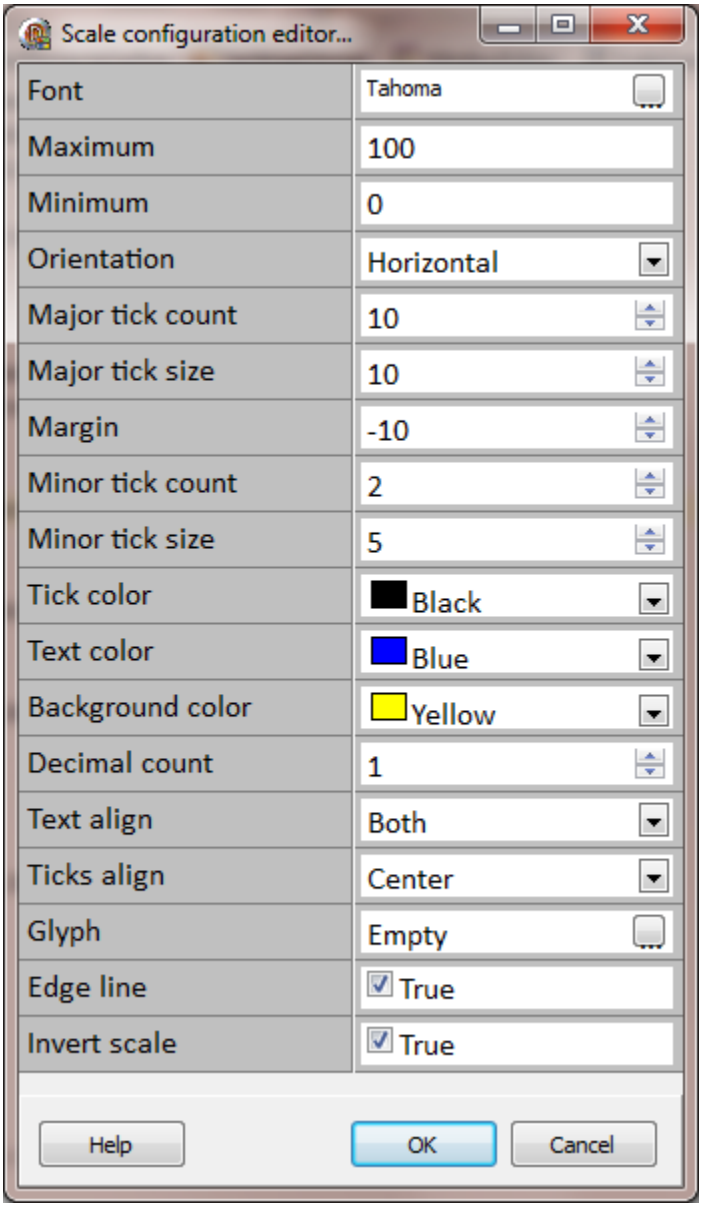
Back to list

Scale



The “Scale” display editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.

Note: When using the editor and a change is made to a property, click on another property to end the edit mode for the changed property and to see the change on the scale.



Font

The text font if the scale is configured to display text.

Maximum/Minimum

This is the maximum/minimum value of the scale. The value can be static, defined in this configuration window or configured to collect the values from the engineering units property of an analog point, at runtime. The static values will be used at design time.

Orientation

The scale can be vertical or horizontal.

Major tick count

The number of major ticks. The tick count is always one more than configured. The ticks can be hidden, for text only scales, by setting the “Major tick size” to zero (0).

Major tick size

This defines the length of the major tick and can be set to zero.

Margin

When the scale is vertical or horizontal and text is displayed, the text is centered on the major tick mark location. The top and bottom of the scaled are at the bounds of the scale graphic element rectangle. This property alters the size of the scale. A negative number decreases the scale size and a positive number increases the scale size.

Minor tick count

The number of minor ticks. The ticks can be hidden, by setting the “Minor tick size” to zero (0).

Minor tick size

This defines the length of the minor tick and can be set to zero.

Tick color

This property defines the color of the ticks. It is the same color value as the pen color.

Text color

This property defines the color of the text. It is the same color value as the foreground color.

Background color

This property defines the color of the scale background. It is the same color value as the background color. To not display a background color, set the “Transparency” property to “Transparent".

Decimal count

This property defines the number of decimal points to display for each major tick division, if text is visible.

Text align

This property defines if the text of the scale is to be displayed. This property is combined with the “Ticks align” property.

Vertical scale
If the ticks align is left or right, the text is aligned to the left or right, automatically. Select “None” to not display the text.

If the ticks align is center, the text can be aligned to the left, right, both or none.

Horizontal scale
If the ticks align is top or bottom, the text is aligned to the top or bottom, automatically. Select “None” to not display the text.

If the ticks align is center, the text can be aligned to the top, bottom, both or none.

Ticks align

The ticks can be aligned to the left/top, right/bottom or center of the scale graphic element rectangle.

Glyph

The scale can display a picture. If the text is "Empty" the scale will not display a picture. If the text is "Loaded" a picture has been loaded for viewing. Select the button in the field to load the editor. Note: The “transparency” must be set to “transparent” for the picture to be seen.

Edge

If the tick align is left or right a line can be drawn from the first major tick to the last major tick.

Invert scale

If the text of the scale is visible the normal progression is maximum value at the top/left and minimum value at the bottom/right. If enabled the scale will be reversed.

Fixed (website only)

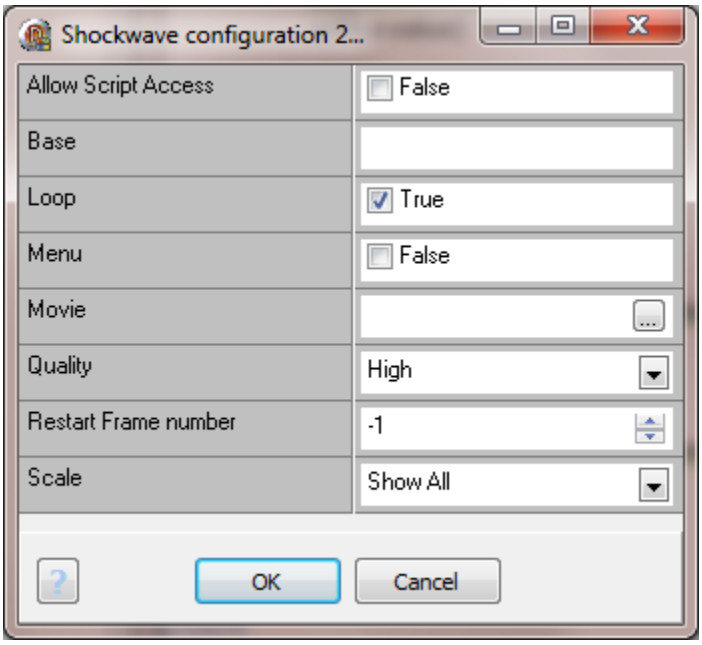
The scale will not scroll if the browser window is scrolled.

Back to list

SHOCKWAVE

Note: Adobe ended support for Adobe Flash Player, December 31, 2020 and Microsoft does not support Flash Player in Windows 10. Web browsers may support Flash Player by embedding Flash Player in the browser program. Shockwave support **will be removed** in a future version of the HMI.

The "Shockwave" display editor is accessed via the main menu "Objects/Edit" or via the right click popup menu "Edit".



Allow script access

Allow outbound scripting from within flash file.

Base

This would be the base directory or URL to resolve relative paths in a flash file.

Loop

If true then animation will repeat, false the animation plays only once.

Menu

If true the menu contains all commands. If false only the 'About' and 'Settings' menus are enabled. **Note:** This behavior depends on the version of Shockwave Flash OCX.

Movie

The URL or path and file name of the flash file (SWF). This is not affected by the 'base' attribute above.

Quality

The quality of the animation.

Restart frame number

If the point.item source evaluates to true the flash file is played. If the value then becomes false the movie is stopped. If the point.item evaluates to true the flash file is resumed. If the value in this attribute is -1 it resumes from the current frame number. Any other number and the flash file resumes from the entered frame number. To restart from the beginning set this value to 0. **Note:** While the movie is commanded to 'stop' any sounds will continue to play and animations may continue.

Scale

Any scaling of the animation.

Notes:

1. Only the “Movie” and “Fixed” property apply to the website graphic.
2. Testing Firefox version 59.0.2, Shockwave Flash version 29.0 r0 was successful.
3. Testing IE version 11.09600.18762IC, Shockwave Flash version 23.0.0.185 was successful.

Back to list

TIME/DATE

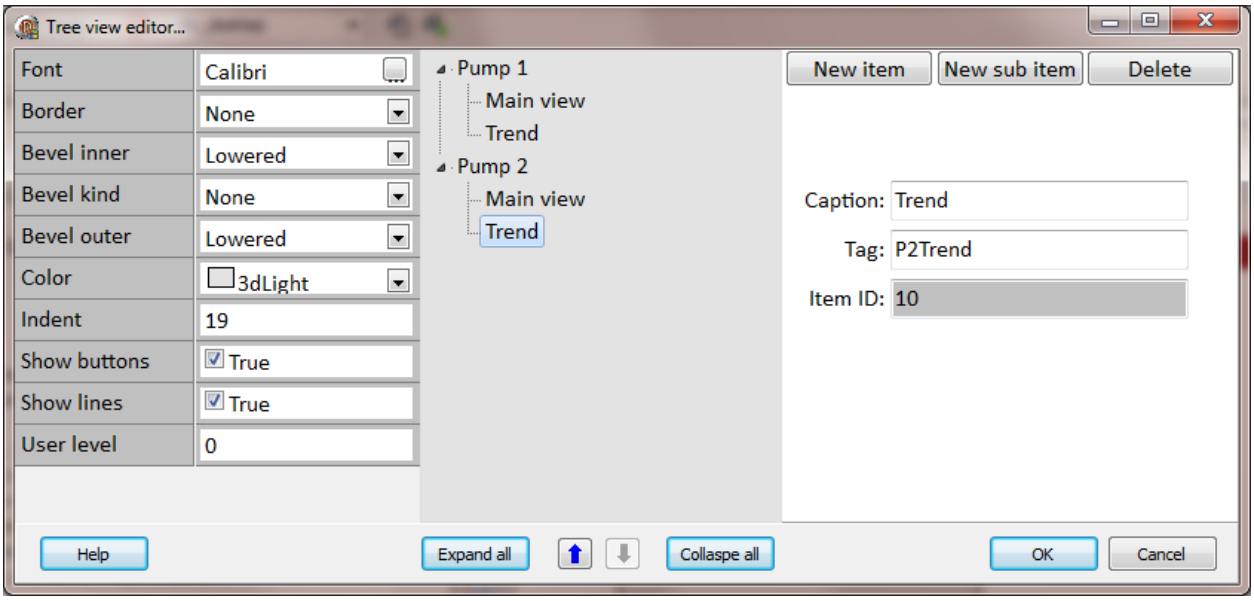
10/25/2013 10:46:53 AM

See animations.

Back to list

TREEVIEW

The Treeview graphic element is static but, can be modified via the graphic script animations.



Font

The treeview font settings.

Border

This defines if the border is visible.

Bevel inner/kind/outer

These properties define how the border will appear.

Color

This defines the color of the treeview. It is the same color property as the foreground color.

Indent

This specifies the amount of indentation, in pixels, when a list of leaves is expanded.

Show buttons

This specifies to display plus (+) and minus (-) buttons to the left side of each parent leaf. The plus (+) and minus (-) might be replaced by other glyphs by the OS.

Show lines

This specifies to display the lines that link child leafs to their corresponding parent leaf.

Expand

When this property is true the treeview will expand all branches when the treeview is first rendered.

New item

This command creates a new leaf at the root of the tree.

New sub item

This command creates a new leaf on a branch from the root of the tree or another sub branch.

Delete

This command deletes a leaf or a leaf and all its branches and leafs.

Caption

This is the text displayed on the treeview for the branch or leaf.

Tag

This is a string passed to the OnTreeviewClick script event.

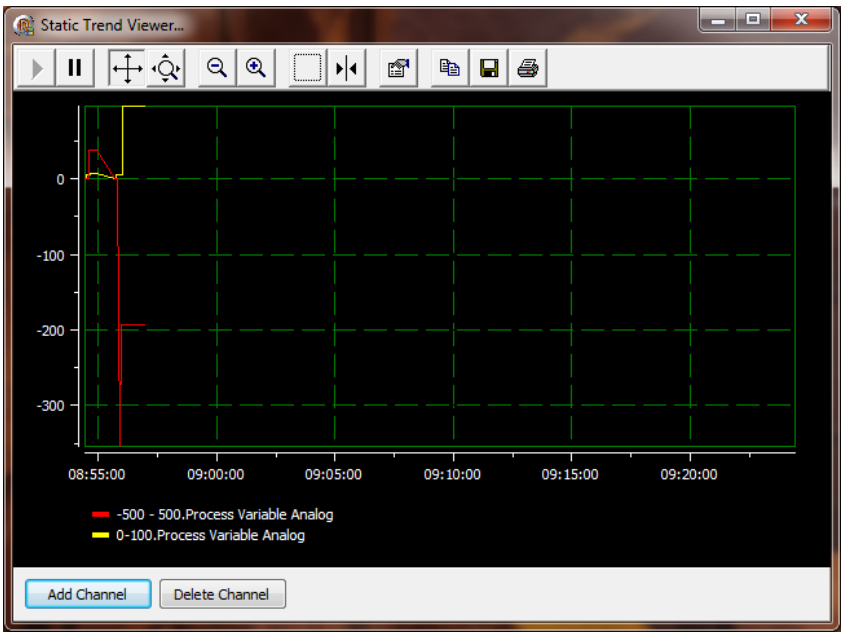
Item ID

Each branch and item has an ID. Each ID is unique and cannot be changed. This value is passed to the OnTreeviewClick script event.

See OnTreeviewClick for handling mouse clicks in the treeview graphic element.

Back to list

TREND



Two types of trends are provided.

Native

Native trends use the "Data logger" and store/retrieve data values from an internal database.

All of the "Native trend" settings are accessed via the native trend animation dialog.

ODBC

ODBC trends use the ODBC protocol to retrieve data values from an external database. The data could be recorded using the ODBC Data Logger or an external program.

All of the "ODBC trend" settings are accessed via the ODBC trend animation dialog.

TREND (STATIC)

This trend graphic element is based on the "Trend" graphic element. This trend is static. The data is loaded and "live" data is not added to the trend. This trend type is used to load static data for display, printing, screen captures, etc.

The "ViewDateRangeTrendHistory" is another option for viewing static trend log data.

Any added, changed functionality or removed trend properties will be defined below. If the property is not defined here, it is defined in the "Trend" graphic element or "ODBC Trend" graphic element.

Start time

This is time to being loading data on the start date. *3

Start date

This is the start date to load data into the trend. *3

End time

This is final time to load data on the end date. *3

End date

This is the end date to load data into the trend. *3

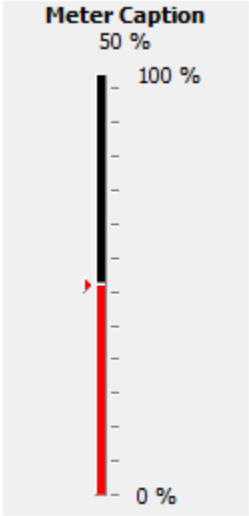
Use trend date for title

If enabled the title above the trend will be a date. If the trend is showing data that is within one day the title will be that date. If the trend is showing data that spans more than one day, the title will be <start date> - <end date>. E.g. 3-1-2017 - 3-3-2017

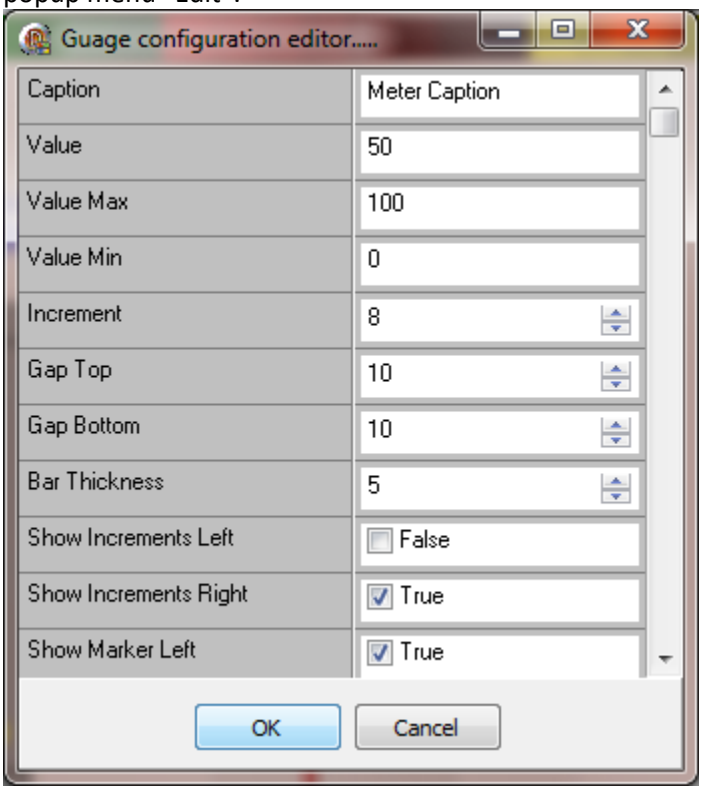
Notes:
1) Use this graphic element with caution. If the amount of data to load, ((data logging frequency X number of days) X number of data points) is large the program/computer may use up all available memory.
2) Displaying the current day can give partial results. The data logger has a buffer and does not write the data to disk until required. It is advised not to use this feature to display the data of a point for the current day. Use a trend graphic element.
3) If the trend type is ODBC and the X-Axis type is "Date/Time" the value entered must convert to a valid date and/or time.

Back to list

VERTICAL GAUGE



The “Vertical gauge” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Caption

The text at the top of the element.

Value

The value of the slider. (Provided to show how the control would appear at runtime with a value)

Value max

The maximum value of the "value".

Value min

The minimum value of the "value".

Increment

The frequency of the tick marks, if shown. Lower numbers are more tick marks.

Gap top

This is the gap between the top of the element and the indicator part of the control. The "show value text" and the caption value affect the "top" of the control from the indicator part.

Gap bottom

This is the gap between the bottom of the element and the indicator part of the control.

Bar thickness

The thickness of the filled bar in the middle of the element

Show increments left

Show tick marks on the left side of the filled bar.

Show increments right

Show tick marks on the right side of the filled bar.

Show marker left

Show the triangle position indictor on the left side of the filled bar.

Show marker right

Show the triangle position indictor on the right side of the filled bar.

Marker color

The color of the marker. The color of the filled bar is the brush foreground color and background color. The foreground color is from the bottom to the position of the value. The background is from the value position to the top of the filled bar. The colors can also be changed via animation.

Text color

The color of all the text in the element.

Panel color

The background color of the panel. The panel can also be transparent.

Digits

The number of digits to display after the decimal. If the value is zero the decimal is not shown.

Show value text

Enable this attribute to display the value at the top of the control below the caption.

Units

The engineering units, if any.

Used inherited units

This overrides the "Units" above and uses the units from the point assigned via animation.

Used inherited ranges

This overrides the "Value Min/Max" above and uses the engineering min/max from the point assigned via animation.

Font name

The font type for the element

Font size

The size of the font for the element.

High range Text margin

The high range text vertical margin. A positive number positions the text lower, a negative number positions the text higher.

Low range Text margin

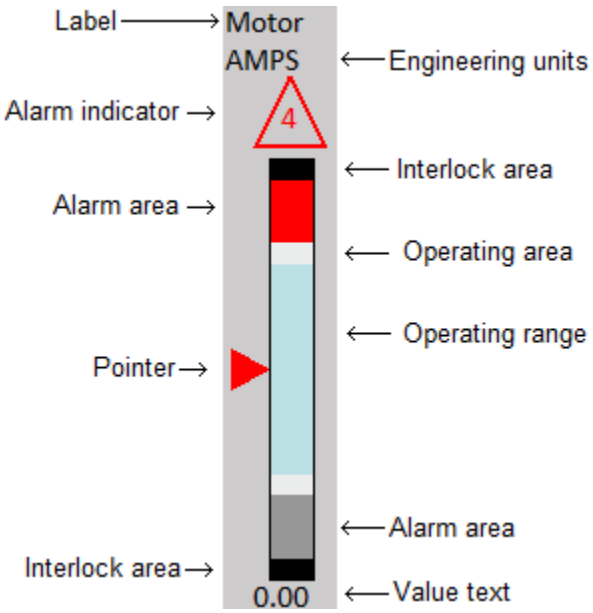
The low range text vertical margin. A positive number positions the text lower, a negative number positions the text higher.

Hide range text

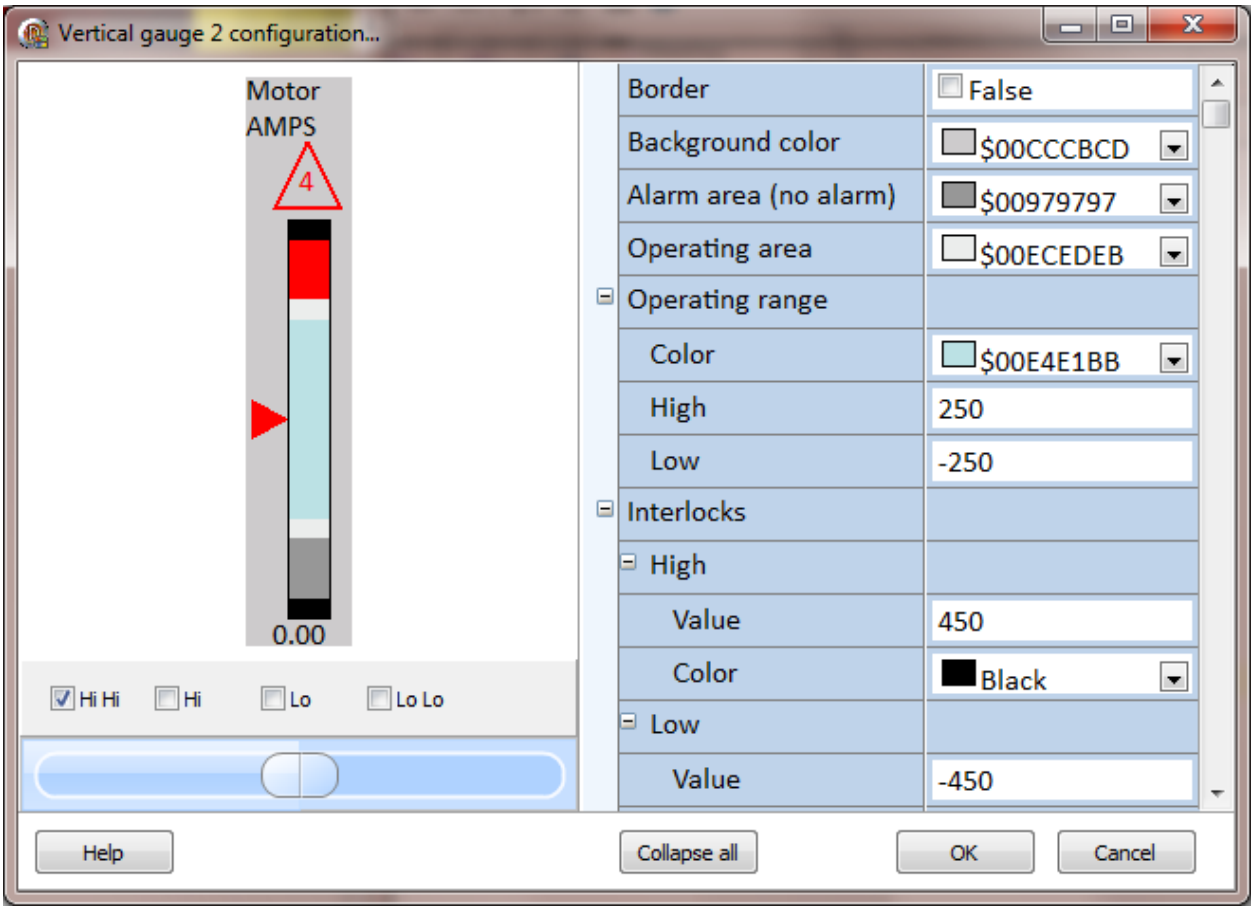
If enabled the range text will not be displayed in the gauge.

Back to list

VERTICAL GAUGE 2



The "Vertical gauge 2" editor is accessed via the main menu "Objects/Edit" or via the right click popup menu "Edit".



Border

If enabled, the gauge will display a border around the edge of the gauge using the pen color and pen width.

Background color

The background color of the gauge. The same as the foreground color setting.

Alarm area (no alarm)

The alarm area color when all alarms are inactive.

Operating area

This is the color for all indicator area not defined by another area color.

Operating range color

The color of the operating range area.

Operating range high/low (engineering units)

These two properties define the operating area of the indicator scale using engineering units.

Interlocks high/low (engineering units)

These areas are at the top and bottom of the scale.
Value: The starting value, in engineering units. The end is the end of the scale.
Color: The color of the area. The color does not change.
Note: To disable the interlock graphic, set the value for high greater than the engineering range and the value for the low, less than the engineering range.

Indicator area, height/width

These properties define size of the indicator area, the bar. The values are in percent of graphic element size.

Indicator area, vertical/horizontal offset

These properties define the position of the indicator bar inside the graphic element in percent of graphic element size.

Indicator area, border width and color

These properties define a border around the indicator area. To not display a border, se the width value to zero (0).

Label

Text

The text to display at the top of the graphic element. To not display any text, clear the property (no characters in the field).

Font

This property defines the text settings for the label text.

Color

The label text color.

Alignment

The label text can be aligned to the left, center or right of the graphic element.

Vertical/horizontal offset

These properties define the offset position of the text label inside the graphic element in percent of graphic element size.

Engineering units

The engineering units (from the point settings) can be displayed below the label text using the same font settings as the label.

Pointer

Color

The pointer color.

Position

The pointer can point to the left of right.

Size

The pointer size.

Use alarm color

If enabled, the pointer will change color to the highest active alarm color. If all alarms are inactive, the color will be the value defined for the pointer.

Value

Visible

If enabled, the indicator value will be displayed, in text, at the bottom of the graphic element.

Font

This property defines the text settings for the label text.

Color

The value text color.

Alignment

The value text can be aligned to the left, center or right of the graphic element.

Decimal count

This property defines the number of digits after the decimal point. Set the value to zero (0) to disable the decimal point and decimal digits.

Color

The value text color.

Vertical/horizontal offset

These properties define the offset position of the value text inside the graphic element in percent of graphic element size.

Alarms

Enable

If enabled, the indicator area defined will be rendered using the values configured. **Note:** This property **DOES NOT** alter the enabled alarm(s) property of the point this gauge is monitoring. This property **ONLY** alters how the gauge is rendered.

Color

The area color when the alarm is active. If Hi Hi and Hi are active the Hi Hi color will be displayed. If Lo Lo and Lo are active the Lo Lo color will be displayed.

Text

This is a single character to be display in the 'Alarm Indicator' area (see below) when the alarm is active.

Alarm indicator

Shape

This property defines the shape of the alarm indicator. If 'None' is selected the alarm indicator will not be displayed.

Font

This property defines the text settings for the label text.

Vertical/horizontal offset

These properties define the offset position of the alarm indicator inside the graphic element in percent of graphic element size.

The slider and checkboxes at the bottom of the window are provided to simulate the value and alarm states of a point for visual testing of the gauge.

Notes:

1) For the website gauge, some attributes, especially text do not appear the same in the editor as when a browsers renders the gauge. Normally the gauges can be made to look the same in the HMI and in a browser window with adjustment to the various properties.
2) The value text offsets are based on the size of the rectangle to contain the text which is based on the size of the font. The offset is not based on the graphic element size as it is with the native element.

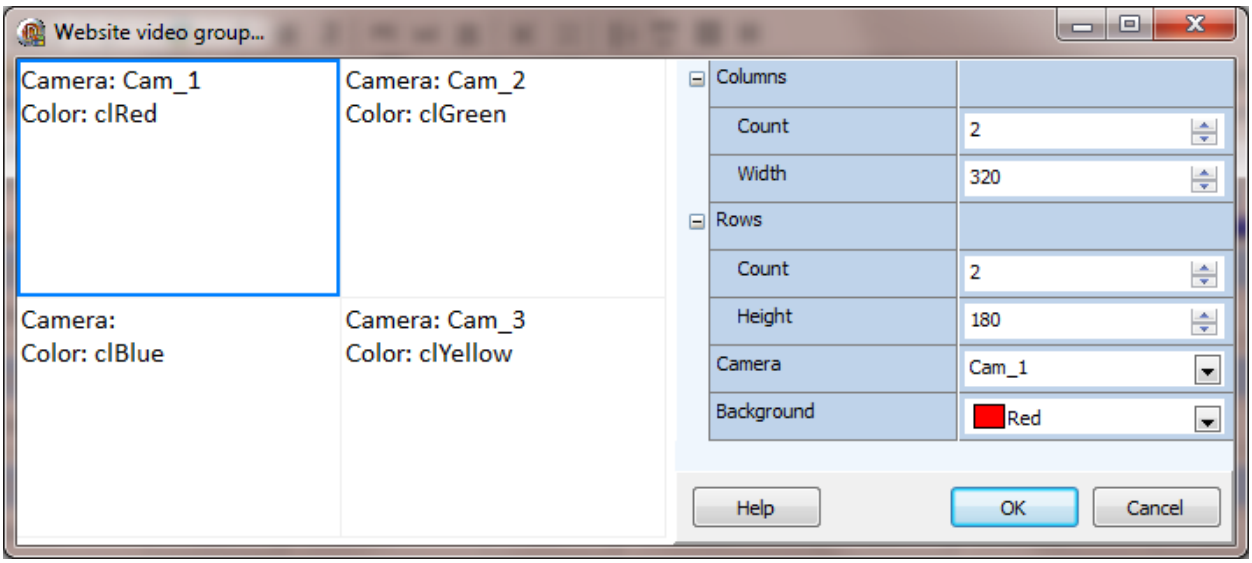
Back to list

VIDEO

All of the “Video” settings are accessed via the video animation dialog.

Back to list

VIDEO GROUP



The video group editor is accessed via the animation configuration dialog.

Website: Web browsers will limit the number of concurrent connections from a web page to a server. Each website video/camera graphic element uses one connection so, the number of videos per page may be limited by the browser. To address the requirement to display multiple cameras on a single page the website video group graphic element is provided. The graphic element uses only one connection per graphic element. **Note:** The video/camera is scaled to fill the cell.

Columns/Rows count	This property defines the number of columns and rows. One video/camera source per cell.
Columns/Rows Width/Height	This property defines the width of the columns and the height of the rows.
Camera	This property defines the camera for each cell.
Background	This property defines the color displayed in the cell when a camera is not defined or not responding.

Page to (website)
Screen to (regular)

If the mouse is clicked in the cell bounds, the configured page/screen will be opened. Note: For regular screens, if script animation is enabled and the left mouse button is pressed in a cell, the column, row and camera name are placed in the script object.

ge.column	=	cell column
ge.row	=	cell row
ge.text	=	cell camera name

Example:

```
procedure OnMouseUp;
begin
 ShowMessage(IntToStr(ge.column) + '~~' +
             IntToStr(ge.row) + '~~~' +
             ge.text);
end;
```

User level

This property defines the minimum user level. This applies to the "Page to/Screen to" command.

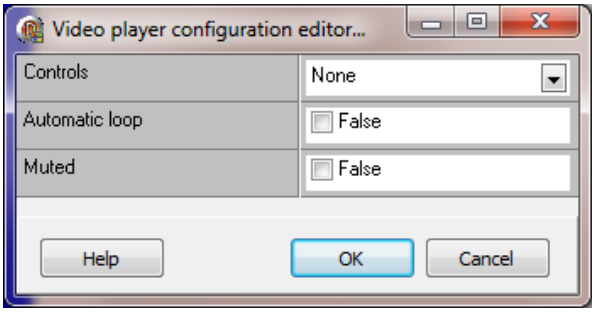
Back to list

VIDEO PLAYER

The “Video player” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.
Windows supports:

Windows Media formats: .asf, .wma, .wmv, .wm
Windows Media Metafiles: .asx, .wax, .wvx, .wmx, .wpl
Microsoft Digital Video Recording: .dvr-ms
Audio Visual Interleave: .avi
Moving Pictures Experts Group: .mpg, .mpeg, .m1v, .mp2, .mp3, .mpa, .mpe, .mpv2, .m3u
Audio for Windows: .wav
CD Audio Track: .cda
Indeo Video Technology: .ivf
QuickTime Movie file: .mov
MP4 Audio file: .m4a

The website video player (HTML5) only supports mp4 and ogg (file extension ogv) video formats. IE and Safari only support mp4. Chrome, Firefox and Opera support both.



Controls

The control bar contains buttons to start, pause, rewind, etc., the video.

None	The control bar is not visible. The video will be played when the window is opened.
Mini	The 'mini' control bar is displayed at the bottom of the graphic element.
Full	The 'full' control bar is displayed at the bottom of the graphic element.
Note:	For the website, any setting other than “None”, the control bar will be visible.

Automatic loop

When the video reaches the end, the video will restart from the beginning.
If enabled, the video will start when the window is opened.

Muted

If enabled, the initial audio state will be muted.

Back to list

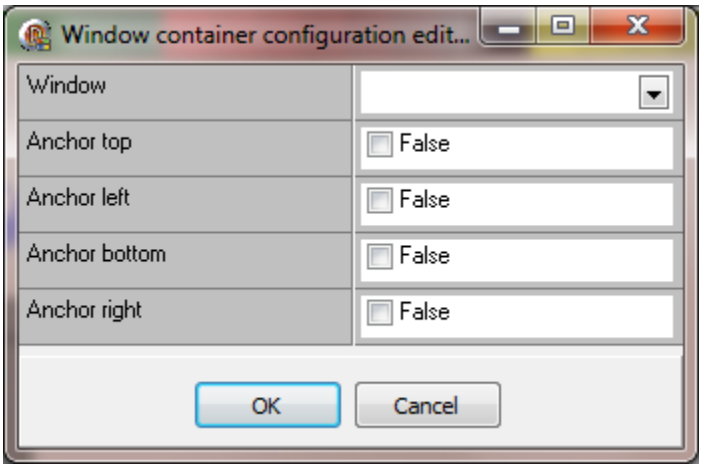
Window container

This element is used to display a window inside of another window.

For example, the project requires a collection of 16 LED indicators to be at the top of all windows and to display the same information. Placing all the LEDs in one window, setting the animations and then referring to the window in all other windows would be a better solution than re-creating the LED 'panel' in each window.

The “Window container” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.

Note: The dimensions of the window container must, at a minimum. be large enough to contain the source window. If the container is too small at runtime. the container will not be visible and a message will be placed in the “Event log”.



Window

The name of the source window to display in the destination window.

Anchor top, left, bottom, right

If enabled the corner of the source window will be anchored to the corner of the destination window.

For example:
To anchor across the complete top of the destination window, enable the top, right and left.

To anchor across the complete left side of the destination window, enable the top, bottom and left.

To anchor across the complete bottom of the destination window, enable the left, bottom and right.

Design time

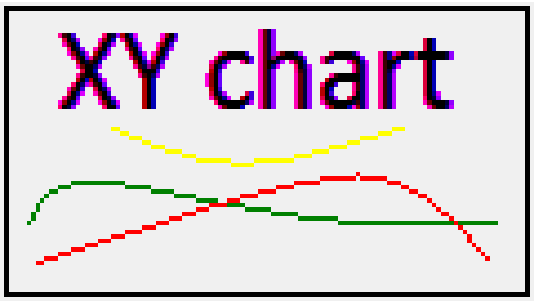
Notes:
The source window image is loaded from disk when:
1) The window is open.
2) The size of the element is modified.
3) When the element is edited via the configuration window.

Runtime

Notes:
1) The source window script settings do not apply.
2) If the hide/show animation is configured for the window container, all the objects of the source window will use the same hide/show configuration as the window container when the window is displayed in the window container.

Back to list

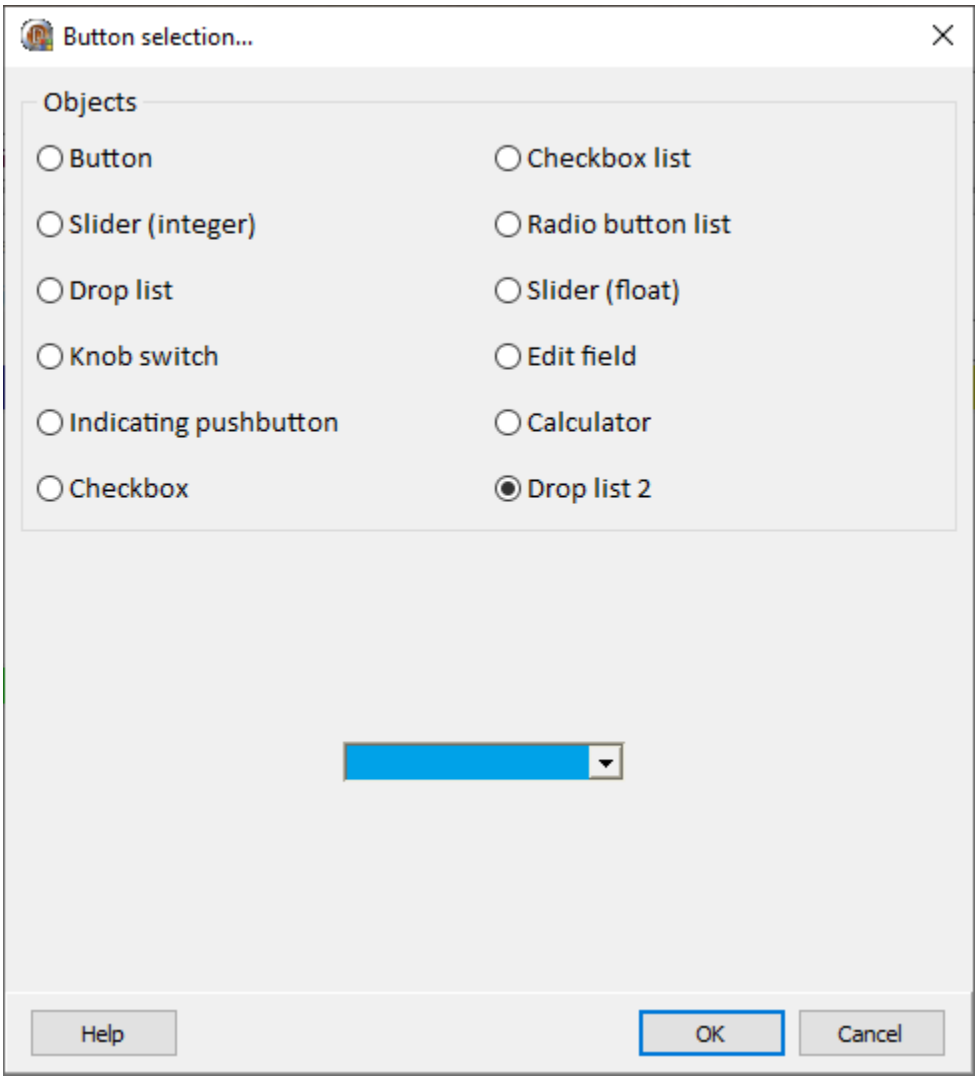
XY CHART



All of the “XY chart” settings are accessed via the XY chart animation dialog.

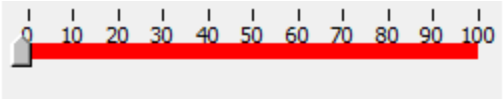
BUTTON OBJECTS

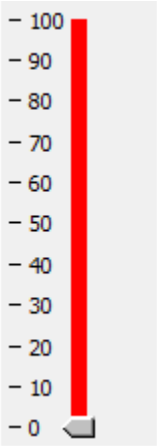
Almost any graphic element can respond to mouse clicks. Refer to the "Animation" section for more information.



The button objects are listed below in alphabetical order.

ANALOG SLIDER

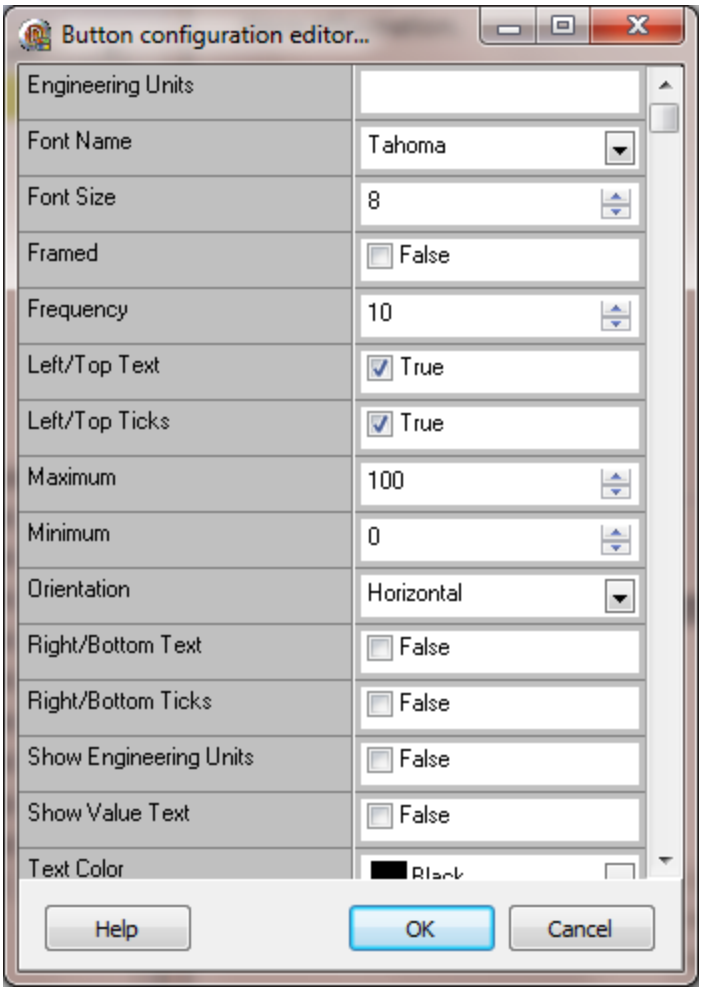




The “Analog slider” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.

There are two sliders. One is for integers and the other is for floating point.

There are two configuration areas for this graphic element. The first is accessed via selecting the control in the window editor and then selecting the Objects/Edit (CTRL+E) menu item and the second is accessed via the animation window.



Decimal count

The number of decimal places.

Engineering units

Text to display near the thumb if enabled.

Font name and size

Text font type and size for numeric scale and value text, if enabled.

Framed

Draw a frame around the control.

Frequency

The number of division for the ticks and scale text.

Left/Top text

If the control is vertical, show text scale values on the left side. If the control is horizontal, show text scale values on the top of the control.

Left/Top ticks

If the control is vertical, show tick marks on the left side. If the control is horizontal, show ticks on the top of the control.

Maximum

The maximum value of the control.

Minimum

The minimum value of the control.

Orientation

The control can be vertical or horizontal.

Page size

The amount the position will change when the “Page up” or “Page down” key is selected.

Right/Bottom Text

If the control is vertical, show text scale values on the right side. If the control is horizontal, show text scale values on the bottom of the control.

Right/Bottom ticks

If the control is vertical, show tick marks on the right side. If the control is horizontal, show ticks on the bottom of the control.

Show engineering units

The engineering units will be displayed near the thumb.

Show value text

The current value will be displayed near the thumb.

Thumb size

The size of the thumb in the control.

Text color

The color of the text on the control.

Thumb type

The thumb style can be Box, Circle, Diamond, Pointer or Square.

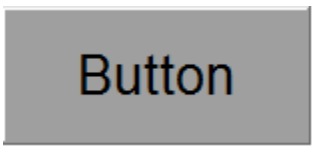
Track width

The width of the bar the thumb slides over.

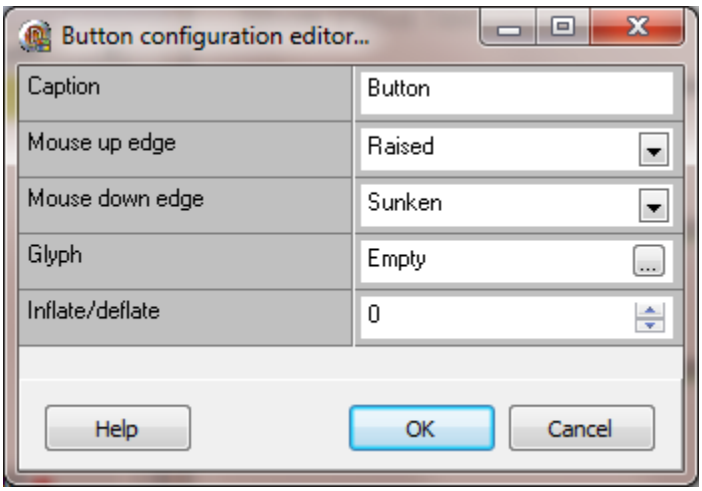
Fixed (website only)

The slider will not scroll if the browser window is scrolled.

Button



The “Button” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Colors

Foreground color = caption (text) color
Background color = inside area of the button

To only change the color of the buttons background at runtime, set the text color at design time (foreground color) and set the button background color at design time. In the animation configuration dialog, select the brush and set the foreground color to the same color as selected at design time and set the background color to the desired color.

Caption

The text displayed in the button

Mouse up/down edge

The border style of the button when the left mouse button is up or down in the button.

Fixed (website only)

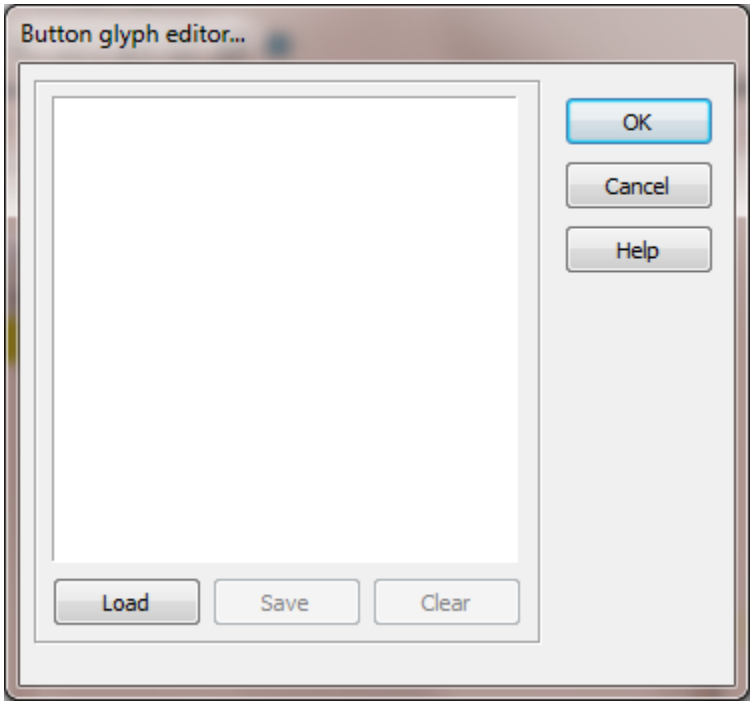
The button will not scroll if the browser window is scrolled.

Glyph

The button can display a picture. If the text is "Empty" the button will not display a picture. If the text is "Loaded" a picture has been loaded for viewing. Select the button in the field to load the editor.

The text in the button is always aligned vertical and horizontal center.

Button glyph editor



Load

Select the load button to import a graphic file.

Save

Select the save button to export the picture to a bitmap.

Clear

Select the clear button to erase the image.

OK

Select the OK button to accept the settings. When the OK button is selected the button will be sized to the dimensions of the bitmap. The button can then be sized as needed.

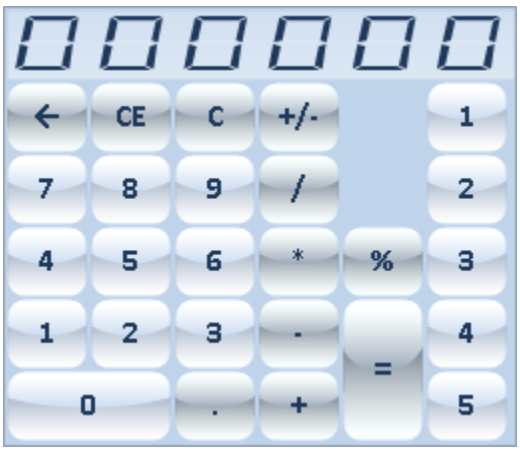
Cancel

Select the cancel button to not make any changes to the picture.

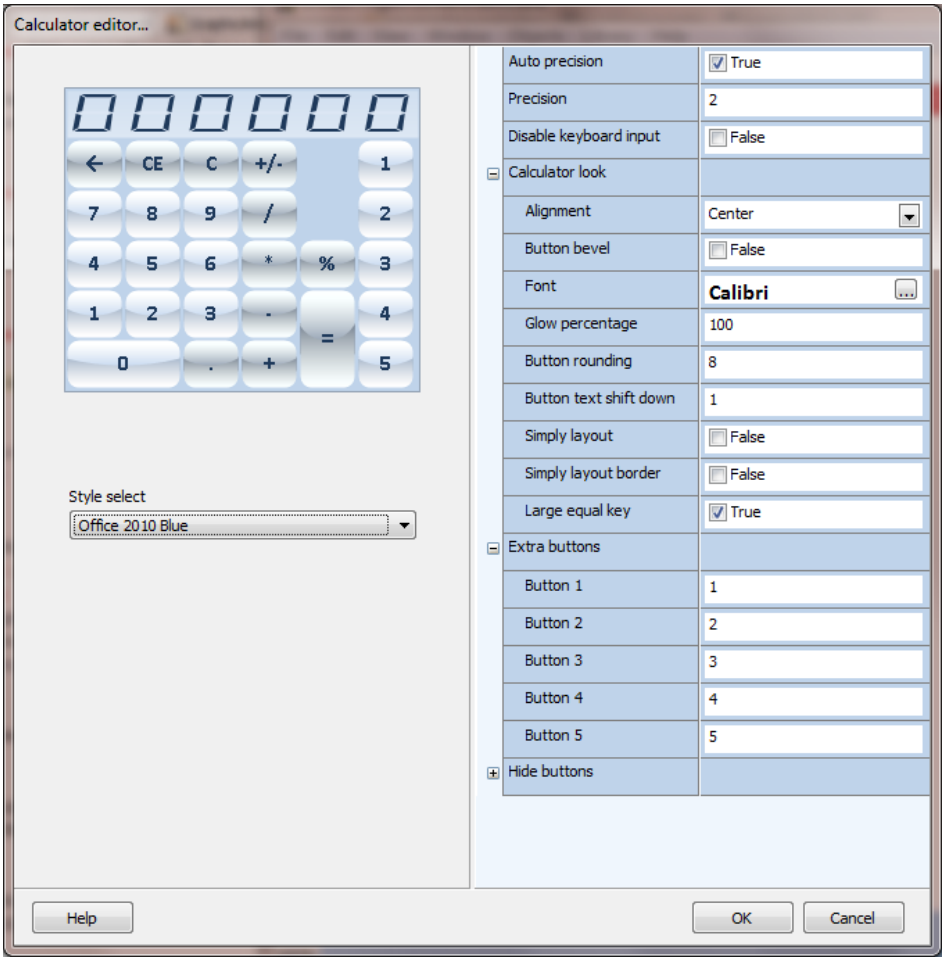
Inflate/deflate

The value can be -10 to 10. If the value is not equal to zero (0) and the left mouse button is pressed inside the button, the button will be rendered with the size (in pixels) increased/decreased, all sides, until the left mouse button is released.

CALCULATOR



The “Calculator” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Note:

When changing values in the configuration editor, the change will not be reflected in the example calculator until the left mouse button is clicked outside the property editor/item.

Auto precision/Precision

These properties configure the calculator to display as integer or float and the result precision.

Disable keyboard input

This property sets if the keyboard can be used for input.

Calculator look

Alignment:	Button text alignment
Button bevel:	Button bevel
Font:	Button font settings.
Glow percentage:	Button glow.
Button rounding:	Button corner rounding
Button text shift down:	Positive value moves the button text down, negative value moves the button text up
Simply layout:	Buttons are rendered without glow settings
Simply layout border:	Show/hide button border with simply layout enabled
Large equal key:	If true the equal/enter key is larger.

Extra buttons

A name in an extra button field creates the field and is displayed on the right side of the calculator. Logic processing for the extra key is via scripting.
See OnCalculatorButtonClick.

Hide buttons

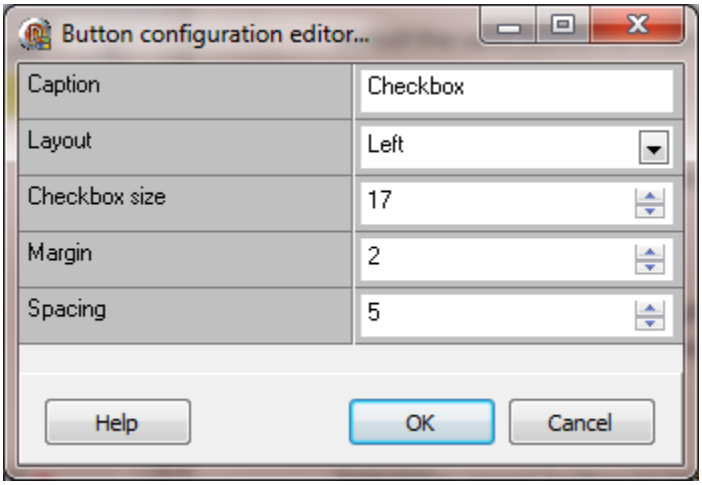
This property allows for buttons to be hidden and disable keyboard input for the button.

Style select

Calculator style selection.

Checkbox

The “Checkbox” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



When the left mouse button is pressed and released in the element the value of the source point is negated. If this is a PLC point a write command is issued.

Colors

Foreground color = caption (text) color

Caption

The text displayed in the button.

Layout

The location of the checkbox in the button.

Checkbox size

The size of the checkbox.

Margin

The distance from the checkbox to the border of the button based on the layout. If the value is -1 the checkbox/caption is centered. Negative numbers move the checkbox up/left, positive numbers move the checkbox down/right.

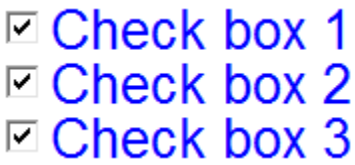
Spacing

This is the space between the checkbox and the text.

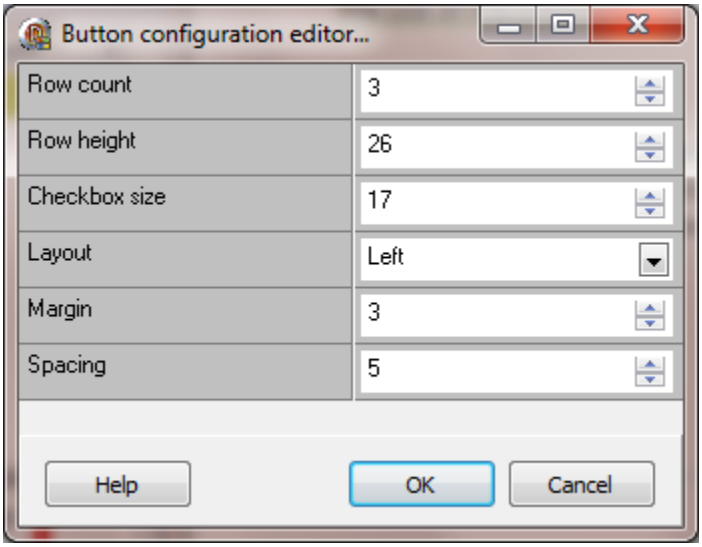
Fixed (website only)

The checkbox will not scroll if the browser window is scrolled.

Checklist



The “Checklist” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



When the left mouse button is pressed and released in a checkbox, the values of the checkboxes are examined and the 16 bit value is written to the point.
The current internal value is used. If the checkbox count is less than 16, the unused bits are not changed. If this is a PLC point a write command is issued.

Colors

Foreground color = caption (text) color

Row count

The number of checkboxes in the list. (2-16).

Row height

Each checkbox is one row in a column. This is the height of each checkbox row.

Checkbox size

The size of the checkbox.

Layout

The location of the checkbox in the row.

Margin

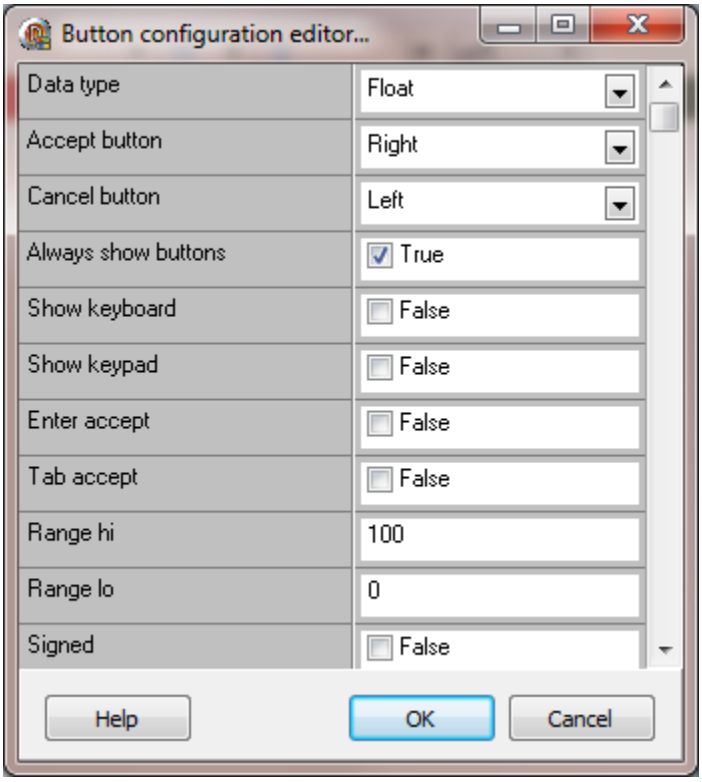
The distance from the checkbox to the border of the list based on the layout. If the value is -1 the checkbox/caption is centered. Negative numbers move the checkbox up/left, positive numbers move the checkbox down/right.

Spacing

This is the space between the checkbox and the text.

Edit Field

The “Edit field” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”. See Edit field animation. Website EditField



Data type

The data type the user can enter in the edit field.

Accept button

If this attribute is enabled, a button will be displayed on the selected side of the text edit field to allow the user to accept the value in the field. A script function can also be executed after internal processing, “OnAccept”.

Cancel button

If this attribute is enabled, a button will be displayed on the selected side of the text edit field to allow the user to cancel edits. The “Escape” key also cancels edits. A script function can also be executed after internal processing, “OnCancel”.

Always show buttons

If this attribute is enabled, enabled button will always be displayed regardless of the edit field focus.

Show keyboard

If this attribute is enabled, when the user clicks the mouse in the edit field, a virtual keyboard will be displayed.

Show keypad

If this attribute is enabled, when the user clicks the mouse in the edit field, the "GetUserInputFloat" dialog will be displayed.

Enter accept

If this attribute is enabled, when the user presses the "Return" or "Enter" keys the edit will be accepted.

Range hi/lo

The value limits if the "data type" is float or numeric.

Signed

The value can be negative if the "data type" is float or numeric.

Maximum length

If the "data type" is not float or numeric, this attribute is the maximum length of the text. This applies to data the user can enter. The edit field will display the complete data from the source regardless of the length.

Decimal count

If the "data type" is float, this attribute is the maximum number of digits after the decimal place.

Tab order

This attribute sets the order when advancing from one control to the next. The tab order is also affected by other controls on the window.

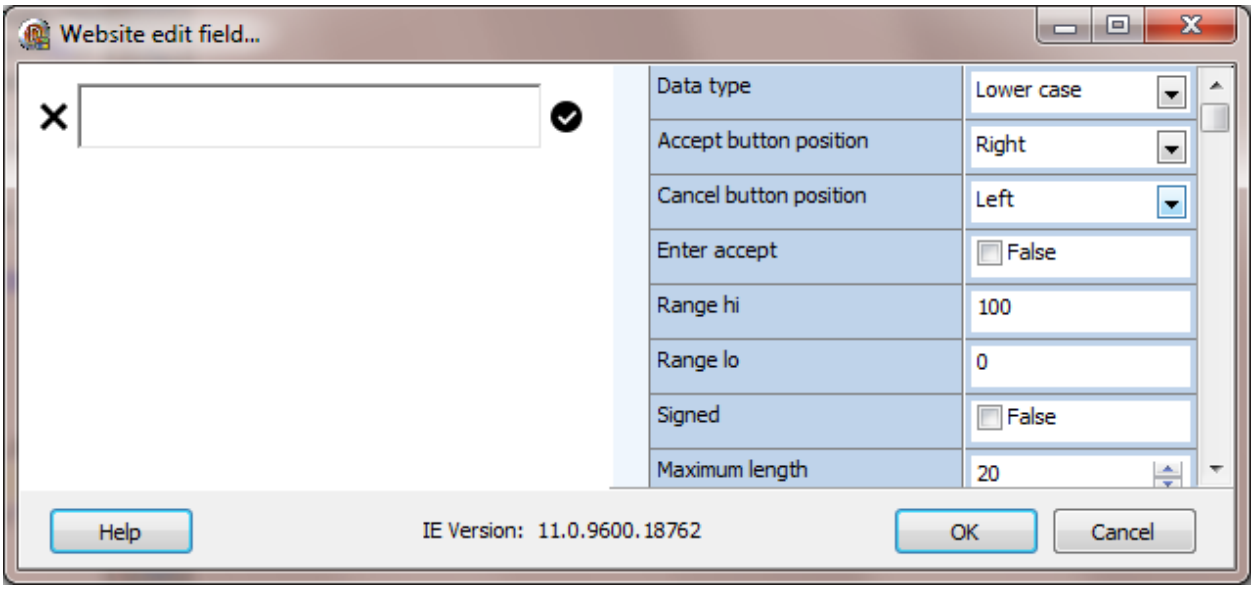
Automatic advance

When the user selects to accept the edit the next control in the tab order will be selected.

Label position/font name/size/color/transparent.

These are attributes for the optional label.

Website Edit Field



The "Edit field" editor is accessed via the main menu "Objects/Edit" or via the right click popup menu "Edit". See Edit field animation.

Notes:

1) When using the editor and a change is made to a property, click on another property to end the edit mode for the changed property and to view the change in the example.
2) Not all browser input restrictions are foolproof. The HMI will validate all inputs when the user accepts the input and the value is transmitted to the HMI from the browser.

Data type

The data type the user can enter. Not all browsers support this property. But, when the user enters a value and accepts the value, the value is sent to the HMI and validated against the data type. E.g. The data type is "Upper case" and the user enters "abc". The HMI will convert the "abc" to "ABC".

Accept button position

If this attribute is enabled, a button will be displayed on the selected side of the text edit field to allow the user to accept the value in the field.

Cancel button position

If this attribute is enabled, a button will be displayed on the selected side of the text edit field to allow the user to cancel edits. The "Escape" key also cancels edits.

Enter accept

If this attribute is enabled, when the user presses the “Return” or “Enter” keys the edit will be accepted.

Range hi/lo

The value limits if the “data type” is float or numeric.

Signed

The value can be negative if the “data type” is float or numeric.

Maximum length

If the “data type” is not float or numeric, this attribute is the maximum length of the text. This applies to data the user can enter. The edit field will display the complete data from the source regardless of the length.

Decimal count

If the “data type” is float, this attribute is the maximum number of digits after the decimal place.

Field width

The buttons and text field are in a container. This property defines the space, in percentage, the text field occupies. The buttons, if enabled, will occupy the remaining space. A text field too large might cause button misalignment.

Fixed

The edit field will not scroll if the browser window is scrolled.

Accept button

✔ The default accept button is 16 X 16. Use this property to specify another button image. The image will be scaled to the height and width settings. If the height or width size is zero, the sizes will be set to imported value.

Cancel button

✕ The default cancel button is 16 X 16. Use this property to specify another button image. The image will be scaled to the height and width settings. If the height or width size is zero, the sizes will be set to imported value.

Height/Width

The height and width of the button image.

Border visible

If this property is true a 1 pixel border will be drawn around the text input field using the pen color.

Edit mode color

The website edit mode is different from the normal PC edit mode. The normal PC edit mode is ended when the value is accepted, cancelled or the mouse is clicked outside the control. Edit mode is enabled, one control at a time.

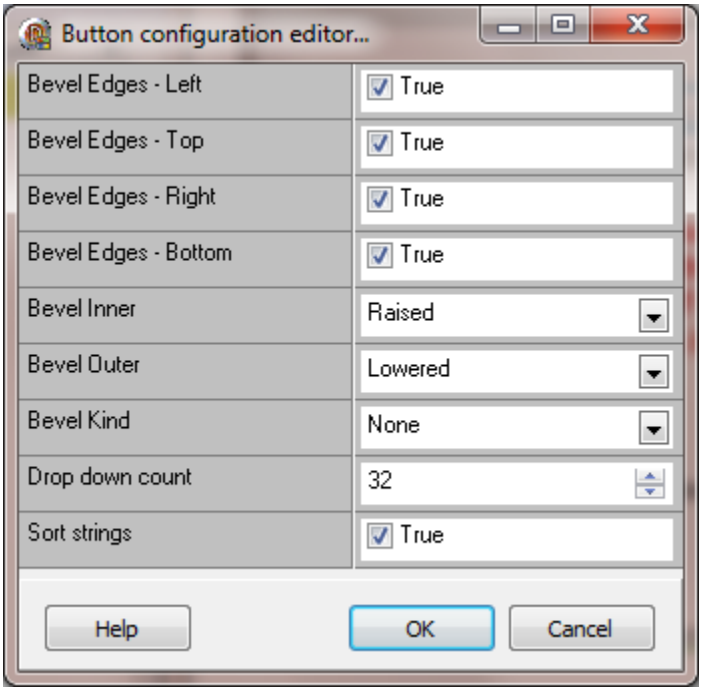
The website edit mode is ended when the value is accepted or cancelled. Multiple edit fields can be in edit mode, concurrently.

The normal control operation, displaying the source value, is active when the control is not in edit mode.

This property provides a visual indication of edit mode. Setting the “edit mode color” to the color of the background will mask the visual indication of edit mode.

Drop list

The “Drop list” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”. See Drop list animation.



Background color

This is the color of the fill.

Pen color

The color of the text.

Bevel edges (Left, Top, Right, Bottom)

Depending on the other selections true shows the edge.

Bevel inner, outer

The drop down list will render the inner or outer edge.

Bevel kind

Select the bevel style.

Drop down count

Determines the number if strings that are displayed when the drop down list appears.

Sort strings

Alphabetically sort the strings for display. Otherwise the strings are shown as entered in the list. Blank strings are not displayed.

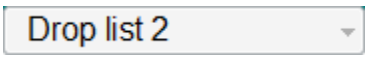
Fixed (website only)

The drop list will not scroll if the browser window is scrolled.

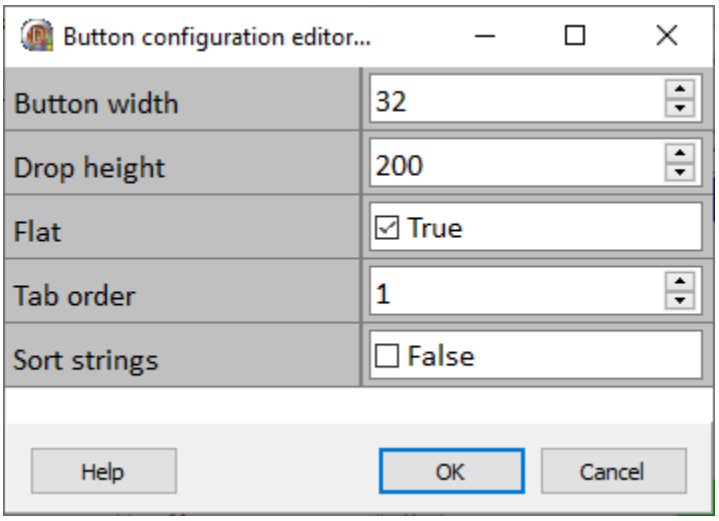
Tab order

This attribute sets the order when advancing from one control to the next. The tab order is also affected by other controls on the window.

Drop list 2



The “Drop list 2” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”. See Drop list animation.



Button width	Button width when “flat” is true.
Drop height	Height, in pixels, when the drop list is selected.
Tab order	This attribute sets the order when advancing from one control to the next. The tab order is also affected by other controls on the window.
Sort strings	Alphabetically sort the strings for display. Otherwise the strings are shown as entered in the list. Blank strings are not displayed.

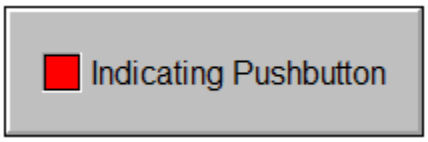
The “Drop list 2” supports some HTML tags for formatting the text displayed for each item. Tags can be combined, for example bold and underline.

Description	Start tag	End tag	Example
Bold	<B>	</B>	<B>text</B>
Underline	<U>	</U>	<U>text</U>
Italic	<I>	</I>	<I>text</I>
Strikeout	<S>	</S>	<S>text</S>
Align left	<P align="left">	</P>	<P align="left">text</P>
Align center	<P align="center">	</P>	<P align="center">text</P>
Align right	<P align="right">	</P>	<P align="right">text</P>

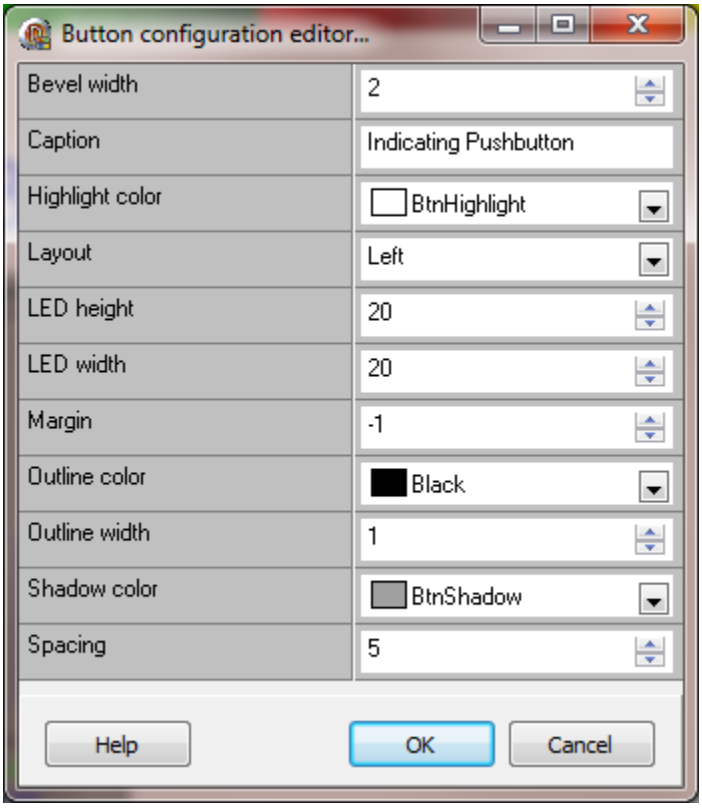
The following special characters are supported:

<	less than <	**>**	greater than >
&	&	**"**	"
™	trademark symbol	**€**	euro symbol
©	copyright symbol		

INDICATING PUSHBUTTON



The "Indicating Pushbutton" editor is accessed via the main menu "Objects/Edit" or via the right click popup menu "Edit".



This graphic element is a hybrid. The indicator part is controlled via the pen, brush, hide/show, etc. animations. The "disable" is to disable mouse commands. If the "Indicating pushbutton" animation is not enabled the mouse commands and flash (if enabled) will not function.

Colors

Pen color	LED color
Foreground color	caption (text) color
Background color	inside area of the button

Bevel width

The width of the bevel around the edge of the button.

Caption

The text displayed in the button.

Highlight color

The color of the top left portion of the bevel.

Layout

The location of the LED in the button.

LED height/width

The height and width of the LED.

Margin

The distance from the LED to the border of the button based on the layout. If the value is -1 the LED/caption is centered. Negative numbers move the LED up/left, positive numbers move the LED down/right.

Outline color

The color of the border around the button.

Outline width

This creates a border around the button.

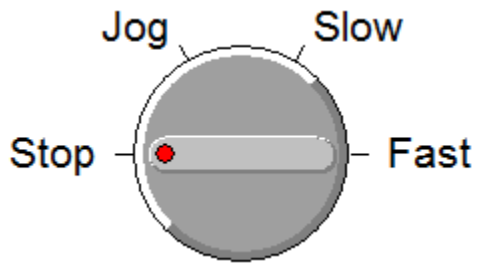
Shadow color

The of the bottom right portion of the bevel.

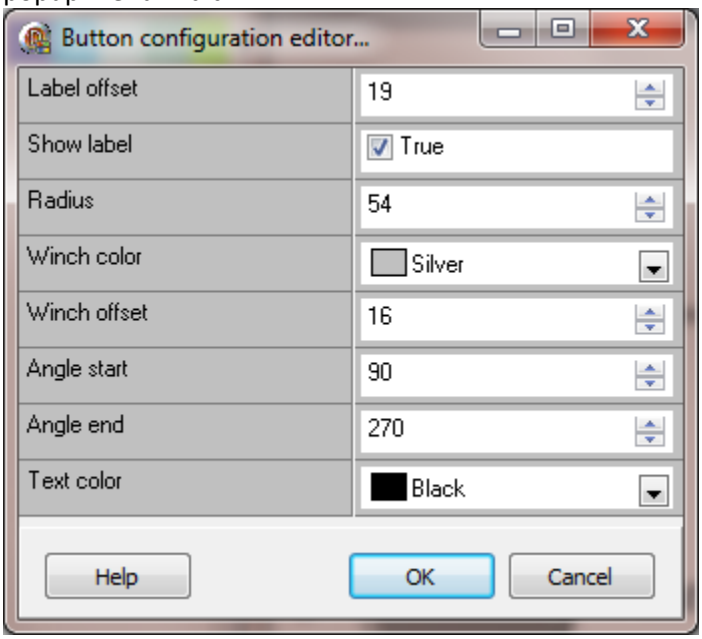
Spacing

This is the space between the LED and the text.

KNOB SWITCH



The “Knob Switch” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Colors

Pen color	button color (circle in the winch)
Foreground color	inside area of the switch
Background color	rectangle around the switch

Label offset

The distance of the text label from the switch edge.

Show label

If true the labels are displayed.

Radius

The size of the switch.

Winch color (bar in the middle of the switch)

The color of the winch.

Winch offset

The size of the winch.

Angle start

This is the start of the switch positions. 0 is 6 o'clock, 90 is 9 o'clock, 180 is 12 o'clock, etc.

Angle end

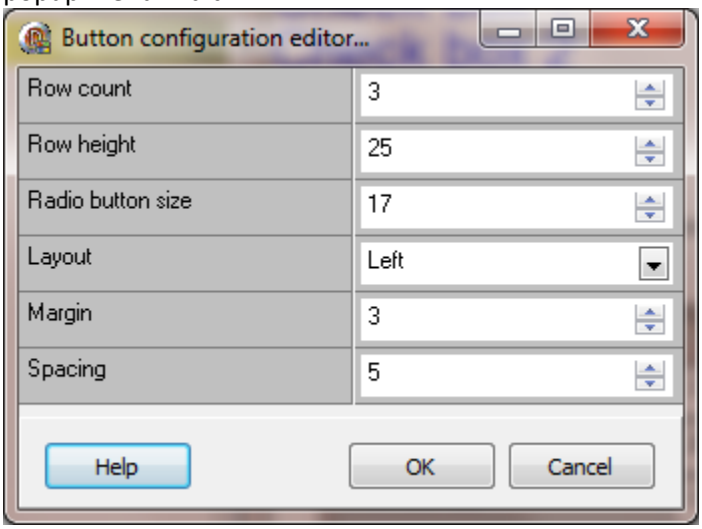
This is the end of the switch positions. 0 is 6 o'clock, 90 is 9 o'clock, 180 is 12 o'clock, etc.

Text Color

The color of the text labels.

Radio button list

The “Radio button list” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



When the left mouse button is pressed and released in a radio button, the values of the radio buttons are examined and the 16 bit value is written to the point.
The current internal value is used. If the radio button count is less than 16, the unused bits are not changed. If this is a PLC point a write command is issued.

Colors

Foreground color = caption (text) color

Row count

The number of radio buttons in the list. (2-16).

Row height

Each radio button is one row in a column. This is the height of each radio button row.

Radio button size

The size of the radio button.

Layout

The location of the radio button in the row.

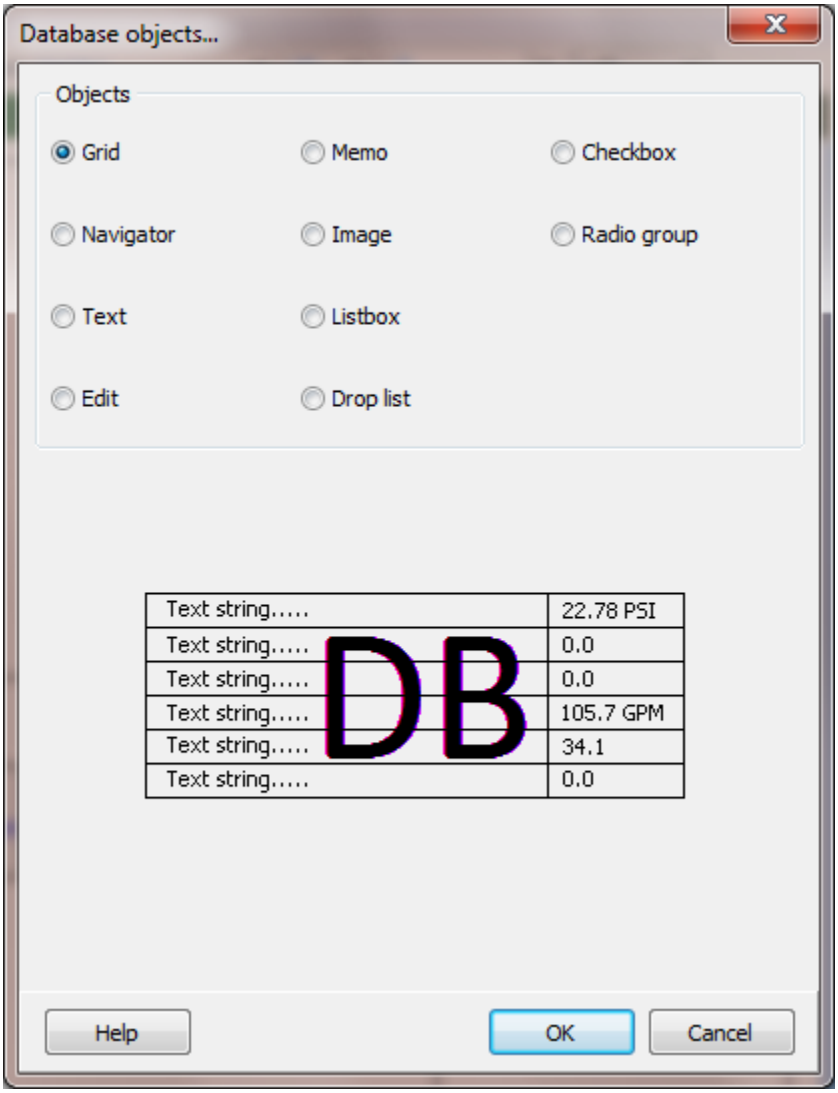
Margin

The distance from the radio button to the border of the list based on the layout. If the value is -1 the radio button/caption is centered. Negative numbers move the radio button up/left, positive numbers move the radio button down/right.

Spacing

This is the space between the radio button and the text.

DATABASE OBJECTS



Database graphic elements

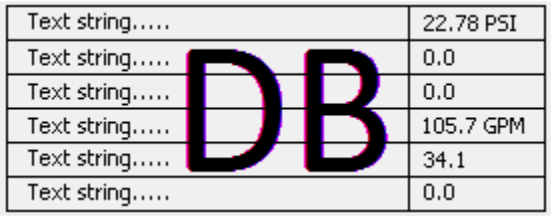
[DBGrid](#)
[DBNavigator](#)
[DBText](#)
[DBEdit](#)

[DBMemo](#)
[DBImage](#)
[DBListbox](#)

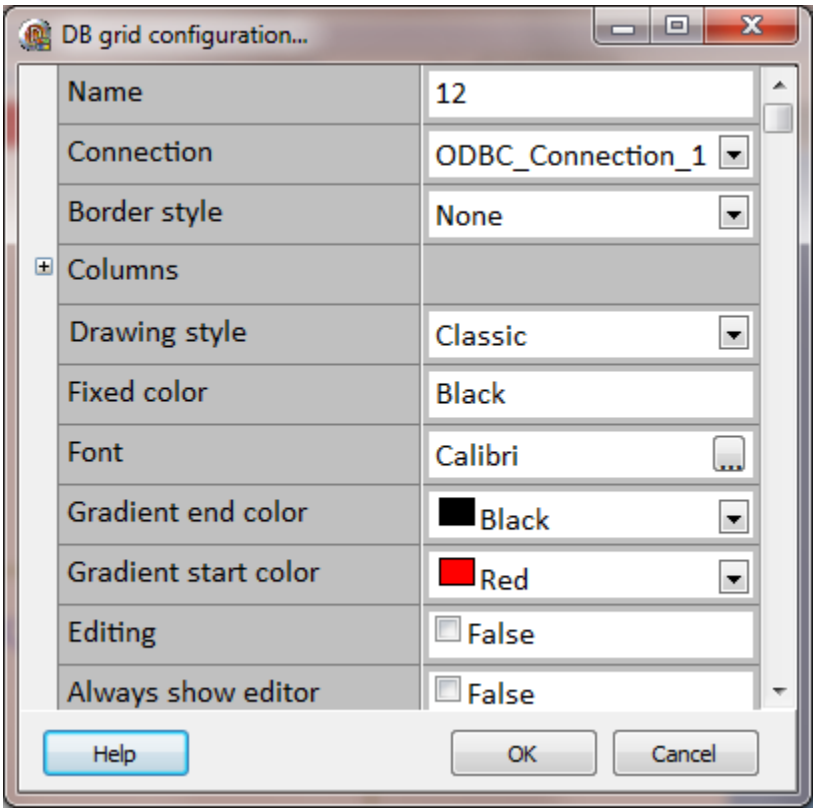
[DBDroplist](#)
[DBCheckbox](#)
[DB Radio group](#)

[Popup keyboard](#)

DATABASE GRID



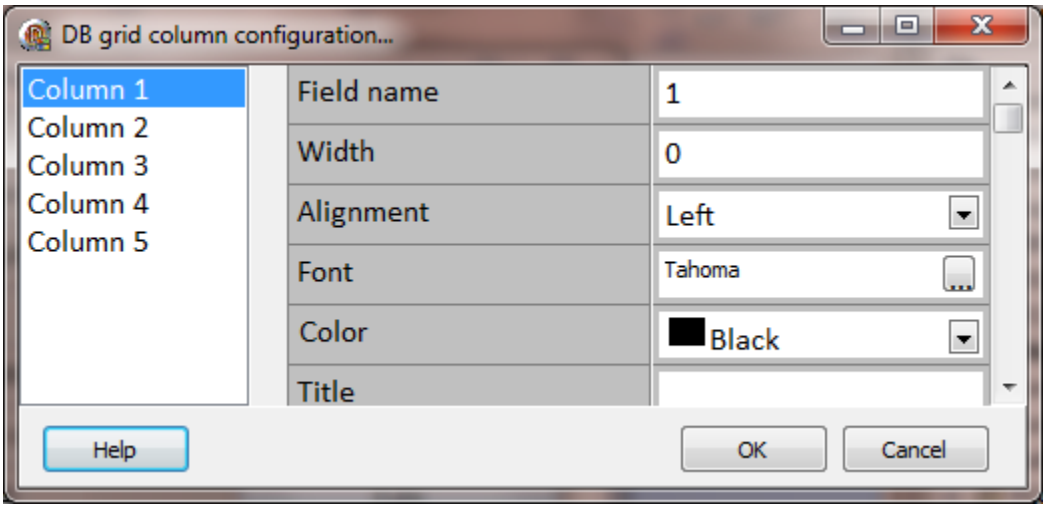
The “Database grid” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Name (optional)	The name of this database grid graphic element.
Connection	This is the connection to a database and a query.
Border style	The border styles are none and single.
Automatic columns	When this property is enabled the column order, selection, etc. are determined by the grid and query results.

Column count — If “automatic columns” is not enabled, this defines the number of columns displayed in the grid and the columns must be defined (below).

Columns



The column count (above) defines the number of columns displayed in the gird.

Field name — The name of the field, in the table, that this column will display in the grid.

Width — The width of the column in the grid.

Alignment — The text alignment for the column.

Font — The font settings used to render the text for this column.

Color — The background color of the column.

Title — This is the text that will appear at the top of the column.

Title font — This is the font for the title.

Title color — This is the background color for the title.

Pick list — This is a list of text values that will be displayed in a “drop list” when the user clicks on a cell in the column. If the grid is read only or editing is not enabled, the drop list will not appear.

Drop count — This is the “count” items that will appear when the pick list is populated. If the list has more items, a scroll bar will be in the drop list. If the list has fewer items, all the items will be visible.

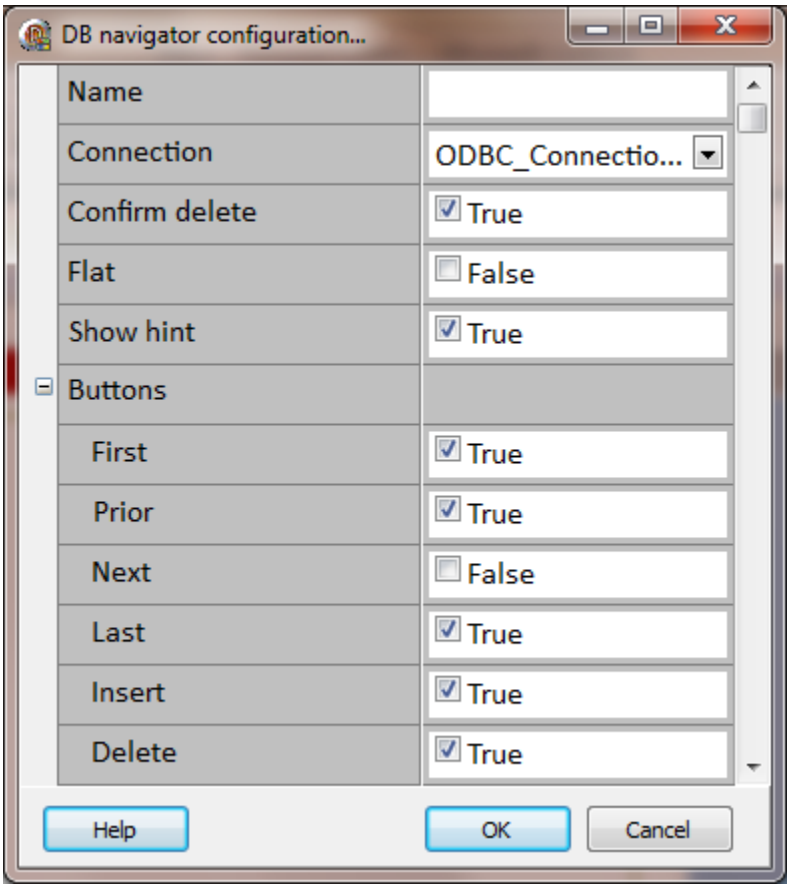
Read only	This is configures this column as read only.
Visible	If this property is enabled the column will be visible, otherwise the column will be hidden.
Drawing style	**Classic:** The grid control uses the standard, un-themed style. **Themed:** The grid control uses the current operating system theme. **Gradient:** The grid control uses gradients for styling.
Fixed color	This is the color of the fixed column/rows.
Font	The font settings for the cell text.
Gradient end color, start color	When the drawing style is “Gradient” this defines the start and end color.
Editing	If this property is enabled the user will be able to alter the table values.
Always show editor	The grid is always in edit mode. This means the user does not have to press ENTER or F2 before editing the contents of a cell . If “Read only” (below) is enabled, the grid will not show the “editor”.
Titles	Titles appear at the top of the columns in the grid.
Indicator	A small pointer appears in the first column to indicate which row is current.
Column resize	Columns that are bound to fields can be resized or moved.
Column lines	Lines appear between the columns of the grid.
Row lines	Lines appear between the rows of the grid.
Tabs	The user can navigate through the grid using the TAB and SHIFT+TAB keys.
Always show selection	The selected cell displays the focus rectangle even when the grid does not have focus.
Confirm delete	A message box appears, asking for confirmation, when the user presses CTRL+DELETE to delete a row in the grid.

Cancel on exit	When the user exits the grid from an inserted record to which the user made no modifications, the inserted record is not posted to the dataset. This prevents the accidental posting of empty records.
Read only	The user cannot edit the grid contents.
Title font	The font settings for the title text.
Default column width	The value applied to all columns if “automatic columns” is enabled. **Note:** This property is only applied when the window is first opened.
Default row height	The value applied to all rows if “automatic columns” is enabled. **Note:** This property is only applied when the window is first opened.
Default alignment	The value applied to all cells if “automatic columns” is enabled. **Note:** This property is only applied when the window is first opened.
Popup keyboard	See here.

Related:
Database objects

DATABASE NAVIGATOR

The “Database navigator” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Name (optional)

The name of this database navigator graphic element.

Connection

This is the connection to a database and a query.

Flat

The buttons are rendered with a “flat” appearance.

Show hint

If this property is enabled and the mouse pointer moves over a button, a “hint” will appear.

Button

First	Move the cursor to the first record.
Prior	Move the cursor to the prior record.
Next	Move the cursor to the next record.
Last	Move the cursor to the last record.
Insert	Insert a new record in the table. **Note:** Not all databases support insert, some will append when insert is called.
Delete	Delete the selected record. **Note:** Not all databases support delete.
Edit	Edit the selected record.
Post	Post any changes to the selected record.
Cancel	Cancel all edits for the current record.
Refresh	Re-fetch the table data from the database.

Button hints

This text will appear, as a hint, when the mouse pointer is moved over a button and the “Show hint” property is enabled. To disable the hint for a button, leave the field blank. To disable hints for all buttons, disable “Show hint”.

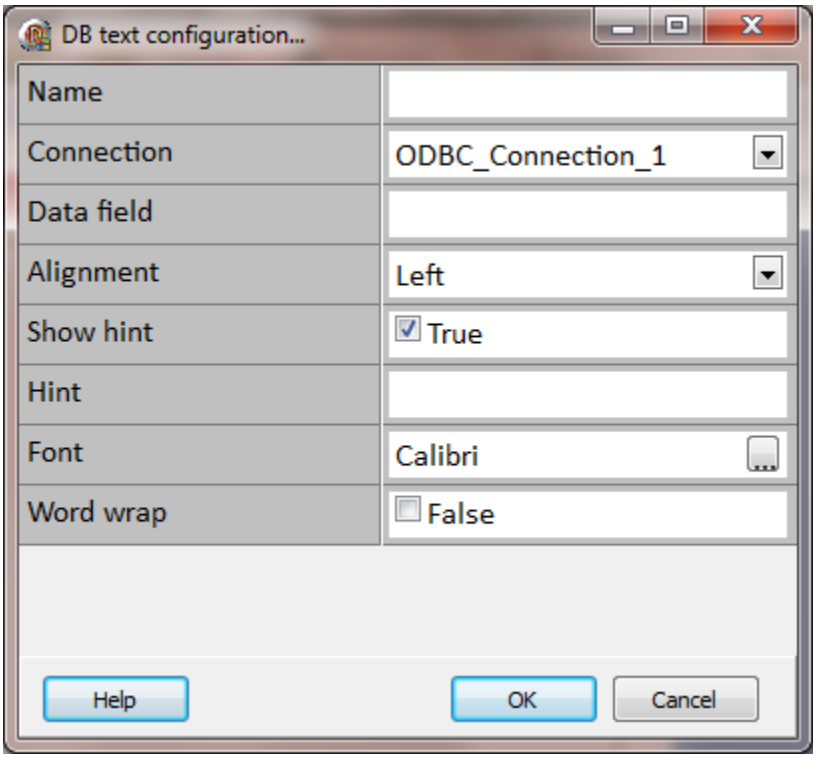
Related:
Database objects

DATABASE TEXT

DBText

This graphic element renders static text from the current record. For editable text, use the "DBEdit" graphic element.

The "Database text" editor is accessed via the main menu "Objects/Edit" or via the right click popup menu "Edit".



Name (optional)

The name of this database text graphic element.

Connection

This is the connection to a database and a query.

Data field

This is the name of the field in the table to be rendered in the graphic element. The text from the currently selected record will be displayed.

Alignment

The text alignment within the bounds of the graphic element.

Show hint

If this property is enabled and the mouse pointer moves over a button, a “hint” will appear.

Hint

This text will appear, as a hint, when the mouse pointer is moved over the graphic element and the “Show hint” property is enabled.

Font

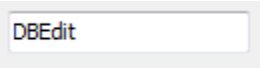
The font settings for the text.

Word wrap

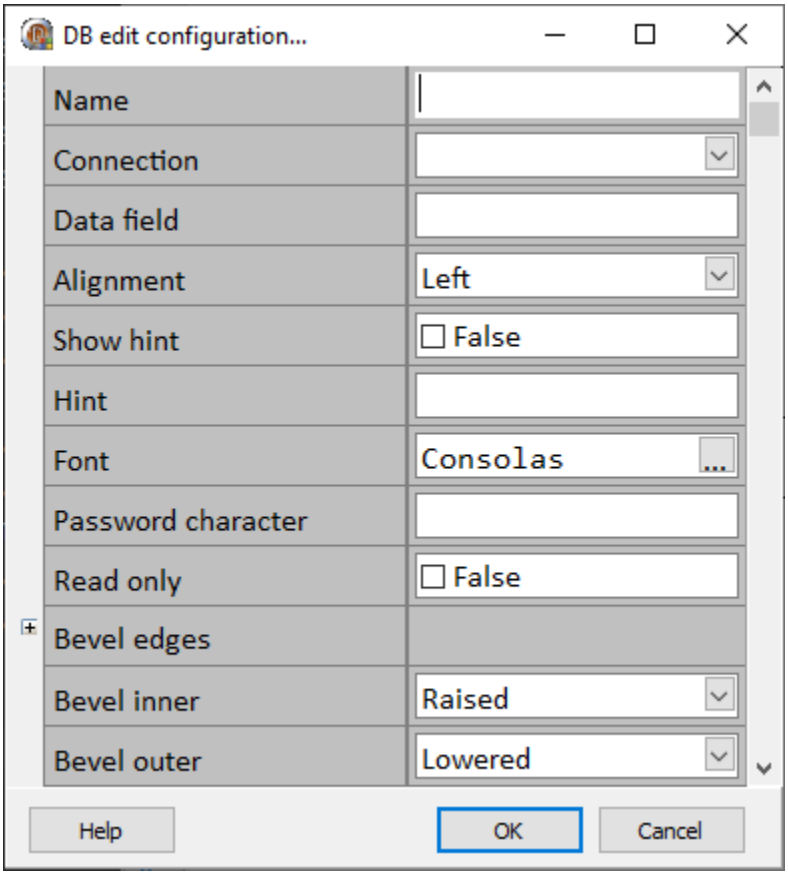
If this property is enabled and the text does not fit on a single line the text will be “wrapped” within the bounds of the graphic element.

Related:
Database objects

DATABASE EDIT



The “Database edit” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Name (optional)

The name of this database edit graphic element.

Connection

This is the connection to a database and a query.

Data field

This is the name of the field in the table to be rendered in the graphic element. The text from the currently selected record will be displayed.

Alignment

The text alignment within the bounds of the graphic element.

Show hint

If this property is enabled and the mouse pointer moves over a button, a “hint” will appear.

Hint

This text will appear, as a hint, when the mouse pointer is moved over the graphic element and the “Show hint” property is enabled.

Font

The font settings for the text.

Password character

If a single character is in this property, each character in the field will be replaced with the character. If this property is blank, the field will be displayed un altered.

Read only

The user cannot edit the contents.

Bevel edges (Left, Top, Right, and Bottom)

Depending on the other selections true shows the edge.

Bevel inner, outer

The edit control will render the inner or outer edge.

Bevel kind

Select the bevel style.

Bevel width

This is the line width of the bevel.

Border style

The border styles are none and single.

Maximum length

This is the maximum number of characters that can appear in the control.

Post on Enter key

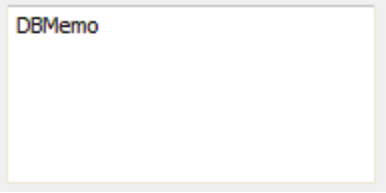
Selecting the “Enter/Return” key, the displayed value will be posted to the database.

Popup keyboard

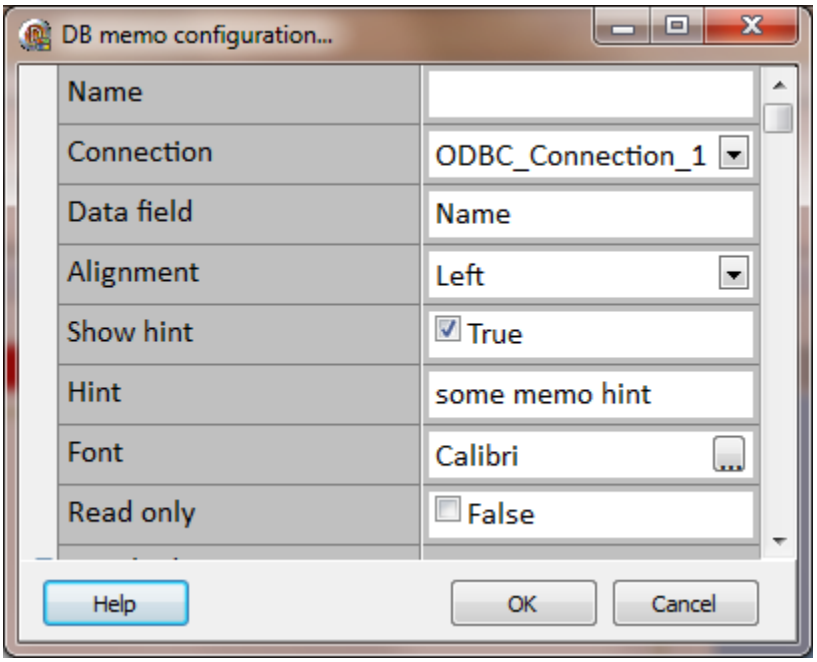
See here.

Related:
Database objects

DATABASE MEMO



The “Database memo” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Name (optional)

The name of this database memo graphic element.

Connection

This is the connection to a database and a query.

Data field

This is the name of the field in the table to be rendered in the graphic element. The text from the currently selected record will be displayed.

Alignment

The text alignment within the bounds of the graphic element.

Show hint

If this property is enabled and the mouse pointer moves over a button, a “hint” will appear.

Hint

This text will appear, as a hint, when the mouse pointer is moved over the graphic element and the “Show hint” property is enabled.

Font

The font settings for the text.

Read only

The user cannot edit the contents.

Bevel edges (Left, Top, Right, and Bottom)

Depending on other selections, true shows the edge.

Bevel inner, outer

The edit control will render the inner or outer edge.

Bevel kind

Select the bevel style.

Bevel width

This is the line width of the bevel.

Border style

The border styles are none and single.

Maximum length

This is the maximum number of characters that can appear in the control.

Color

This is the background color of the memo.

Hide selection

Determines if the visual indication of the selected text remains when focus shifts to another control.

Want returns

If enabled the user can insert carriage returns in the text.
Note: If using the "Popup keyboard" and this property is enabled and the "Close on enter key" property is enabled in the popup keyboard, the "enter/return" will be inserted into the memo and the keyboard will close.

Want tabs

If enabled the user can insert tabs in the text.

Word wrap

If this property is enabled and the text does not fit on a single line the text will be "wrapped" within the bounds of the graphic element.

Scroll bars

This property determines if the memo will display scroll bars.

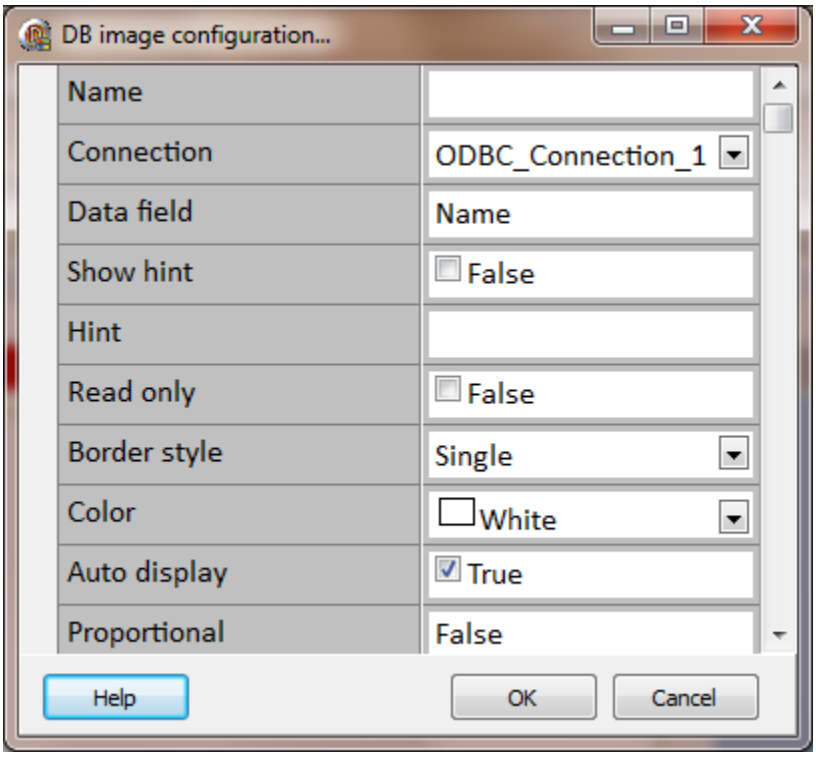
Popup keyboard

See here.

Related:
Database objects

DATABASE IMAGE

The “Database image” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Name (optional)

The name of this database image graphic element.

Connection

This is the connection to a database and a query.

Data field

This is the name of the field in the table to be rendered in the graphic element. The image from the currently selected record will be displayed.

Show hint

If this property is enabled and the mouse pointer moves over a button, a “hint” will appear.

Hint

This text will appear, as a hint, when the mouse pointer is moved over the graphic element and the “Show hint” property is enabled.

Read only

The user cannot edit the contents.

Border style

The border styles are none and single.

Color

This is the background color of the memo.

Auto display

If Auto display is true, the image automatically displays new data when the underlying BLOB field changes (such as when moving to a new record).

If Auto display is false, the image clears whenever the underlying BLOB field changes. To display the data, the user can double-click on the control or select it and press Enter.

Change the value of Auto display to false if the automatic loading of BLOB fields seems to take too long.

Proportional

Indicates if bitmaps and metafiles should be changed, without distortion, so that they fit the bounds of the database image.

Set Proportional to True to ensure that the bitmap or metafile can be fully displayed in the database image without any distortion. The distortion may appear when the Stretch property is set to True.

When Proportional is True, bitmaps that are too large to fit in the database image are scaled down (while maintaining the same aspect ratio) until they fit in the database image. Bitmaps that are too small are displayed normally. Proportional can reduce the magnification of the bitmap, but does not increase it.

When the database image resizes, the bitmap resizes as well.

To resize the bitmap so that it fits exactly in the database image, even if that causes distortion, use the Stretch property instead.

Center

Determines if the image is centered in the image control.

When the image does not fit perfectly within the image control, use Center to position the image. When Center is true, the image is centered in the control. When Center is false, the upper left corner of the image is positioned at the upper left corner of the control.

Quickdraw

Specifies if the image is displayed using a palette.

Set QuickDraw to specify whether a customized palette should be used when displaying field values. If false, a palette is used, to provide the best possible image quality at the expense of additional processing time. If true, no special palette is used, which is faster, but, results in poorer picture quality, especially with 256-color images on a 256-color video driver.

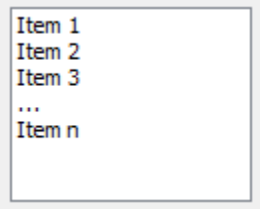
Stretch

Determines if bitmaps and metafiles assume the size and shape of the database image.

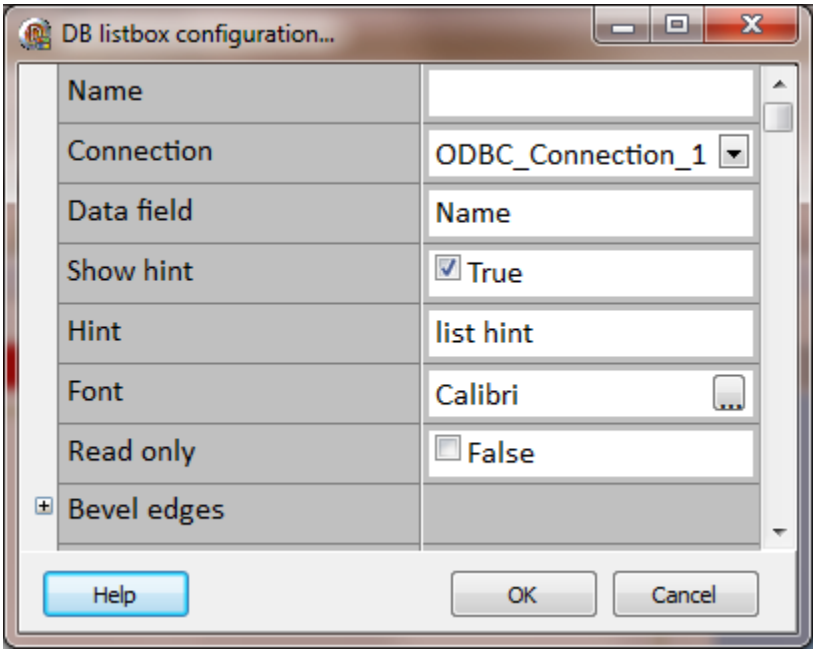
Set Stretch to true to cause the picture to assume the size and shape of the image control. When the image control resizes, the picture resizes also. Stretch resizes the height and width of the image independently. Thus, unlike a simple change in magnification, stretch can distort the image if the image control is not the same shape as the image.

Related:
Database objects

DATABASE LISTBOX



The “Database listbox” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Name (optional)

The name of this database listbox graphic element.

Connection

This is the connection to a database and a query.

Data field

This is the name of the field in the table to be rendered in the graphic element. The text from the currently selected record will be displayed.

Show hint

If this property is enabled and the mouse pointer moves over a button, a “hint” will appear.

Hint

This text will appear, as a hint, when the mouse pointer is moved over the graphic element and the “Show hint” property is enabled.

Font

The font settings for the text.

Read only

The user cannot edit the contents.

Bevel edges (Left, Top, Right, and Bottom)

Depending on other selections, true shows the edge.

Bevel inner, outer

The edit control will render the inner or outer edge.

Bevel kind

Select the bevel style.

Bevel width

This is the line width of the bevel.

Border style

The border styles are none and single.

Color

This is the background color of the list box.

Auto complete

Determines if the user can give focus to items by typing in the list.

Item height

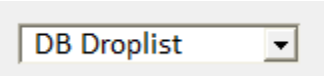
This is the height of each item in the list box.

Items

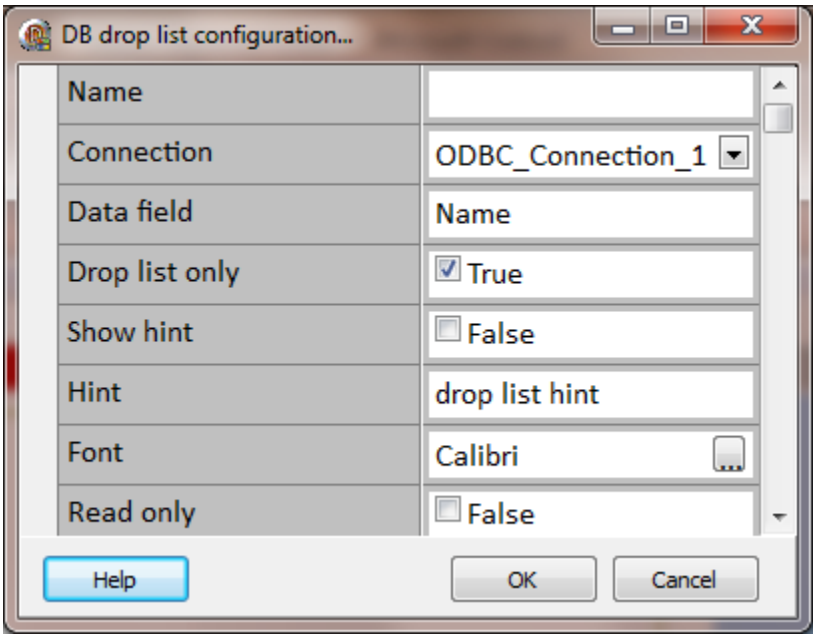
The items in the list box. Enter each item on a single line in the editor.

Related:
Database objects

DATABASE DROPLIST



The “Database droplist” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Name (optional)	The name of this database droplist graphic element.
Connection	This is the connection to a database and a query.
Data field	This is the name of the field in the table to be rendered in the graphic element. The text from the currently selected record will be displayed.
Drop list only	If enabled, the droplist will contain the items configured in the “items” property. The user can select one of the items. If not enabled, the items will be listed, the user can select an item or type a value in the field.
Show hint	If this property is enabled and the mouse pointer moves over a button, a “hint” will appear.
Hint	This text will appear, as a hint, when the mouse pointer is moved over the graphic element and the “Show hint” property is enabled.

Font	The font settings for the text.
Read only	The user cannot edit the contents.
Bevel edges (Left, Top, Right, and Bottom)	Depending on other selections, true shows the edge.
Bevel inner, outer	The edit control will render the inner or outer edge.
Bevel kind	Select the bevel style.
Bevel width	This is the line width of the bevel.
Border style	The border styles are none and single.
Color	This is the background color of the droplist.
Auto complete	Determines if the user can give focus to items by typing in the list.
Item height	This is the height of each item in the droplist.
Items	The items in the droplist.

Several methods to populate the drop list are provided.

A) Enter each item on a single line in the editor.
B) Script global hive
=SG(<hive name>)
All the items of the hive will be used as the drop list items.
Example: =SG(Pump Pressure)
Note: Do not include a hive item name. Including a hive item name will execute option "D", below.
C) Execute an SQL "Select" and the fields of the record result will be used as the drop list items. The drop list connection is the database.
=SQL(<statement> ~ [Water])
The last values(s), in the bracket [], defines field values (comma separated) to exclude from the drop list and must contain at least one value, the record search/key to match. In the example below it is "Water".
Example: =SQL(SELECT * FROM PumpDropList WHERE Pump='Water' ~ [Water,600])
Notes:
1) The SQL portion can be tested at configuration time in the "ODBC connections test" dialog.
2) If the result does not produce any fields, no items will be added.
D) Execute an SQL "Select" from a script global hive and item.
=SG(<hive name>.<item name>)
Example: =SG(Pump Pressure.Water)

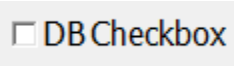
Notes:

1) If a hive item name is **not** included option “B”, above, will be executed.
2) The string in the hive.item must use the option “C” format.
3) See option “C” for more information.

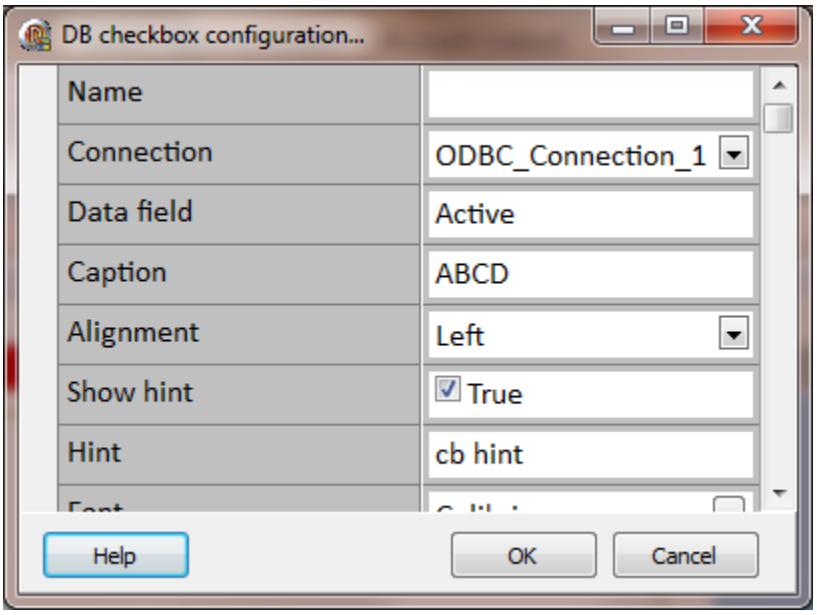
Sorted	If this property is enabled the items in the droplist will be sorted in descending order.
Auto drop down	Specifies if the droplist drops down automatically in response to user keystrokes if “Read only” and “Drop list only” are not enabled. **Note:** The control does not support monitoring mouse clicks when in edit mode (control text can be altered). The keyboard, if it can be displayed, will appear when the mouse is moved over the control. The control **must** be selected for editing. The keyboard appearing does not select the database droplist graphic element.
Popup keyboard	See here.

Related:
Database objects

DATABASE CHECKBOX



The “Database checkbox” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Name (optional)

The name of this database checkbox graphic element.

Connection

This is the connection to a database and a query.

Data field

This is the name of the field in the table to be rendered in the graphic element. The text from the currently selected record will be displayed.

Caption (No caption (blank))

An optional caption to be placed next to the checkbox.

Show hint

If this property is enabled and the mouse pointer moves over a button, a “hint” will appear.

Hint

This text will appear, as a hint, when the mouse pointer is moved over the graphic element and the “Show hint” property is enabled.

Font

The font settings for the text.

Checked text

Specifies the field value that corresponds to the checked state of the check box.

Use “checked text” to specify the field value represented by the check box when it is checked. If the value of the “checked text” property is equal to the data in the field of the current record of the dataset, the database check box appears checked. When the user checks the database check box, the field value is set to “checked text”.

“Checked text” can represent more than one value in a semicolon-delimited list of items. If any of the items matches the contents of the field of the current record in the dataset, the check box appears checked. For example, set the value of “checked text” string like this:

True; Yes; On;

If the contents of the associated field is the string true, Yes, or On, the check box is checked. The value of the field is compared to “checked text” in a case-insensitive comparison. If the user selects a check box where “checked text” represents more than one value, the first item in the list is assigned to the field.

See **notes**

Unchecked text

Specifies the field value that corresponds to the unchecked state of the check box.

Use “unchecked text” to specify the field value represented by the check box when it is unchecked. If the value of the “unchecked text” property is equal to the data in the field of the current record of the dataset, the database check box appears unchecked. When the user unchecks the database check box, the field value is set to “unchecked text”.

“unchecked text” can represent more than one value in a semicolon-delimited list of items. If any of the items matches the contents of the field of the current record in the dataset, the check box appears unchecked. For example, set the value of “unchecked text” string like this:

False; No; Off;

Note: If the contents of the associated field is the string false, No, or Off, the check box appears unchecked. The value of the field is compared to “unchecked text” in a case-insensitive comparison. If the user unchecks a check box where “unchecked text” represents more than one value, the first item in the list is assigned to the field.

If the contents of the field of the current record matches a string specified as the value of the “checked text” property, the check box appears unchecked. If the contents of the field matches no string in either “checked text” or “unchecked text” , the check box appears gray.

Note: If the “DataField” of the database check box is a logical field, the check box is always checked if the contents of the field is true, and it is always unchecked if the contents of the field is false. The values of the “checked text” and “unchecked text” properties have no effect on logical fields.

Read only

The user cannot edit the contents.

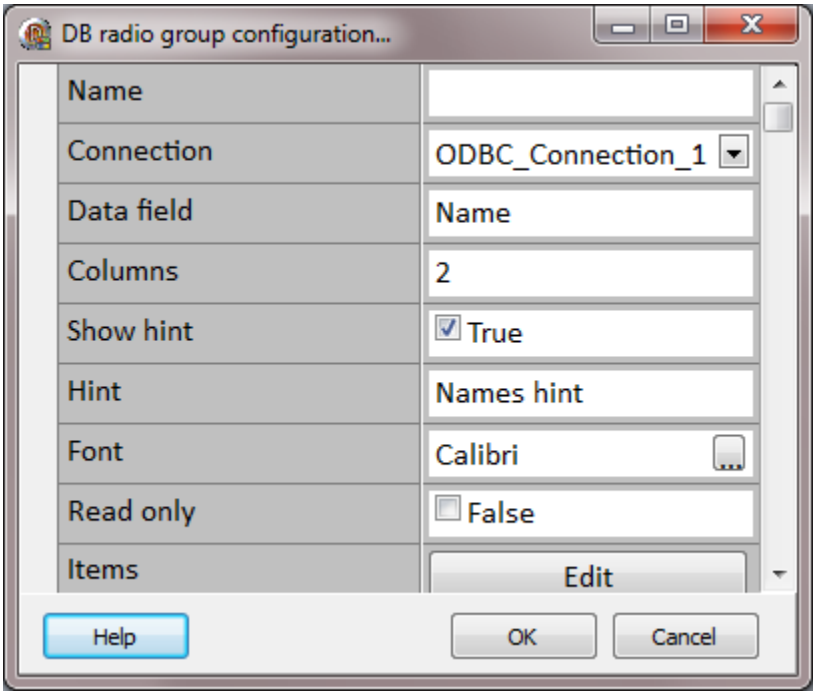
Word wrap

If this property is enabled and the text does not fit on a single line the text will be “wrapped” within the bounds of the graphic element.

Related:
Database objects

DATABASE RADIO GROUP

The “Database radio group” editor is accessed via the main menu “Objects/Edit” or via the right click popup menu “Edit”.



Name (optional)

The name of this database checkbox graphic element.

Connection

This is the connection to a database and a query.

Data field

This is the name of the field in the table to be rendered in the graphic element. The text from the currently selected record will be displayed.

Caption (No caption (blank))

An optional caption to be placed above the radio buttons.

Columns

This is the “count” columns used to display the radio buttons.

Show hint

If this property is enabled and the mouse pointer moves over a button, a “hint” will appear.

Hint

This text will appear, as a hint, when the mouse pointer is moved over the graphic element and the “Show hint” property is enabled.

Font

The font settings for the text.

Read only

The user cannot edit the contents.

Items

This lists the radio buttons in the radio group.

Values

This determines the values of the radio buttons.

When the user selects a radio button, the "value" of that button is written to the linked field in the database. By default, the value of a button is simply the caption that appears next to it on the screen, as determined by the “Items” property (above).

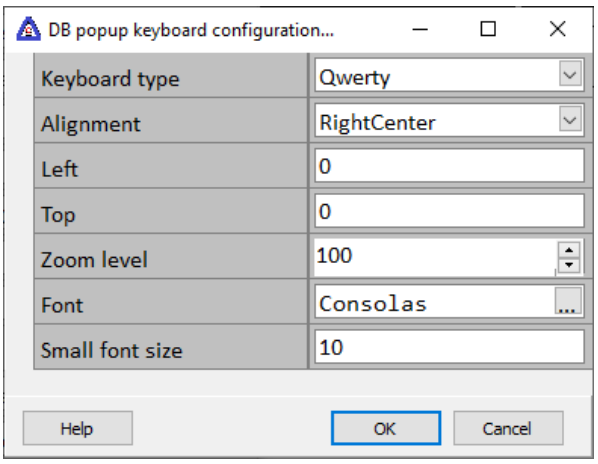
In some cases, a need for the values of the radio buttons to differ from their captions. For example, the radio buttons are used to represent a database field whose content can be "Y" or "N", and need the buttons' captions to be "Yes" and "No". In this case, enter "Yes" and "No" in the “Items” list, and enter "Y" and "N" in the “Values” list.

Color

This is the background color of the radio group.

Related:
Database objects

DATABASE POPUP KEYBOARD



The "Database Popup Keyboard" can be displayed when the mouse is clicked in a database graphic element that supports runtime text editing.

Keyboard type

The type of keyboard to display.
None, Qwerty, Azerty, Dvorak, Numeric, Cellphone, Qwertz, and OS keyboard

OS keyboard is the keyboard provided by the OS. If the OS keyboard is selected, all of the following properties do not apply.

Alignment

This defines where the keyboard will appear on the monitor.

Left, Top, Right, Bottom, and Center, define the keyboard position in relationship to the graphic element.

LeftTop, LeftCenter, LeftBottom, CenterTop, CenterCenter, CenterBottom, RightTop, RightCenter, and RightBottom define the keyboard position in relationship to the monitor.

Custom is used to specify the position of the keyboard on the monitor. Left and top, below, the location of the left/top corner of the keyboard.

Left/Top

These properties are utilized two ways.
Refer to "Alignment", above for the "Custom" alignment.

For all other alignment configurations, these two properties define an offset to the calculated position.

Zero equals no offset.

Left, a positive number moves the keyboard to the right and a negative number moves the keyboard to the left.

Top, a positive number moves the keyboard down and a negative number moves the keyboard up.

Zoom

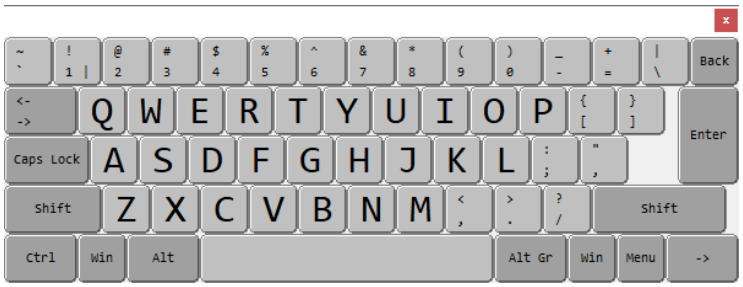
The keyboard can be scaled up/down.
A value of 100 equals no zoom (scaling).
Values above 100 increase the size of the keyboard.
Values below 100 decrease the size of the keyboard.

Font

The font to be used for the keyboard keys.

Small font size

The font size to be used for some of the keyboard keys. Below the regular font size is 24 (assigned in the font property) and the small font size is 8.



Close on enter key

When the “Enter/Return” key is selected the keyboard will close. If the database control is in edit mode, the database field will be updated with the control value before the keyboard is closed.

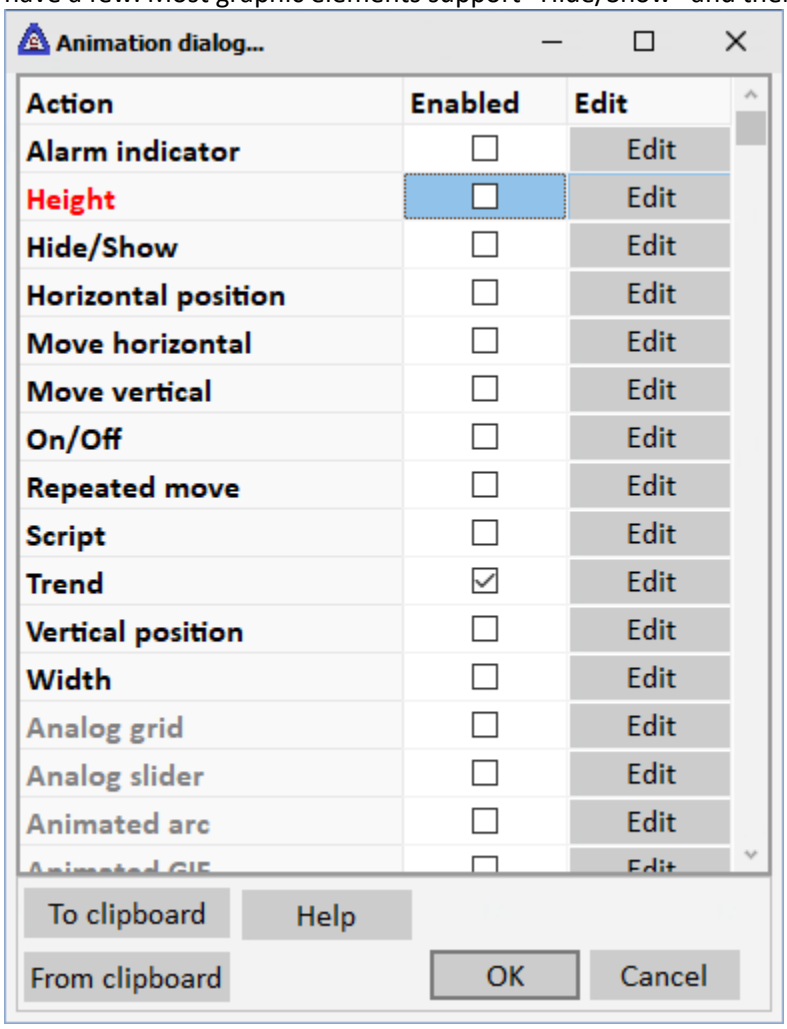
Inactivity time

If the keyboard is inactive for X time (in seconds), the keyboard will close. A 0 (zero) value disables the timer. Switching from a control to another control will reset the timer with the selected controls “Inactivity time” value.
If the control is in edit mode, the database will **not** be updated before the keyboard is closed.

Related:
Database objects

ANIMATIONS

There are many types of animations. Many objects have several animations. Some objects only have a few. Most graphic elements support “Hide/Show” and then one or two more.



Not all animations types apply to all element types. For example, the "Time/Date" cannot be applied to a circle graphic element.

Animations by name list

If the animation name is grey, the animation for the object is not applicable.

If the animation is red, the animation has a data source configured, but is **not** enabled.

If the "Enable" checkbox is not checked, the animation will not be active at runtime.

Most animation windows will have a source button. Use this button to select the tagname.item number monitored for the animation. When configuring the mouse actions, the mouse commands window is displayed.

To/From Clipboard buttons

If selected the elements animation configuration will be copied to the clipboard. This allows the duplication of the animation configuration from one type of element to another.

For example, a circle has a hide/show animation. A rectangle needs the same animation. Double click on the circle, set the animation if needed, and select the "To Clipboard" button. Select the OK or cancel as is appropriate. Double click the rectangle element. When the animation window appears select the "From Clipboard" button. The animation configuration from the clipboard will be pasted into the rectangle element.

Not all animations types apply to all element types. e.g. the "Time/Date" cannot be applied to a circle.

Most animation windows will have a digital panel and an analog panel. The correct data fields are saved based on the item number data type of the tagname selected.

Items B0 - B15 are not applicable to all point types. Based on the device communication protocol some or all of the items might be defined. Please refer to the 'port' settings for the port used in the source address of the point.

Digital Panel

When the item type is digital and the value is false, the object is rendered at runtime as it was designed at design time. If the value is true the settings in the animation dialog are used to render the object.

Analog Panel

When the item type is analog and the runtime value of the tagname.item monitored is less than the bottom value (last row) the design time configuration of the object is displayed. At runtime the animation engine starts at the first row, the top row and checks the comparisons against the item value and if the condition is true the settings are applied and subsequent rows are not processed.

To place a bitmap on the window, if it is on the clipboard, select paste and the object will be created on the window or drag and drop a file.

Multiple animations can be applied to an object using the same or different tagname item pairs.

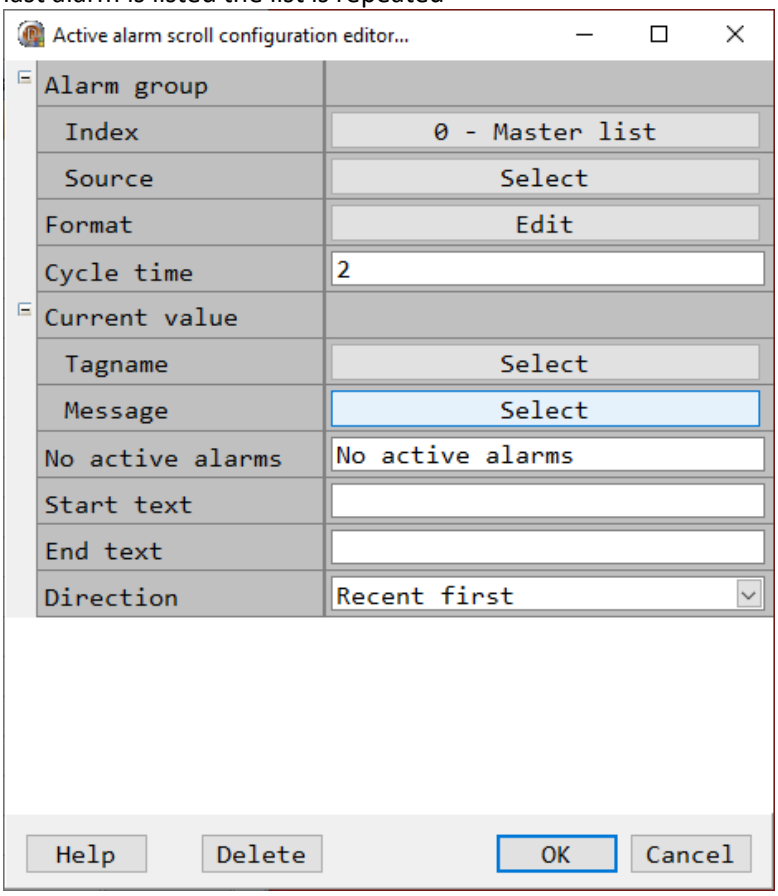
The most used items are:

5000	Process variable analog
5007	Process variable digital
5008	Percent of Full Scale

Animations by name

Active alarm scroll

This animation is used to “scroll” a list of alarms. Each active alarm is listed, one a time, and when the last alarm is listed the list is repeated

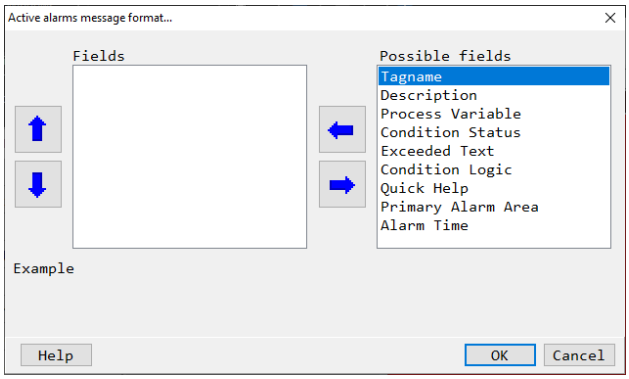


Alarm group

Index Specifies the alarm group to use. 0 (zero) is the master list. When the next property “Source” is specified, this property is ignored.

Source Specifies the alarm group to use from a script global.

Format Specifies the alarm properties to use to format the displayed message.



Cycle time The number of seconds each message will appear before the next message is displayed.

Current value (optional)

The script animation is the last animation executed. These properties can be used to take action in a script based on the tagname and/or message.

Tagname If specified, the tagname of the alarm being displayed is stored in the specified script global.

Message If specified, the formatted message of the alarm being displayed, is stored in the specified script global.

No active alarm When the selected list does not contain any active alarms the configured text will be displayed. (optional)

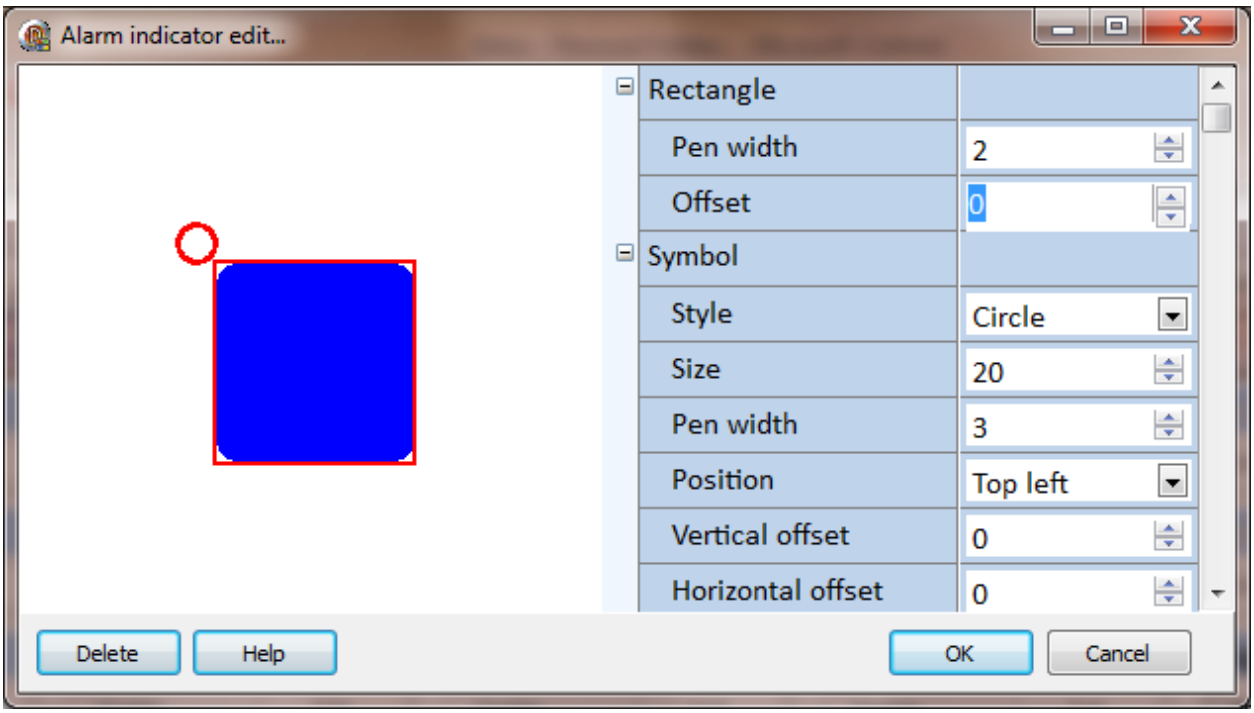
Start text A text message to display before the first alarm in the list is displayed. (optional)

End text A text message to display after the last alarm in the list is displayed. (optional)

Direction The selected alarm group list can be “scrolled” forward or backwards.

Recent first Newest alarm to oldest alarm
Oldest first Oldest alarm to newest alarm

ALARM INDICATOR



The alarm indicator animation is used to display a symbol, near or on a graphic element, containing an animation connected to a point in alarm. The animation supports a rectangle symbol and/or an "icon" symbol.

Note: When using the editor and a change is made to a property, click on another property to end the edit mode for the changed property and to view the change in the example animation.

Rectangle

Pen width	This is the pen width of the rectangle. Set the property value to 0 (zero) and the rectangle symbol will not be rendered.
Offset	This values defines an offset from the graphic element to render the rectangle.

Symbol

Style	The symbol can be a circle, square or diamond shape.
Size	This property defines the size of the symbol. Set this property value or the pen width property value to 0 (zero) and the symbol will not be rendered.
Pen width	This is the pen width of the symbol. Set this property value or the size property value to 0 (zero) and the symbol will not be rendered.
Position	This property defines the graphic element corner the symbol will be rendered.

Vertical/Horizontal offset

These properties define an offset applied to the symbol before the symbol is rendered.

Fill

This property, if enabled, will fill the symbol with the applicable alarm color.

Active Active/acknowledged

Each alarm object has three possible states: 0 = no alarm, 1 = active alarm and acknowledged and 2 = active alarm.

Color

This property defines the symbol color when the alarm object is in the defined state.

Flash

This property defines if the symbol will flash (visible/invisible).

Points

Some graphic elements can be configured for multiple animations and each animation might be connected to a different point. For example, a graphic element might be configured for hide/show, pen and brush animations.

Note: When a graphic element has multiple animations with points configured with alarms, the highest alarm state is used to render the symbol.

All

This property defines all connected points, for the graphic element animation, will be used to determine the correct symbol state.

Select

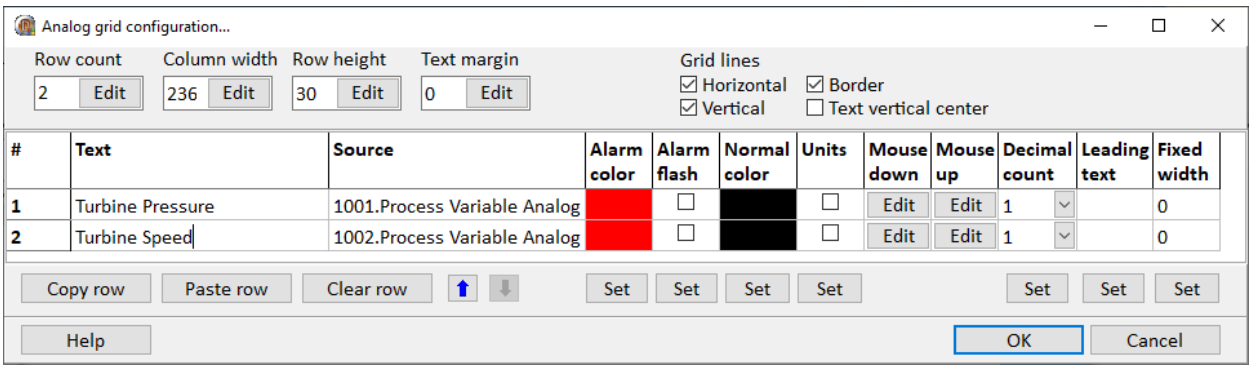
If the “All” property is not enabled, this edit item will be enabled and the points to monitor for symbol state can be selected. If the “All” property is enabled, this property does not apply.

Alarm state

This editor is used to rendered the example with one of the three alarm states.

Back to list

ANALOG GRID



The grid contains rows of text and an indicator rectangle. The "Set" button below the “On” and “Off” color columns will display a color selection dialog. If "OK" is selected the color will be set for all the rows. The "Set" button below the remaining columns will set all the fields in the column to the value of the field in the first row. The “SetCell” script command can also be used to set cell properties.

Notes:

1. The foreground color is the text color.
2. Transparent is not applicable to this object.
3. When setting the user level for mouse actions the level selected applies to all rows.

Row count

This is the number of rows in the grid. (1 - 128)

Column width

This is the width of the indicator column. (4 - 512) The text column is the remainder of the width of the gird.

Row height

This is the height of each row. (4 - 512)

Text margin

The number of pixels to offset the text from the left or right column boundary.

Grid lines

Horizontal, Vertical, Border. If enabled the grid/border lines will be displayed.

Text vertical center

The text will be centered in the cell. Otherwise, the text will be aligned at the top of the cell.

Fixed (website only)

The grid will not scroll if the browser window is scrolled.

For each row

Text The text to display in the first column

Source The source point must be a host digital or host digital pointer.

Alarm color This is the color of the indicator column when the source is in alarm. (Points must have alarms configured)

Alarm flash If the source is in alarm and this is enabled the value will flash between the alarm color and normal color. **Note:** Flash is not enabled for web pages.

Normal color This is the color of the indicator column when the source is not in alarm.

Units If the enabled, points engineering units will be appended to the value. (<value> space <engineering units>)

Mouse down This provides for mouse commands when the mouse is pressed in the indicator for the row. **Note:** Mouse down is not enabled for web pages.

Mouse up This provides for mouse commands when the mouse is pressed and released in the indicator for the row.

Decimal count

To display large numbers, as whole numbers, set the decimal count to -1. The floating point value will be truncated and displayed.

Leading text If this field is not blank the text entered will be prepended to the value.

Fixed width

If the "leading text" field is not blank, the leading text will be prepending to the value. The displayed text might be longer than the fixed width if the leading text is greater than one character.

Examples:
Value 123

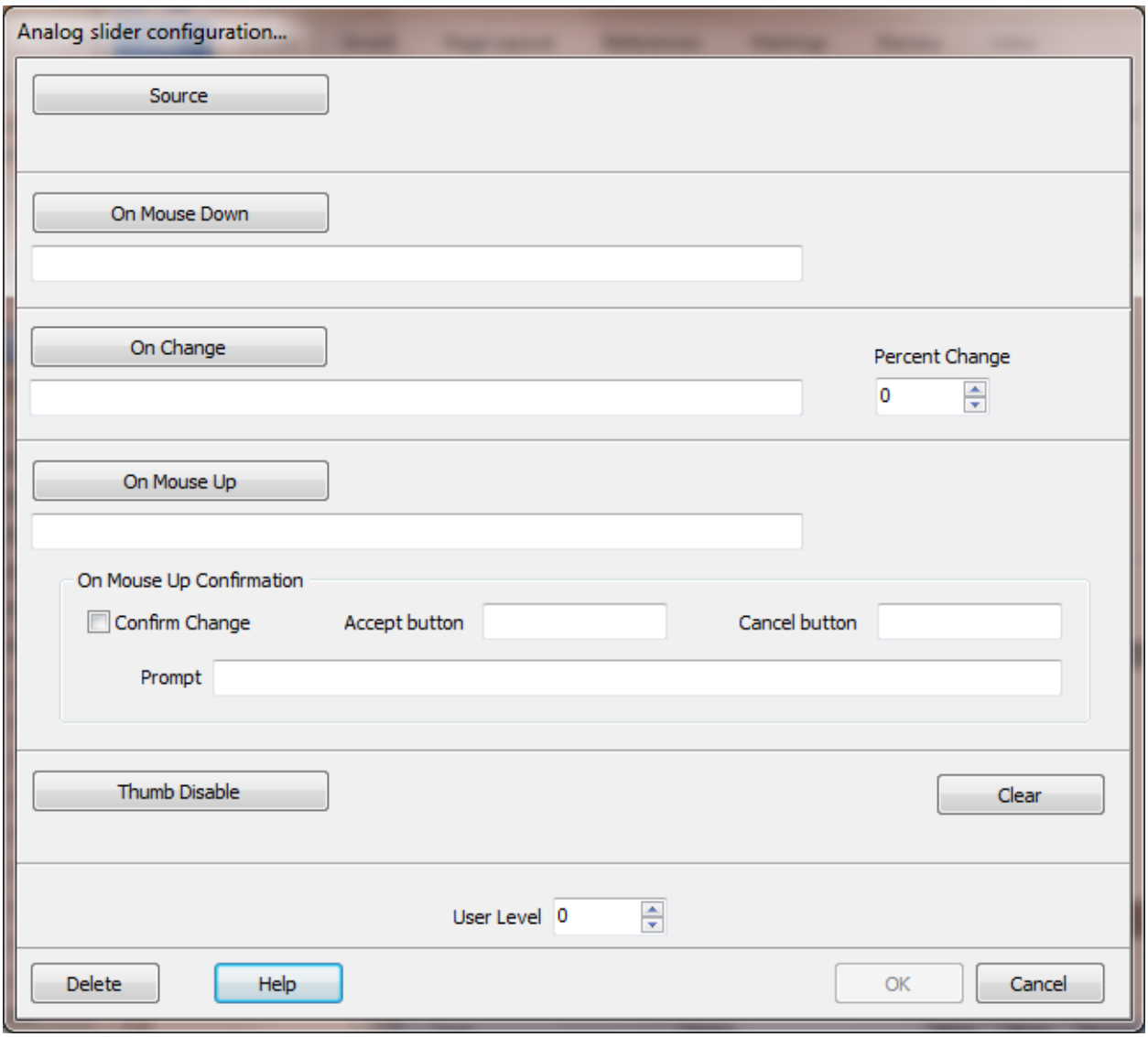
Leading text	Fixed width	Displayed text
0	6	000123
0	2	123
AB	6	ABAB123
A	6	AAA123
EX:	0	EX:123

The order of value to text display processing:

1) Convert the value to a string using the “decimal count” property.
2) If the “leading text” is not blank
    a. If the “Fixed width” property is zero (0), prepend the leading text and proceed to step 3.
    b. If the text length is less than the “Fixed width” property the leading text is repeatedly prepended to the text until the length of the text is greater than or equal to the “Fixed width” property.
3) If the “Units” property is enabled, append the point engineering units.

Back to list

ANALOG SLIDER



Source

The source point must be an analog or host analog pointer. The value of the slider position is stored in the source.

All the mouse and change actions below apply the "User Level" property. The logged on user must have an access level at least as high as the value in the field.

On Mouse Down

If configured this script is called anytime the left mouse button is pressed in the control.

On Change

If configured this script is called anytime the position of the control changes. If the "Percent Change" field is not zero (0) then the script is only called once the amount of change is greater than X percent of full range.

On Mouse Up

If configured this script is called anytime the left mouse button is released after having been pressed in the control. If the mouse pointer is outside the control rectangle the slider position is restored to the value when the mouse button was pressed. If the "Confirm Change" checkbox is checked a confirmation window will appear. If the user selects the "accept" button the script is called. If the user selects the "cancel" button the script is not called and the slider position is restored to the value when the mouse button was pressed.

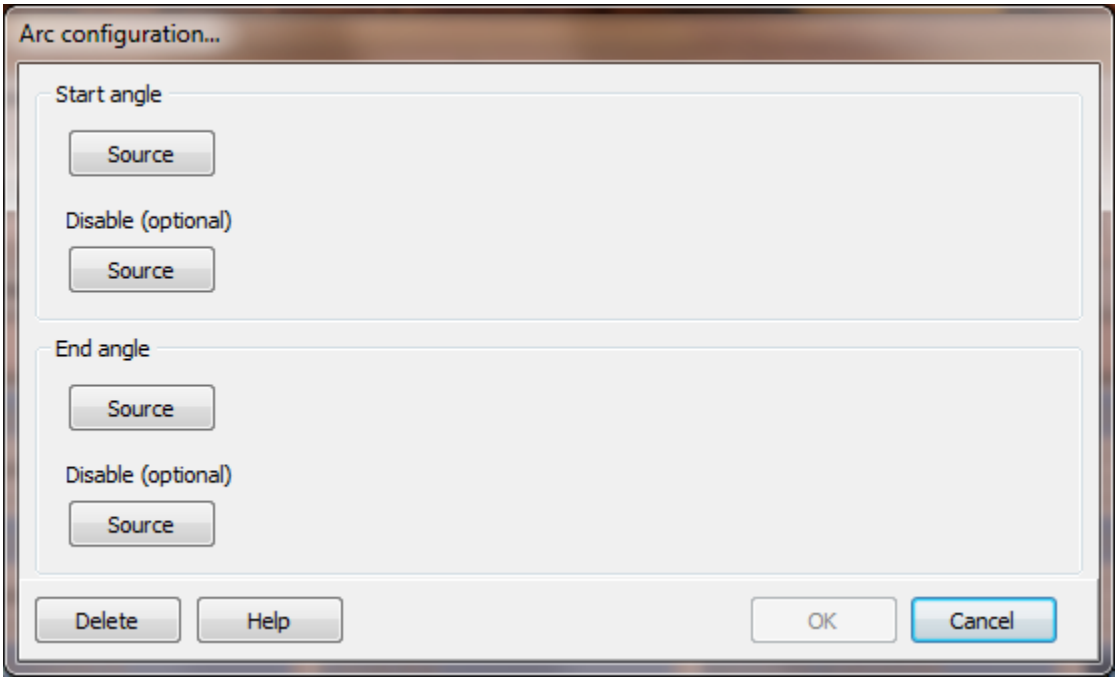
Note: The automatic write to the source tag is disabled when an On Mouse Up script is selected. If assistance is needed, contact support.

Thumb Disable

A digital point to enable/disable the thumb of the slider. The clear button removes the source.

Back to list

ANIMATED ARC



Start/End angle Source

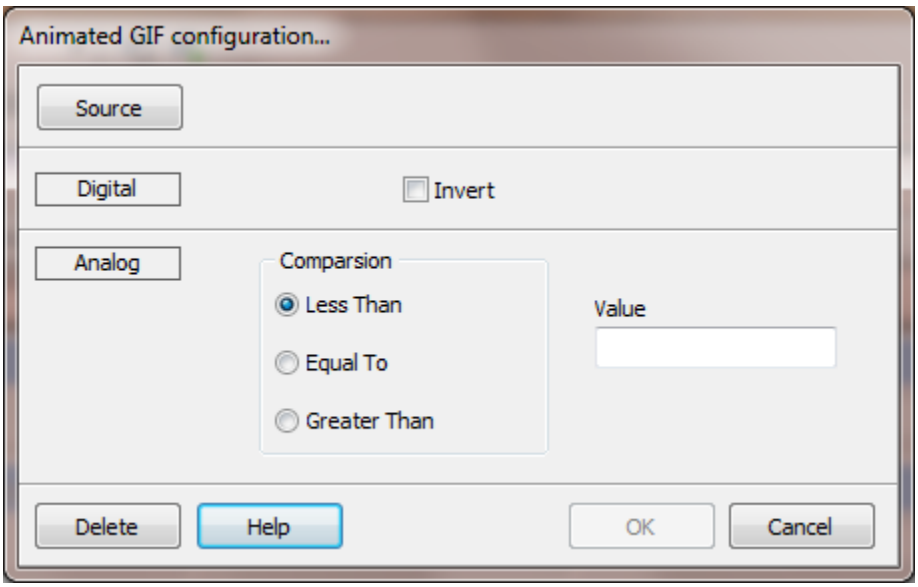
The analog tagname.item used to control the arc starting or ending angle. The value must be 0-359. Use the point scaling feature if needed. The value used for the start/end point is <input value modulus 360>.

Start/End angle Disable (optional)

The digital tagname.item used to disable the changing of the start/end angle. If the input is true the angle is not changed.

Back to list

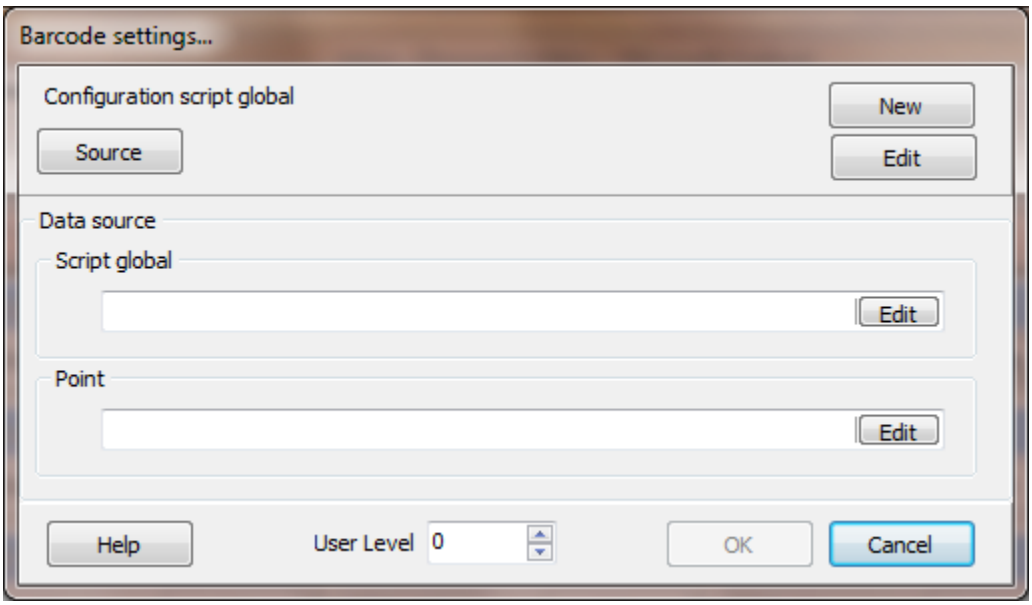
ANIMATED GIF



Source	The source point used to "run" the animation.

Back to list

Barcode



For configuration of the barcode settings see here. A "Configuration script global" must be defined for the barcode to be displayed. If more than one barcode graphic element uses the same configuration values, one script global can be used.

Data source

The source can be a script global or analog point. If a script global is specified the point reference is ignored.

New/Edit

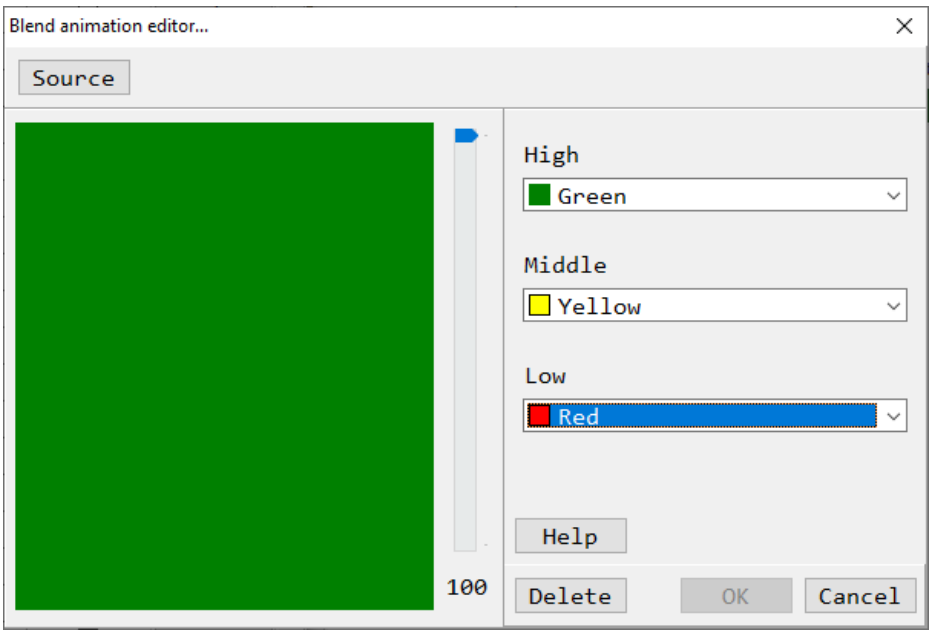
"New" creates a new script global with all the attributes defined and the "Edit" button opens the script global editor.

User level

The "user level" is applied if the "PrintOnClick" attribute is enabled.

Back to list

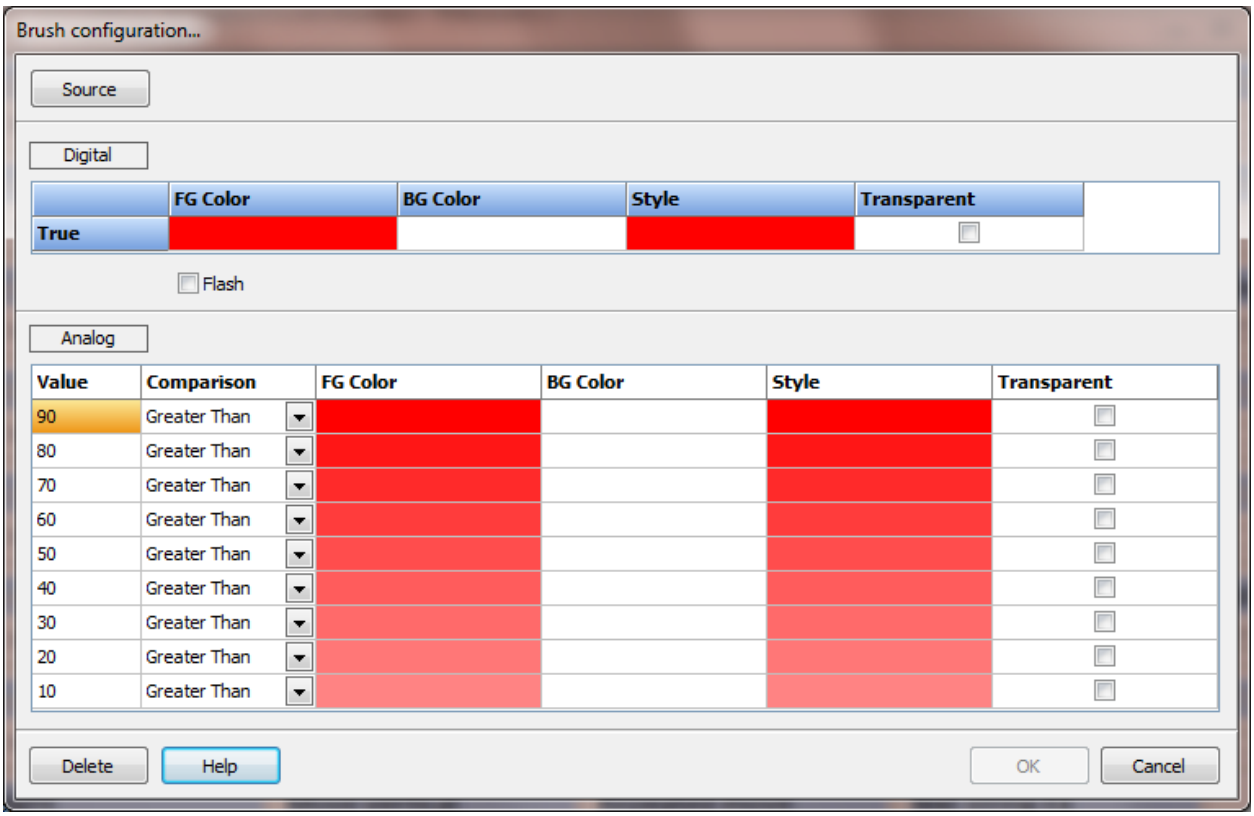
Blend



This animation will “blend” the “High” and “Middle” or the “Middle” and “Low” colors using the source analog value and the foreground color will be set. The value must be between 100-0.

Back to list

Brush

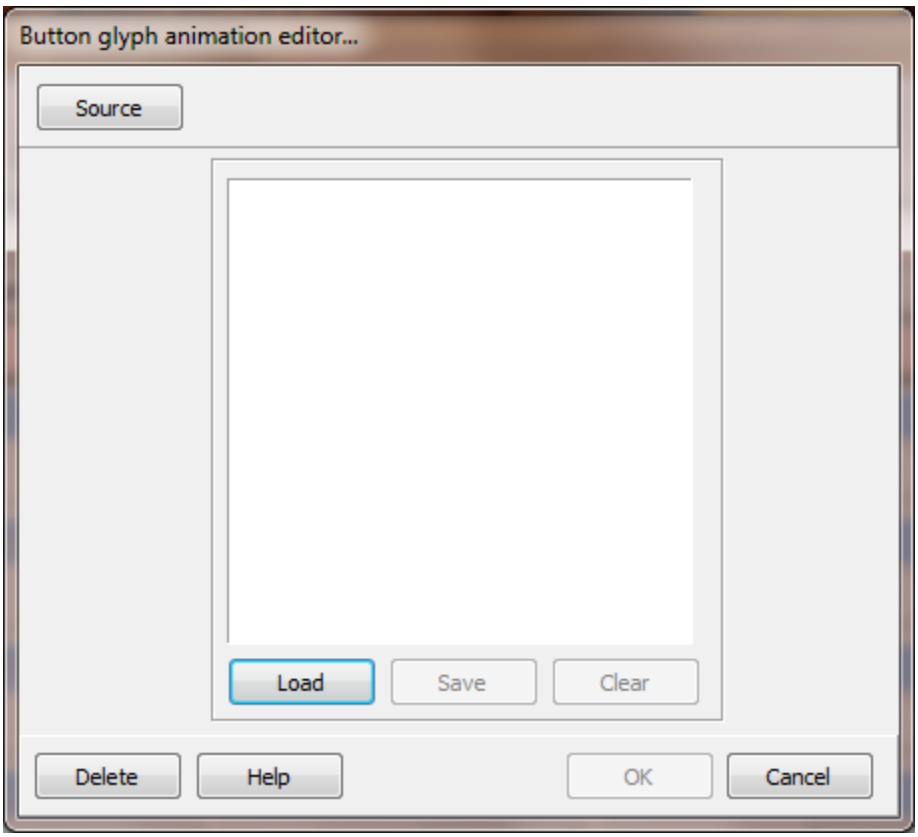


This is used to configure the objects brush attributes. The foreground and background color, style and transparent state can be configured. When the source is digital, the flash attribute can be enabled. When enabled and the source value is true the brush will flash between the design time and animation settings.

The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

Back to list

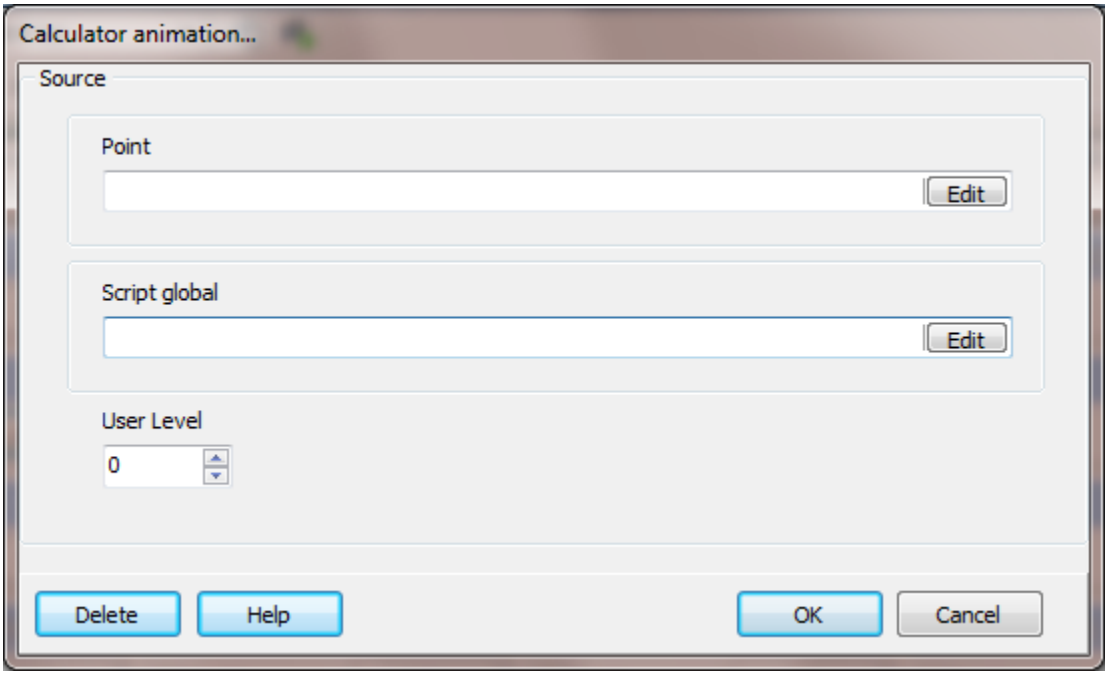
Button glyph



If the button has a glyph, the glyph selected at design time will be displayed if the source tagname/item is false, the glyph selected for animation will be displayed if the source tagname/item is true.

Back to list

CALCULATOR



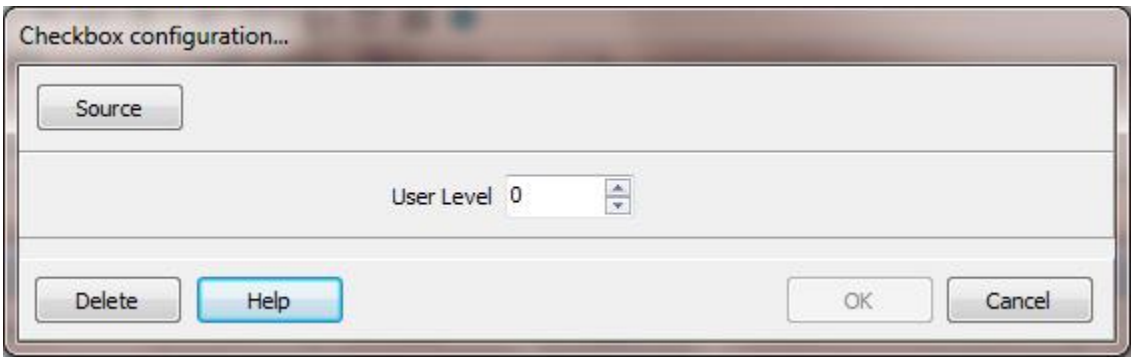
Source

Point: The point to write when the enter/equal key is selected.
Script global: The script global to write when the enter/equal key is selected.

The graphic scripting command "OnCalculatorButtonClick" can also be used.

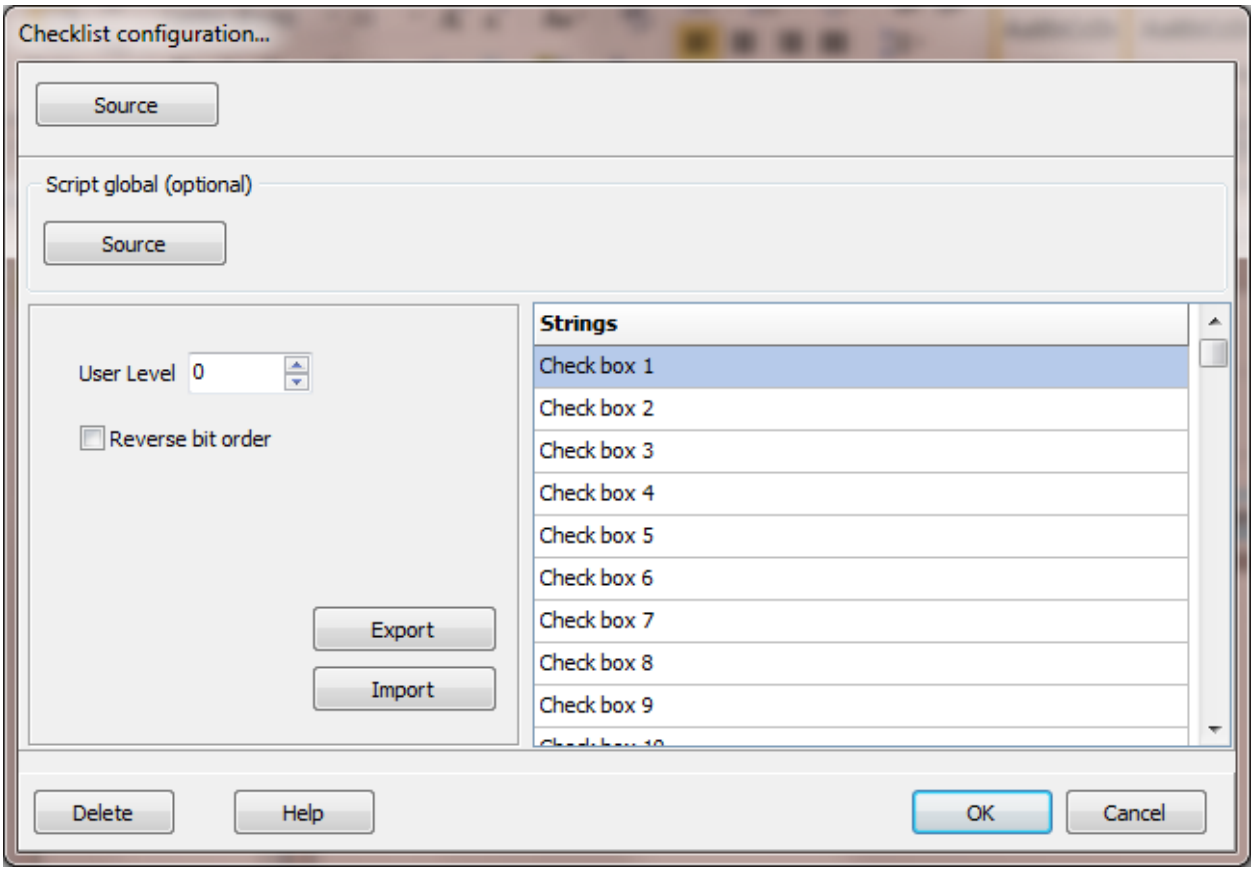
Back to list

Checkbox



Source The source must be a digital type tag.

Checklist



Source The source must be an analog type tag.

String source (optional)

The checklist strings can be configured via the static editor at configuration or at runtime the checklist strings can be loaded from the selected script global section when the window is opened. The section is selected at configuration.

The names in the section must be 1 - 16. <section name>.1, <section name>.2 ... <section name>.16.

When a script global is used:
1. The 'Row count' above is ignored. The number of rows is determined by the highest number name in the section. (16 maximum)
2. A missing name does not delete a checkbox. The highest number in the section determines the row count.
3. If the bit order cannot match the checkbox order, use an 'Analog Host' point as the source and call a script in the 'On Mouse Up' animation to do any needed processing of the bit order. The 'On Mouse Up' animation is called after the source point is updated.

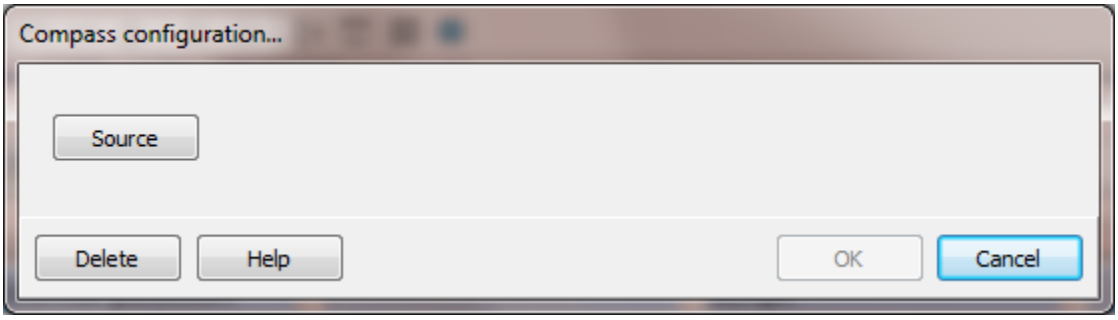
Warning: If the source point is located in an external device, the data might not have been written to the device before the 'On Mouse Up' animation is called.

Reverse bit order

The normal bit order is left to right, 15, 14, 13, etc. Checkbox 1 is the top.
Checkbox 1 = bit 0
Checkbox 2 = bit 1
Checkbox 3 = bit 2
...
If this attribute is enabled the bit order is reversed. From left to right, 0,1,2,3, etc.
Checkbox 1 = bit 15
Checkbox 2 = bit 14
Checkbox 3 = bit 13

Back to list

Compass

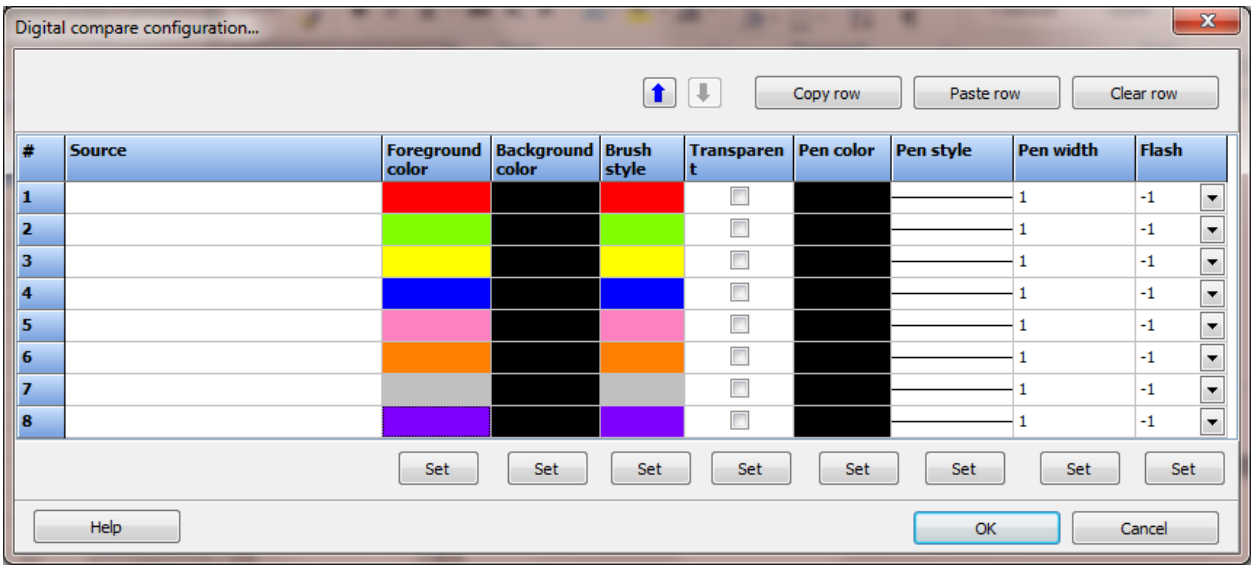


Source The source must be an analog type tag.

See the static configuration for additional settings.

Back to list

DIGITAL COMPARE



When the animation is executing it starts at row one (1) and continues down the rows searching for a true condition. If a true condition is found the attributes for the row are applied to the graphic element and scanning stops. If the source field is blank scanning stops and the design time attributes are applied to the graphic element. If a true condition is not located the design time attributes are applied to the graphic element.

Source

The digital item used to select the row attributes. The tag type can be analog or digital. The item type must be digital. The rows are scanned top to bottom. If a source is not present for a row the scanning stops. A row that does not have a source can be used for flash values, see below, but logic scanning for a true value stops when a true value is found or the source is blank.

Foreground color/Background color/Brush Style/Transparent

The brush settings.

Pen color/Pen style/Pen width

The pen settings.

Flash

If the source value is true the configured attributes are applied to the graphic element. This field is a value between -1 and 8.

-1	No Flash
0	Configuration attributes
1 - 8	Row 1 - 8

When -1 is selected flash is disabled.
When 0 is selected the attributes applied to the graphic element will toggle between the settings for the row and the design time settings.
When the value is 1 - 8 the attributes applied to the graphic element will toggle between the settings for the row and the number of the row selected. The selected row does not need to contain a source address.

Example 1:
Row one is used for a valve closed indication and is set to red.
Row two is used for a valve opened indication and is set to green.
Row three is used for a jog indication and is set to green. The flash is set to 1. When row three is true the graphic element will flash from red to green.

Example 2:
Row one is used for a motor running indication and is set to green.
Row two is used for a motor stopped indication and is set to red. The flash is set to 8.
Row eight does not have a source but the color is set to yellow. When row two is true the graphic element will flash from red to yellow.

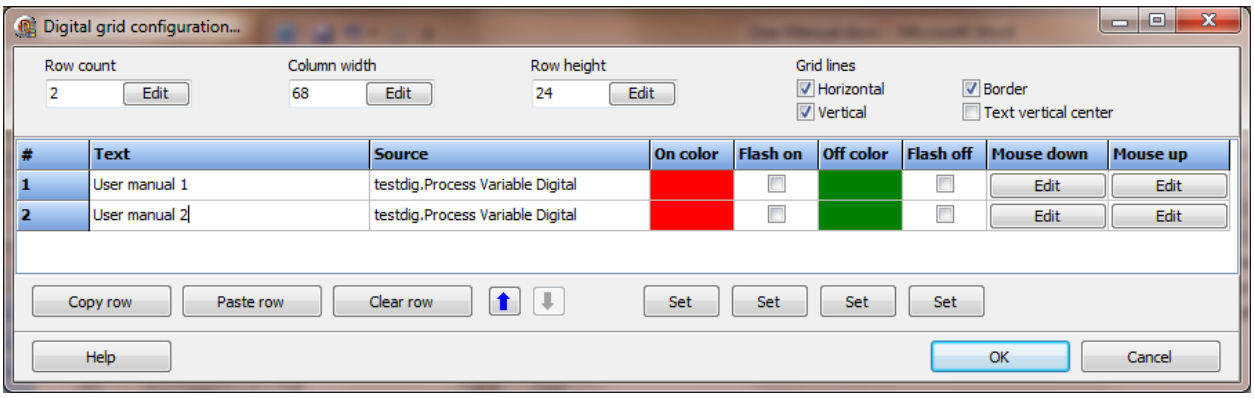
Example 3:
Row one is used for a motor running indication and is set to red. The flash is set to 0.
When row one is true the graphic element will flash from red to the design time settings. For this example, if the only the brush or pen is used it might be easier to use the brush or pen animation. If both the brush and pen are animated this animation is easier.

Set

When the "Set" is selected for a column a dialog will appear allowing for a selection that will be applied to all rows or the value of the first row will be applied to all rows.

Back to list

DIGITAL GRID



The grid contains rows of text and an indicator rectangle. The "Set" button below the “On” and “Off” color columns will display a color selection dialog. If "OK" is selected the color will be set for all the rows. The "Set" button below the Flash “On” and “Off” column will set all the checkboxes in the column to the state of the check box in the first row. The “SetCell” script command can also be used to set cell properties.

Notes:

1. The foreground color is the text color.
2. Transparent is not applicable to this object.
3. When setting the user level for mouse actions, the level selected applies to all rows.

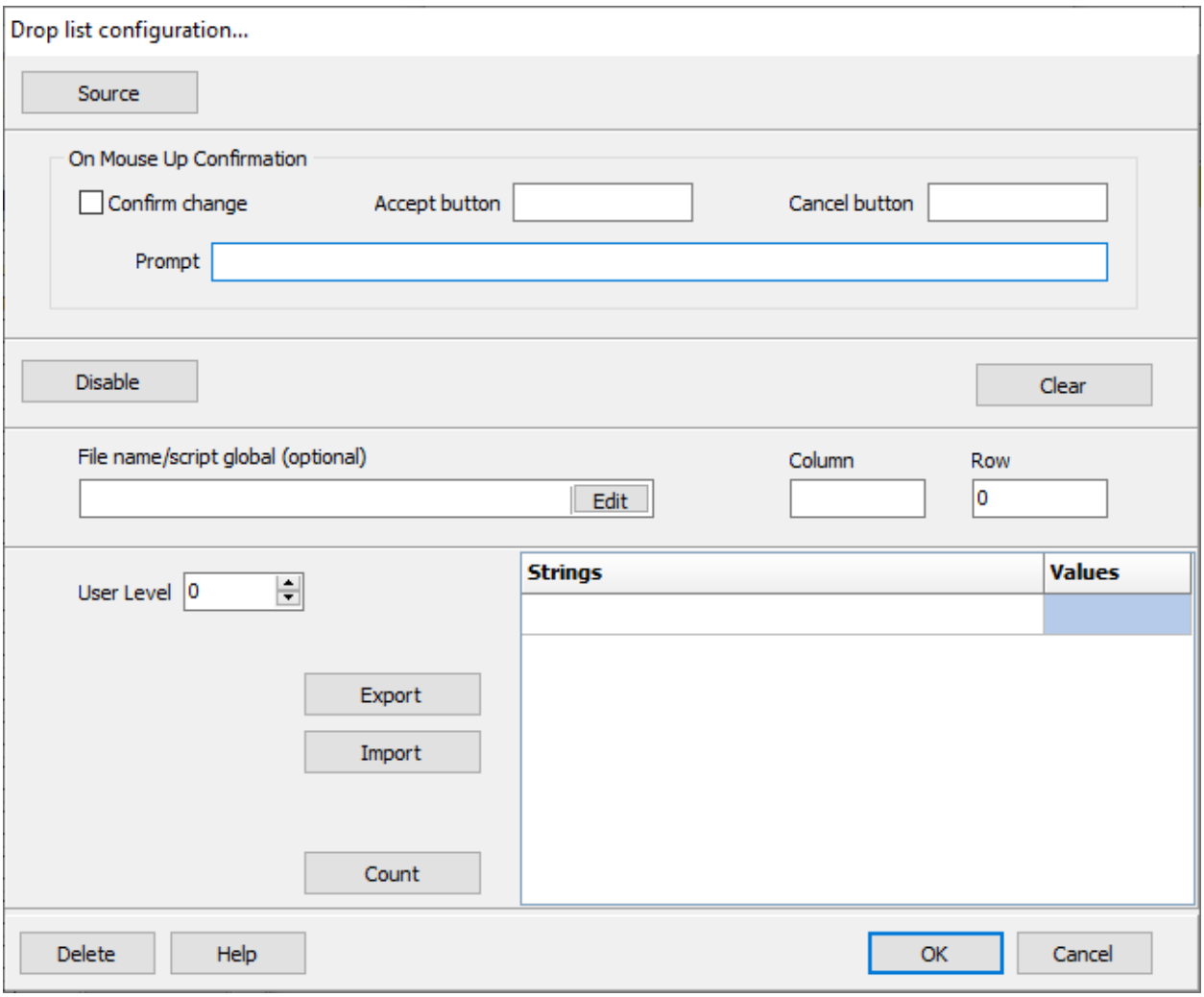
Row count	The number of rows in the grid. (1 - 128)
Column width	The width of the indicator column. (4 - 512) The text column is the remainder of the width of the gird.
Row height	The height of each row. (4 - 512)
Grid lines	Horizontal, Vertical, Border. If enabled the grid/border lines will be displayed.
Text margin	The number of pixels to offset the text from the left or right column boundary.
Text vertical center	The text will be centered in the cell. Otherwise, the text will be aligned at the top of the cell.
Fixed (website only)	The grid will not scroll if the browser window is scrolled.

For each row

Text	The text to display in the first column
Source	The source point must be a host digital or host digital pointer.
On color	The color of the indicator column when the source value is true.
Flash on	If the source is true and this is enabled, the indicator will flash between the on color and off color. **Note:** Flash is not enabled for web pages.
Off color	The color of the indicator column when the source value is false.
Flash off	If the source is false and this is enabled, the indicator will flash between the on color and off color. **Note:** Flash is not enabled for web pages.
Mouse down	This provides for mouse commands when the left mouse button is pressed in the indicator for the row. **Note:** Mouse down is not enabled for web pages.
Mouse up	This provides for mouse commands when the left mouse button is pressed and released in the indicator for the row.

Back to list

Drop list



See here for static configuration.

Source

The source point must be an analog type. The user entered value is stored in the source. If the point is external, "access rights" must be "read/write" for proper operation.

On Mouse up confirmation

If configured this option is called anytime the left mouse button goes up after being pressed in the control. If the mouse pointer is outside the control rectangle the value is not changed and the On Mouse Up animation is not called. If the "Confirm Change" checkbox is checked a confirmation window will appear. If the user selects the "accept" button the On Mouse Up animation is called. If the user selects the "cancel" button the On Mouse Up animation is not called and the list item is restored to the previous value.

Disable

Select a point to prevent the user changing the selection. If the value of "source" changes, the control will reflect the change.

File name/script global (optional)

Notes:
1) If this field is blank, the "strings" property is used.
2) If this field is not blank and the file is invalid/corrupt, the "Strings" property is used.
3) If this field is not blank and the script global is invalid/corrupt, the "Strings" property is used.

File name

File extension "txt"
This is the name of a two column file in the project directory that is read to populate the drop list before it is displayed.
Note: The text file must be two columns, comma separated. No header, no extra data.

File extension "xls"
This is the name of an "Excel" file in the project directory that is read to populate the drop list before it is displayed. The first sheet of the xls file is used for data.

Column:	The column containing the string name. The column + 1, contains the value for the string name. ABCD...Z are the permitted characters.
Row:	The starting row. Rows are imported until a blank cell in the string name column or a blank cell in the value column is found.

Script global

The script global points to the script global that contains the drop list/item values to load in the drop list.

The format is **=SG(<section name> . <item name>)**.
Example: =SG(Combo_1.Manufacture)

When the drop list is loaded, the configured script global item is read. It must contain another script global section name.

Example
=SG(Combo_1.Manufacture) is the configured value.

Three other script global sections exist, each with several items.

Chevy

Item name	Item value
Silverado	7
Corvette	5
Impala	6

Ford

Item name	Item value
F150	9
Crown Vic	8
Pinto	10

Audi

Item name	Item value
Sedan A3	1
SportBack SQ5	3
Sedan S3	2

Set the configured script global to "**Chevy**" and the drop list will be loaded with the values in script global section named "**Chevy**".

Strings — These are the strings that appear in the list. The "Drop list 2" supports some HTML tags for formatting.

Values — The value of "source" is compared to the values in the list. When a match is found the value of the corresponding string is shown in the control. If a value match is not found the content is not specified.

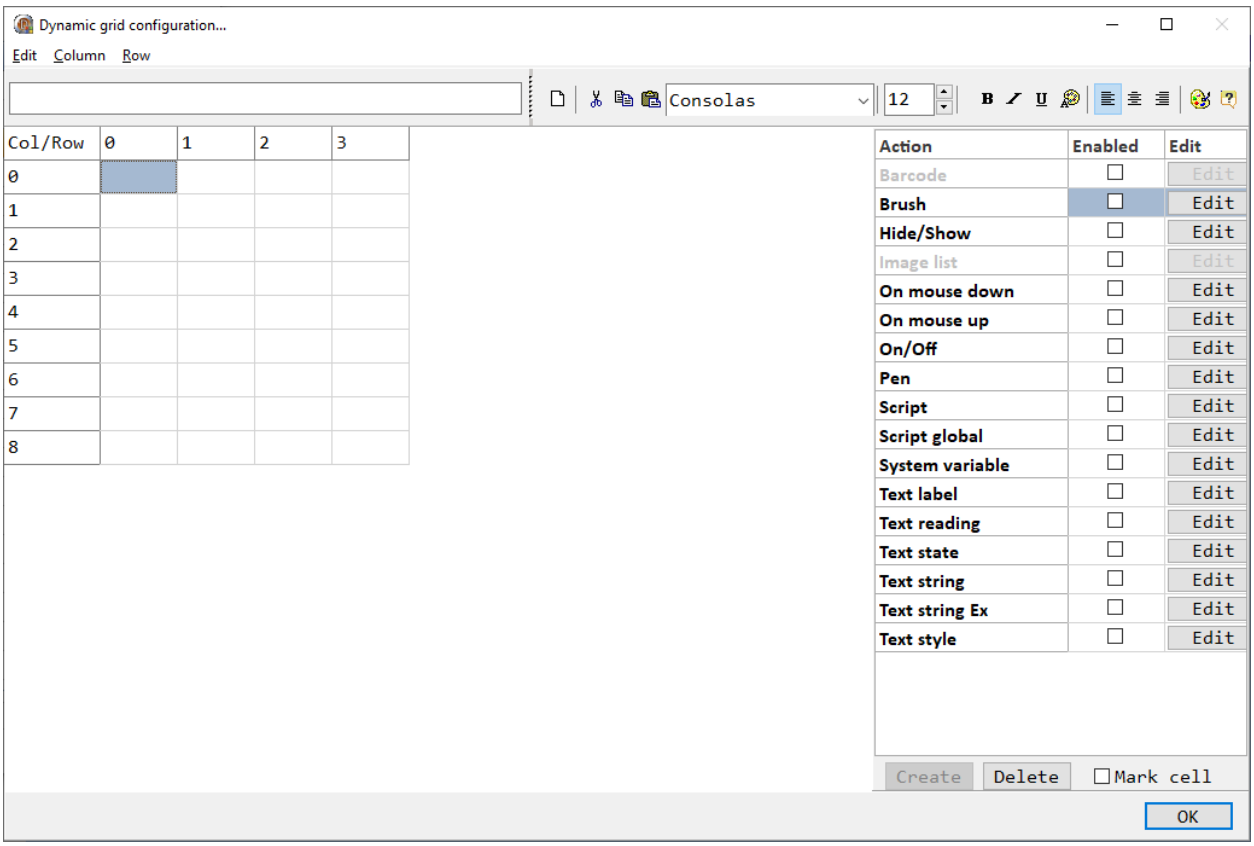
Export — This exports the string grid as a two column comma separated file.

Import — This will import a two column comma separated file to the string grid.

Count — This is the number of strings/values in the list. (1 - 32767)

Back to list

DYNAMIC GRID



The grid can contain cells of text and optional graphic elements with animations.
Text, image list and barcode graphic elements are supported. The text graphic element supports all the text animations.

The grid can be populated via graphic element scripting. Or a combination of animations and script control.

Options

Block mouse up	If the left mouse button is pressed in a cell, not containing one of the graphic elements, and when the mouse button is released, if the pointer is not over the same cell the OnMouseUp event is not called. If this property is enabled the event is called. See mouse events.
Border color	The pen color of the border.
Border width	If the value is greater than zero (0) a border using the “border color” will be rendered around the dynamic grid element.

Ignore cell bounds	If the text/image does not fit the cell bounds, the text/image is clipped. If this property is enabled the text/image is not clipped and the text/image might be visible outside the cell. Note: This option is **not** applied at design time. At design time all cells are "clipped".
Margins Left/top/right/bottom	Creates a margin between a cell side and the grid line.
Resize graphic	The graphic element width/height needs to be, at a minimum, large enough to contain all the cells. If this property is true and the graphic element is not large enough, the graphic element size will be increased to contain all the cells when the OK button is selected in the editor. **Note:** If the graphic element is too small at runtime the cells will not appear correctly.
Show animations	At design time, if the cell has any animations a rectangle will be rendered around the border of the cell. "Mark cell" can be used to temporarily enable/disable.

Merging

To prevent complications with merging a few protections are present.

1) Only empty cells can be merged. After the cells are merged text and/or animations can be added to the merged cell.

2) Unmerging a cell places the cell contents in the left/top cell position.

3) The merged cells are referenced using the top/left column/row of the cell.

4) Merged cells cannot be merged with other cells.

Mouse events

The grid provides two mouse events OnMouseDown and OnMouseUp.

The grid can contain a graphic element in a cell or a cell can be "static" (empty or contains text).

a) If a cell has a graphic element and the script animation is enabled, the cell script will process any configured mouse events.

b) If a cell has a graphic element and the script animation is **not** enabled, the grid script animation, if enabled, will process any configured mouse events.

c) If a cell is static (empty or text only), the grid script animation, if enabled, will process any configured mouse events.

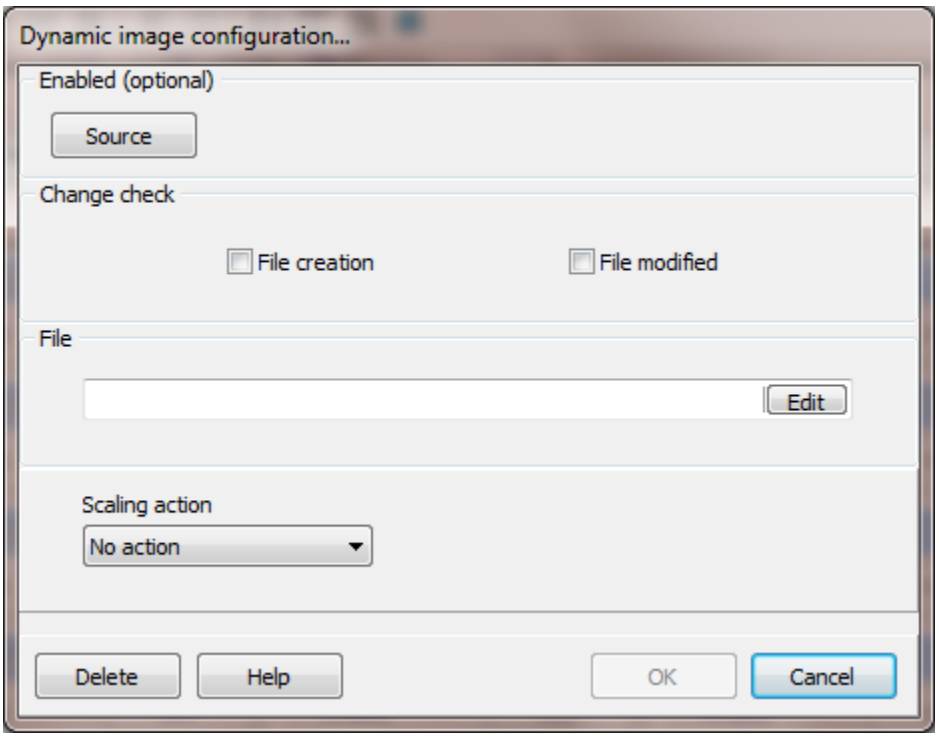
The column, row number and cell text are placed in the graphic element (ge) script object before the mouse down/up event is called.

Miscellaneous

a. While editing a cell, to enter a carriage return/line feed, use CTRL & ENTER keys.
b. The design time fixed column and row are not rendered at runtime.
c. If using the brush animation the background color is used to fill the cell.
d. Press the CTRL key and select one of the alignment buttons. The button will shift to vertical alignment. Left = Top, Center = Center, Right = Bottom
e. Barcode, the barcode color is the font color.
f. Press the CTRL key and select the “Cell color” button sets the “color picker” dialog color to the gird color.

Back to list (complex objects)
Back to list (animations)

DYNAMIC IMAGE



This animation monitors an image file and updates the image on the screen when the file changes.

Source (optional)

A digital reference used to enable/disable changes to the graphic element image regardless of a change in the image file. If a source point is not configured the animation will be active when the screen is open.

Change check

The “change check” logic always looks at the file size for change. This might not be adequate because the file size might not change when the image changes. For example, only a color change might not alter the size of the file.

File creation
This enables the change detection logic to utilize a change in the file creation date/time as a change in the image.

File modified
This enables the change detection logic to utilize a change in the file modification date/time as a change in the image.

Scaling action

This only applies to JPEG and bitmap images. WMF/EMF and PNG are rendered in the full graphic element bounds.

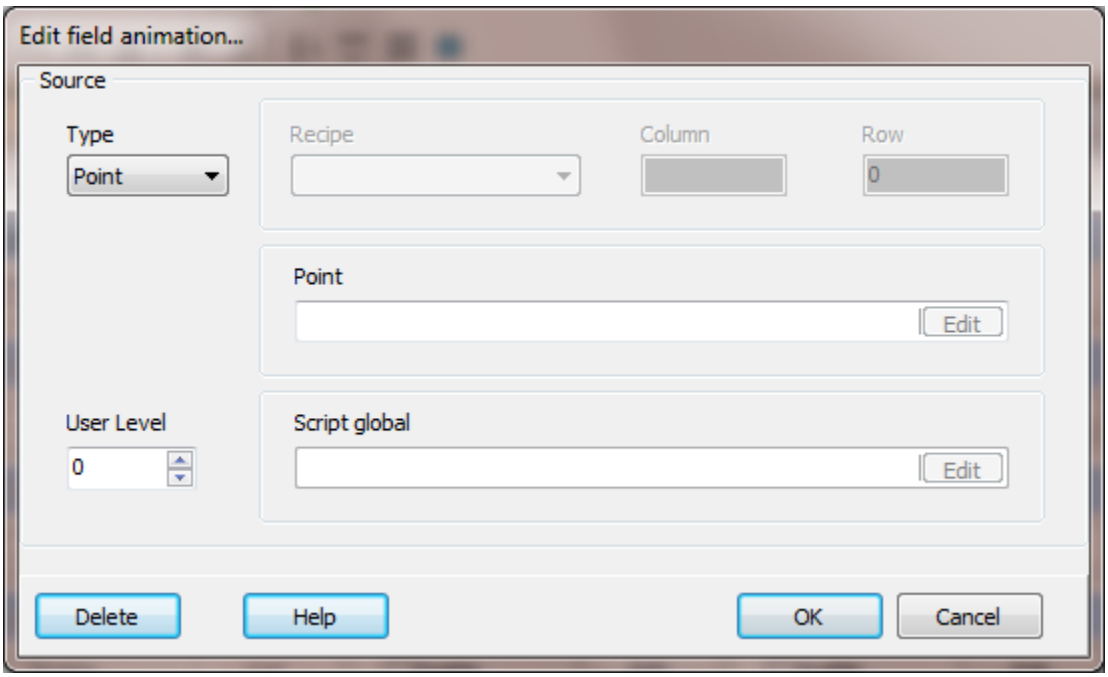
No Action:
The image will be imported and centered in the graphic element bounds.

Crop:
The bitmap will be aligned to the top/left of the graphic element and if the image is larger than the graphic element, the bottom and right will be cropped to fit the graphic element.

Expand, Shrink:
The bitmap will be expanded or shrunk to fill the graphic element bounds.

Back to list

EDIT FIELD



This is an edit field used to display/edit the contents of a cell in a recipe, a point value or a script global.

Type

Recipe The source is the cell/column-record of a recipe that is displayed and can be edited.

Point The source is a point.item that is displayed and can be edited if it is a host point or the point has an access type of read/write.

Script global
The source is a script global that is displayed and can be edited.

Recipe

The name of the recipe containing the cell to display/edit.

Column XLS: The column is an uppercase letter that is the column. A = column 1, B = column 2, etc.
ODBC: The column is the field name for the column.

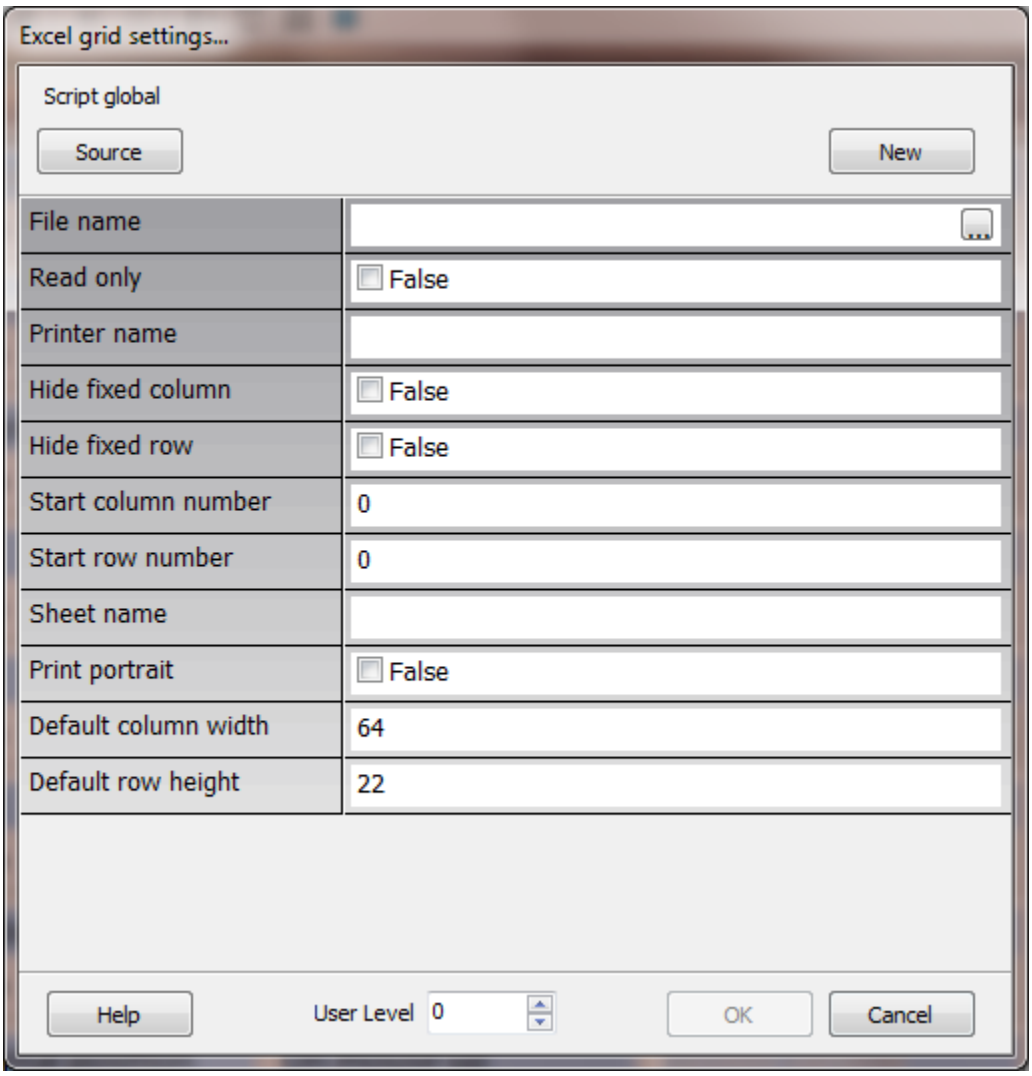
Row The row is the row/record number. The row index begins at 1 (one) and increases.

Point The point.item to use as the source and destination.

Script global
The script global section.item to use as the source and destination.

Back to list

EXCEL GRID



This is a string grid used to display/edit the contents of an Excel file or CSV (comma separated values) file. Excel is not required to be installed on the computer. This grid does not use Excel. The grid cannot perform macros or calculations.

The object uses a 'script global' section to control and report operations.

Note: A grid exported to XLS/XLSX might contain some visual differences. For example, a date field displayed as '27-Jun' might appear as '6/27/2012'. A negative value '(87.50)' might appear as '-87.5'. The data will be correct.

Item names

File name

This is the complete path and file name for the Excel or CSV file. This can be a shortcut and will be resolved.

Notes:
1) XLS, XLSX and CSV files are supported.
2) Do not use this graphic element to view or edit a file that is in use by one of the ODBC ports or an active custom log file.
3) This attributes only applies when the file to load is an Excel formatted file. These attributes do not apply to CSV files.

Read only

If 'True' the grid cannot be edited, 'False' the grid can be edited. Any changes to the text of the grid are not saved to disk until a 'Save' command is executed.

Printer name

If blank, the default printer is used. If not blank the name entered is the name of the printer to use when the print command is executed.

Hide fixed column

If 'True' the first column will be a regular column If 'Read only' is not true the column will be editable.

Hide fixed row

If 'True' the first row will be a regular row. If 'Read only' is not true, the row will be editable.

Start column number

This is the column in the Excel file to start the load or save. **Note: 3**

Start row number

This is the row in the Excel file to start the load or save. **Note: 3**

Sheet name

The 'sheet name', in the Excel file, to import or export. If the name is blank the default sheet is used. **Note: 3**

Print portrait

The default printing orientation is 'landscape'. If this is enabled the output will be in 'portrait' orientation.

Default column width

When the file type to load is Excel (XLS or XLSX) the column width is collected from the Excel file. The “Default column width” is applied to CSV files.

Note: The column width can be set for each column or a subset of the columns for CSV and Excel files. Add script global items to the source script global to set a column width override value for a column.

For example, to set column 0 and column 3 to 80 pixels add two items, “ColWidth0” with a value of “80” and “ColWidth3” with a value of “80”. If an item named “ColWidthX” is not found the column width will be determined from the Excel column attributes for XLS or XLSX files or from the “Default column width” attribute for CSV files.

Default row height

When the file type to load is Excel (XLS or XLSX) the row height is collected from the Excel file. The “Default row height” is applied to CSV files.

Cell formatting

When the file type to load is Excel (XLS or XLSX) the cell formatting will be imported (font color, size, etc.).

Status

This field is used to report the status of grid commands. If the command was successful the status will be blank. If the command fails the status will be 'Exception occurred'.

Changed

This field is used to indicate if at least one field of the grid has been edited and not saved. If 'True' at least one field has been edited and not saved to disk. If 'False' editing, has not occurred. When a 'save' command is executed this field is set to 'False'.

Commands

These commands are used to control the operation of the grid. The command text must match exactly (case sensitive) to the commands listed below. When the command begins to execute the 'Status' field will be updated and this item will be cleared.

Load

This loads the grid from the file in the 'Filename' field. When the grid is first displayed this command is automatic. If this command is called the current contents of the grid are discarded and the file contents are loaded.

Save

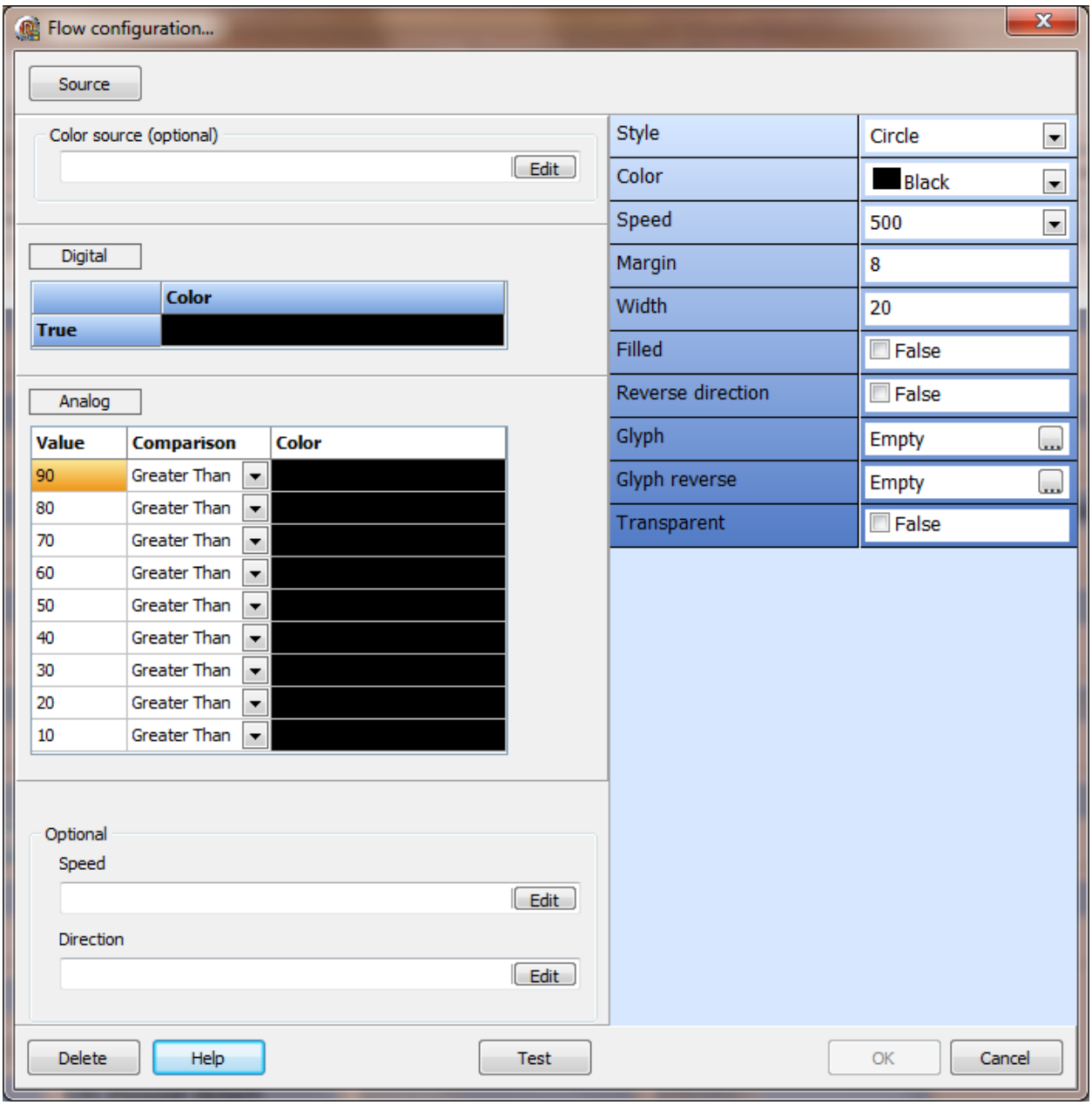
This saves the grid contents to the file in the 'Filename' field. If the file exists, it is overwritten. If the file does not exist, it is created. If the path to the file does not exist the command will fail.

Print

This prints the grid contents to the printer selected in the 'Printer name' field.

Back to list

Flow



This graphic animation is used to show flow in a vertical or horizontal rectangle. The attributes on the right side of the window are the static settings and some attributes can be overridden by dynamic configurations.

Not all settings apply to all styles.

If the width of the rectangle is greater than the height the flow will be horizontal, otherwise it will be vertical.

The "Test" button uses the static settings to simulate the animation.

Source

A digital reference used to display the flow animation or the rectangle without flow using the design time settings.

Static settings

Style

The type of indicator in the rectangle to represent flow.

Color

The color of the indicator selected. These settings do not apply if the style is "glyph".

Speed

A value between 50 and 2000 milliseconds, in 50 millisecond increments, that controls the "flow rate" in the element. Shorter time is faster flow.

Margin

The flow indicator is centered in the rectangle. This value is used to create a "gap" between the indicator and the rectangle sides. The value is applied to both sides of the rectangle.

Width

The width of the selected indicator. The height is set by the size of the rectangle minus the margin.

Filled

If enabled and the style supports "filling" the indicator will be filled with the selected color. This does not apply to bitmaps.

Reverse direction

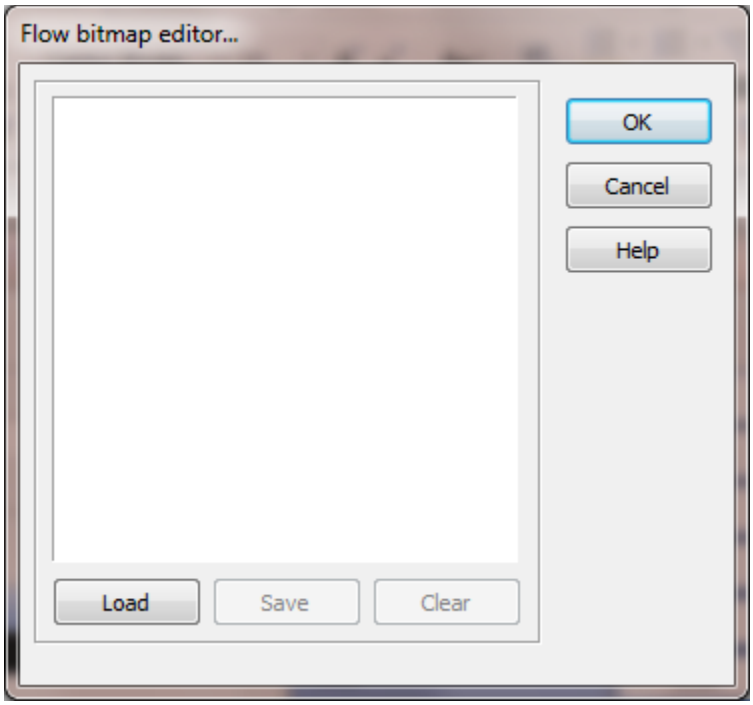
The normal flow indication is left to right or top to bottom. If this is enabled the flow will be right to left or bottom to top.

Glyph/Glyph reverse

A user created symbol can be used. If required, if "Glyph reverse" is not selected the selected "glyph" will be "flipped" and used for the reverse direction.

Note: If the flow is vertical and the glyph indicates direction, create the glyphs on the horizontal plane. The program will rotate the glyphs for the correct flow direction.

Glyph editor



This is used to select a glyph.

Transparent

Only applies if the style is "Glyph".

All of the following attributes are optional.

Color source

If the point.item is digital the false color will be the static color and the true color will be the selected color.

If the point.item is analog the comparison will select the color and the static color is not used.

Speed

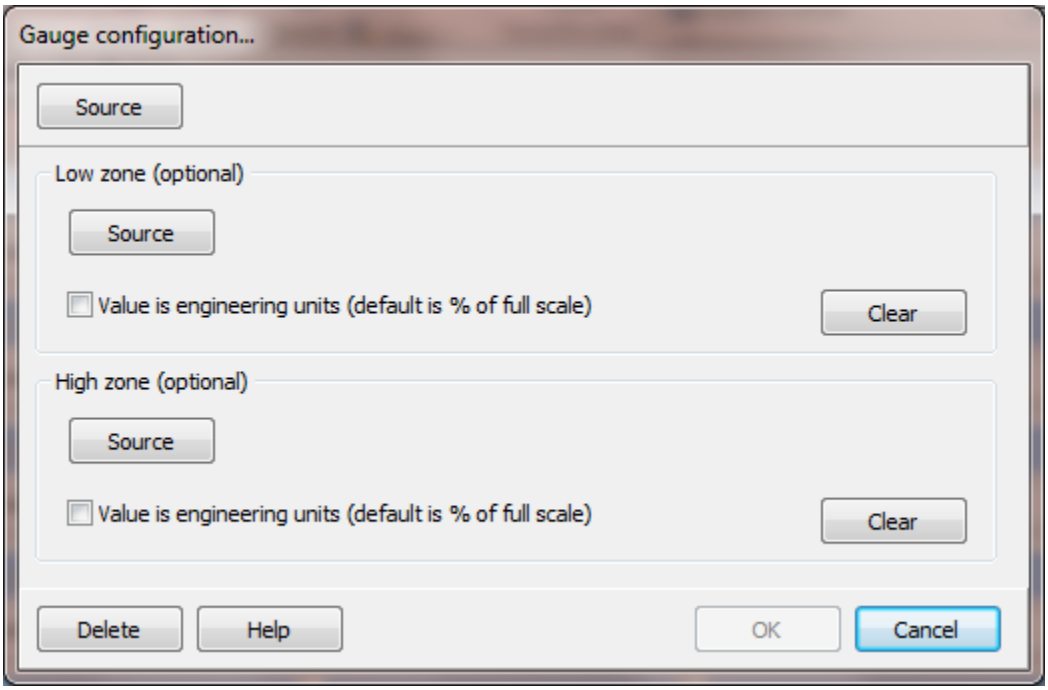
The selected point.item will determine the speed. The value is limited between 50 and 2000 milliseconds, in 50 millisecond increments. A value outside the limits of 50 - 2000 defaults to 500.

Direction

A true value "toggles" the static direction; a false value is the static direction.

Back to list

GAUGE



This is an analog gauge. Select the analog tagname.item to monitor. Some gauges have a scale that is colored to indicate when the value is 'in range', a low zone and high zone. For example, the low zone might be yellow, and the high zone might be red. The area between the two zones is green. The optional zone fields are to allow the zone range to change based on a tag value. The default for the value is percent of full scale. Enabling the check box for the source value allows the source to be in engineering units.

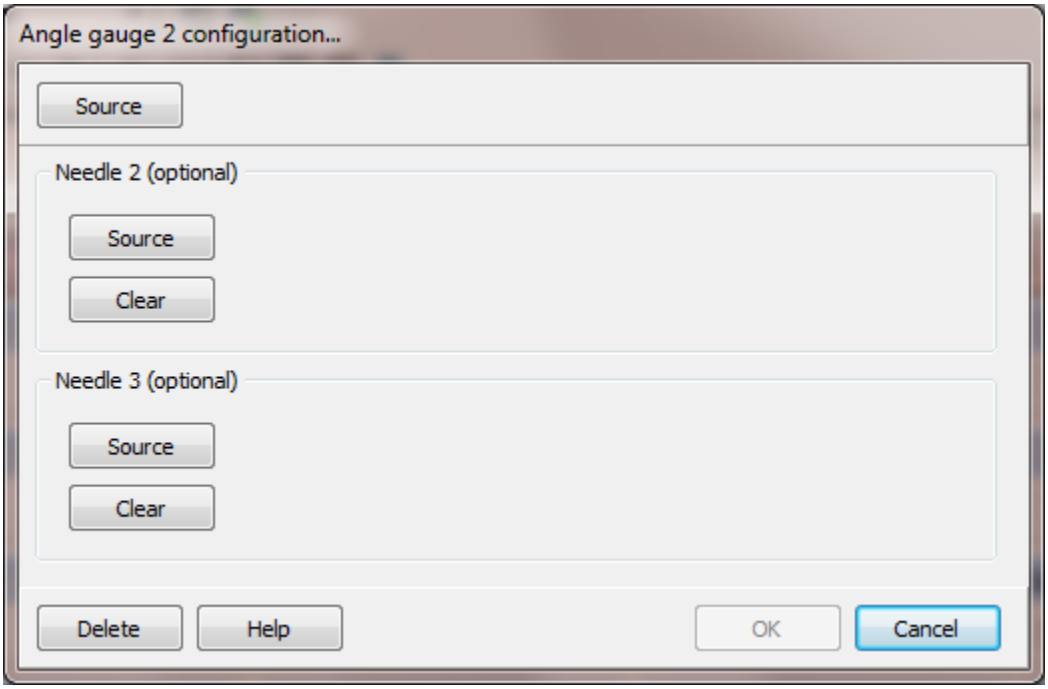
Back to list

Gauge

This is for the website analog gauge 2, vertical gauge 2 and pointer gauge. Select the analog tagname.item to monitor.

Back to list

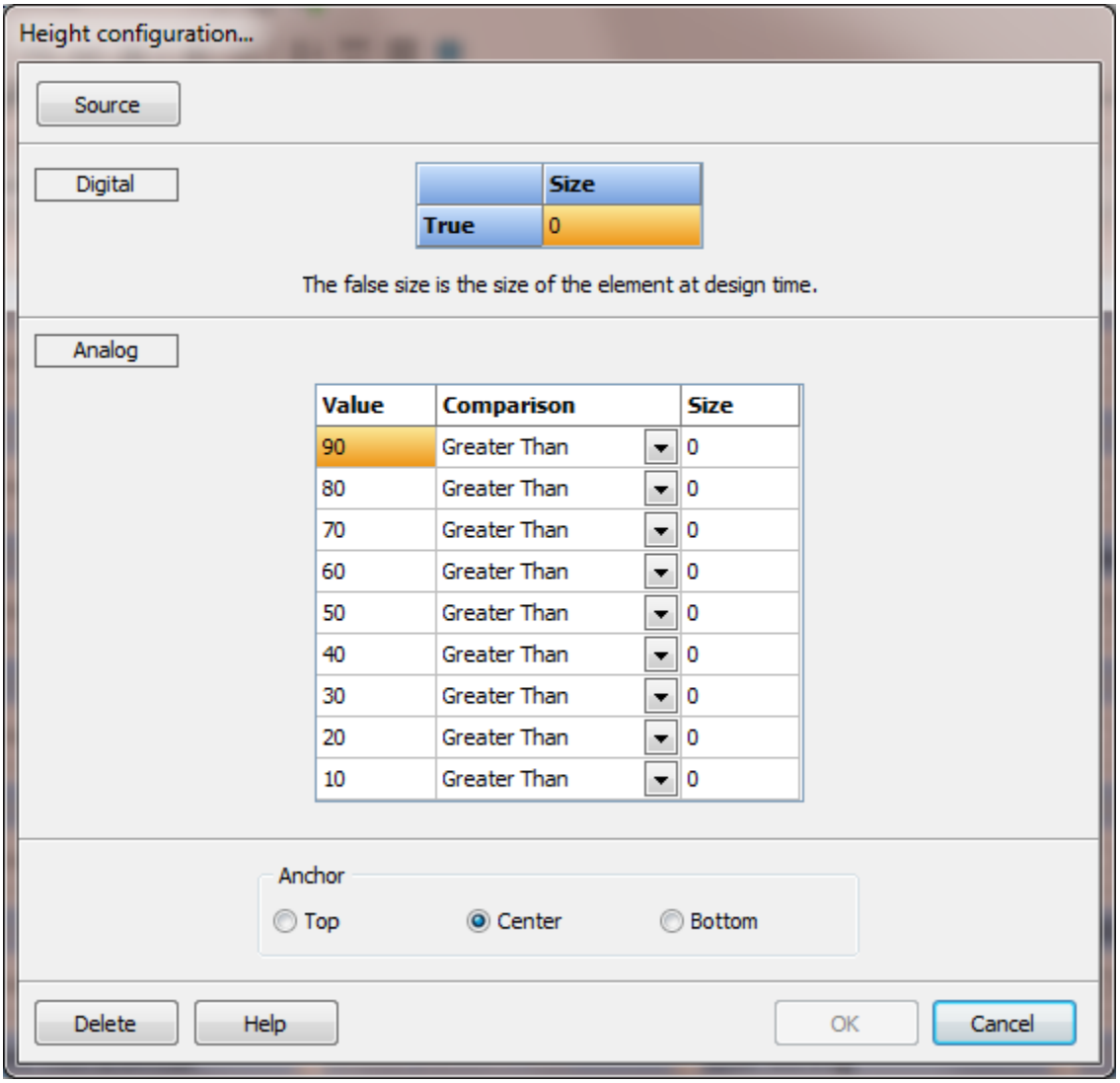
Gauge 2



The analog gauge 2 supports one to three needles. The first needle must be defined and the second and third needle are optional.

Back to list

HEIGHT

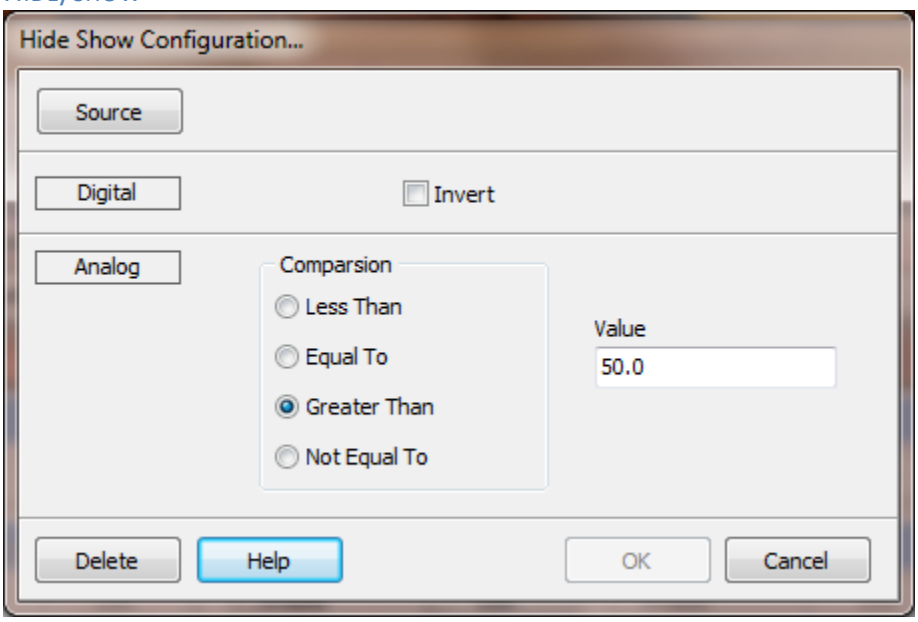


This is used to configure the height of an object.

The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

Back to list

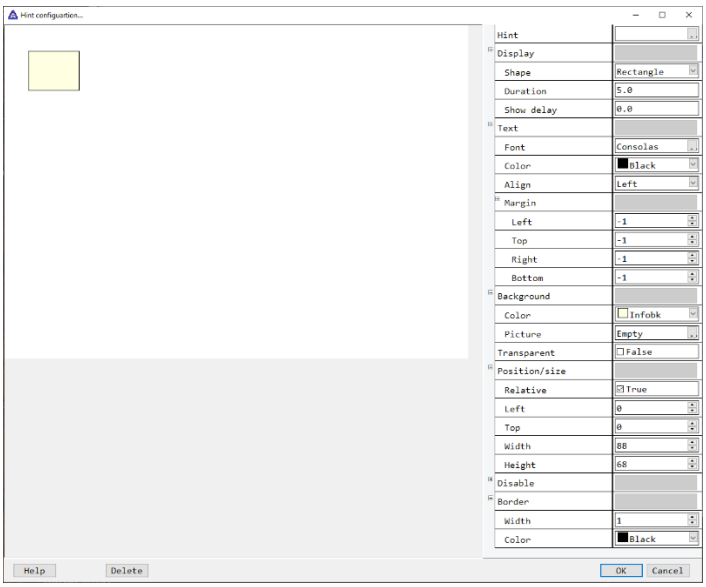
HIDE/SHOW



This causes an object to be visible or hidden. If the object is hidden all other animations are not processed. When the condition is true the graphic element is hidden.

Back to list

Hint



This animation displays a message (text and/or image) when the mouse is over a graphic element. The hint can also be controlled at runtime via scripting.

Hint If not empty, the text to display.

Display

Shape The shape of the hint area, rectangle, diamond or circle.

Duration The amount of time to display the hint. The value is rounded to the nearest ½ (0.5) second.
0 = infinite (while the pointer is over the graphic element the hint will be visible)

Show delay When the mouse is moved over the graphic element, the amount of time to wait before displaying the hint. The value is rounded to the nearest ½ (0.5) second.

Text

Font The font name.

Color The font color.

Align Horizontal alignment of the text, left, center or right. Use “Margin” to vertically align the text.

Margin(s) This expands or shrinks the “bounds” area for text. Negative values shrink a boundary and positive values expand the boundary. For example, the hint text is top justified/aligned. To move the text down, use the “Margin top” and a greater negative number will move the text further down.

Background

Color The color to fill the hint area if “Transparent” is not enabled.

Picture A picture can be used for the background. The image will be scaled to fit the bounds.

Transparent If enabled, the background will not be filled with the background color and the text background will be transparent.

Position/size

Relative If enabled, the hint area is relative to the graphic element. If not enabled, the top/left are screen coordinates.

Left/top The left and top of the hint bounds.
Width/height The width and height of the hint bounds.

Disable

Point If the value of the configured point is true the hint will not be displayed.

Script global If the value of the script global is 1 (one) the hint will not be displayed.

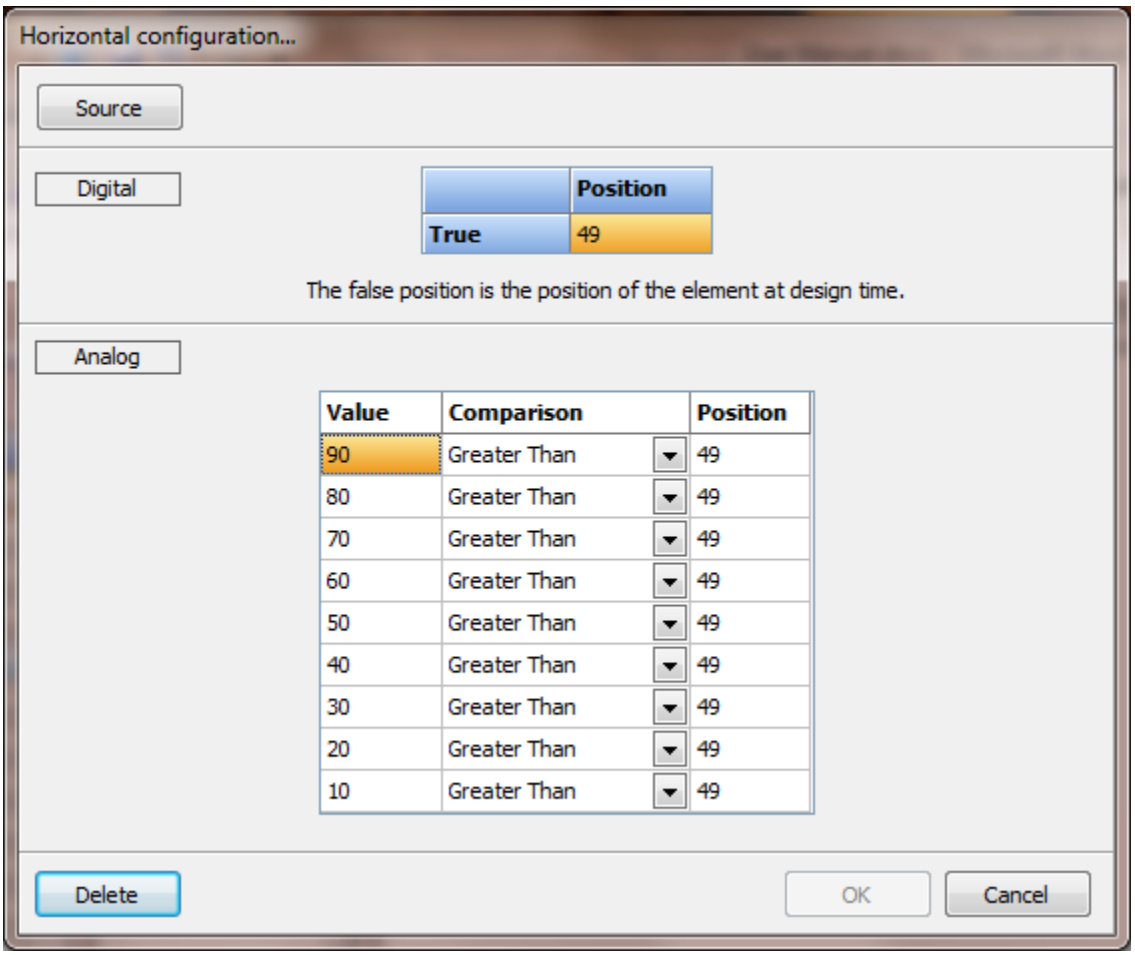
Border

Width The width of the border around the hint shape. 0 (zero) is no border.

Color The color of the border.

Back to list

HORIZONTAL POSITION

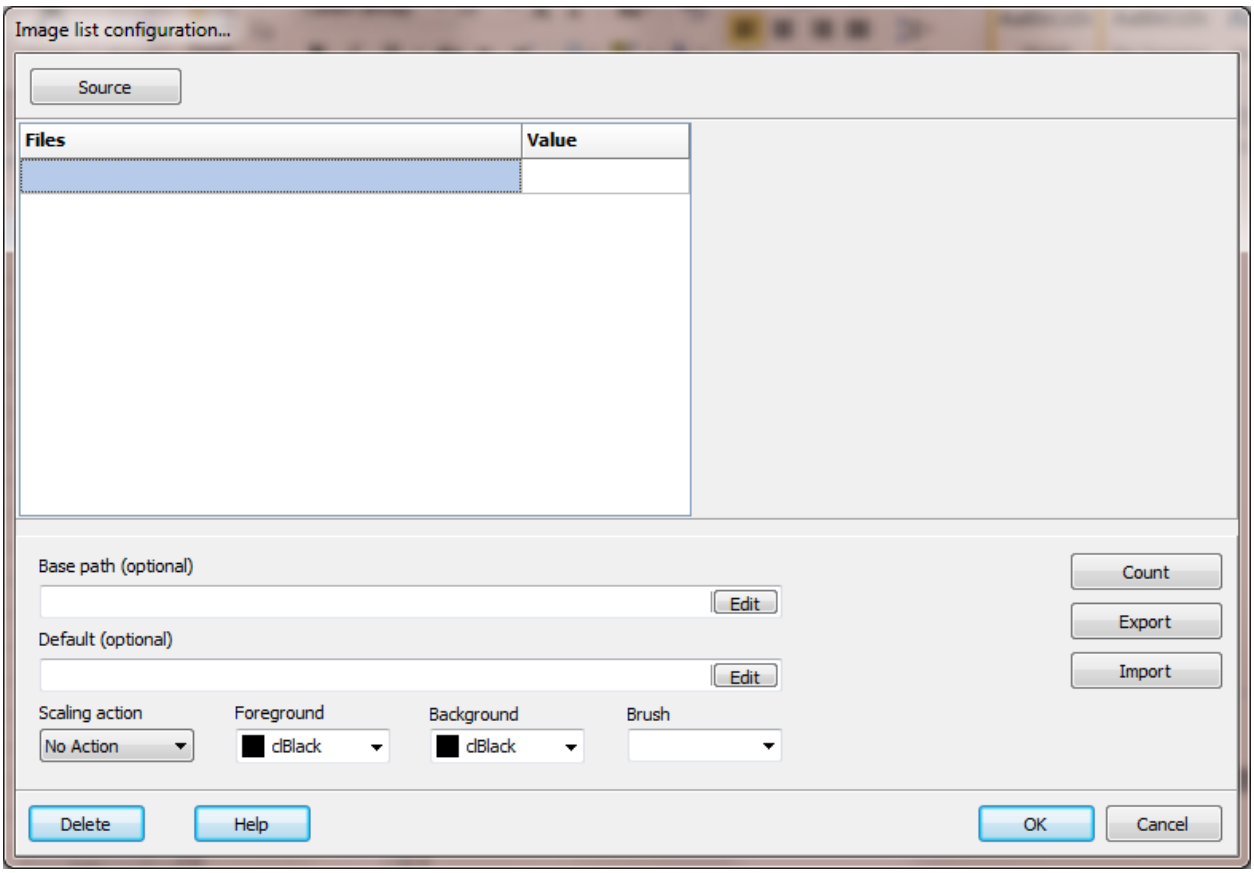


This is used to configure the horizontal position of an object.

The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

Back to list

IMAGE LIST



Source

The source point.item must be an analog type. The user entered value is stored in the source.

Files

These are the files containing the images to display. The file must be a fully qualified path. See the "base" below to use a prefix.

Values

The value of "source" is compared to the values in the list. When a match is found the value of the corresponding file is displayed. If a value match is not located, the default image is displayed. If the default image is not configured or does not does not exist, an image is not displayed

Base (optional)

This is an optional base path for the location of the image files.
Each file could be a full path:

Files	**Value**
C:\images\project 1\A.bmp	1
C:\images\project 1\B.bmp	34
C:\images\project 1\C.bmp	45

Or the base parameter could be set to "C:\images\project 1\" and then the file entries could be:

Files	**Value**
A.bmp	1
B.bmp	34
C.bmp	45

Default (optional)

This is a path to the image to display if the source value does not match a configured value. If this field is blank and a value does not match, an image will not be displayed.

Scaling action

The display size of the file image might not be the same size as the image in the window. This provides direction on how the image should be displayed

No Action:
The image will be imported into the window and centered in the graphic element.

Crop:
The image will be aligned to the top/left of the graphic element and if the image is larger than the client area of the graphic element the bottom and right will be cropped to fit the graphic element.

Expand, Shrink:
The image will be expanded or shrunk to fill the area of the graphic element.

Foreground color

This is the color of the fill when the image does not fill the rectangle.

Background color

This is the color of the fill when the image does not fill the rectangle.

Brush

This is the brush used to fill the rectangle when the image does not fill the rectangle.

Note: Web pages: scaling, colors and brush are not applied to web pages. The image will be rendered by the browser within the bounds of the graphic element. The Z-order is maintained for image list.

Count

This is the number of files/values in the list. (1 - 32767)

Export

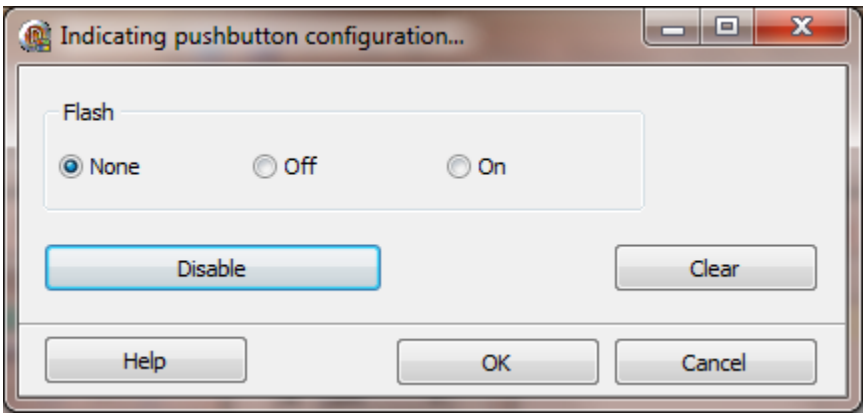
This exports the string grid as a two column comma separated file.

Import

This will import a two column comma separated file to the string grid.

Back to list

INDICATING PUSHBUTTON



Flash

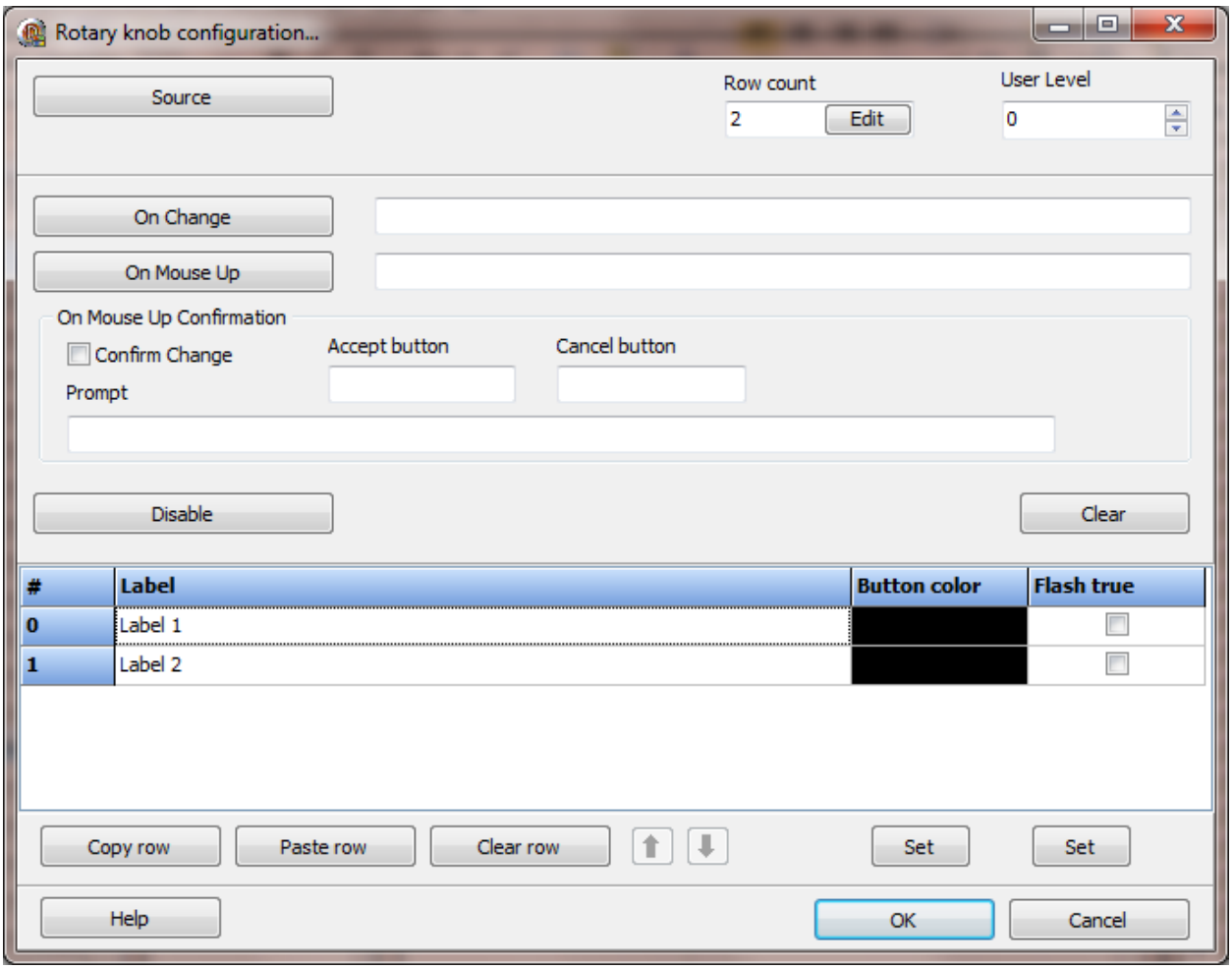
If “Off” or “On” is selected the LED will flash.
1. The pen animation must be valid and enabled.
2. If the animation is digital the 'True' color and the design time color are used.
3. If the animation is analog and the 'state' is true the configured 'state' pen and the design time pen are used.
4. If the animation is analog and the 'state' is false the configured pen in the bottom row of the grid and the design time pen are used.

Disable

This is used to disable all mouse actions for the control. The indication is via the other animations.

Back to list

KNOB SWITCH



Source

The source point must be an analog or host analog pointer. The index value (value in # column) is placed in the source point.

Row count

The number of switch positions.

User Level

The logged on user must have a user level equal to or greater than this value to activate the control.

On Change (Optional)

The script to execute when the switch position is changed.

On Mouse Up (Optional)

If configured this script is called anytime the left mouse button is released after having been pressed in the control. If the mouse pointer is outside the control rectangle the switch position is restored to the value when the mouse button was pressed. If the "Confirm Change" checkbox is checked a confirmation window will appear. If the user selects the "accept" button the script is called. If the user selects the "cancel" button the script is not called and the switch position is restored to the value when the mouse button was pressed.

Disable (Optional)

If this point is configured and the point value is true, changes to the switch by the user are disabled. Changes to the switch source tag will result always result in a change to the switch position. If disabled, the switch becomes an indicator only.

Each row

Label

The text to display at the position.

Button color

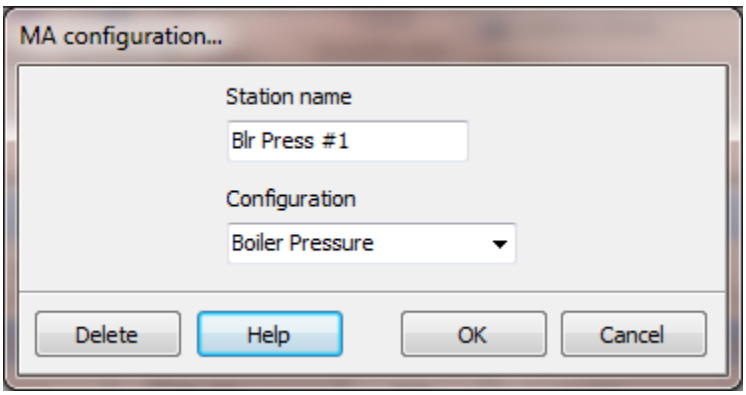
The color of the button.

Flash true

If enabled the button color will flash between the design time color and the button color when the switch is in this position.

Back to list

MA STATION



The MA (Manual/Auto) is a compound object. This allows for setting a station name and default configuration. The configuration can be changed via scripting.

The name of the station is only a place holder for the configuration name. When a window opens it reads the assigned station name from the MA station graphic element and checks for the station name in a list of station name/configuration name pairs. If the station name is found it returns the configuration name and the station uses the configuration for operations.

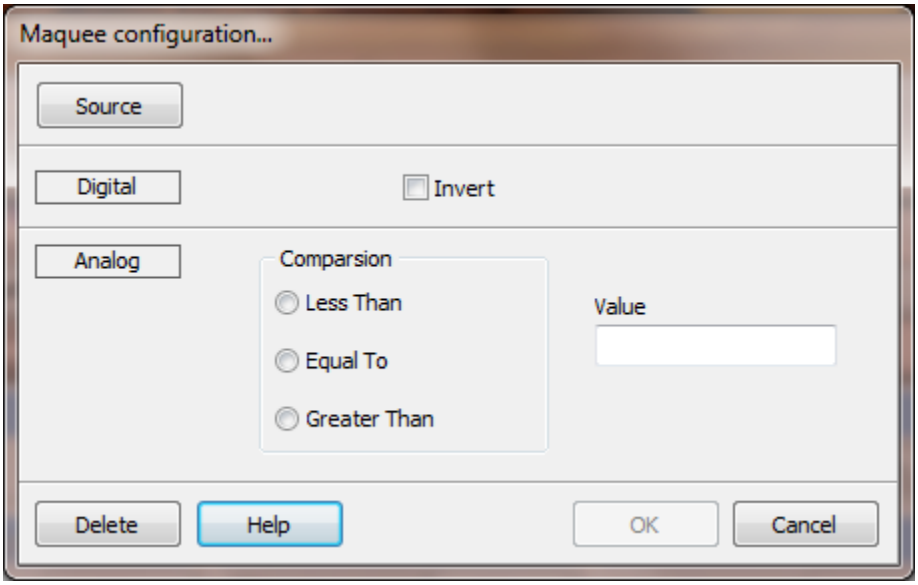
If the name is not found it uses the configuration name selected at design time.

If via a script command the station name/configuration name is set then all stations with that station name will use the configuration set, when the window opens. This only applies when the window opens. After the window is open the station name/configuration name pair can be changed without change to existing open windows

As long as:
A. All stations have unique station names.
OR
B. The station name/configuration name pair is not modified via scripting.
OR
C. The station name/configuration name is always set, via scripting, before the window is open.
AND
D. The same station name is not used twice in the same window.

Back to list

Marquee

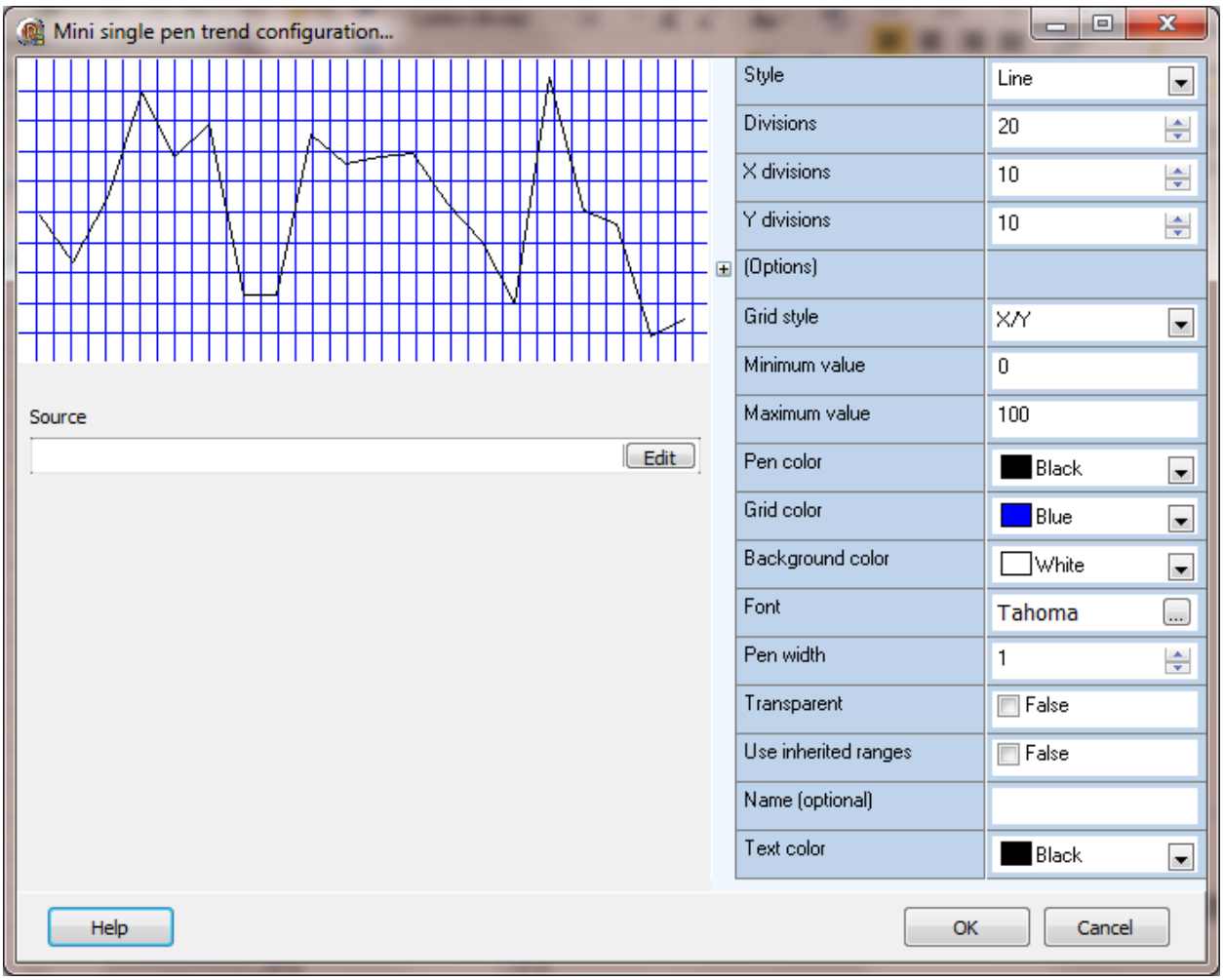


Source

The source point.item used to control when the marquee scrolls. If the source point.item is digital and not inverted the marquee will scroll when the point.item is true. If the point.item is analog, the marquee will scroll when the selected comparison evaluation is true.

Back to list

MINI SINGLE PEN TREND



This graphic element displays the previous 2 - 100 seconds of a point as a trend.
Note: The 'Mini single pen trend enable" in the configuration settings must be enabled for points to store the values.

Style

The style of the trend. (Bar, Line, Scatter, 3D bar, Filled)

Divisions

The number of samples displayed. Default is 60.

X divisions

The number of vertical divisions on the grid. This is used to draw the grid lines. Lower numbers = more divisions = more vertical lines. The value is limited to the width of the trend.

Y divisions

The number of horizontal divisions on the grid. This is used to draw the grid lines. Lower numbers = more divisions = more horizontal lines. The value is limited to the height of the trend.

Options

These options apply when the "style" is "Bar or "3D bar"

Values	Display the value above the bar.
Transparent	Transparent text
Rotated	Rotate the text 90 degrees. (Does not apply to all fonts.)
No zeros	If the sample value is 0 the text will not be displayed.

Grid style

This attribute sets the grid lines.

None	No grid lines will be drawn.
X – Vertical	Only vertical lines.
Y – Horizontal	Only horizontal lines.
X/Y	Vertical and horizontal lines.

Minimum/maximum value

The minimum and maximum values for the grid.

Pen color

The color of the pen.

Grid color

The color of the grid lines. (Foreground color)

Background color

The background color of the grid.

Font

The font settings when the style is "Bar or "3D bar".

Pen width

The pen width when the "style" is "Line".

Transparent

The background color is not applied to the trend.

Used inherited ranges

This overrides the "Minimum/maximum value" above and uses the engineering minimum/maximum from the assigned point.

Brush color

The color of the brush.

Name (Optional)

The name is used when a command to change a pen at runtime is executed. Trend names can be duplicated. If a change command is used, the first trend with a name matching the name in the command will be used for the command.

Text color

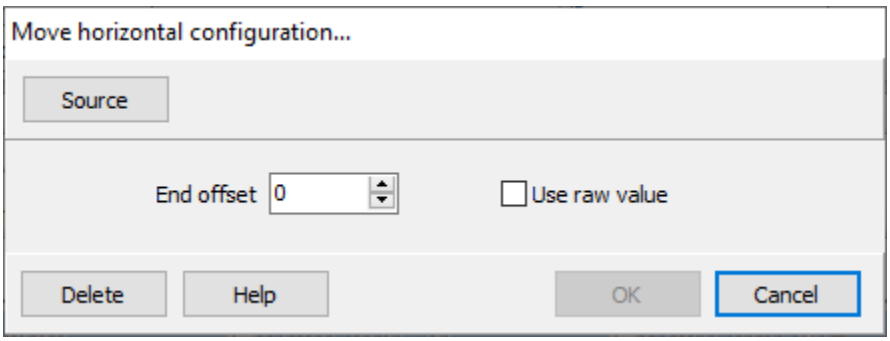
The color of the text when the trend style is "Bar or "3D bar".

Fixed (website only)

The mini trend will not move if the browser window is scrolled.

Back to list

Move horizontal



This animation is used to move an object horizontally from the design position to the offset position based on the "percent full scale" (PFS) of the selected analog point.

This description also applies to "Move Vertical", replace horizontal with vertical.

End offset

This value defines the maximum pixel amount the graphic element will move from the starting position. The source address is the percent of full scale for the point (0-100%). The movement is down (vertical) or to the right (horizontal) from the starting (design) position. This property does not apply when "Use raw value" is enabled.

Examples:

End offset = 200
PFS = 0% the graphic element will not move
PFS = 50% the graphic element will move 100 pixels
PFS = 100% the graphic element will move 200 pixels

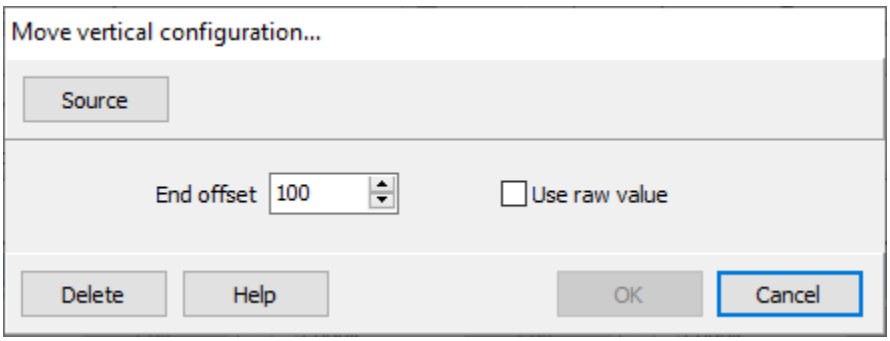
End offset = 300
PFS = 0% the graphic element will not move
PFS = 33% the graphic element will move 99 pixels
PFS = 100% the graphic element will move 300 pixels

Use raw value

This is the amount, in pixels, to move the graphic element. The points "process value analog" value is used. The value can be negative.
Negative values move the graphic element up/left from the starting (design) position.
Positive values move the graphic element down/right from the starting (design) position. Note: The value can move the graphic element out of bounds, will be off screen, not visible.

Back to list

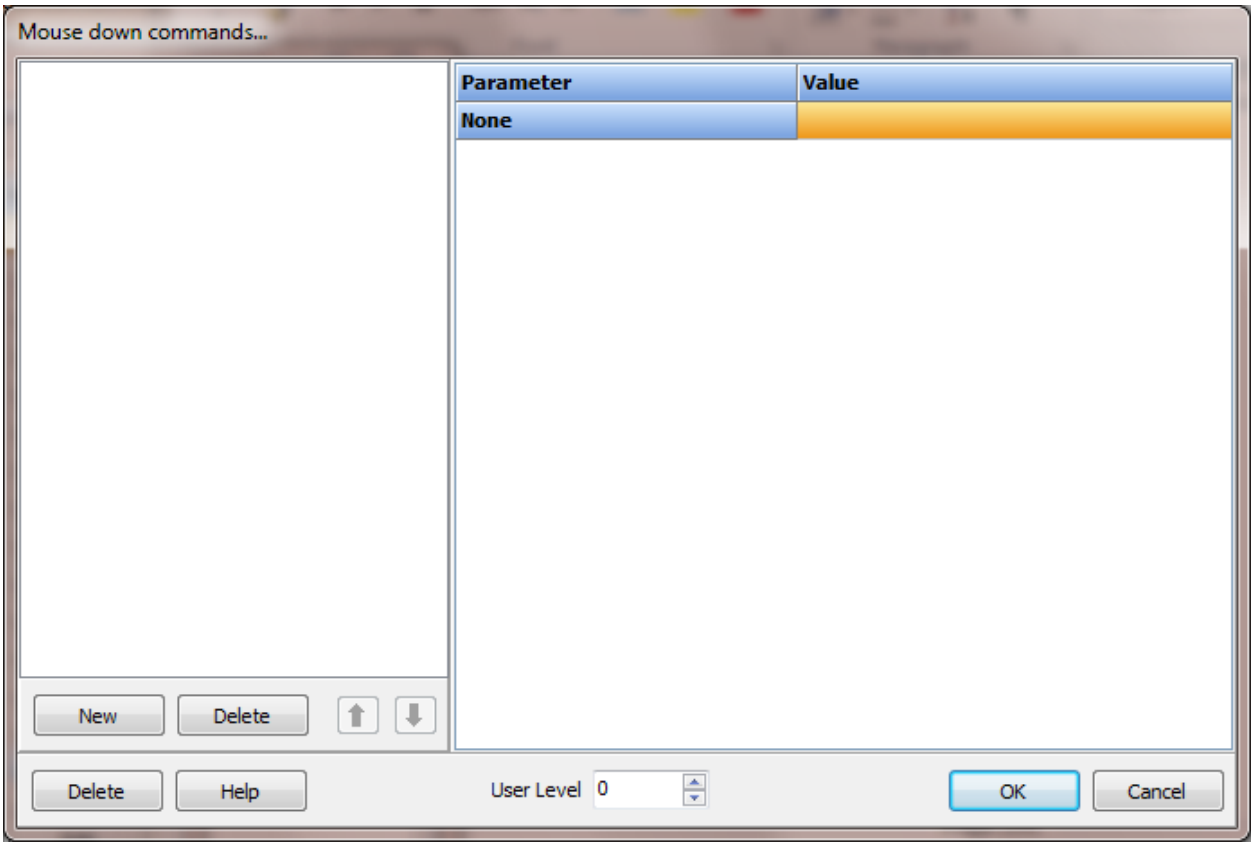
Move vertical



This is animation is used to move a graphic element vertically from the design position to the offset position based on the "percent full scale" (PFS) of the selected analog point. See "Move Horizontal" for more information

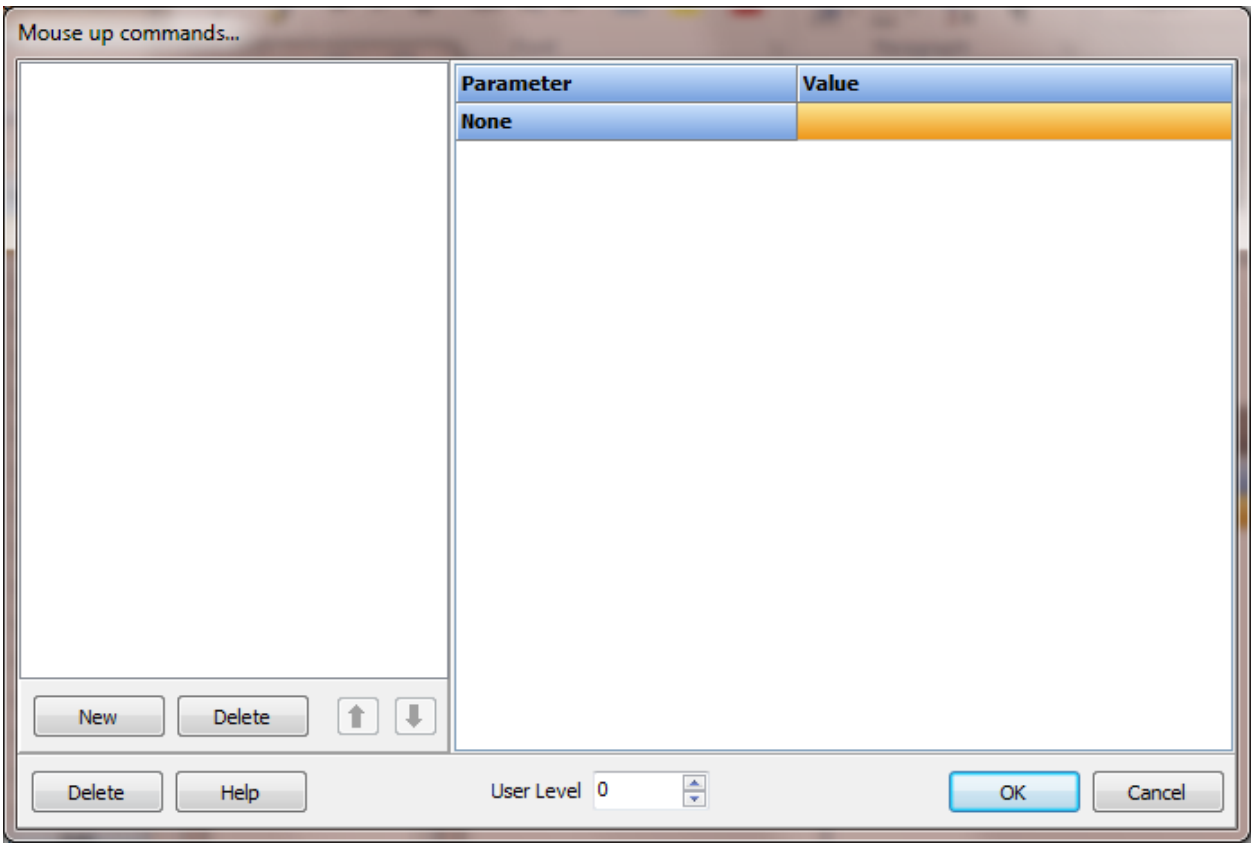
Back to list

On mouse down



When the left mouse button is pressed in a graphic element the configured actions are executed. See the section on "Mouse Commands" for a description of all mouse commands.

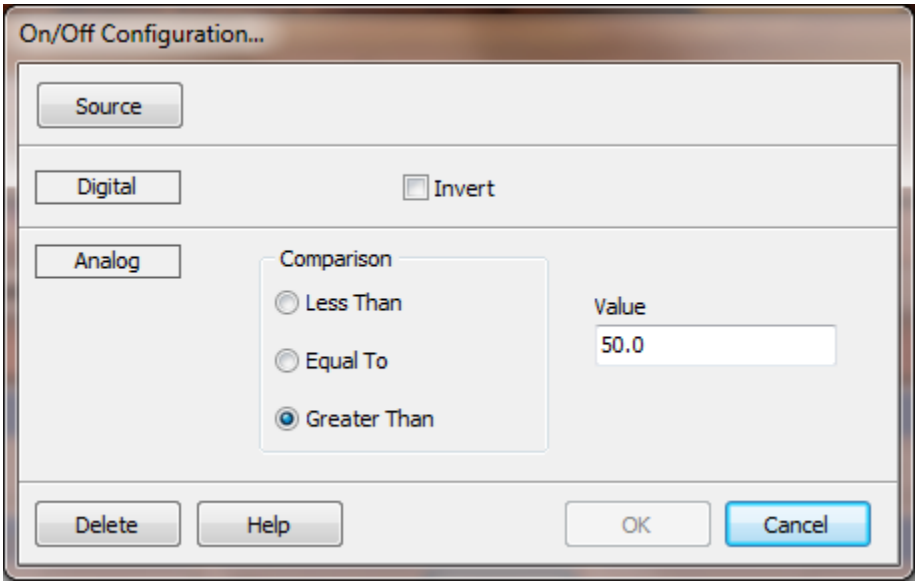
On mouse up



When the left mouse button is released (the same graphic element it was pressed in) in a graphic element the configured actions are executed. See the section on "Mouse Commands" for a description of all mouse commands.

Back to list

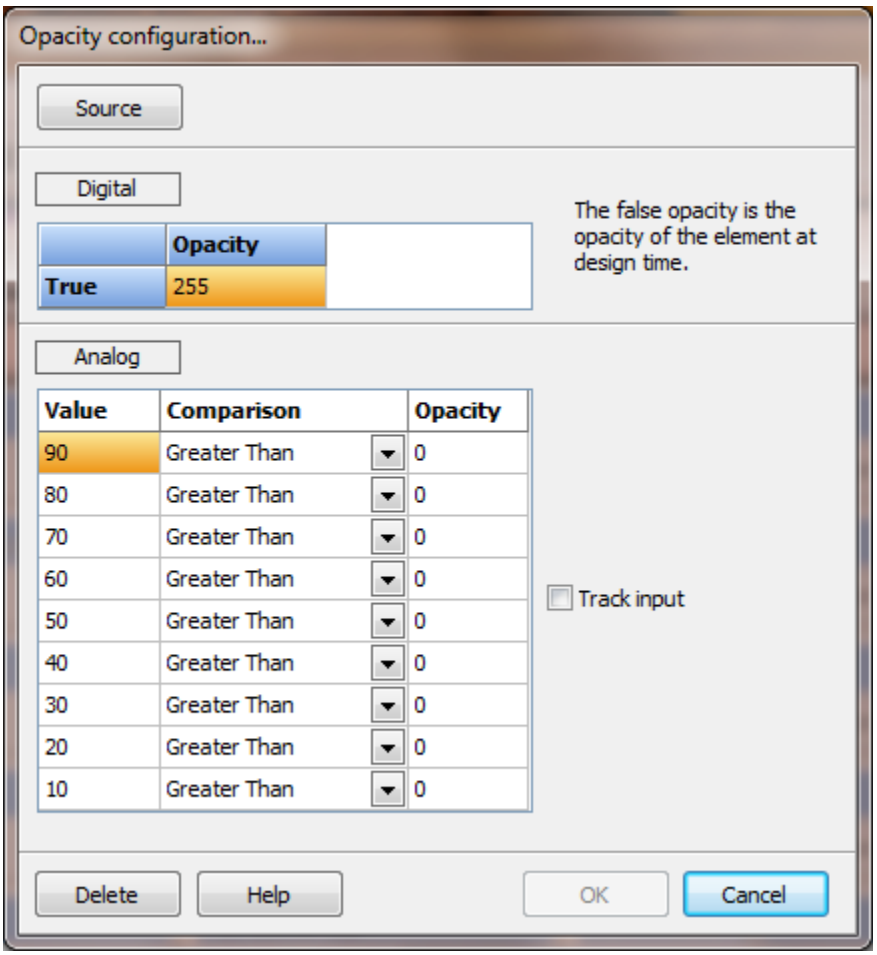
On/Off



This causes an object to flash visible and invisible.

Back to list

OPACITY

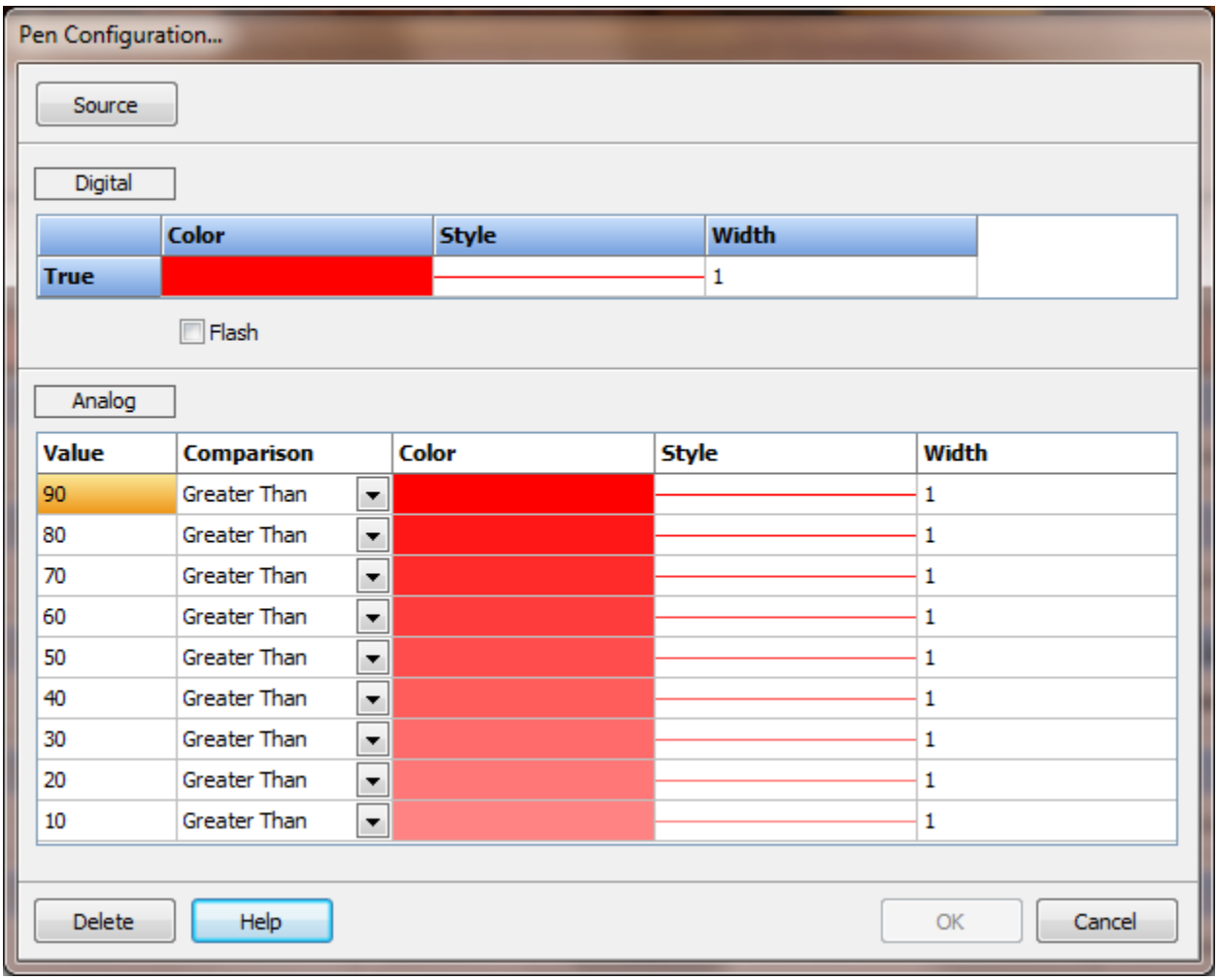


This animation is used to configure an object's opacity. Rectangles, round rectangles, circles, line polygons, free polygons and, text support opacity from 0 - 255. The "Track Input" checkbox on the analog panel is used to vary the opacity of the graphic element by the amount of the analog value. The comparisons are ignored if the checkbox is enabled.

The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

Back to list

PEN

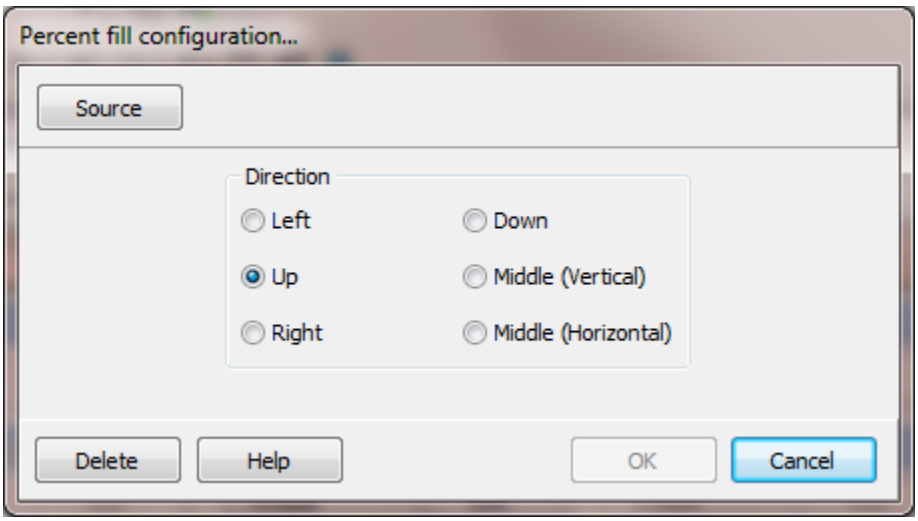


This is used to configure the objects pen attributes. The color, style and width are configurable. When the pen width is not 1 the style will not apply. When the source is digital the flash attribute can be enabled. When enabled and the source value is true the pen will flash between the design time and animation settings.

The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

Back to list

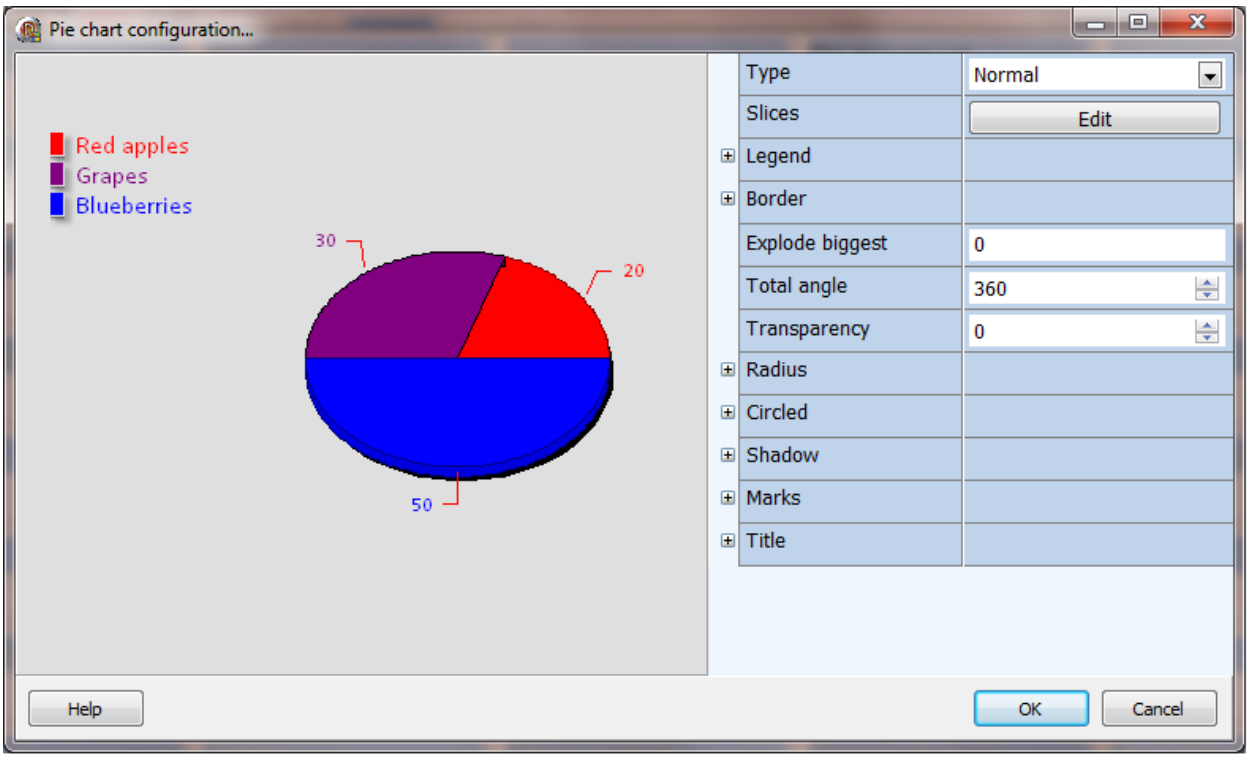
PERCENT FILL



The object will fill in the direction configured. The "Percent Full Scale" item is used.

Back to list

PIE CHART



The pie chart supports up to 10 slices. The slice data value can be connected to a point, script global, fixed value or calculated value based on pie slice values. The color of the pie chart panel is the foreground color.

Type

Normal

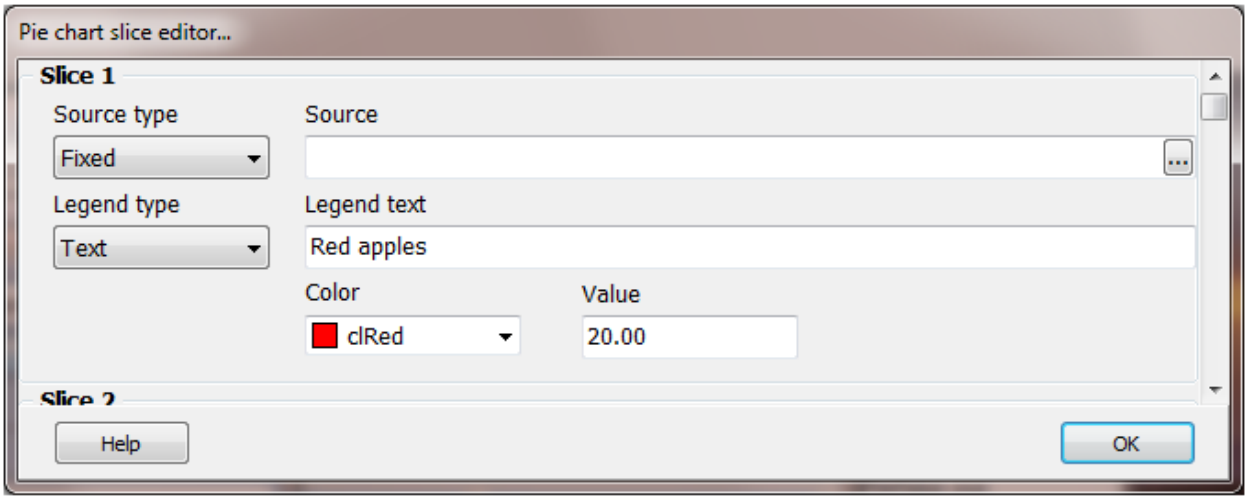
The pie chart uses the data values to determine the pie slice size.

Balance percent

All the pie slice values must be in the range of 0 – 100. The chart sums the values of the point, script global, and fixed values to determine the balanced point slice value. Balanced slice value = (100 – (sum of other slices))

Slices

The pie chart supports up to 10 slices.



Source type

The pie chart uses the data values to determine the pie slice size.

Disabled: The slice is not processed at runtime.
Point: The slice size is determined by the point value.
Script global: The slice size is determined by the script global value.
Balance point: The chart sums the values of the point, script global, and fixed value slices to determine the balance point slice value. The first slice configured as a balance point will be used and other slices configured as balance points are ignored.
Fixed: The slice size is determined by the value field.

Source

The source field is required if the source type is point or script global.

Legend type

Tagname: The legend text will be the tagname.
Point description: The legend text will be the tagname point description.
Text: The legend text will be the text entered.

Legend text

If the legend is visible and the legend type is “Text” this is the text for the legend. The legend text is also used at design time in the pie chart configuration dialog.

Color

This property defines the color of the pie slice.

Value

The value field is used for the “Fixed” type and for rendering the slices at design time in the pie chart configuration dialog.

Legend

Visible

If this property is true, the legend will be visible.

Font

This property defines the font properties of the legend text.

Font series color

If this property is true, the legend text for each slice will be the color of the slice.

Vertical spacing

This property defines the number of pixels between each slice legend.

Shadow

If this property is true, the legend border will display a drop shadow. The transparent property must be false for the drop shadow to appear.

Text style

Plain:	Legend text only
Left value:	Value and legend text
Right value:	Legend text and value
Left percent:	Value as percent and legend text
Right percent:	Legend text and value as percent
X value:	Slice index
Value:	Slice value
Percent:	Slice value as percent
X and value:	Slice index and value
X and percent:	Slice index and value as percent

Transparent

If this property is true, the legend panel will not be rendered.

Legend (Dividing lines)

Visible

If this property is true, a line dividing each slice legend will be rendered.

Style

This property defines the pen style of the diving line.

Width

This property defines the pen width of the dividing line.

Color

This property defines the pen color of the dividing line.

Legend (Position)

Position

This property defines the position of the legend on the pie chart panel.

Resize chart

If this property is true, the pie chart will be resized based on the legend size and position.

Margin

This property defines spacing between the pie chart and the legend.

Custom

If this property is true, the legend top and left will be positioned on the panel at the coordinates defined by the legend position left and top properties.

Legend (Symbols)

Visible

If this property is true, a symbol for each slice on the legend will be rendered.

Continuous

If this property is true, the symbols will not be divided with a space; the symbols will be connected.

Squared

If this property is true, the symbols will be square otherwise the symbols will be rectangular.

Width

This defines the width of the symbol in pixels or percentage. See the next property.

Width units

The symbol can be configured with a fixed width or as a percentage of the legend width. See the previous property.

Position

The symbol, if visible, can be placed on the left or right side of the legend text. See the "Text style" property.

Legend (Title)

Title

The legend can display a title. For no legend title, this property should be blank.

Position

This is the legend title position property, left, right or center.

Font

This property sets the font properties of the legend title text.

Color

This property defines the background color of the legend title. See the next property.

Transparent

If this property is true, the legend title text area will be transparent. See the previous property

Border

Visible

If this property is true, the pie chart panel border will be rendered.

Width

This property defines the pie chart panel border width.

Color

This property defines the pie chart panel border color.

Explode biggest

If this property is true, the pie chart will separate the largest slice from the pie chart.

Total angle

This property defines the total angle (0-359), the pie chart will use to render the slices.

Transparency

This property defines the transparency value of the pie chart. Note: The “3 Dimensions” property must be true for transparency to be applicable.

Radius (Vertical/Horizontal)

If the property is not 0 (zero) the pie chart will be rendered with the value entered.

Circled

Circled

If this property is true, the pie chart will be rendered as a circle. Otherwise the pie chart will use the height and width of the panel to render the pie chart.

3 Dimensions

If this property is true, the pie chart is rendered using 3 dimensions(X, Y and Z).

Rotation

The first slice is rendered at 0 (zero) degrees. The zero degree position is the middle top of the pie chart (12 o’clock). All other slices are rendered moving clockwise in the pie chart.

Shadow

Visible

If this property is true, a shadow is drawn around the pie chart using the color property. See the next property.

Color

This property defines the shadow color. See the previous property.

Marks

"Marks" consist of the line and text rendered for each pie slice.

Visible

If this property is true, the "marks" will be rendered.

Font series color

If this property is true, mark text is rendered using the pie slice color.

Style

Value:	Slice value
Percent:	Slice value as percent
Label:	Legend text only
Label and percent:	**L**egend text and percent
Label and value:	**L**egend text and value
Legend:	Duplicates the legend setting
Percent total:	Percent and total value
Label and percent total:	Legend text, percent and total value
X value:	Slice index
X and Y value:	Slice index and value
Series title:	Pie chart
Point index:	Slice index
Percent relative:	Relative percent

Color

This property defines the background color. See next property.

Transparent

If this property is true, the mark text area will be transparent. See the previous property.

Callout leg size

This property defines the line length that connects to the mark text.

Multi line

If this property is true, the mark text may be rendered on multiple lines.

Font

This property sets the font properties of the mark text.

Line

Visible

If this property is true, the line from the pie chart to the mark text will be visible.

Color

This property defines the mark line color.

Width

This property defines the mark line width.

Distance

This property defines the distance from the pie chart to the start of the line.

Title

Title

This property defines the chart title. For no chart title, this property should be blank.

Color

This property defines the background color of the title. See next property.

Transparent

If this property is true, the chart title area will be transparent. See the previous property.

Font

This property defines the font properties of the chart title.

Alignment

This property defines horizontal alignment of the chart title text.

Shadow visible

If this property is true, and the chart title is not transparent, a shadow is drawn around the pie chart title.

Border

Visible

If this property is true, the chart title border will be rendered.

Color

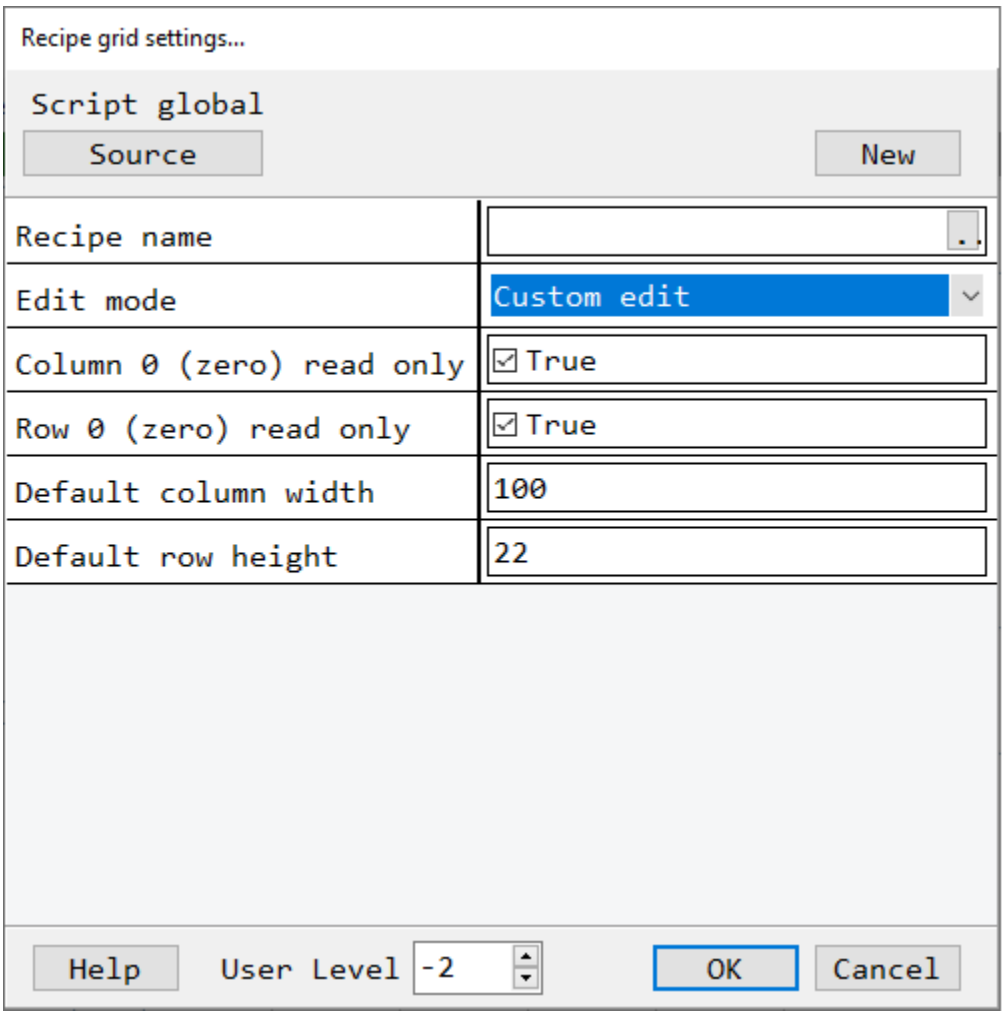
This property defines the chart title border color. See next property.

Width

This property defines the chart title border line width. See the previous property.

Back to animations list
Back to complex objects list

Recipe grid



The “Recipe grid” can be used to view a recipe and if needed edit the recipe ingredient values. Recipes are loaded to memory at runtime start. This grid can be used to alter the recipe in memory.

Script global	A script global section is used to control and monitor the recipe grid at runtime. For example, changing the recipe name at runtime and loading the grid allows for one recipe grid graphic element to be used for multiple grids.
Recipe name	The recipe name. This is provided for easy name change. At runtime the recipe specified in the “script global” section is used.

Edit mode

This provides for three edit modes.

Read only
The grid cannot be used to alter the recipe.

Edit validation
The grid in-place editor is used. When the user ends the edit a script function, "OnRecipeValidate" is called.

Custom edit
When the user presses the left mouse button in a grid cell a script function, "OnRecipeClick" is called.

The script must be in the graphic element and the script enabled animation set true.

Column/Row 0 (zero) read only

ODBC column/row zero should **never** be altered at runtime. If either is altered it may cause failure. For Excel, both can be altered, but not advised.

It is advised not to alter column zero or row zero for ODBC or Excel.

Default column width

Note: The column width can be set for each column or a subset of the columns. Add script global items to the source script global to set a column width override value for a column.

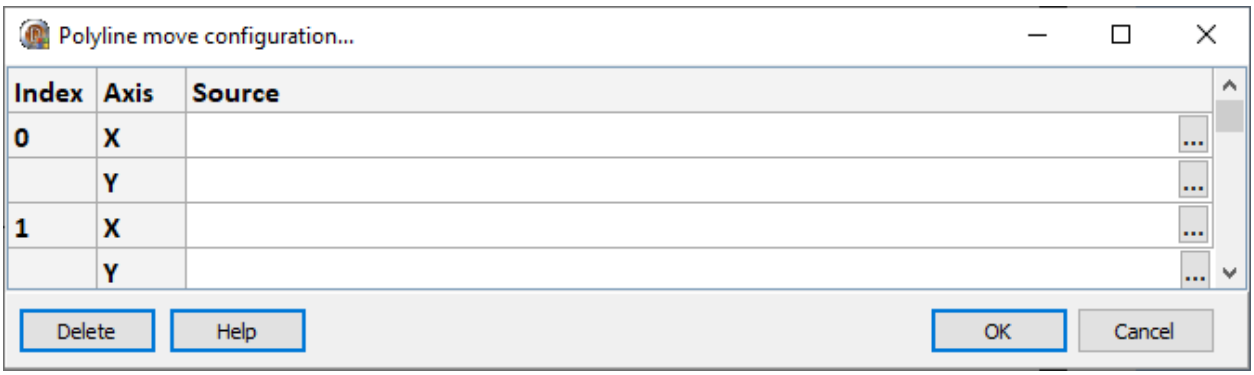
For example, to set column 0 and column 3 to 80 pixels add two items, "ColWidth0" with a value of "80" and "ColWidth3" with a value of "80". If an item named "ColWidthX" is not found the default column width will be applied.

Default row height

The row height.

Back to animations list
Back to complex objects list

POLYLINE MOVE



Each point of the polyline can be configured for an X-axis (horizontal), Y-axis (vertical) or not configured. The "process variable analog" (5000) property is the data source. The value is the number of pixels, positive for down/right, negative for up/left, to move the point from the current point location.

Notes:

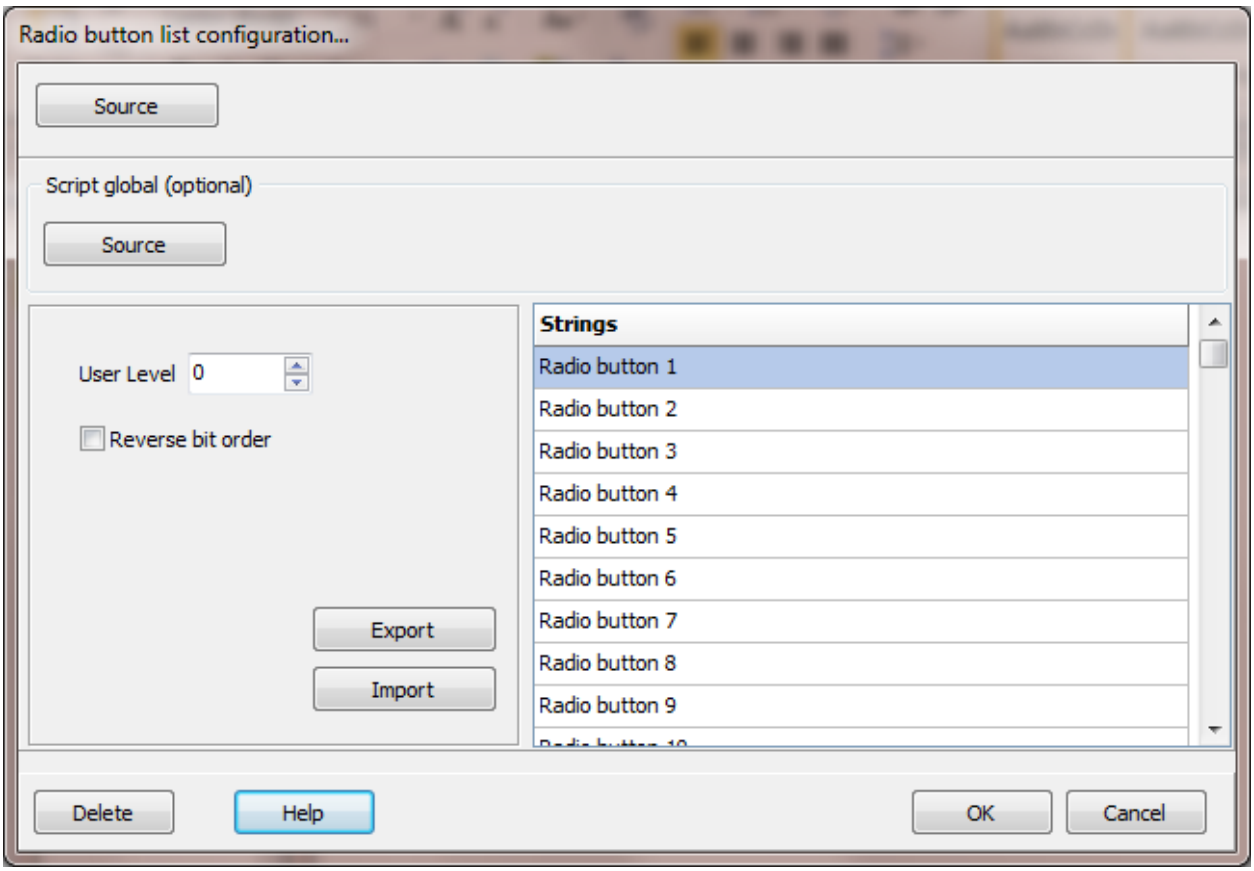
1) The value can move the graphic element out of bounds, will be off screen, not visible.
2) The other positioning animations are processed before this animation and may change the position of the complete polyline.

Source

The source must be an analog type tag.

Back to list

RADIO BUTTON LIST



Source

The source must be an analog type tag.

String source (optional)

The radio button strings can be configured via the static editor at configuration or at runtime the radio button strings can be loaded from the selected script global section when the window is opened. The section is selected at configuration.
The names in the section must be 1 - 16. <section name>.1, <section name>.2 ... <section name>.16.

When a script global is used:
1. The 'Row count' above is ignored. The number of rows is determined by the highest number name in the section. (16 maximum)
2. A missing name does not delete a radio button. The highest number in the section determines the row count.
3. If the bit order cannot match the radio button order, use an 'Analog Host' point as the source and call a script in the 'On Mouse Up' animation to do any needed processing of the bit order. The 'On Mouse Up' animation is called after the source point is updated.

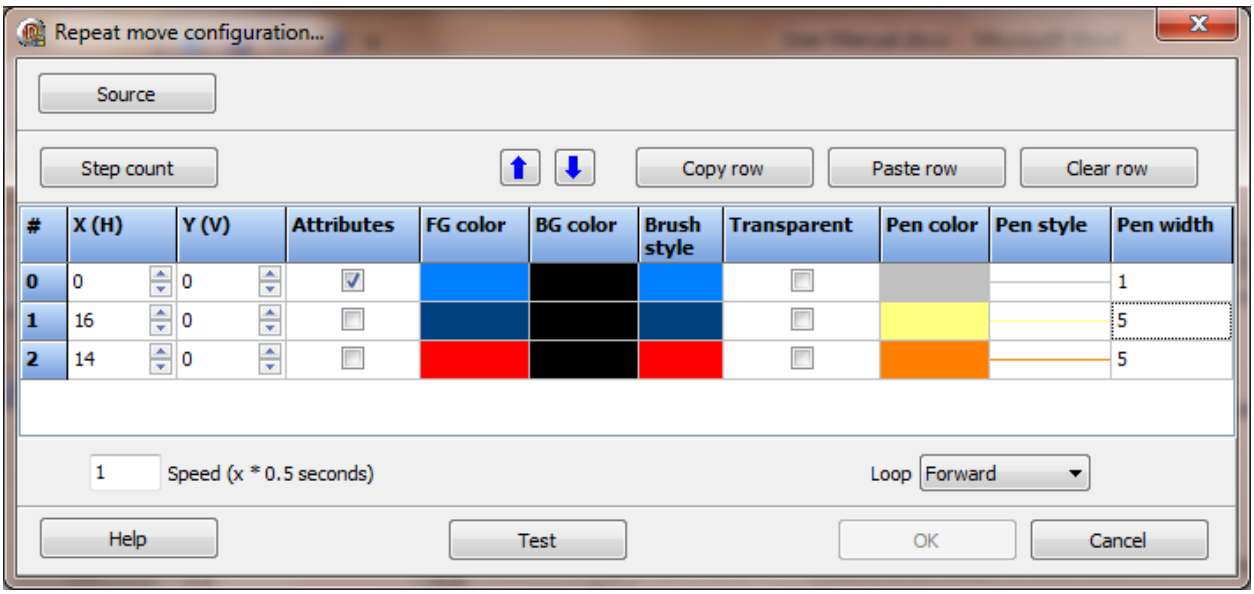
Warning: If the source point is located in an external device, the data might not have been written to the device before the 'On Mouse Up' animation is called.

Reverse bit order

The normal bit order is left to right, 15, 14, 13, etc.. Radio button 1 is the top.
Radio button 1 = bit 0
Radio button 2 = bit 1
Radio button 3 = bit 2
...
If this attribute is enabled the bit order is reversed. From left to right, 0,1,2,3, etc.
Radio button 1 = bit 15
Radio button 2 = bit 14
Radio button 3 = bit 13

Back to list

REPEATED MOVE



The animation is used to move an element on the screen in programmed steps.

Source

The source must be a digital value.

Step count

The number of steps the graphic element moves before it reaches the last step and either reverses or repeats. See loop below.

X (H), Y (V)

The horizontal and vertical amount to move the element from the last position. Positive values move the element to the right and down. Negative values move the element to the left and up.

Attributes

When this checkbox is enabled the color, style, etc. attributes are applied to the element when the step number is equal to the row number. Otherwise, the design time attributes are applied to the graphic element.

Speed

This is the rate the steps progress. The fastest speed is a step every 0.5 seconds. 1 = 0.5, 2 = 1.0, 3 = 1.5, etc.

Loop

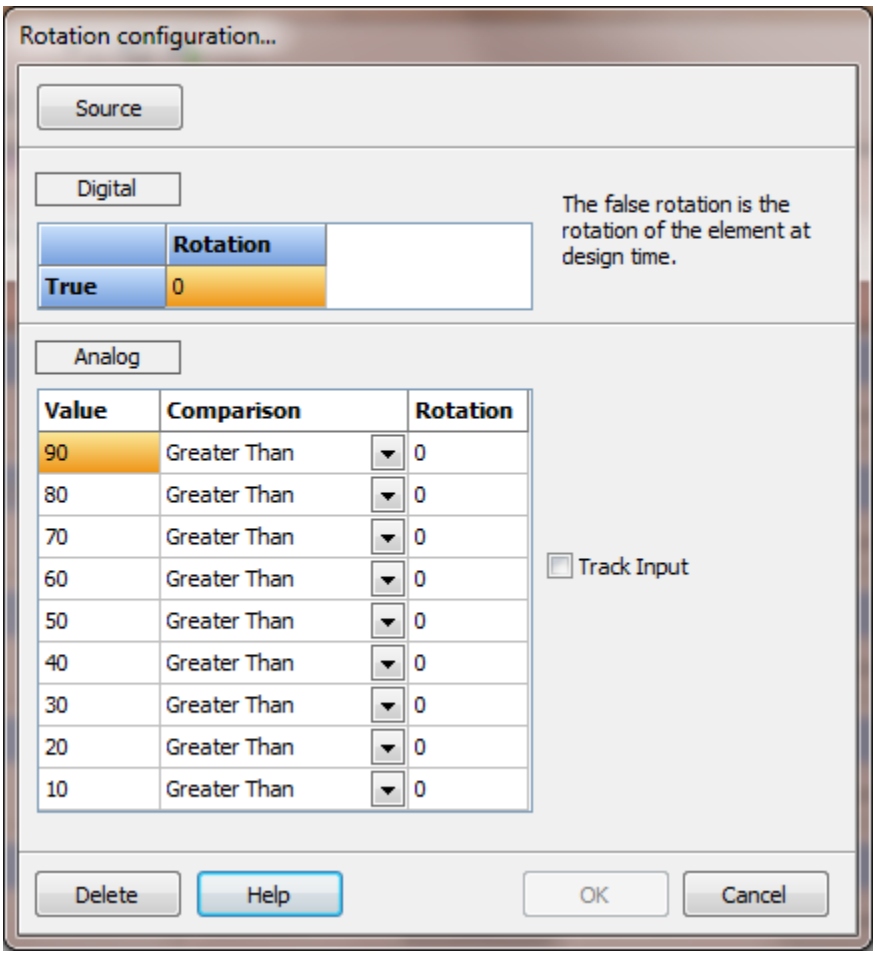
When the progression reaches the last step this attribute determines the next action.

Forward = the element returns to the design time position and repeats the steps. Looping, repeating.

Reverse = the element moves along the steps in reverse order. Reverses, forward, backward, forward, backward...

Back to list

ROTATION

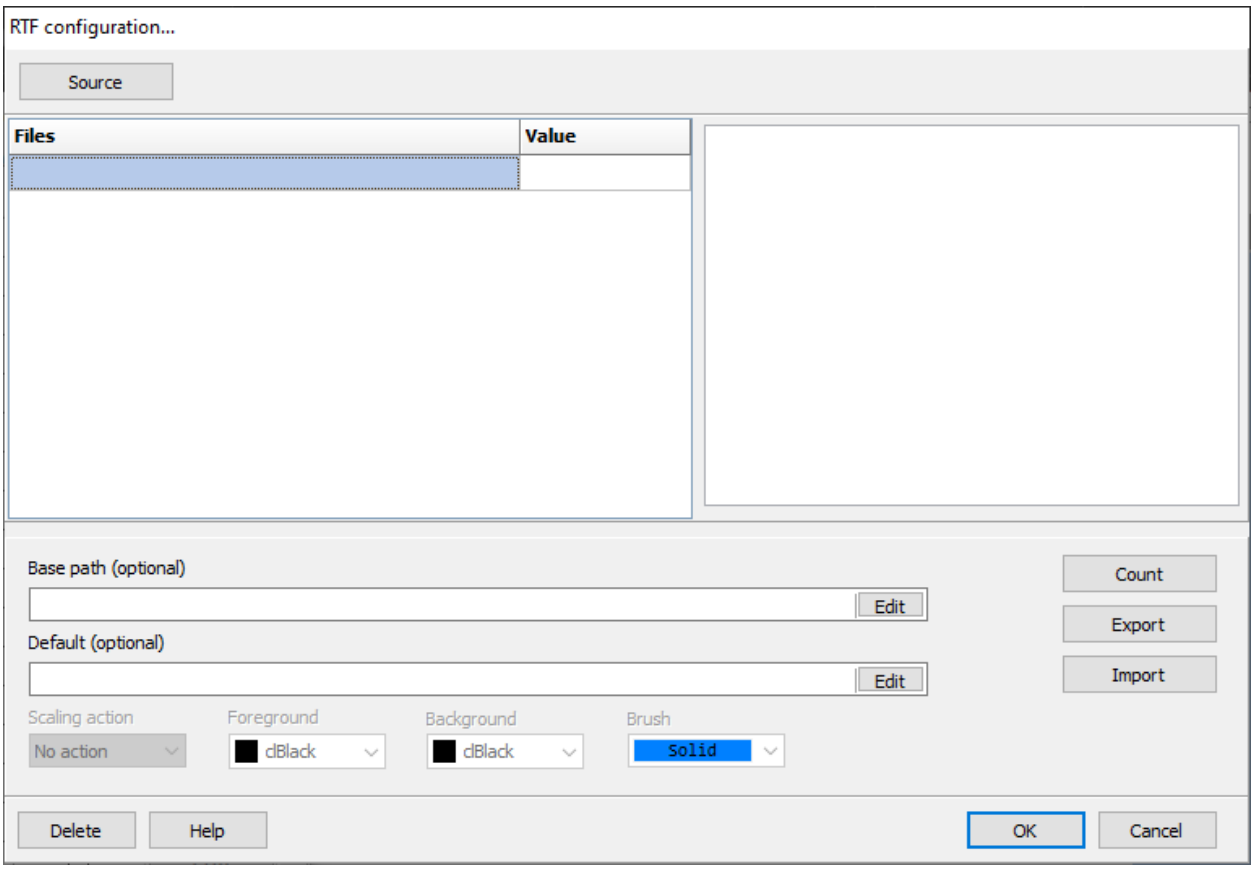


This is used to configure an objects rotation. Text, polygons and bitmaps can be rotated. For a rectangle or line to be rotated, make a polygon of the same dimensions and rotate it. The "Track Input" checkbox on the analog panel is used to rotate an object, clockwise, the amount of the analog value. The comparisons are ignored if the checkbox is enabled.

The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

Back to list

RTF



Source

The source point.item must be an analog type. The user entered value is stored in the source.

Files

These are the files containing the images to display. The file must be a fully qualified path. See the "base" below to use a prefix.

Values

The value of "source" is compared to the values in the list. When a match is found the value of the corresponding file is displayed. If a value match is not located, the default RTF file is displayed. If the default RTF file is not configured or does not does not exist, an RTF file is not displayed

Base (optional)

This is an optional base path for the location of the RTF files.
Each file could be a full path:

Files	**Value**
C:\images\project 1\A.rtf	1
C:\images\project 1\B. rtf	34
C:\images\project 1\C. rtf	45

Or the base parameter could be set to "C:\images\project 1\" and then the file entries could be:

Files	**Value**
A. rtf	1
B. rtf	34
C. rtf	45

Default (optional)

This is a path to the RTF file to display if the source value does not match a configured value. If this field is blank and a value does not match, an image will not be displayed.

Count

This is the number of files/values in the list. (1 - 32767)

Export

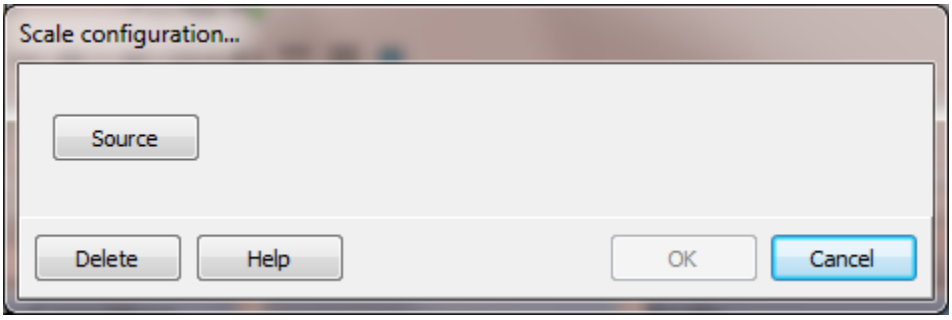
This exports the string grid as a two column comma separated file.

Import

This will import a two column comma separated file to the string grid.

Back to list

Scale



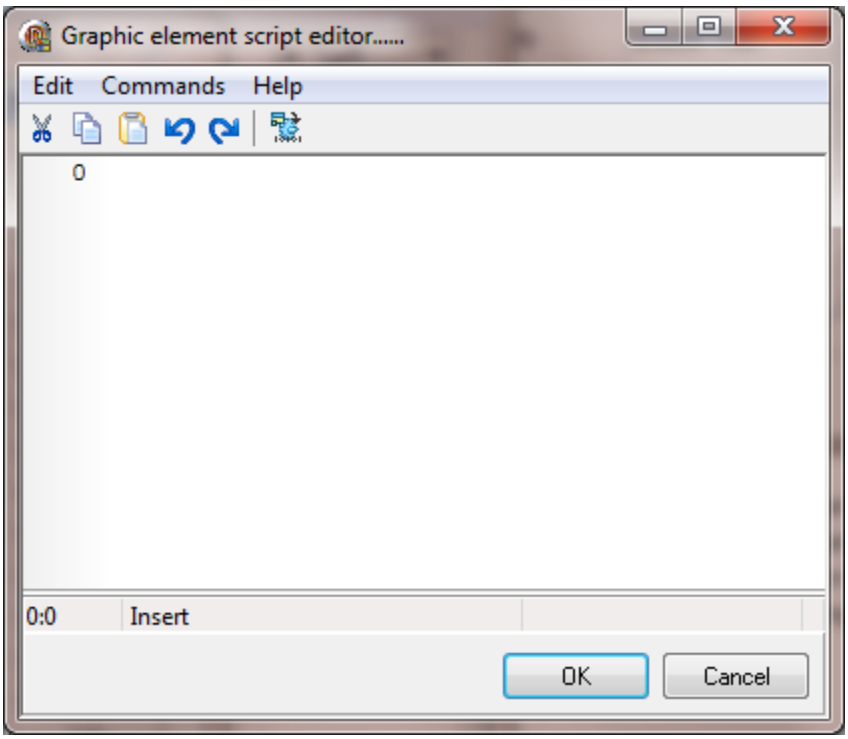
This animation is used to configure a scale graphic element.

The animation is needed for only two reasons.

#1 The maximum and minimum settings are collected from an analog point.
#2 One of the other animations, e.g. pen color, foreground color, etc., is required. If this animation is not enabled the scale is considered "static" and is rendered using the design time settings. If one of the other animations is required and a point does not exist to provide the minimum and maximum values, create a dummy analog host point.

Back to list

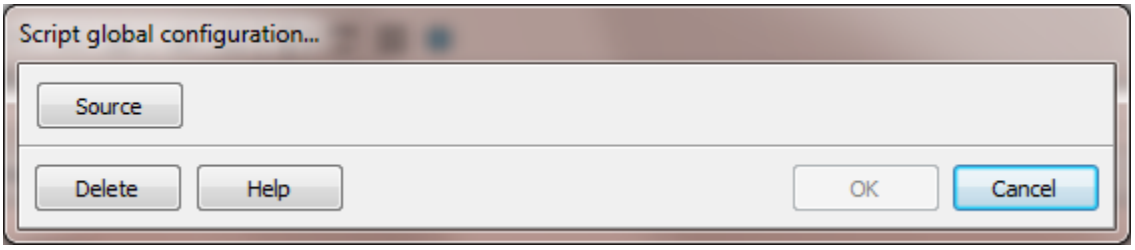
Script



Scripting is covered in the scripting section. Graphic element scripting should only be used if using the animations will not accomplish the design goal. The built-in animations are much faster for the program to execute.

Back to list

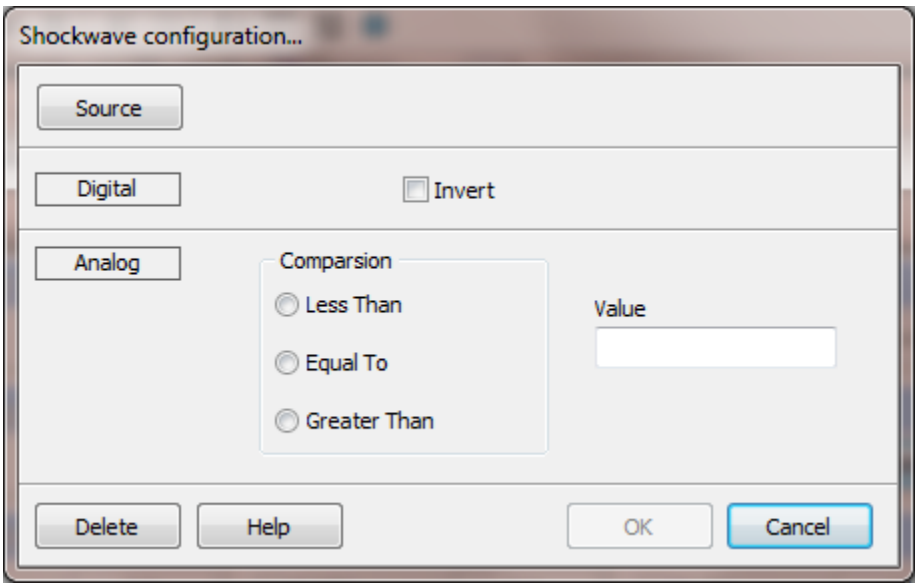
Script global



This provides a method to display the value of a script global.

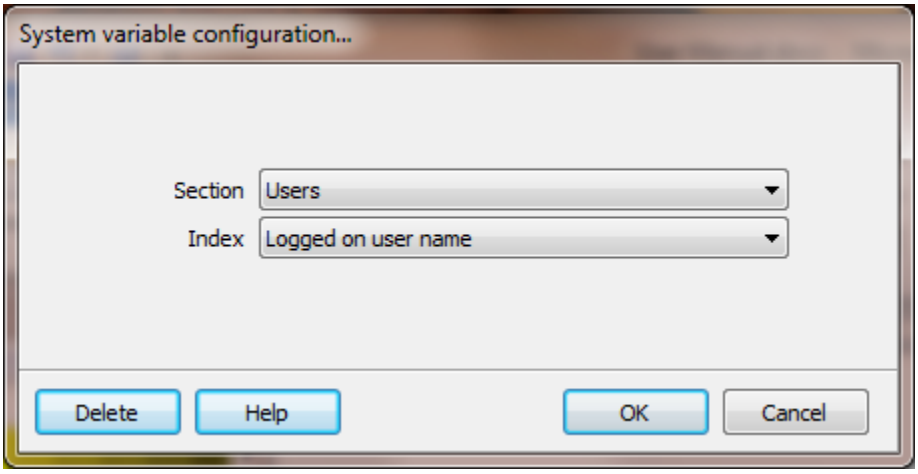
Back to list

Shockwave flash



Back to list

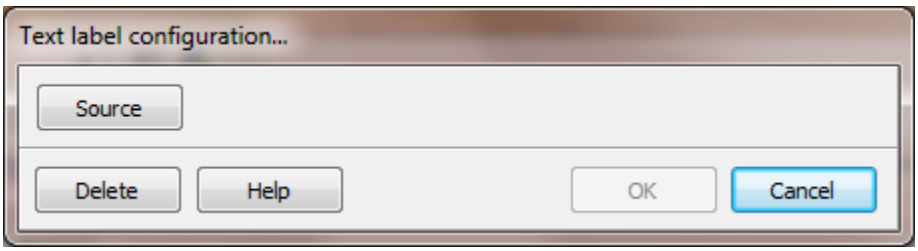
System variable



This provides a method to show various internal values.
i.e. alarm counts, event counts, the last 1-15 alarms or events.

Back to list

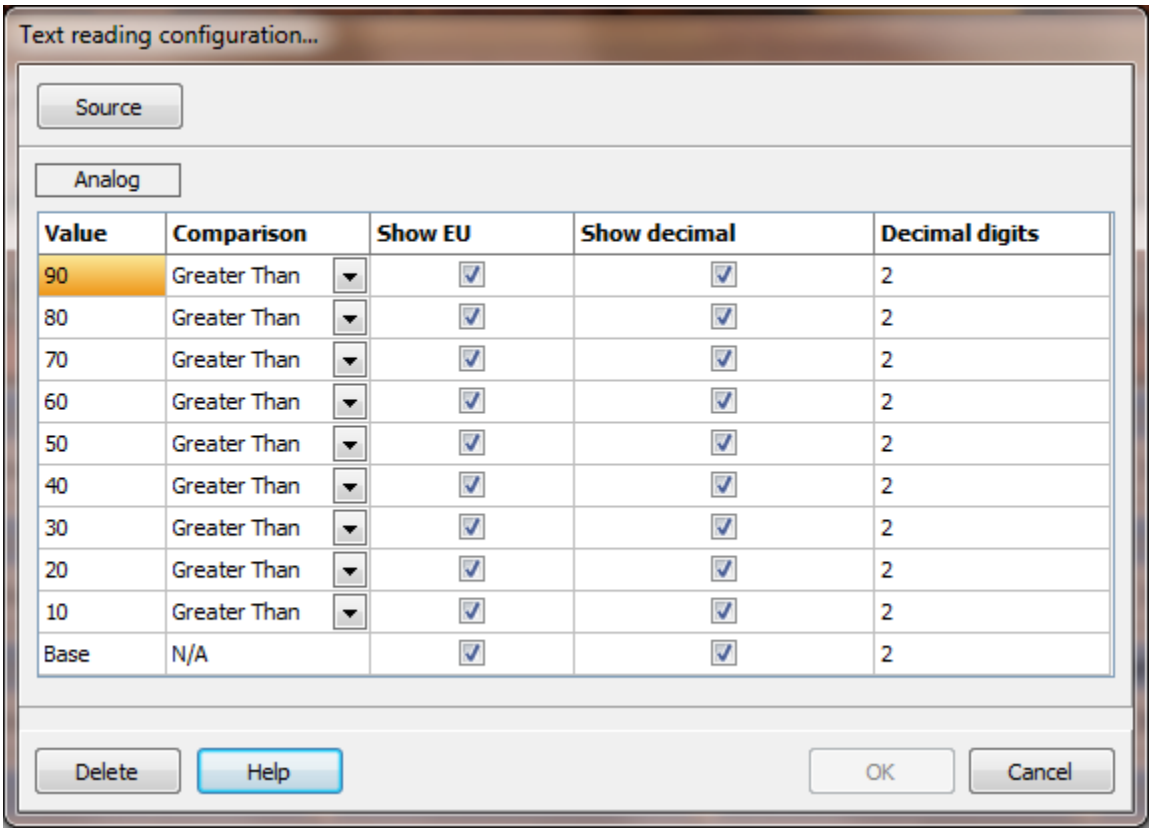
TEXT LABEL



This provides a method to show the text value of a point configuration item or other text source.

Back to list

TEXT READING

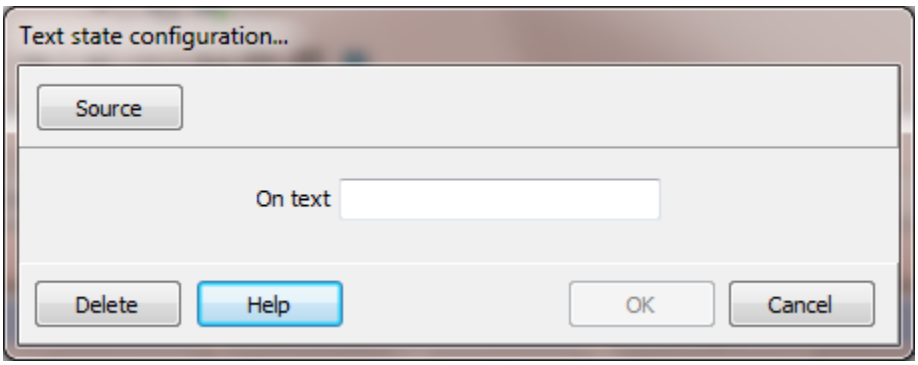


This is used to configure a text object to display the value of the item monitored.

To display large numbers, as whole numbers, select the show decimal point checkbox and set the decimal point count to -1. The floating point value will be trunc'd and displayed.

To display the value, regardless of the value, set the first row comparison to N/A. The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

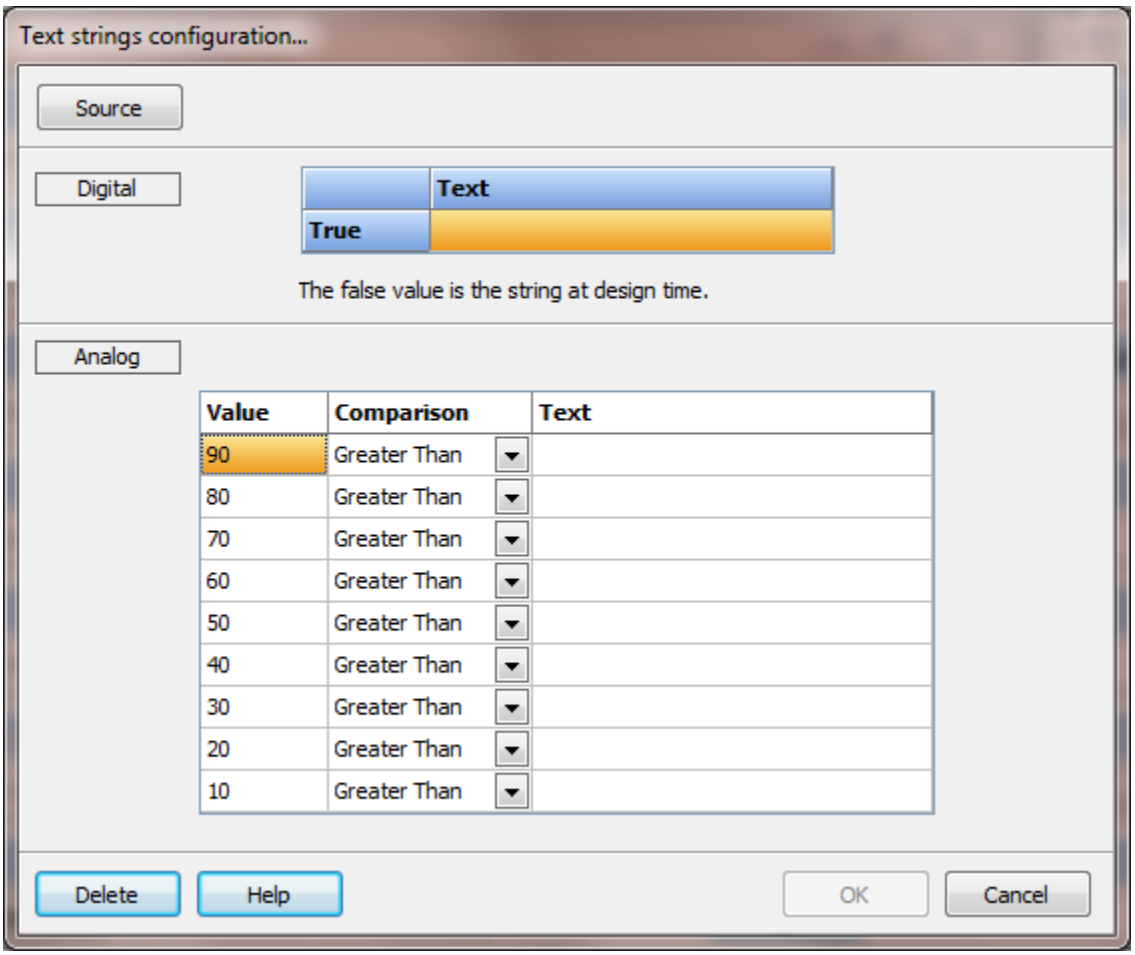
TEXT STATE



This animation is used to configure a text object to display a string based on the value of a monitored digital item. When the item is false, the configuration time text is displayed. When the monitored item is true, the “On text” text is displayed.

Back to list

TEXT STRING

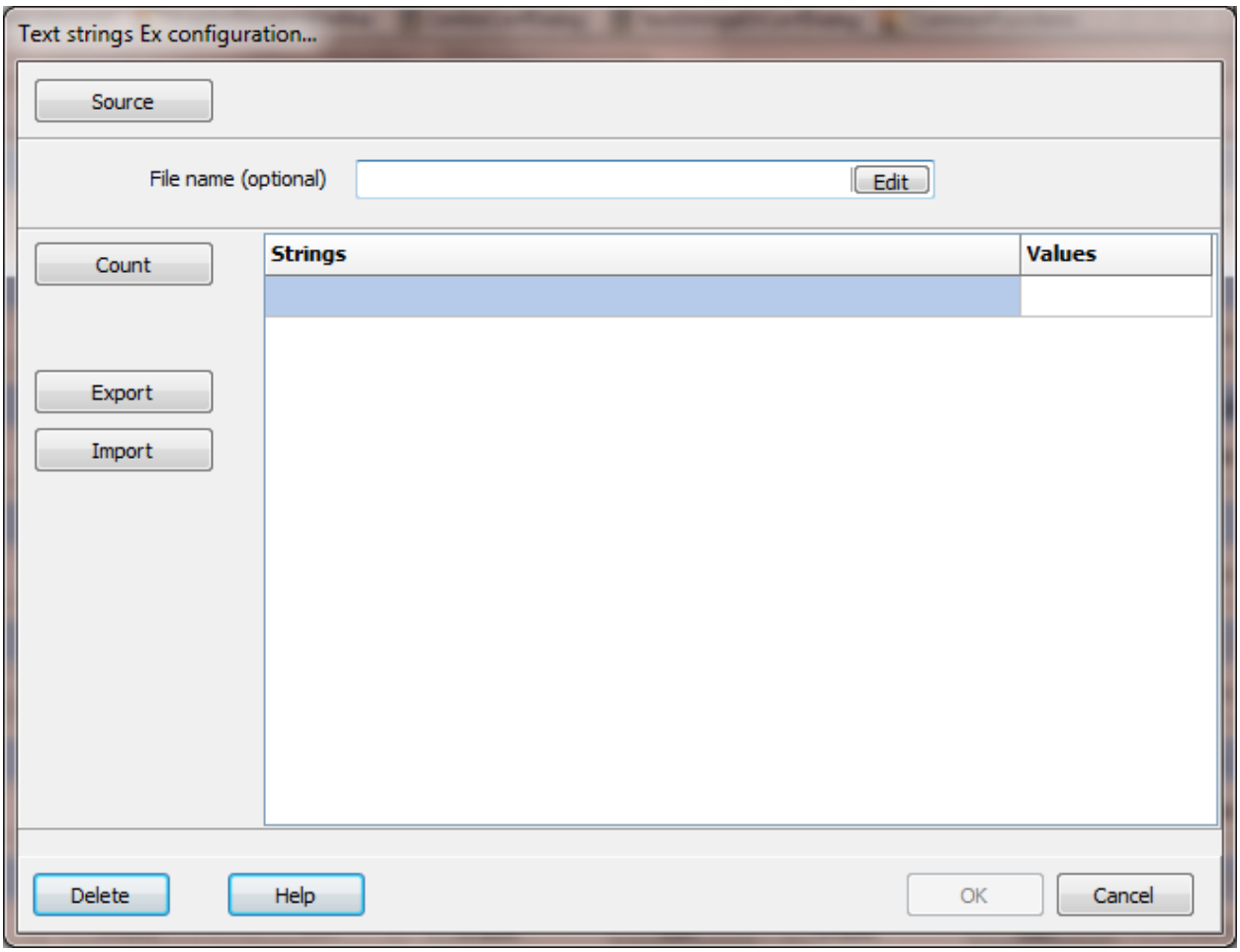


This is used to configure a text object to display a string based on the value of the monitored item.

The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

Back to list

TEXT STRING EX



This is used to configure a text object to display a string based on the value of the monitored item. Also see: Text String and TextState. At runtime if a match for the monitored item and a value in the list is not found the text will not be altered.

Source

The source point must be an analog type. The user entered value is stored in the source.

File name (optional)

This is the name of a two column file in the project directory that is used to populate the string/values list.

Notes:

1) If this field is blank, the "strings" property is used.

2) If this field is not blank and the file is missing/invalid/corrupt, the "Strings" property is used.

3) The text file must be two columns, comma separated. No header, no extra data.

Count

This is the number of strings/values in the list. (1 - 32767)

Export

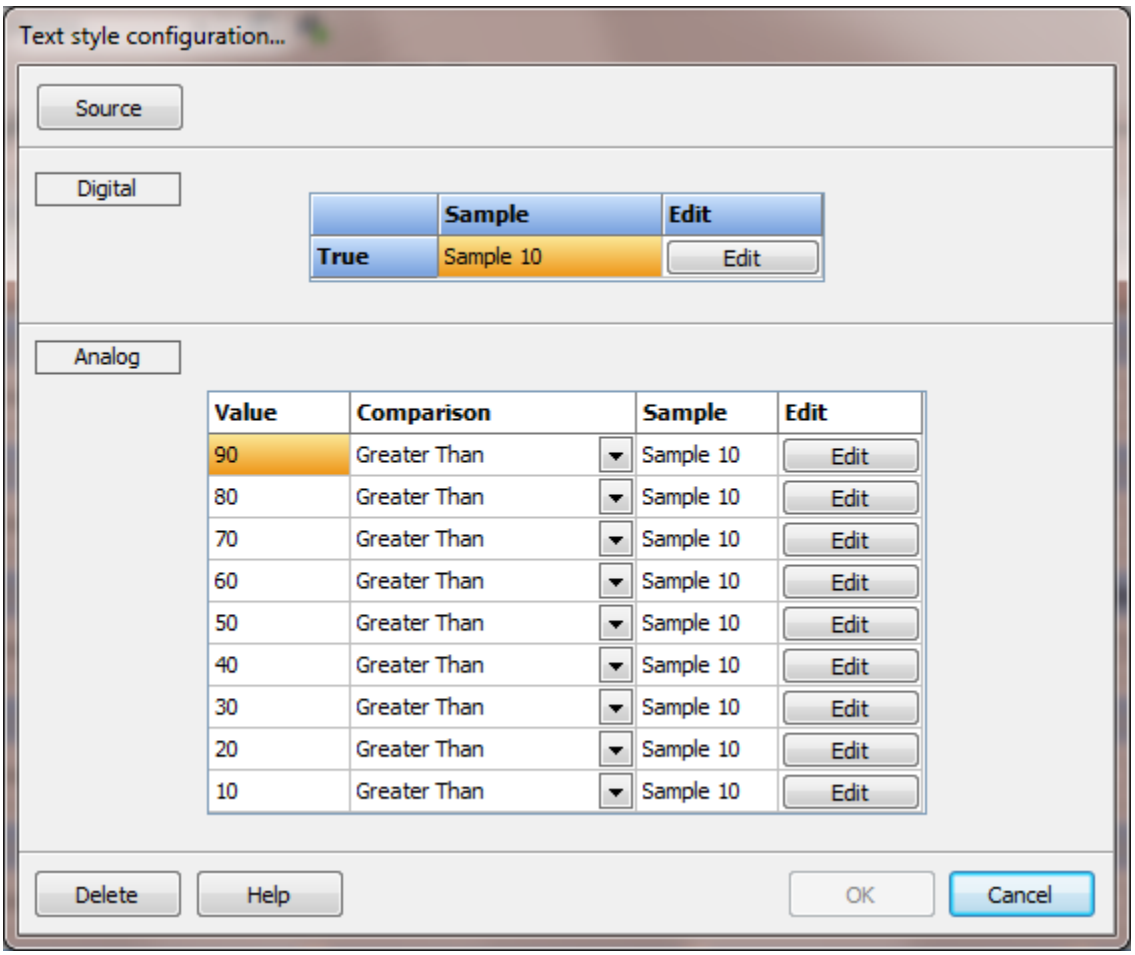
This exports the string grid as a two column comma separated file.

Import

This will import a two column comma separated file to the string grid.

Back to list

TEXT STYLE

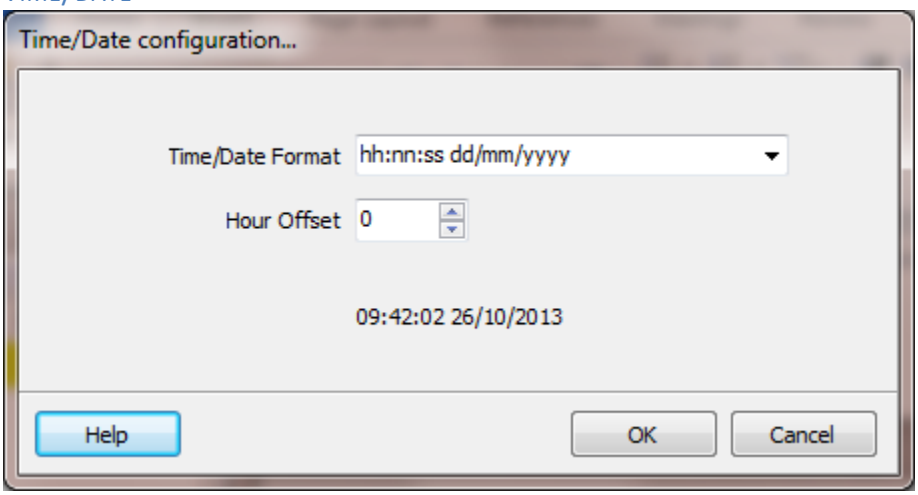


This is used to configure a text object attributes. Text style, font, size and other attributes can be configured. The text color is configured via the brush animation. The color combo box on the text edit dialog is ignored.

The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

Back to list

TIME/DATE



Mask

c	Displays the date using the format given by the OS "Short Date",followed by the time using the format given by the OS "Long Time". The time is not displayed if the date-time value indicates midnight precisely.
d	Displays the day as a number without a leading zero (1-31).
dd	Displays the day as a number with a leading zero (01-31).
ddd	Displays the day as an abbreviation (Sun-Sat).
dddd	Displays the day as a full name (Sunday-Saturday).
g	Displays the period/era as an abbreviation (Japanese and Taiwanese locales only).
gg	Displays the period/era as a full name.
e	Displays the year in the current period/era as a number without a leading zero (Japanese, Korean and Taiwanese locales only).
ee	Displays the year in the current period/era as a number with
a	leading zero (Japanese, Korean and Taiwanese locales only).
m	Displays the month as a number without a leading zero (1-12). If the m specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.
mm	Displays the month as a number with a leading zero (01-12). If the mm specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.
yy	Displays the year as a two-digit number (00-99).
yyyy	Displays the year as a four-digit number (0000-9999).
h	Displays the hour without a leading zero (0-23).
hh	Displays the hour with a leading zero (00-23).
n	Displays the minute without a leading zero (0-59).
nn	Displays the minute with a leading zero (00-59).
s	Displays the second without a leading zero (0-59).
ss	Displays the second with a leading zero (00-59).
z	Displays the millisecond without a leading zero (0-999).
zzz	Displays the millisecond with a leading zero (000-999).
am/pm	Uses the 12-hour clock for the preceding h or hh specifier, and

displays 'am' for any hour before noon, and 'pm' for any hour after noon. The am/pm specifier can use lower, upper, or mixed case, and the result is displayed accordingly.

a/p Uses the 12-hour clock for the preceding h or hh specifier, and displays 'a' for any hour before noon, and 'p' for any hour after noon. The a/p specifier can use lower, upper, or mixed case, and the result is displayed accordingly.

'xx' Characters enclosed in single or double quotes are displayed as-is, "xx" and do not affect formatting.

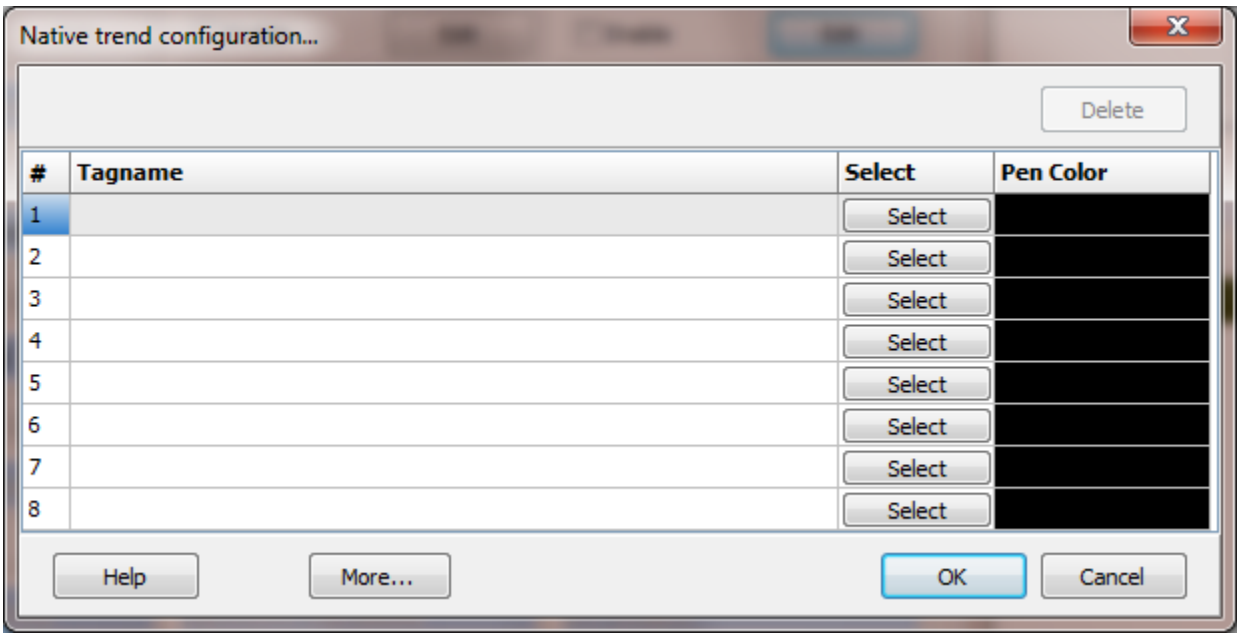
Format specifiers may be written in upper case as well as in lower case letters--both produce the same result.

Hour offset

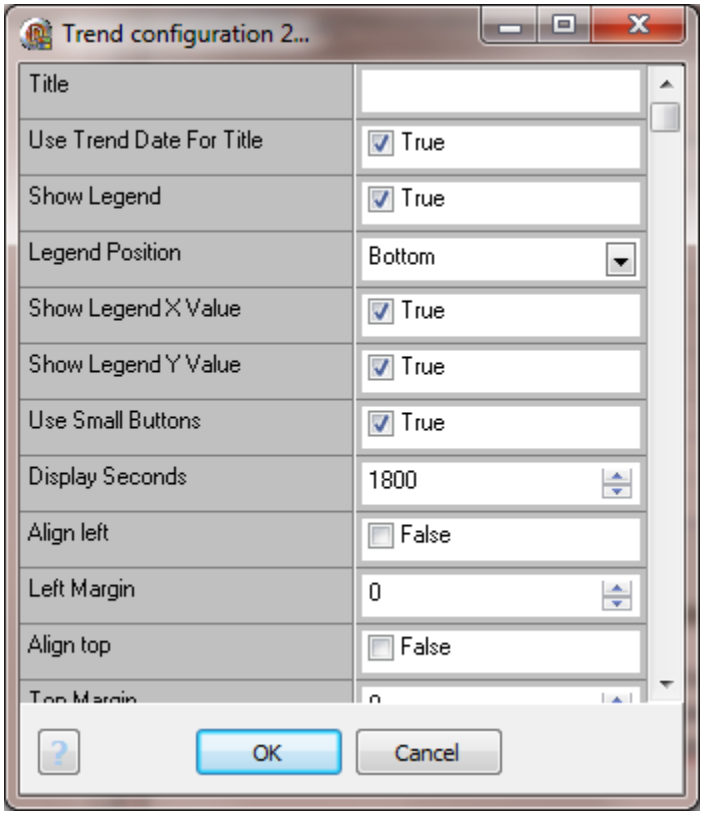
The offset from the PC clock, in hours.

Back to list

NATIVE TREND



Each trend can display a maximum of thirty-two (32) pens.
Select the tagname to trend and select the pen color.
See "Trend (static)" if using a trend (static) graphic element.
To view past days trends, use the ViewTrendHistory mouse command.



Title

This is a string displayed at the top of the trend. It can be blank.

Use trend date for title

If enabled the title above the trend will be the date of the trend currently displayed. If enabled the title attribute is ignored.

Show legend

The legend displays the pen tagname and if enabled the X and Y values. If enabled, the legend will be displayed according to the settings.

Legend position

The legend can be displayed on the bottom, left, right or top area of the trend element.

Show legend X value

If the legend is enabled and this attribute is enabled the last "X" axis trend value for the pen will be displayed after the tagname in the legend.

Show legend Y value

If the legend is enabled and this attribute is enabled the last "Y" axis trend value for the pen will be displayed after the tagname in the legend.

Use small buttons

The buttons for runtime trend viewing adjustment will be displayed in with small buttons.

Runtime user level

When the window containing the trend is opened, the logged on user “user level” is compared against this value. If the logged on user level is less than this value, the user will not be able to edit the trend runtime properties. (The toolbar “Properties” button and popup menu “Edit” item will not be visible.)

Display seconds

The trend can be zoomed to many levels. This attribute is the span of the "X" axis when the trend is initially displayed. The default value is 1800 seconds (30 minutes). The limits are 60 to 86,400 seconds (1 minute to 1 day)

Align left, top, right, bottom

The trend element can be anchored (locked) to the any or all four sides of the window.

Left, top, right, bottom margin

If the align attribute is set this defines the amount of space (in pixels) to offset the selected edge of the trend.
Example:
The "align left" attribute is enabled.
The "left margin" is 100.
When the trend is displayed in the window the left edge of the trend will be 100 pixels from the left edge of the window.

Y axes decimal points

The number of digits shown after the decimal point. (0-16)

Filename

If this field is not empty, it is the file name (without extension) to store and load the trend settings. These are the settings that are configured when the trend is displayed at runtime. The file is saved/loaded from the project directory.
Prior to version 5.0.0.1
The file is saved/loaded from the "Miscellaneous" path configured in the project settings.

Trend name (Optional)

The name is used when a command to change a pen, add/remove limit or band, at runtime is executed. Trend names can be duplicated. If a change command is used, the first trend with a name matching the name in the command will be used for the command.

Font color

The font color for all fonts in the trend.

Font color (printing), Background color (printing)

This is the color for the fonts/background when printing via the print button in the trend.

Use print colors

If enabled, when the print command is issued via the print button in the trend, the trend will print with the colors selected. Frequently, the trend on the monitor uses a dark background and a lighter color for fonts. For printing, this option provides for the colors to reverse creating better print output.

Print orientation

If enabled, when the print command is issued via the print button in the trend, the trend will print with the orientation selected.

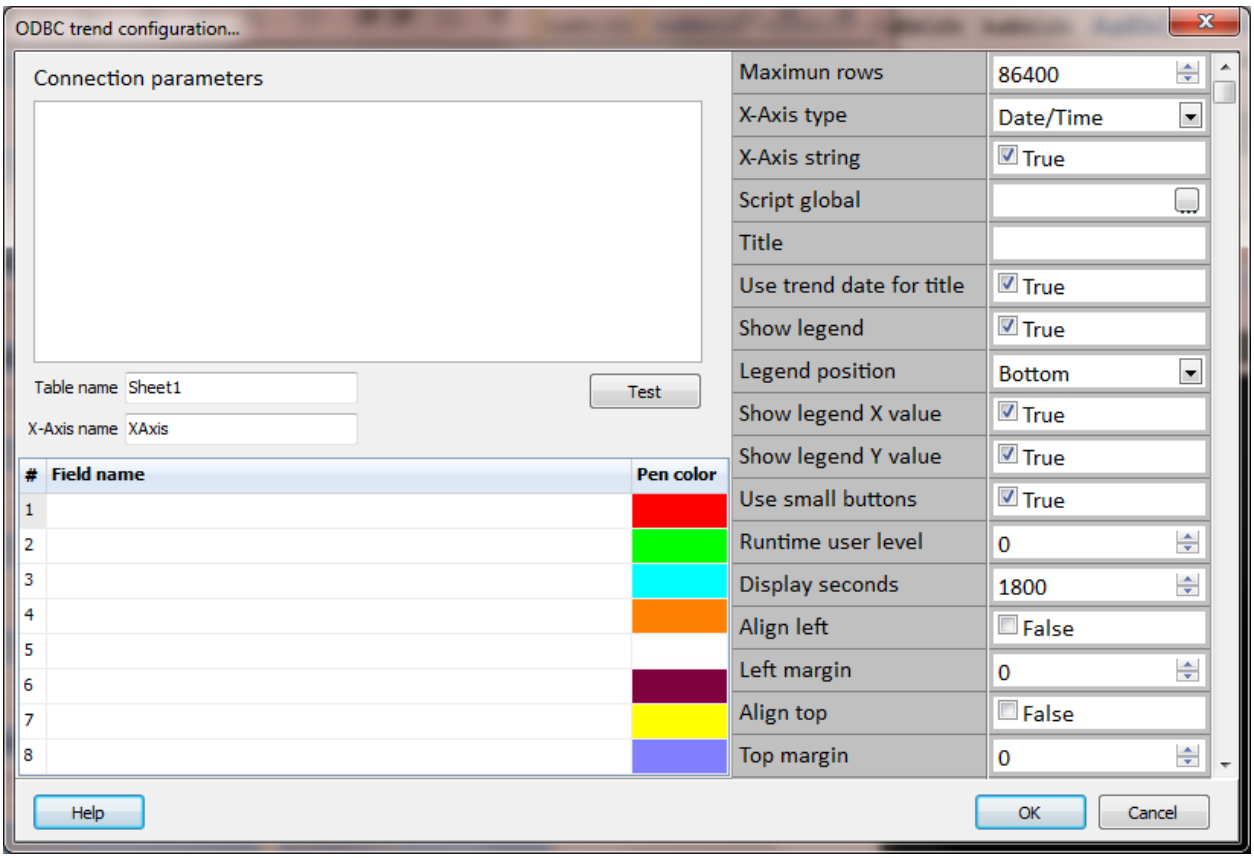
Cursor – “Empty” text

When the HMI runtime program is not executing the data logger is not collecting/storing data for trends. If the trend “cursor” is positioned on a second without data, the trend cursor hint displays the time and “Empty”. e.g. 10:23, Empty.

If this property is not blank the “, Empty” text will be replaced with the configured text.

Back to list

ODBC TREND



Each trend can display a maximum of eight pens.

Connection parameters

These are the values used to connect to the ODBC data source. Connection parameters are described here.

Table name

This is the table name containing the trend values in the database. The database name is specified in the connection parameters.

X-Axis name

This is the field name for the "X-Axis" of the trend.
Note: The X-Axis field value must be an increasing value for the trend to function properly. In other words, the value in each row must be greater than the value in the preceding row and lesser in value than the subsequent row.

Row index	X-Axis value
R - 1	< X
R	X
R + 1	> X

Field name and color

Select the field name to trend and select the pen color.

Maximum rows

When the trend is first displayed the connection to the database is established and the trend is loaded with data from the database. This value defines how many rows, from the last row reading backwards, to load into the trend. A value of zero (0) defines to read all rows. Note: Too large of a value may cause an out of memory condition and/or cause the user interface to stall while the data is being loaded. Rows loaded to the database, after the initial trend load, will be added to the trend up to a maximum value (not the "maximum rows" value). If the maximum value is too low, contact support for assistance.

X-Axis type

The X-Axis of the trend is usually time based and stored as a 64-bit floating point value in the IEEE-754 standard format. See X-Axis name above for data requirements.

Type	**Description**
Date/Time	64-bit floating number, number of days since December 30, 1899 and time of day. The value must be between -657434.0 and 2958465.99999.
Number	An incrementing value.
UNIX time	UNIX date-and-time values are encoded as the number of seconds since midnight at the start of January 1, 1970. The value is converted to a Date/Time.

Note: If using ODBC data logging for data storage, one method to create a timestamp for the X-Axis:

1) Create an analog host point.
2) The script for the analog host point is one line; **result:=Now;**
3) Use the PT command to log the value in the X-Axis field.
4) Be sure set to set the "X-Axis is string" property true.

X-Axis is string

The X-Axis field type can be 'text' or 'number'. The string '123' appears to be a number but, it really is an encoding of the value 123. The HMI ODBC data logger uses strings. Other database systems may use numbers. The settings must be correct for the database/table/field.

Script global

The script global property provides a method to load the connection parameters for the trend. This property is the section name of the script global. The trend loading logic reads all the items (regardless of the item name) and applies the item values to the connection parameters. The connection parameters field is ignored if a script global section name is configured.

There are three reserved item names (case sensitive).
Table name
Maximum rows
X-Axis name

When the section is loaded and either or both reserved name is detected, the item value is applied. See above for each for each name represents.

Caution: All other items are applied to the connection parameters, regardless of item name.

Note: Each line (item) of the connection parameters should be in a separate item.

Example:
The trend is loaded and another data set needs to be displayed in the same trend.
Set the script global items (connection parameters) to the desired values and call the "SetTrendPen" with the correct values. The database connection will be terminated, the trend cleared of data, the connection parameters loaded, connection to the database established and the data loaded to the trend.

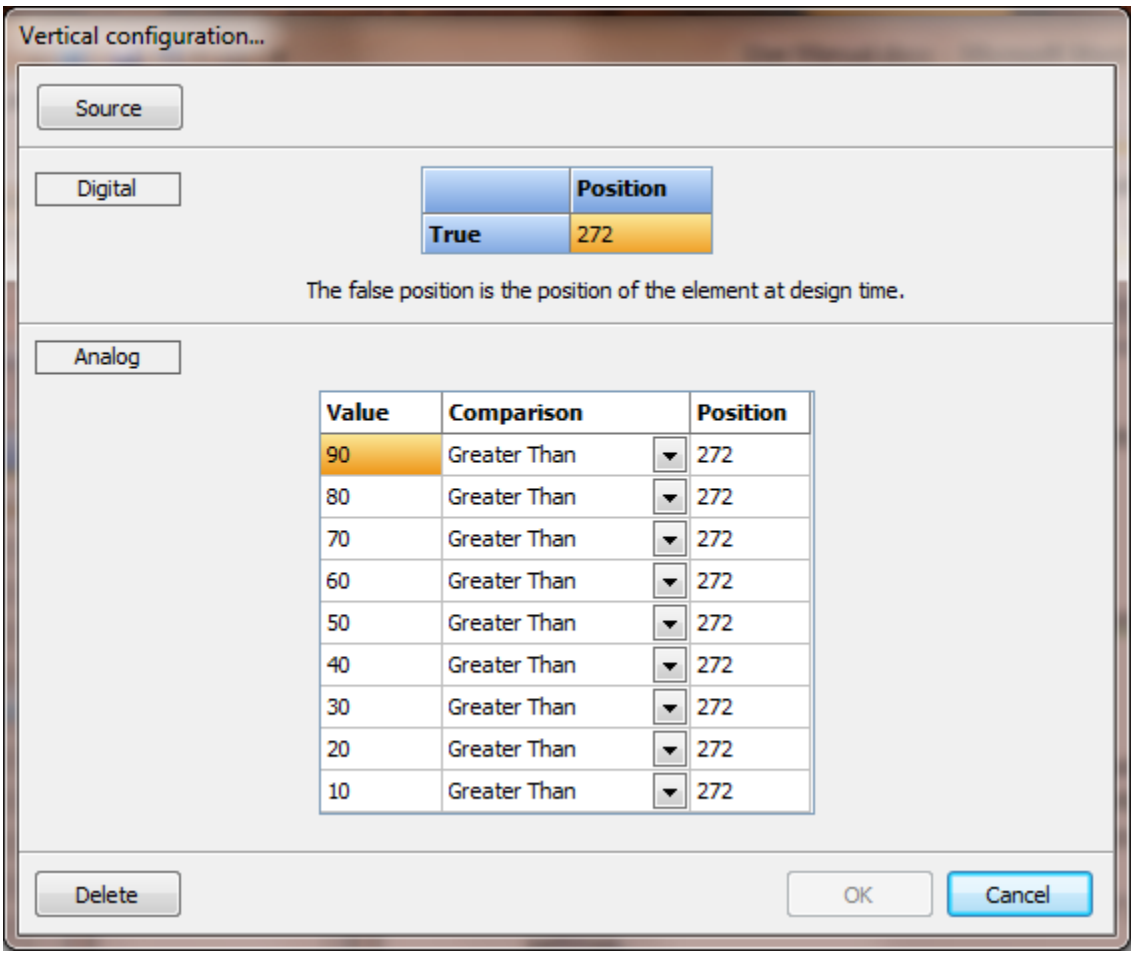
Example script global:

Name	Value
DatabaseLine	Database=C:\logFile.xlsx
Line1	DriverID=ODBC
Line2	ODBCDriver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *.xlsb)}
Line3	User_Name=
Line4	Password=
Line6	ODBCAdvanced=ReadOnly=False
Line7	LoginTimeout=10

Additional settings for ODBC Trend (static).

Back to list

VERTICAL POSITION

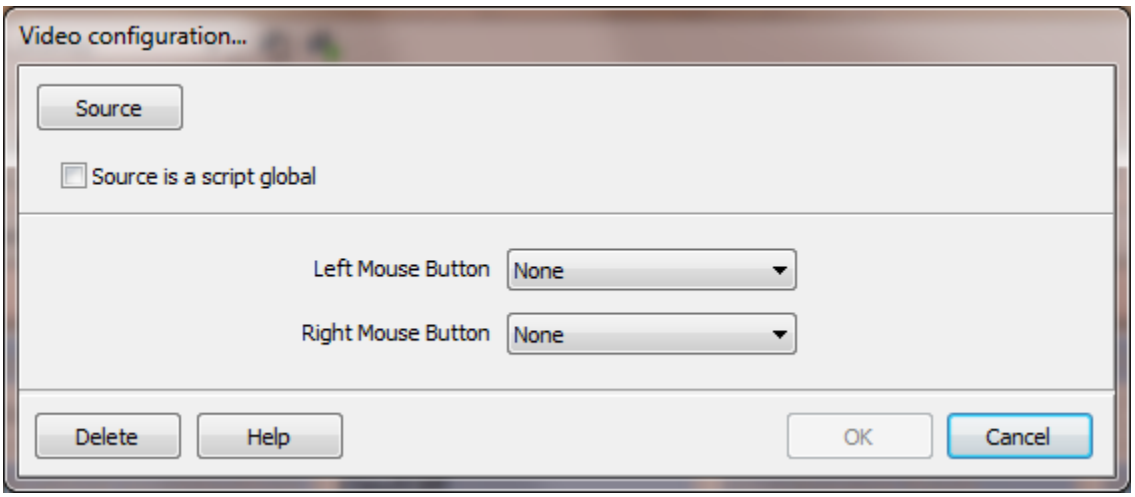


This is used to configure the vertical position of an object.

The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

Back to list

Video



Provides a method to display images from a video camera or encoder.

Source

The source video to supply the video images.

Source is a script global
The name of the source video is the value of the selected script global if this attribute is enabled. This provides a method to change the source video used before the window is opened. For example: a project has two pumps with the same I/O configuration and both have one video. Using tag pointers to access the I/O for graphic elements and script globals as a pointer to the video name, one window can be created and maintained. Setting the tag index and the script global configures the window to show the selected data at runtime.

Left Mouse Button, Right Mouse Button

None

When the mouse button is pressed in the element no action will be taken.

Capture

A frame from the video will be captured and saved to disk.

Record

When selected the state of recording the video will be toggled. Start to stop or stop to start.

Note: If recording and the video device is a VFW connection recording will be terminated when the window is closed. If the video device is an IP camera/video

encoder the recording will not terminate unless there are no other open windows displaying the video and motion detection is not recording.

Note: VFW cameras can only be viewed in one window at a time. IP cameras/video encoders do not have this limitation.

Note: For mouse down/up commands place a button over the video element or that part of the element the user to must click on. Set the button to transparent and use the button mouse commands.

RUNTIME WARNING

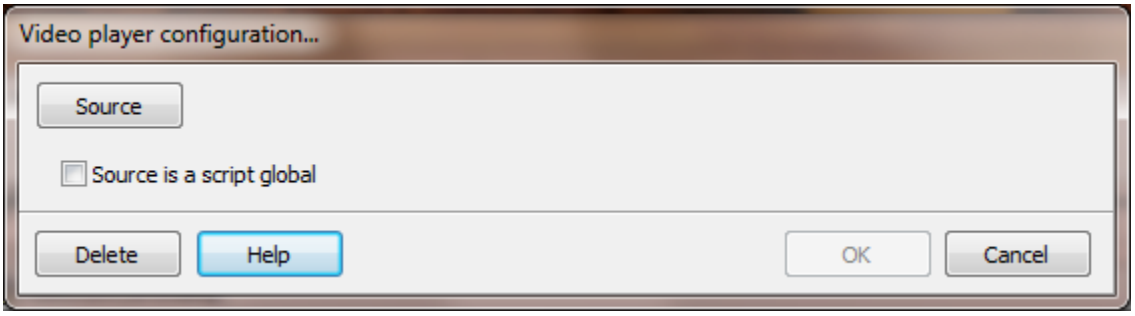
#1 If an attempt to view a camera output (VFW) in more than one element at a time during runtime, errors will be generated and will result in an access violation. A video element referencing a camera (VFW) can be configured more than once in separate windows. Make sure both windows cannot be opened and viewed at the same time. IP cameras/video encoders do not have this limitation.
#2 If the video input device is not connected and functioning properly, the program may respond very slowly if the window containing the video animation is displayed. If this occurs, select the window and

press and hold the ALT + F4 keys until the window closes
or
press and hold the left mouse button in the window close box until the window closes
or
press and hold the left mouse button on the window graphic used to close the window until the cursor changes to a "hand" and then release the mouse button.

RUNTIME WARNING

Back to list

VIDEO PLAYER



Source

The name/path of the file to playback in the player.

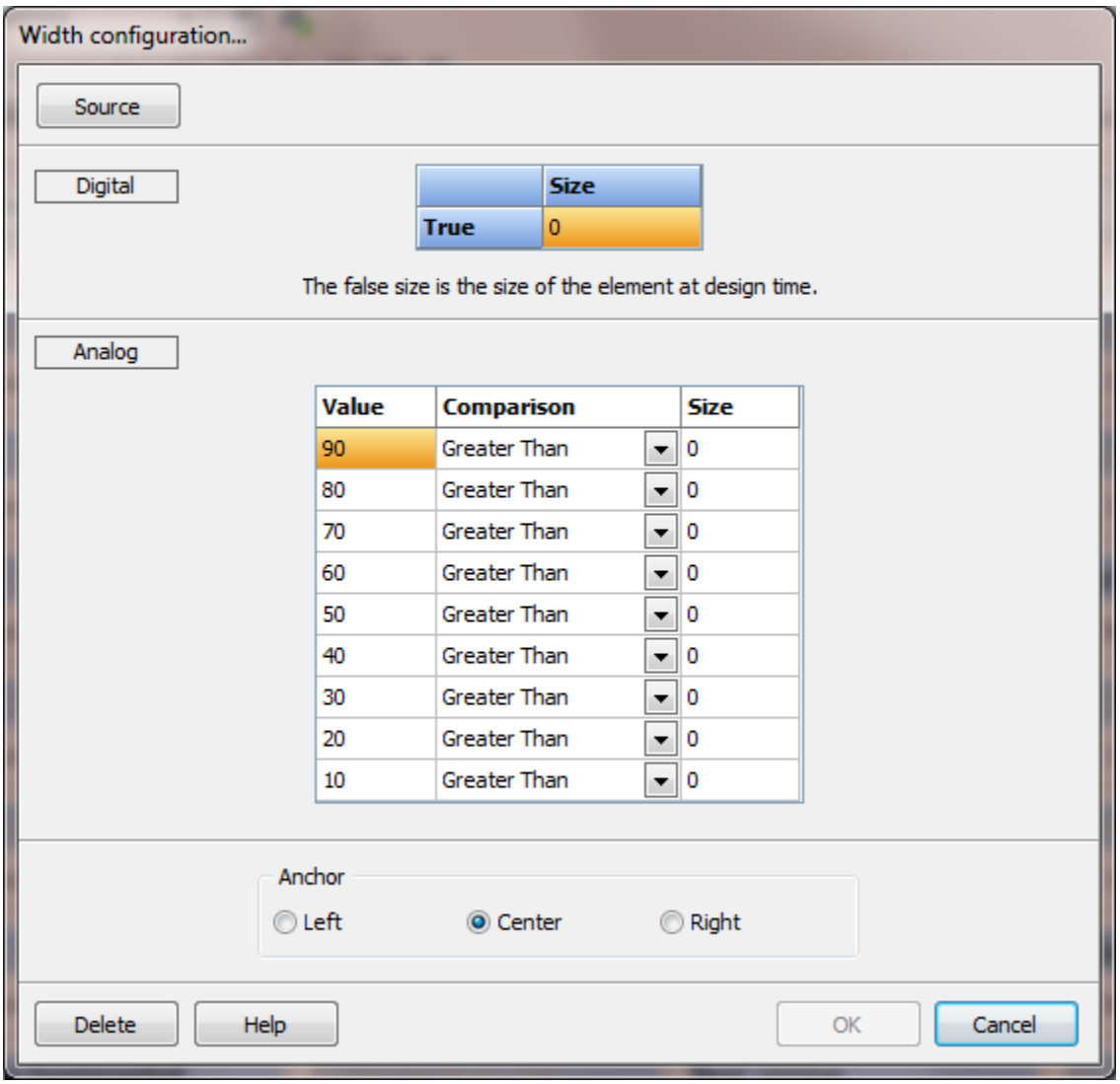
Note: To use a URL as the source for the video playback, enable the “source is script global” property and enter the URL in a “script global.item name” .

Source is script global

The file name/path is the value of the selected script global if this attribute is enabled. This provides a method to change the file name used before the window is opened or before the 'Play' button is selected.

Back to list

Width

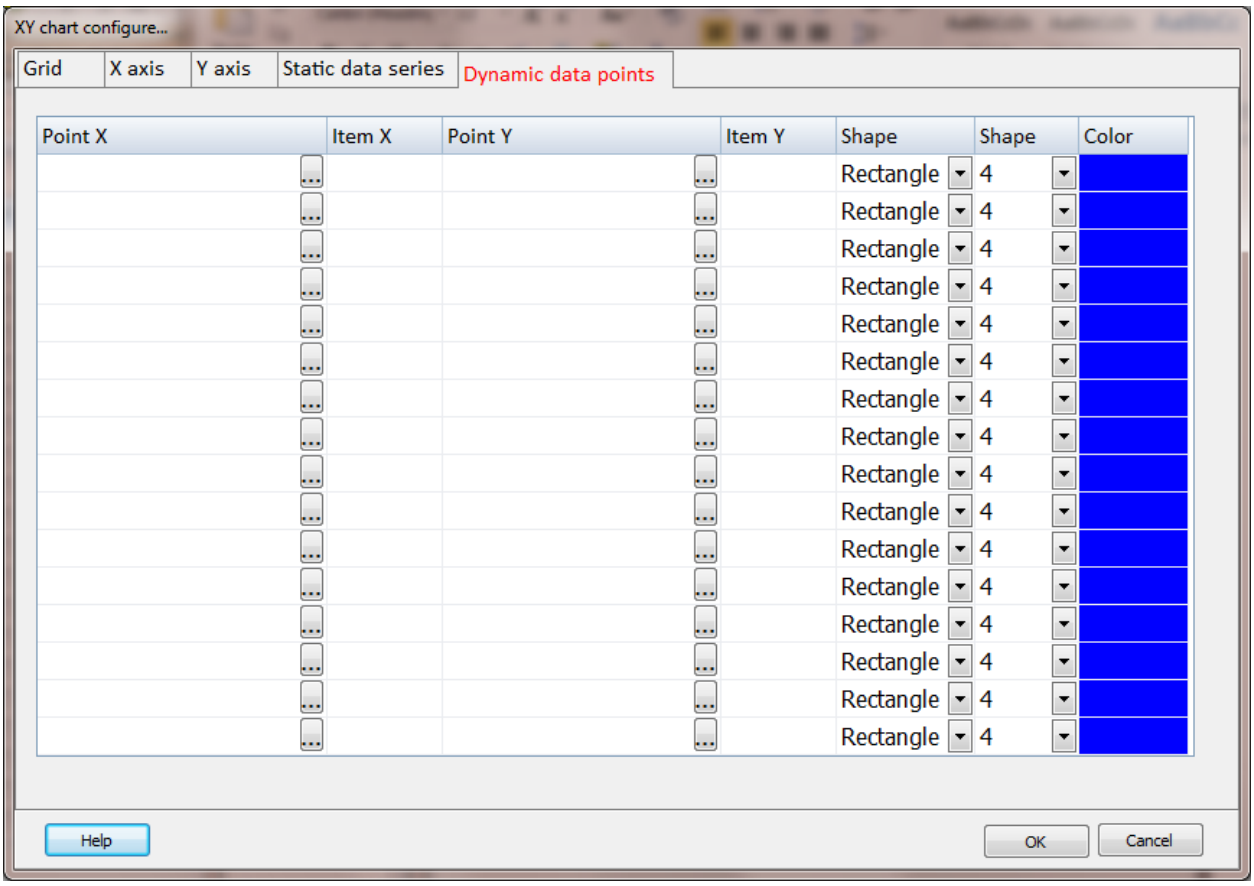


This is used to configure the width of an object.

The logic scans from top to the bottom. If N/A is used for any comparison, the scanning halts and the settings for the row are used.

Back to list

XY CHART



This is an XY chart that can contain many dynamic data points and many static data series.

Dynamic data point: a point on the chart that consist of an X value and Y value that are plotted on the chart as one point.

Static data series: a series of X, Y points on the chart used to show a series of points.

For example, one series could show a high range and another for the low range. Then a series of dynamic data points could be plotted to show current values.

For example, one series could show a series of values representing a characterized curve and then a dynamic data point is used to indicate the location on the curve for an output.

Note: The X and/or Y divisions need not be linear. The X and Y divisions are drawn in an even distribution. Adjusting the margins or the size of the grid element to get the first and last positions to line up with the edge of the grid might be required. If an edge lining up is desired make sure the inside rectangle, the grid adjusted with the margins) is evenly divisible by the axis tick count - 1.
Example: inside rectangle height = 336, (tick count - 1) = 14, 336 / 14 = 24 good.
Example: inside rectangle height = 341, (tick count - 1) = 14, 341 / 14 = 24.357 bad.

Grid

Margins, Top, Bottom, Left, Right

This provides a space around the edge of the chart. Used to provide space for the X and Y tick marks and values if enabled. See below.

Border, pen width, color

If the pen width is greater than 0 (> 0) a border is drawn around the chart using the pen width and color selected.

Background color

This is the color of the complete chart area. If the margins are zero this color will not be visible. The background color, inside the chart, is the brush color.

X axis, Y axis

File

This is the file that contains the tick points for the axis. The file path is the project path and in the directory 'XYChart'. The file is a comma separated text file (.txt) with each line representing one tick point. Each line has three values.
Tick value: a number, used in data calculations.
Displayed value: a string, displayed at the tick mark. If the 'Tick values' is not enabled this string is ignored.
Tick size: 'S' for short or 'L' for long. The 'S' length is 1/2 the tick length (see below).
The first line is the bottom/left of the chart and the last line is the top/right of the chart.
(value, displayed value, tick size)

Example:
0.0, Empty, L
5.0, 1/8, S
10, 1/4, L
...
40.0,Full, L

It is not required for the data points to be evenly distributed. The spacing of the ticks will be evenly distributed on the chart. The values can be any value as long as each lines value increases from the line before.

Example: 0, 2, 4, 6, 8 or 1, 45, 46, 47, 80. Bad, 1, 2, 5, 3, 7
A minimum of two lines per file is required.

Tick values

If enabled the values will be displayed.

Tick length

The length of the long tick mark. Setting this value to 0 will not display tick marks.

Font

The font for the tick values.

Grid lines

If enabled, a line will be drawn across the gird at each tick mark.

Grid/tick color

The color of the tick mark and the grid lines.

Tick value align (Y axis only)

If the tick value is enabled (see above) this selects the alignment of the text.

Static data series

File

This is the file that contains the data points for the series. The file path is the project path and in the directory “XYChart”. The file is a comma separated text file (.txt) with each line representing one series point. Each line has two values. (X, Y)
X value: the X value or horizontal value.
Y value: the Y value or vertical value.

Example:
0.0, 55.5
7, 22
10, 19

A minimum of one line is required.

Shape

Each data point displayed on the chart can have a shape. Rectangle, circle, diamond or none.

Shape size

The size of the shape.

Pen width

The pen width of the line that connects the data points. Setting the value to 0 will prevent the line from being drawn.

Note: If the pen width is zero (0) and the shape is none, the data series will not be drawn.

Color

The color of the data points and connecting line if enabled.

Dynamic data point

Data point

Each point consist of two values. An X value and a Y value. Each value can use as a source a:
1) point.item pair
2) constant value

To use a constant value place an equal sign (=) and the value in the point and item field, the two fields must match. Example =10 in both fields.

Shape

Each dynamic data point displayed on the chart can have a shape. Rectangle, circle, diamond or none.
Note: If the shape is none, the point will not be displayed.

Shape size

The size of the shape.

Color

The color of the dynamic data point.

Back to list

Mouse Commands

Mouse commands are commands that are mainly used in the user created graphics screens. The commands are also used by the task scheduler, the scheduler and any other function of the program that allows configuration of actions similar in nature.

Most mouse commands "wrap" scripting commands. When a mouse command wraps or has the same attributes as a script command and has the same function, the mouse command will be described in the scripting section.

All the mouse commands are listed below in alphabetical order.

Acknowledge command

Same as script AcknowlegeCommand.

Beep

Same as script Beep.

Capture screen

Same as script CaptureScreen.

Capture window

Same as script CaptureWindow.

Clear port counters

Same as script ClearPortCounters.

Close active alarm window

Same as script CloseAlarmWindow.

Close alarm log window

Same as script CloseAlarmLogWindow.

Close all user windows

Same as script CloseAllUserWindows.

Close event log window

Same as script CloseEventWindow.

Close port diagnostic window

Same as script ClosePortDiagnosticWindow.

Close tag monitor window

Same as script CloseTagMonitorWindow.

Close window

Same as script CloseWindow.

Close window 2

This closes the window that the graphic element is located on. If the 'Browser to front' is 'true' and a browser window is open, it will become the front window and then the window will close.

Custom Log Copy

Same as script CustomLogCopy.

Custom Log Flush

Same as script CustomLogFlush.

Custom Log Log

Same as script CustomLogLog.

Custom Log Save

Same as script CustomLogSave.

Custom Log View

Same as script CustomLogView.

Execute report

Same as script ExecuteReport.

Force logon

Same as script ForceLogon.
Note: When a user is not "logged on" all commands are disabled.

FTP set setting

Same as script FTPSetSetting.

Global set

Same as script GlobalSet.

Javascript

Website only. See Website Javascript.

Kill a process

Same as script KillAProcess.

Kill a process 2

Same as script KillAProcess2.

Launch application

Same as script LaunchApplication.

Load recipe

Same as script LoadRecipe.

Log event

Same as script LogEvent.

Log off

Same as script LogOff.

Log on

Same as script LogOn.
Note: When a user is not "logged on" all commands are disabled. This single command, in a mouse down or mouse up animation is permitted. If more than one command is configured for the mouse down/up animation, all commands will be disabled.

Mute

Same as script Mute.

Navigate

Same as script Navigate.

ODBC data logger

Same as script ODBCDatalogger.

ODBC data logger pause

Same as script ODBCDataloggerPause.

ODBC data logger set refresh

Same as script ODBCDataloggerSetRefresh.

Omni retrieve report

Same as script OmmiRetrieveReport.

Omni view report

Same as script OmmiViewReport.

Open active alarm window

Same as script OpenAlarmWindow.

Open alarm log window

Same as script OpenAlarmLogWindow.

Open browser window

Same as script OpenBrowserWindow.

Open drive status window

Same as script OpenDriveStatusWindow.

Open event log window

Same as script OpenEventWindow.

Open monitor window

Same as selecting the “Monitor” button on the runtime panel.

Open port diagnostic window

Same as script OpenPortDiagnosticWindow.

Open scheduler monitor window

Same as script SchedulerOpenMonitorWindow.

Open script monitor window

Same as script OpenScriptMonitorWindow.

Open tag monitor window

Same as script OpenTagMonitorWindow.

Open user log addition window

Opens the user log window for log entry addition.
If 'Allow category change' is enabled the user can select the category for the new log entry. If categories are not defined, this attribute has no effect. The category field is optional.

Open user log viewer window

Opens the user log viewing window.

Open window

Same function as the script function OpenWindow with the additional “close window“ attribute (calling window will be closed).

Open window ex

If the “Close window” attribute is true, the calling window will be closed. If a window does not open, the calling window will not be closed.

Open window once

The selected user window is opened. If the "Bring to front if already open" attribute is true and the window is open, it will be moved in the Z order to the front and a second window will not be opened. If the attribute is not set, the names of other windows are not examined.

If the 'Close window' attribute is true, the calling window will be closed. If a window does not open, the calling window will not be closed.

Open window user select window

Same as script OpenWindowUserSelect with added “close window“ attribute.
If the “Close window” attribute is true, the calling window will be closed. If a window does not open, the calling window will not be closed.

Play sound

Same as script PlaySound.

Play sound 2

Same as script PlaySound2.

Print screen

Same as script PrintScreen.

Print screen active window

Same as script PrintScreenActiveWindow.

Pulse boolean

Same as script WriteValuePulse.

Queue script

Same as script ExecuteScript.

Quit runtime

Same as script QuitRuntime.

Recipe save sheet

Same as script RecipeSaveSheet.

Reload recipe sheet

Same as script RecipeReloadSheet.

Reset boolean

Same as script WriteValue with a value of zero (0).

Save camera loop

Same as script CameraSaveLoop.

Send keys

Same as script SendKeys.

Set boolean

Same as script WriteValue with a value of one (1).

Set MA station configuration name

Same as script SetMAStationConfigurationName.

Set mouse position

Same as script MousePositionSet.

Set port read enable

Same as script SetPortReadEnable.

Set system window position

Same as script SetSystemWindowPosition.

Set task state

Same as script SetTaskState.

Set timer field

Same as script TimerSet.

Set trend pen

Same as script SetTrendPen.

Set window color

Same as script SetWindowColor.

Set window date

The user is queried for a date via the GetUserInputDate script command. If the accept button is selected the SetWindowDate script command is executed.

Silence acknowledge command

Same as script SilenceAcknowledgeCommand.

Silence command

Same as script SilenceCommand.

Start camera recording

Same as script StartCameraRecording.

Stop camera recording

Same as script StopCameraRecording.

String set

Same as script StringSet.

String set Ex

This is similar to StringSet except the value of the string is collected from a script global.

Toggle boolean

This is a "WriteValue" with the value inverted from the tagname/item value. The point access rights must be "Read/Write."

View trend history

This command is used to display a trend calendar selection window. The window has two parts, calendar/date selection and the trend viewing area.

Also see the ViewDateRangeTrendHistory script command.

First pen (optional)

If the mouse command is on a window with more than one trend element this identifies the trend to be used. If blank, the first trend element in the graphic element list is used. If the window only has one trend element this field can be blank.

Window left/top/width/height

The size and position of the display window. If width or height is 0 the default/last value is used.

Keep window on top

If enabled, the window will be set to be the top window. More than one window open with this attribute set will cause unpredictable window behavior.

Open maximized

The window will be opened filling the screen. The window position/size above as well as the last open position/size are ignored if this attribute is enabled.

Days back

This is used to limit the number of days, from the current date, that are accessible. 0 = infinite

Border style

Windows provides for several window border styles. None, single, sizeable and dialog.

Global script section/item (Optional)

The script global selected must contain a date with the format of MM-DD-YYYY. Example 06-11-2012. The date must match the format. This can be used to open the trend history to a specific date. For example; the date could be set, the border style to 'dialog', and the 'disable calendar' could be enabled. This would display the trend history for the only date selected. If trend logs do not exists for the date, an error window will be displayed.

The default date, this field is empty, displayed is the day before the current date.

Disable calendar

If enabled the calendar control will not allow for changing the date.

Zoomed in

If enabled, the chart will be zoomed in on the X and Y axis to show all the data loaded for the day. User zooming remains enabled. This attribute only applies for the horizontal trend.

Automatic close time

The maximum number of seconds the window is to be open. If the value is 0 (zero), the window will not automatically close.

Channel title override

If enabled, the channel title will be the tagname and the item.

Write boolean value : Ask user 2

This is a "WriteValue". The standard two button selection window is displayed. If a tagname and item are configured for the button and the user selects the button the value (True or False) will be written to the tagname/item. If a tagname is not present a write command is not executed. Each button also has a "log" string. If the string is not blank the string will be written to the 'Event log.'

Write boolean value : Ask user 3

This is a "WriteValue". The standard three button selection window is displayed. If a tagname and item are configured for the button and the user selects the button the value (True or False) will be written to the tagname/item. If a tagname is not present a write command is not executed. Each button also has a "log" string. If the string is not blank the string will be written to the 'Event log.'

Write boolean value: Ask user 4

This is a "WriteValue". The standard four button selection window is displayed. If a tagname and item are configured for the button and the user selects the button the value (True or False) will be written to the tagname/item. If a tagname is not present a write command is not executed. Each button also has a "log" string. If the string is not blank the string will be written to the 'Event log.'

Write float value 2: Ask user

The same as "Write Float Value: Ask User" except the low and high limit reference a tagname/item pair.

Write float value: Ask user

The user is queried for a value via the GetUserInputFloat command. If the accept button is selected and the value is in range the write command is executed.

Write float value percent: Ask user

The user is queried for a percent value via the GetUserInputFloat command. The value must be between 0 - 100 and the high/low limits configured. If the accept button is selected and the value is in range the write command is executed.

The final value is the percent of the engineering range of the tagname.

Write integer value 2: Ask user

The same as "Write Integer Value: Ask User" except the low and high limit reference are a tagname/item pair.

Write integer value: Ask user

The user is queried for a value via the GetUserInputInteger command. If the accept button is selected and the value is in range the write command is executed.

Write integer value percent: Ask user

The user is queried for a percent value via the GetUserInputFloat command. The value must be between 0 - 100 and the high/low limits configured. If the accept button is selected and the value is in range the write command is executed.
The final value is the percent of the engineering range of the tagname.

Write value fixed

Same as script WriteValue with the value set in the mouse command configuration.

Write value percent

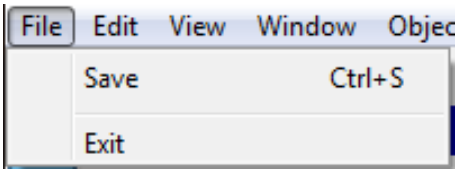
This is a WriteValue command. The value is the percent of the engineering range of the tagname. The value must be between 0 - 100.

Back to list

Menus

This will cover the menus in a graphic editor window from left to right.

File



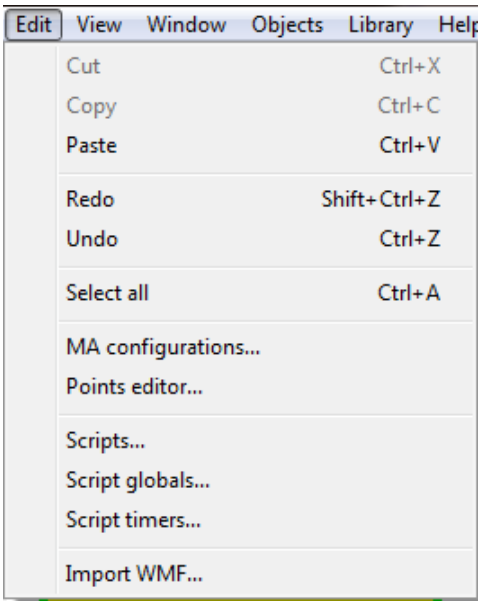
Save

Saves the graphic screen to disk.

Exit

Closes the editing window.

Edit



Cut/Copy

These menu items perform the normal actions of a modern program.
Cut, copies the selection to the clipboard and deletes the selection.
Copy, copies the selection to the clipboard.

Paste

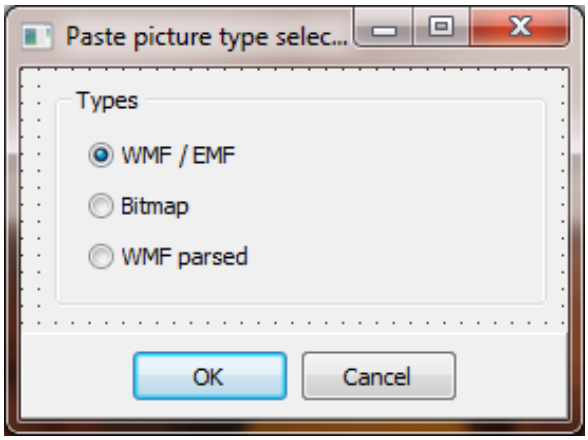
The contents of the clipboard will be pasted to the window. If the contents are of a type native to the program they will be pasted as native. If the clipboard contains a bitmap or JPEG it will be imported and converted to a native bitmap element. If the clipboard contains text it will be imported and converted to a native text element.

If the contents of the clipboard are native elements, the elements to be pasted, are created in the window using one of two methods.

If the "SHIFT" key is not down, the pasted elements are centered on the location of the last mouse down click.

If the "SHIFT" key is down, the pasted element locations are not altered and are pasted to the same location as copied. If the source and destination window are the same, the "new" pasted elements will be on top of the original (copied) elements.

If the program detects an EMF/WMF file a dialog will appear. The HMI can import the image using one of three methods.



See the "Import WMF" menu item below.

Redo

Most drawing actions are recorded in a first in last out list. If "Undo" is selected the last action is "rolled back". Redo "rolls forward" in the list. If undo is selected 5 times in a row, without performing any other action, redo can be selected 5 times, if no other actions are performed. See "Undo" below.

Undo

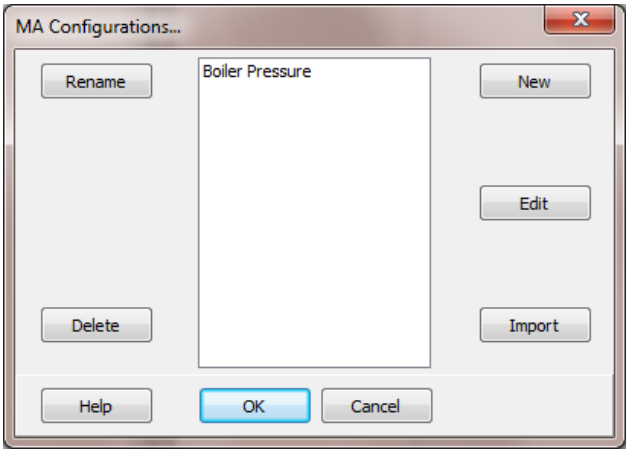
Most drawing actions are recorded in a first in last out list. To undo the last action, select undo. It might be possible to undo many previous actions if the action chain has not been cleared. Some drawing actions require the action list to be cleared. If "undo" is selected and before any other drawing action is performed it is

possible to "redo" the last action by selecting the redo menu item. See "Redo" above.

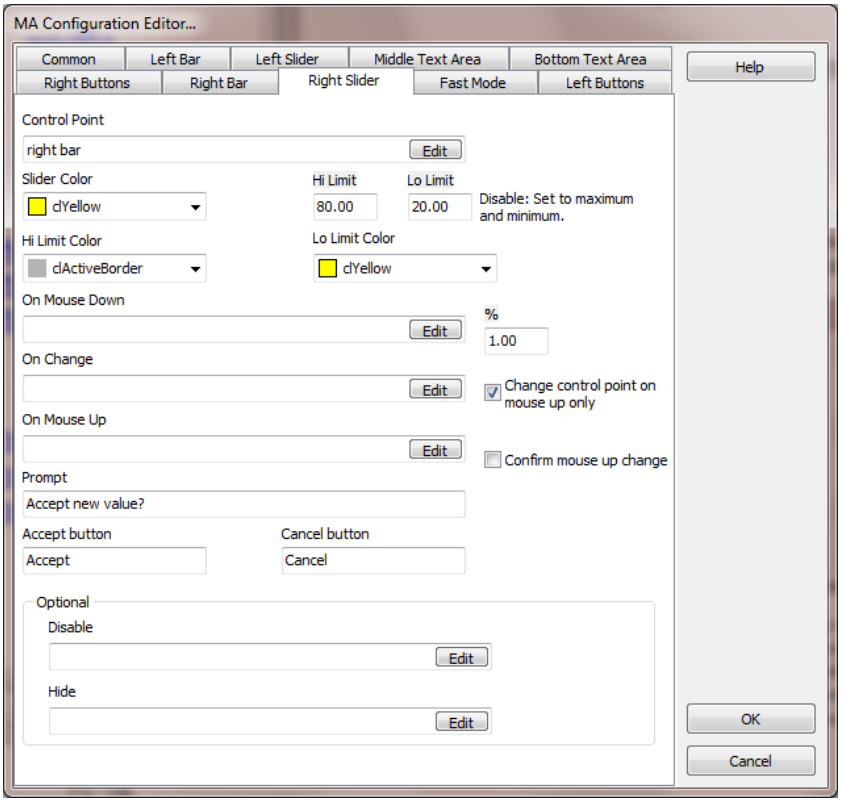
Select All

If the current tool selection is the "arrow" all objects on the screen will be selected. If the current tool selection is another tool only those elements that are of the same type will be selected.

MA Configurations



The Manual/Auto (MA) configuration is not contained in the graphic element to allow an MA to be used on a screen and the screen reused. The MA configuration used for an MA element can be set via scripting via the SetMAStationConfiguration command. The graphic element has an initial MA configuration selection.



Bottom Text Area

The bottom text area can show limited text on the left and/or right side. Selecting a tagname and digital item will display the text in the value row 0 or 1. Selecting a tagname and analog item will display the text in row 0 through 7. If the control value is less than 0, the row 0 text will be displayed. If the control value is greater than 7, the row 7 text will be displayed.

Common

Name

The name of the MA configuration.

Caption

The string, if any, to be displayed at the top of the MA element.

Font

The font name and color to use for text.

Font Style

The font style to apply to all text on the MA station.

User Level

The logged on user must have a user level equal to or greater than this value to activate the controls in this MA configuration. If the user level is not sufficient a beep will be played and the control action will be prevented.

Fast Mode

On the left side of the MA are four possible buttons, grouped in two pairs. Each pair has an up arrow button and a down arrow button. If the left slider is enabled the top two buttons will be connected to the left slider. If the right slider is enabled the bottom two buttons will be connected to the right slider. The top button of the pair will increase the slider and the bottom button of the pair will decrease the slider. The buttons have two modes: Fast Mode and Slow Mode. If the checkbox is not checked all buttons operate in slow mode. If the checkbox is checked all the buttons operate in fast mode.

Fast and slow mode are the amount of change each time one of the buttons is pressed. The percentage is percent of full scale of the data point connected to the slider.

Example
The fast mode is set to 5.00%.
The slow mode is set to 1.00%.

The left slider is connected to a data point and the range of the data point is 100.

The current value of the data point is 50.
If fast mode is enabled, the checkbox is checked, and the increase button for the left slider is pressed the data point would be changed to 55.

The current value of the data point is 50.
If fast mode is not enabled, the checkbox is unchecked, and the increase button for the left slider is pressed the data point would be changed to 51.

Name

The name of the checkbox. If this field is blank the fast mode checkbox will not be visible and the increase/decrease buttons will operate in slow mode

Fast Mode

Enter the percent of change per button click when fast mode is active. If the value is less than are equal to 0 the value will be set to 5.

Slow Mode

Enter the percent of change per button click when fast mode is not active. If the value is less than are equal to 0 the value will be set to 1.

Inhibit Fast Mode (Optional)

Select a data point to enable or disable the fast mode. If the checkbox is inhibited (disabled) slow mode is selected regardless of the checkbox checked state. (true = inhibit)

Left Bar, Right Bar

Monitored Point

The analog data point to monitor for the bars percent of fill.

Bar Color

The bar has two colors. The foreground and the background. The foreground color is the color of the moving bar. The background color is the color of the unfilled portion of the bar.

Edge

The border of the fill bar.

Direction

Down:
The bar will fill from the top to the bottom.

Middle:
The bar will fill from the middle and fill toward the top if the percent of full scale is greater than 50 and fill towards the bottom if the percent of full scale is less than 50.

Up:
The bar will fill from the bottom to the top.

Tick Color

The color of the tick marks on the side of the bar.

Hi Hi, Hi, Lo, Lo Lo Alarm color

The monitored point has four alarm points. Each alarm can be enabled or disabled. If the alarm is enabled a small symbol will be drawn next to the bar at the alarm setting value. Select the color for each alarm point.

Left Buttons

See **Fast Mode** for a description of button operation.

These are optional.

Left Slider, Right Slider

The buttons can be configured to also execute a script when pressed.

Increase On Click:
When the increase button is clicked execute the selected script.

Decrease On Click:
When the decrease button is clicked execute the selected script.

Left Slider, Right Slider

Control Point

The analog data point the slider reads for position and the data point written to when the slider is moved by the user.

Slider Color

The color of the slider indicator.

Hi Limit Color

The color of the high limit indicator.

Lo Limit Color

The color of the low limit indicator.

Hi Limit

The maximum value the slider can be moved by the user. To disable set the value to the high range of the control point.

Lo Limit Color

The minimum value the slider can be moved by the user. To disable set the value to the low range of the control point.

On Mouse Down (Optional)

A script to execute when the mouse button is pressed in the slider indicator.

On Mouse Change (Optional)

A script to execute when the mouse button changes the position of the slider.

On Mouse Down (Optional)

A script to execute when the mouse button is released after the mouse was pressed in the slider indicator.

%
When the slider is moved, after the mouse button is pressed in the slider indicator, this value determines how much the slider must change position, in percent of full scale, before a write command is issued to the control point. If the "Change control point on mouse up only" attribute is checked this setting is ignored

Change control point on mouse up only

When enabled the a write to the control point will not occur until the mouse button is released. See the "Confirm mouse up change" attribute.

Confirm mouse up change

When enabled the write to the control point does not occur until the confirmation dialog is executed. If the result of the confirmation dialog is false a write to the control point is not executed and the on mouse of script, if configured will not execute. See the "Change Control point on mouse up only" attribute. This attribute can also be used without the "Change Control point on mouse up only" attribute set.

Prompt

The text to display in the prompt area of the configuration dialog, if the "Confirm mouse up change" attribute is enabled.

Accept Button

The text to display in the accept button of the confirmation dialog, if the "Confirm mouse up change" attribute is enabled.

Cancel Button

The text to display in the cancel button of the confirmation dialog, if the "Confirm mouse up change" attribute is enabled.

Disable (Optional)

The control point to disable the slider. (true = disabled)

Hide (Optional)

The control point to hide the slider. (true = hide)

Middle Text Area

The middle of the MA station has three lines of text. The lines can be connected to the sliders, the fill bars or no connection. If connected the value of the connection is shown along with a caption.

If the connection is a slider and the mouse button is pressed on the line of text a dialog will appear to allow setting the value of the slider.

Connection

The control element on the MA station to connect to the text readout.

Color

The text color of the line.

Slider disabled (On Slider configuration tab)

If the control element this line of text is connected to is disabled, the mouse down action is disabled. Otherwise it is always enabled.

Caption

The caption for the line of text.

Disable Mouse

When connected to a slider and this attribute is enabled the mouse will not perform any actions. The user clicking on the text will not cause any change.

Note: When a text line is connected to a slider and the slider is not disabled or hidden and the user presses the mouse in the text area, a GetUserInputFloat dialog is displayed to allow the user to enter a value that is limited by the range of the slider low and high limits. If the value is in range and the user selects the accept button the value is written to the slider tagname and the OnMouseUp script of the slider is called if it is assigned. If the user selects the cancel button the OnMouseUp script is not called. The OnMouseDown and OnChange scripts of the slider are not called via the middle text area.

Right Buttons

Caption

The string, to be displayed in the button.

On Click (Optional)

The script to execute when the button is clicked.

Control Point (Optional)

The digital data point the button send the data value to when the button has been clicked.

Data Value

The button will perform the action selected.

Set Bit:
The button will write to the data point specified at the control point a "1".

Reset Bit:
The button will write to the data point specified at the control point a "0".

Toggle Bit:
The button will write to the data point specified at the control point a "1" and then a "0". The length of time between the 1 and the 0 is variable. It depends on such thing as the communication protocol, the physical media, etc..

Disable (Optional)

The control point to disable the button. (true = disabled)

Points editor

”Points” and the points editor are covered in the section “Points”.

Round chart themes

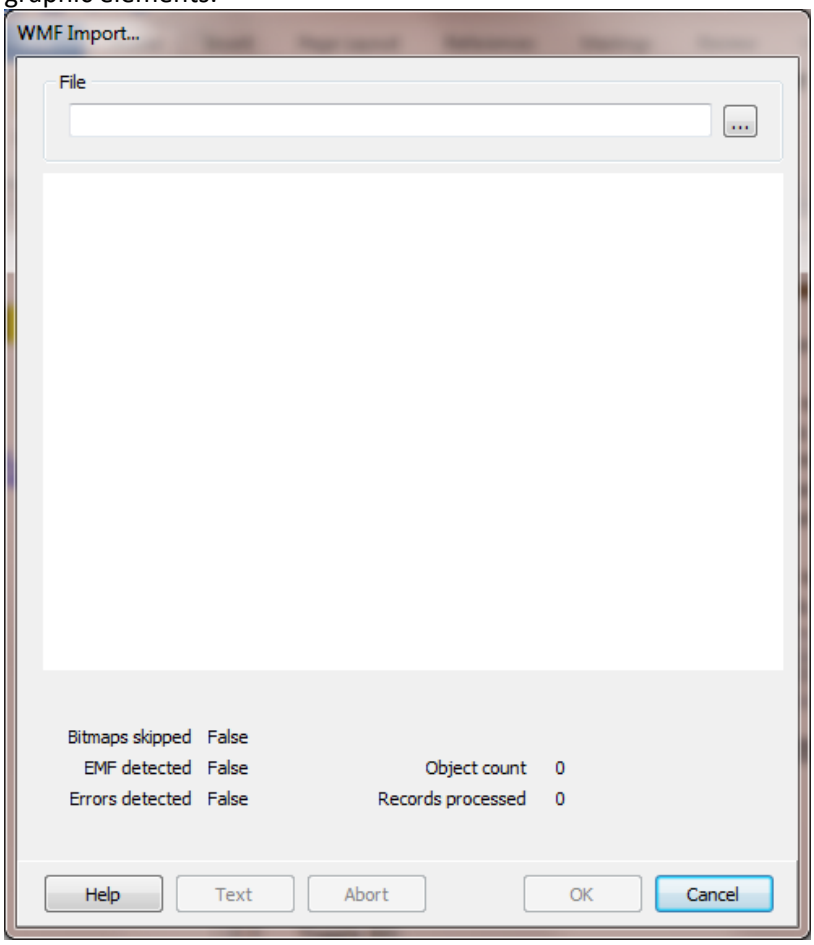
The “Round chart them editor is covered in the section “Round trend chart”.

Scripts

”Scripts”, ”Script globals” and ”Script timers”, are covered in the section “Scripting”.

Import WMF

This provides for the importing and parsing of EMF/WMF into native graphic elements.



A WMF (Windows Meta File) can be pasted into the drawing as a WMF element or it can be parsed, when pasted/imported, in an attempt to create native HMI objects from the WMF data structure. **CAUTION:** The parsing of a WMF file can create 1000s of graphic elements. When the parsing is complete and the user selects the OK button the native graphics elements are grouped and placed in the top left of the window.

When a WMF is parsed some record types are not converted. Bitmap, embedded EMF (Enhanced Meta File) and text are all skipped in the conversion. If using an WMF file and the parsing result is not satisfactory, please contact support.

When a WMF is pasted/imported as a WMF it can be scaled by selecting and moving one of the selection handles. When it is parsed into native HMI elements and grouped, use the scale command to scale the group.

File

Select a WMF file to import.

Bitmaps skipped

If true, bitmap records in the WMF structure were detected and skipped.

EMF detected

If true, embedded EMF data was detected in the WMF structure was detected and skipped.

Errors detected

If true, some kind of error was detected in the WMF structure. Please send the file to support.

Object count

After the WMF is parsed this is the count of native objects created.

Records processed

This is the count of records detected in the WMF structure and processed

Text button

While the WMF structure is being parsed a log is created. To view the log, select this button.

Abort button

WMF structures can be very large. Selecting this button will stop the processing of the WMF structure.

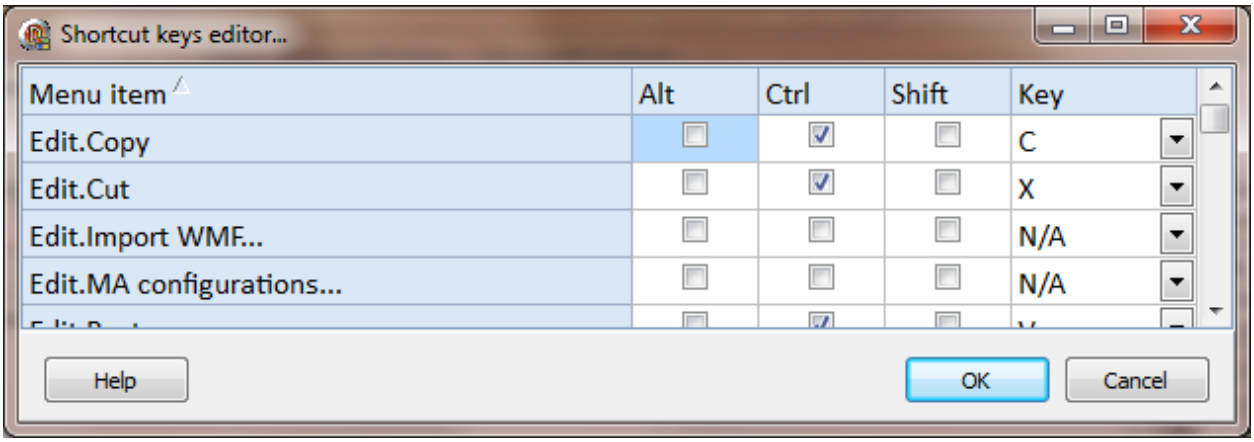
OK button

If after the WMF structure is parsed and at least one object was created select this button to add the object to the window.

Cancel button

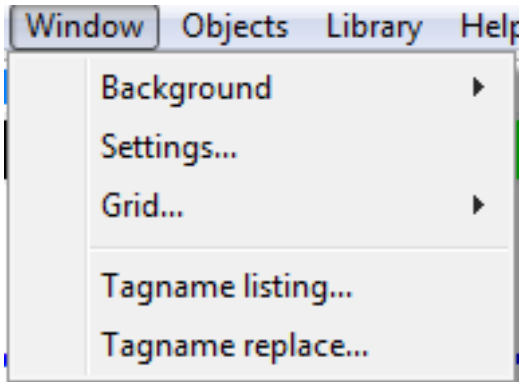
Select this button to exit. Any imported data will be discarded.

SHORT CUT KEYS…



This feature provides a method to customize menu shortcut keys.

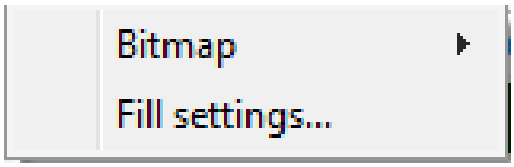
Window



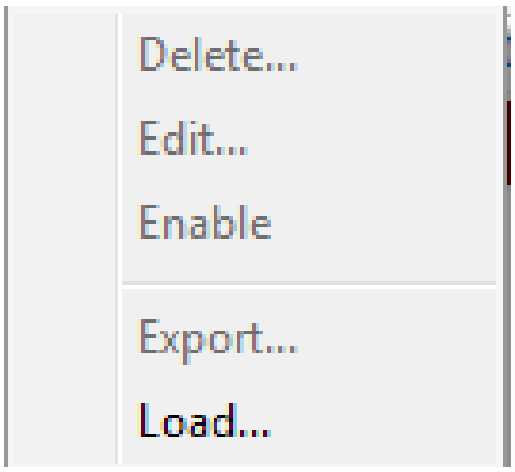
Background

The window background can be filled with a solid color or a brush pattern with a fore ground and back ground color. It can also dislpay a bitmap in the background.

Bitmap



Fill settings...



Delete

If a bitmap image is not loaded into the background, this menu will be disabled. If enabled and selected, the internal background bitmap will be deleted.

Edit

If selected, the HMI bitmap editor opens. The bitmap editor is covered under “Bitmap editor” in this manual.

Enable/Disable

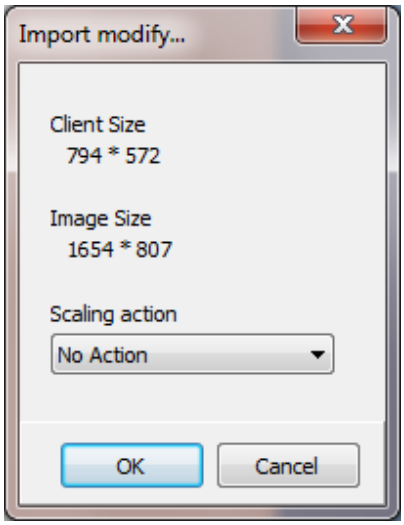
If a bitmap is loaded, it can be enabled to display or disabled. Disabling the bitmap does not delete it

Export

If this menu is enabled and selected, the internal background bitmap can be saved to a bitmap file.

LOAD

A bitmap can be loaded to the internal background bitmap. When the bitmap is selected the file type can be EMF, BMP, EMF, JPG or JPEG. When the bitmap is selected, if it is not the exact size of the client area of the window (see settings below), a dialog will appear allowing for several actions.



No Action:
The bitmap will be imported and centered to the window.

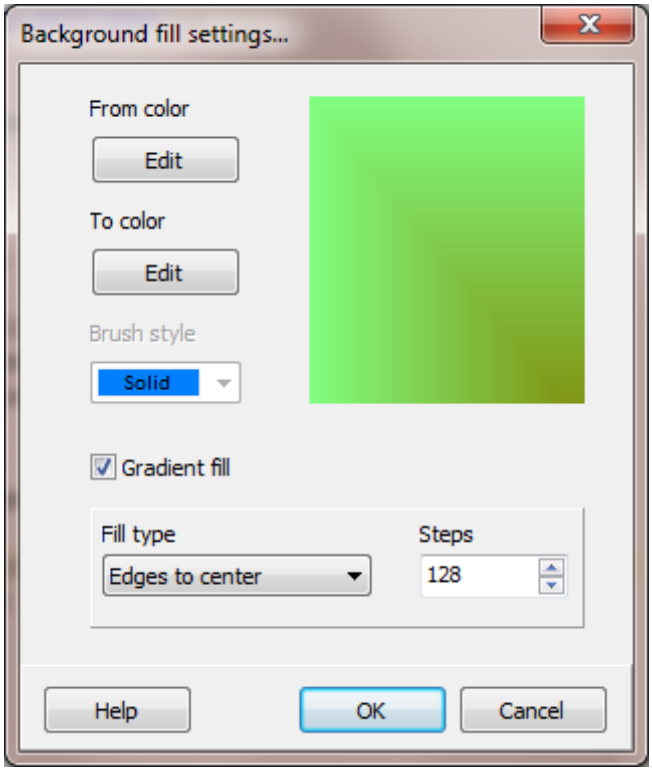
Crop:
The bitmap will be aligned to the top/left of the window and if the bitmap is larger than the client area of the window, the bottom and right will be cropped to fit the window.

Expand, Shrink:
The bitmap will be expanded or shrunk to fill the client area of the window.

Note: If on import/load the bitmap does not fill the client area, the current fill settings are applied, followed by the internal bitmap generation. So, first apply any desired fill settings, if applicable, and then import/load the bitmap.

FILL SETTINGS

This provides for the selection of the foreground color. If a brush pattern is selected a background color can also be selected.



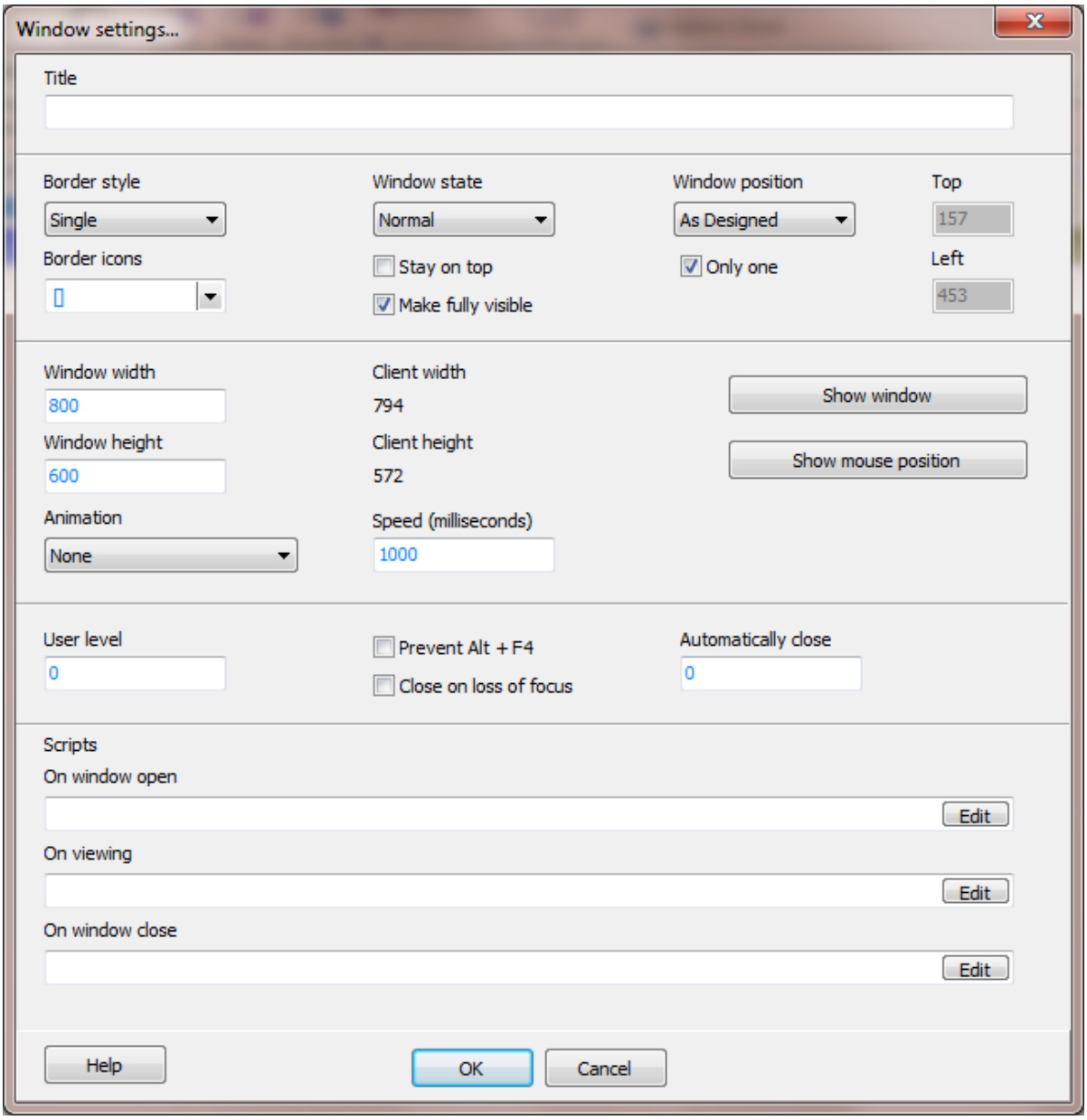
Foreground/From color:
The foreground color or the “from” color if the window is filled with a gradient.

Background/To color:
The background color or the “to” color if the window is filled with a gradient.

Brush style :
The window can be configured to fill with a pattern or brush style. The brush is not used if a gradient is configured.

Gradient fill :
The window can be filled with a gradient using the foreground and background color, the “Fill type” and the number of steps (color changes) .

SETTINGS



Title

This string will appear in the title bar of the window if the window is configured to show the title bar.

Border style

Windows provides for several window border styles. None, single, sizable and dialog. These styles can be viewed by selecting the "Show Window" button in the middle of the screen.

Border icons

These are the normal "Windows" icons in the top right corner of the window. If the "Maximum" or Minimize" is enabled the "System" must be enabled.

Stay on top

This instructs the window to stay on top of all other windows. More than one window open with this attribute set will cause unpredictable window behavior.

Make fully visible

On multi-monitor systems when the window is positioned the window may span several monitors. If this attribute is enabled the form will be repositioned to fully fit on the monitor, if possible. Note: The main monitor is the target monitor for the window when this attribute is enabled.

Window position

Select the position to open the window. As designed opens the window in the last position it was during configuration.

Only one

If this attribute is enabled the program will allow only one window with this name to be opened.

Window state

This determines how the window is to be displayed when opened. Normal is the configured size. Maximized, the window will fill the main screen. Minimized will show the window as a small bar on the bottom of the screen (as determined by Windows).

Window width/height

This is the "normal" size of the window, including any border and title bar. The border style and the "normal" size of the window determine the client width and height. Refer to the client size when working with importing to the background.
Caution: When changing the window size verify all the graphic elements on the window fully fit within the desired window size before adjusting the window size.

Animation

The window opening can be animated. The possible animations are: None, Center, Blend, Left to right, Right to left, Top to bottom, Bottom to top, Top left to bottom right, Bottom left to top right, Top right to bottom left and Bottom right to top left.

Animation speed

This property is the time in milliseconds, to play the animation.

Show window

View the window with the selected settings. When the window is visible, click the mouse button to close the window.
Note: The size of the window is limited to the monitor size.

Show mouse

This button will display a window. The window will show the mouse coordinates and monitor index while the window is open. It can be used to determine the top/left for a window position.

User level

At runtime the logged on user must have at least the level entered to view the window.

Automatically close

The number of seconds the window will be open. If the value is 0, the window will not automatically close.

Prevent Alt + F4

If this attribute is enabled, the window will not close when the Alt+F4 key combination is used.

Close on loss of focus

If this attribute is enabled, the window will close if the "focus" is moved to another window; another window becomes the active window.

Scripts

On Window open

If selected, the script will run once when the window is opened. It will run each time the window is opened. The script is executed as the next to last action for opening a window. The last action is to check the user level. If the logged on user level is not equal to or greater than the user level the window is destroyed.

On viewing

If selected, the script will run while the window is opened. The script will be put in the queue and executed. When the script is complete it will be placed in the queue and executed. This is repeated until the window is closed. The script should execute about every second. The number of windows open and other scripts executing can affect the frequency of execution.

On window close

If selected, the script will run once when the window is closed. It will run each time the window is closed. **Warning:** The window may be closed before the script executes.

Regions

Regions are used to remove the visibility of an area(s) of the window or create an elliptic shaped window. Script globals are used to define the desired regions.

The script global section name for a window is the window name and '_Region'.
Example: 'Main_Region'

The items define the region(s).

To create a round window create an item with a 'W' as the first character. Only the first item with a 'W' is processed.
Example: 'Window_Region'
The value of the item is the bounds of the elliptic.
Example 20,20,200,200 //left, top, width, height

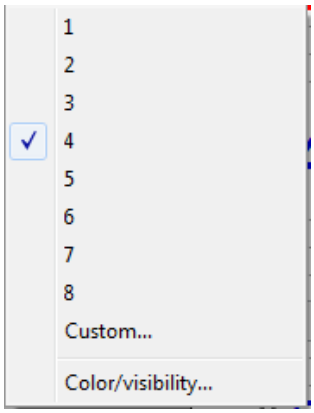
Hiding (removing visibility) an area of a window and be achieved with a rectangle, an elliptic or polygons. All items with the correct first character will be processed.

Rectangle, first character 'R'
Example 'R1'
The value of the item is the bounds of the rectangle.
Example 20,20,200,200 //left, top, width, height

Elliptic, first character 'E'
Example 'E1'
The value of the item is the bounds of the elliptic.
Example 20,20,200,200 //left, top, width, height

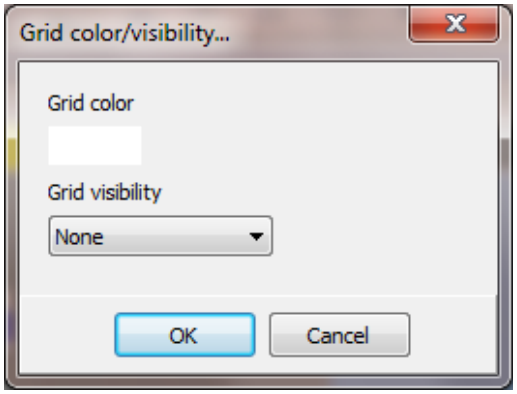
Polygon, first character ‘P’
Example ‘P1’
The values are pairs of points and at least three points are required. The polygon is automatically closed. The pair is the X (horizontal) and Y (vertical)
X1,Y1, X2,Y2, X3,Y3, Xn,Yn
Example 20,20,40,20,40,40 //point 1, point 2, point 3, point n

Grid



The drawing in the window can be drawn with the element points aligned to a grid. The grid can be set from 1 (no grid) to 100. The default is 4. Select the custom menu item to enter values greater than 8.

Color/Visibility



The grid points can be displayed. Select the desired color.

None:

The grid pixels are not visible.

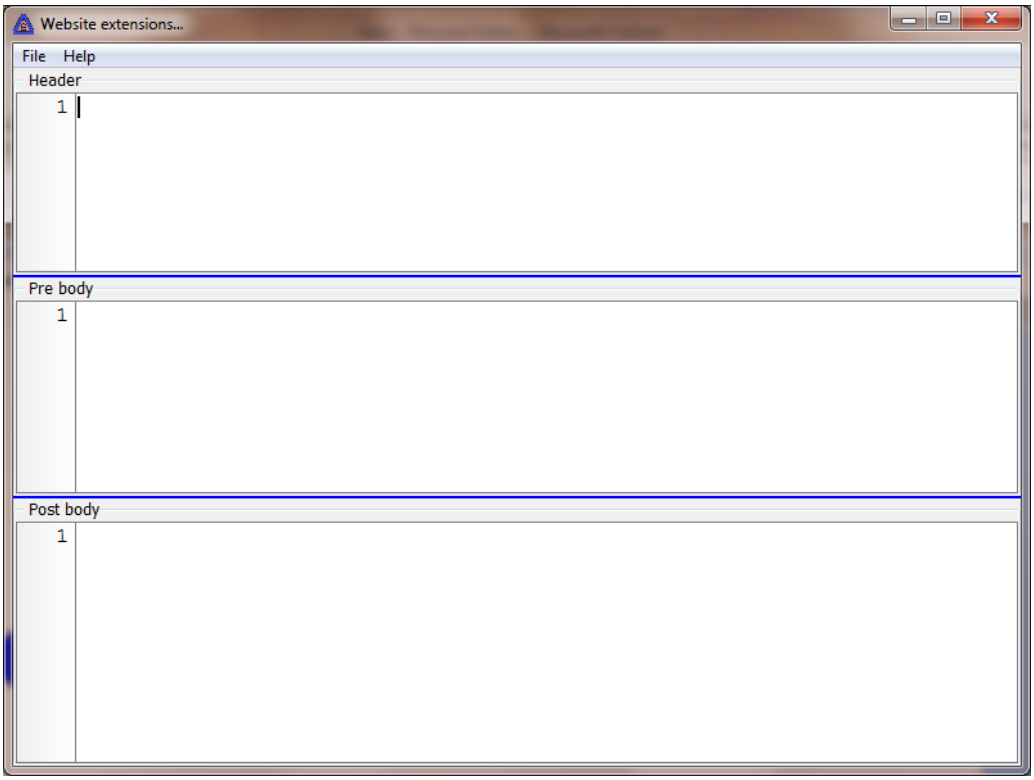
Above:

The grid pixels will be drawn after the background and before the elements are rendered. The graphic elements will cover the pixels.

Below:

The grid pixels will be drawn after the background and after the elements are rendered. The grid pixels will cover the graphic elements.

HTML (Advanced)



This is for website pages and provides for code to be inserted in the generated web page.

Header

The text in this field is placed/inserted before the **</HEAD>** statement in the header section of the page.

Pre body

The text in this field is inserted just after the first **<BODY>** statement.

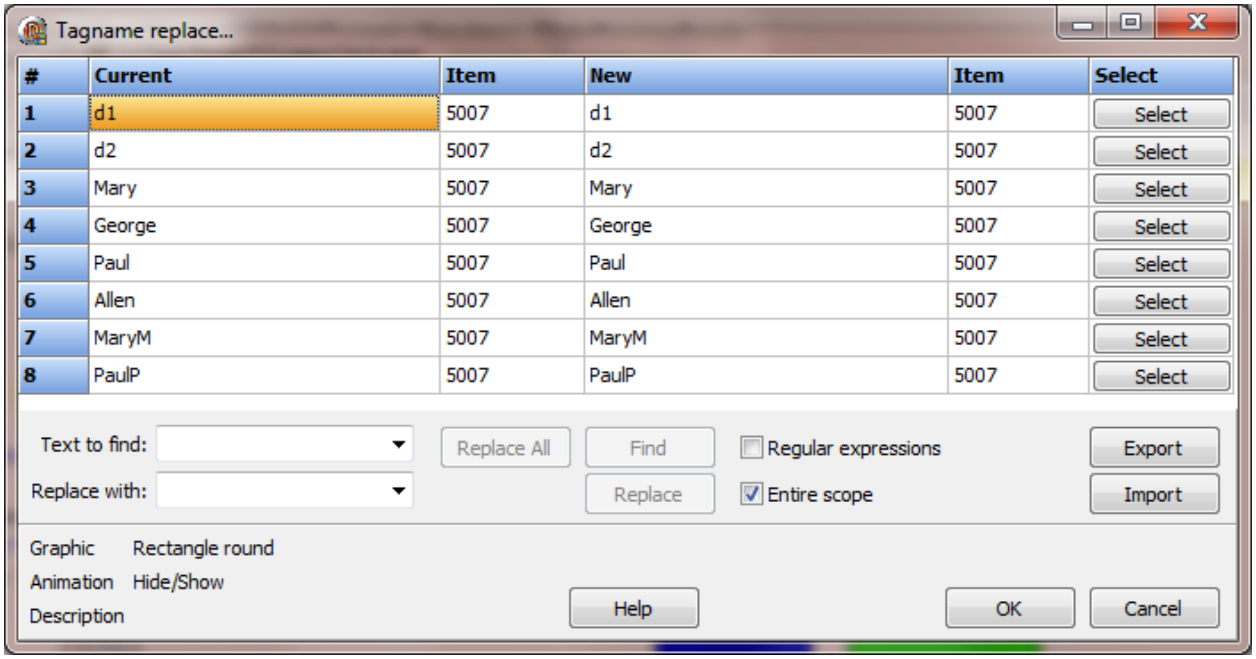
Post body

The text in this field is inserted just before the **</BODY>** statement.

Tagname Listing

This displays a list of all the selected elements animation tagnames. If no elements are selected it displays the animation tagnames for all the elements in the window. Selecting a tagname in the displayed list will select all elements using the tagname in an animation.

TAGNAME REPLACE



This displays a list of all the selected element animation tagnames. If no elements are selected the window displays the animation tagnames for all the elements in the window.

The fixed column, first column from the left, shows a graphic element number. A graphic element can have more than one animation. The animations for each element are grouped together. If a " symbol is displayed in the column, the animation belongs to the element above.

The first tagname/item from the left is the current configuration. The next set, tagname/item, is also the current tagname/item when the window is opened.

The second pair can be changed by typing in the cell or selecting the "select" button for the row. The item type for the new item must match the current item. **Note:** If the tagname is in a script it can be changed using the ”Select“ button.

The graphic element type and the animation type are displayed at the bottom of the window. The description field is for the current tagname for the selected row.

Text to find

When searching for text or replacing, this is the text string used.

Replace with

This is the text used to replace what was search for, above, excluding scripts.

Note: To delete an item from the “Text to find” or “Replace with” drop list, press and hold the CTRL key while selecting the drop list item.

Replace all

Select this to search all the new tagnames and replace the text found, excluding scripts.

Find

Select this to search, starting at the selected cell.

Replace

Select this to replace the found text, excluding scripts. After the text is replaced the search continues down the column.

Entire scope

When using the “Find” command the search starts at the current row and searches down the column. When the search reaches the end the search stops. If “Entire scope” is enabled the search will begin again from the first row and search (down) until the currently selected row is reached.

Regular expressions

Use this attribute to execute a more complicated search.

Character	Description
^	A circumflex at the start of the string matches the start of a line.
$	A dollar sign at the end of the expression matches the end of a line.
.	A period matches any character.
*	An asterisk after a string matches any number of occurrences of that string followed by any characters, including zero characters. For example, bo* matches *bot*, *bo*, and *boo*, but not *b*.
+	A plus sign after a string matches any number of occurrences of that string followed by any characters, except zeros. For example, bo+ matches *boo* and *booo*, but not *bo* or *be*.
[ ]	Characters inside square brackets match any character that appears in the brackets, but no others. For example, [bot] matches *b*, *o*, or *t*.
[^]	A circumflex at the start of a string inside square brackets means NOT. Hence, [^bot] matches any characters except *b*, *o*, or *t*.
[-]	A hyphen inside square brackets signifies a range of characters. For example, [b-o] matches any character from *b* through *o*.
{ }	Braces group characters or expressions. Groups can be nested, with a maximum number of 10 groups in a single pattern. For the Replace operation, groups are referred to by a backslash and a number, according to the position in the "Text to find" expression,

	beginning with 0. For example, given the text to find and replacement strings, Find: {[0-9]}{[a-c]*}, Replace: NUM\1, the string 3abcabc is changed to NUMabcabc.
\	A backslash before a wildcard character tells the editor to treat that character literally, not as a wildcard. For example, \^ matches ^ and does not look for the start of a line.

Export

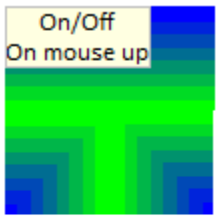
The tagnames can be exported to an Excel spreadsheet (XLS) or a Comma separated file (CSV).

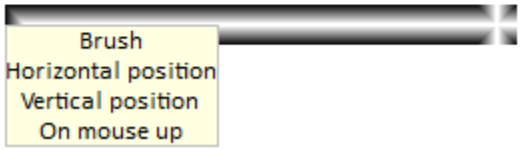
Import

The tagnames can be imported from an Excel spreadsheet (XLS) or a Comma separated file (CSV). The tagnames must be in the first column.

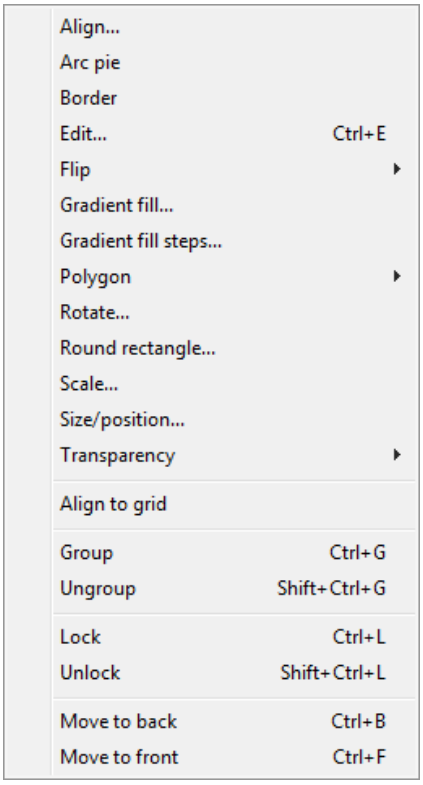
Display Animation Hint

When selected, a hint box will appear above each graphic element configured with an animation, listing all enabled animations. The hint box will disappear the next time the window is updated.



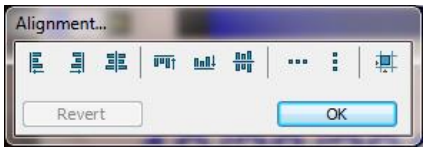


Objects



Align

When one or more graphic elements have been selected this menu item will be enabled. An alignment palette will appear.



The selections are align: left, right, horizontal centers, top, bottom and vertical centers. The next two selections are “distribute horizontal” and “distribute vertical”. The last selection is “align to grid”.

Arc Pie

When one or more arc graphic elements have been selected, this menu item will be enabled. This item toggles the "wedge" attribute. An arc is a section of a complete circle. When "wedge" is active a line will be drawn from the center point of the circle to both end points of the arc, and the "wedge" will be filled with the brush color and pattern as configured.

Border

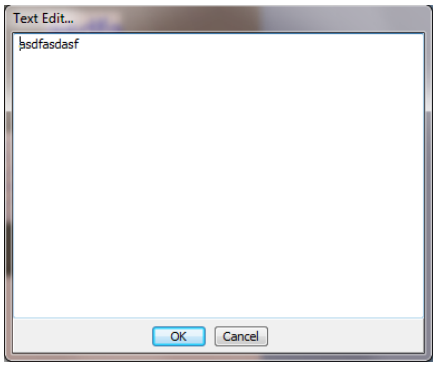
This will toggle the border attribute of rectangles. When enabled the brush is set to clear. The rectangle is drawn using the foreground and background color and the pen width and style. The foreground color and background color indicate the colors that appear on the left and top or the right and bottom of the rectangle respectively. The pen width parameter specifies the width of the frame.

To create a raised effect, the foreground should be a light color and the background should be a shadow color. To create a depressed effect, the foreground should be the shadow color and the background should be the light color. To create a beveled effect, draw a frame immediately inside another frame with the colors reversed.

Edit

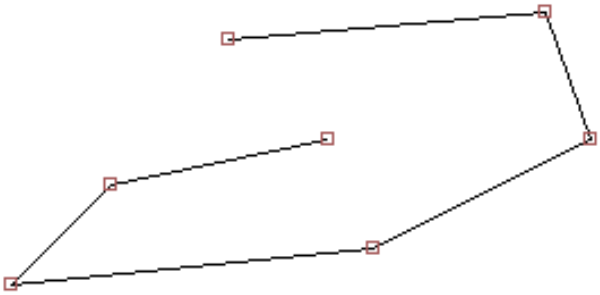
Some objects have additional attributes that can be configured. Some of the object editors are discussed in “Complex Objects” and “Button Objects”.

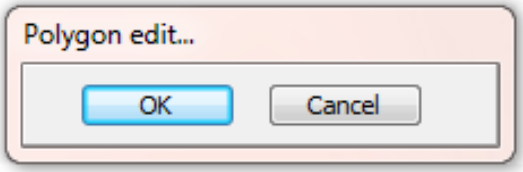
Text



When one text graphic element has been selected this menu item will be enabled. Select edit to change the text.

Polyline, Polyline Freehand





When one polygon graphic element has been selected this menu item will be enabled.

A polyline is a group of lines. The first line is two points. All subsequent lines use the last line end point as the start point of the next line.

The points can be moved.
1. Press the left mouse button in a point selection rectangle and without releasing the mouse button move the mouse and the point will move.

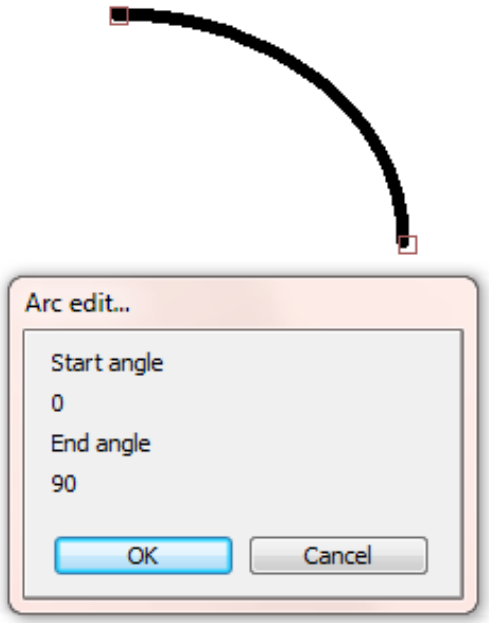
2. While holding done the "shift" key click the mouse button in the desired point selection rectangles. This allows for the selection of multiple points. Once the selection is made press the left mouse button in one of the selected point selection rectangles and without releasing the mouse button move the mouse and the points will move.

3. When a point or points have been selected the arrow keys can be used to move the points.

To delete a point, select the point and press the "DEL" key. The polygon must have at least two points. If the line consists of only two points the points cannot be deleted.

To add a point, click the left mouse button on the line at the location of the desired new point.

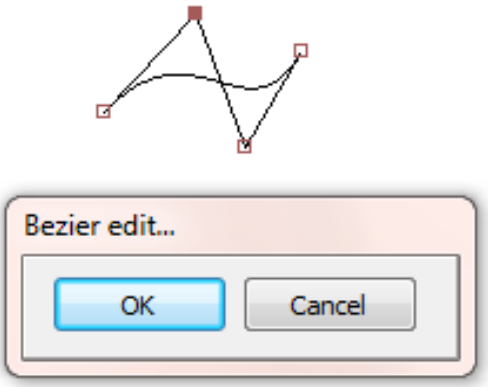
Arc/Wedge



When one arc graphic element has been selected this menu item will be enabled.

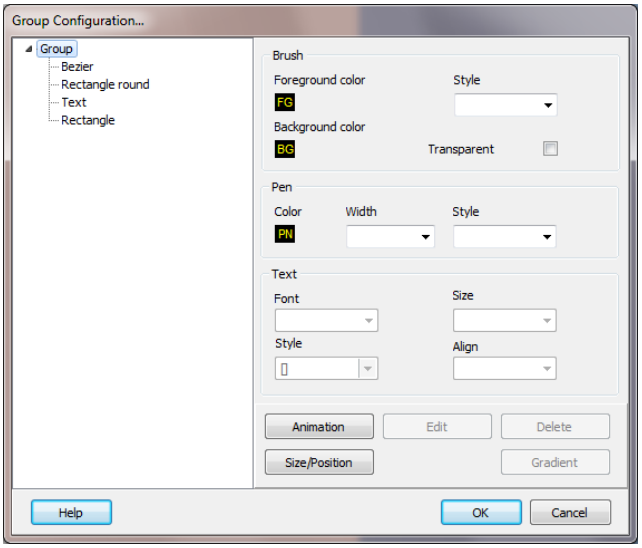
An arc is a portion of a circle with a start angle and an end angle. Press the left mouse button in a point selection rectangle and without releasing the mouse button move the mouse and the endpoint will move.

Bezier Line



When one Bezier graphic element has been selected this menu item will be enabled. When editing this control it responds like a polygon.

Groups



The group editor is composed of several sections.

The left section (Group) list all the elements of the group by element type.

The right sections (Brush & Pen) list attributes that are commons to most elements.

The right section (Text) only applies to those graphic elements that contain text.

The right bottom section (Animation, Edit, etc. buttons) is used for extended editing of the graphic element of a group.

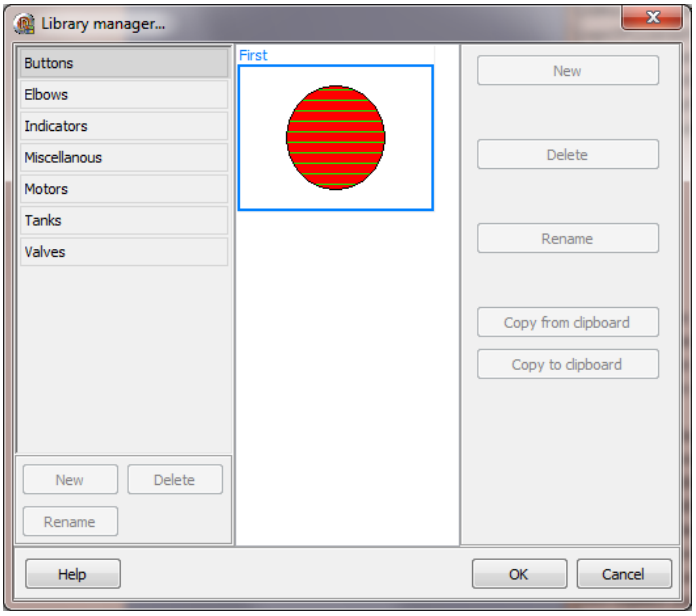
When an element is selected from the 'Group' section' the other sections update and display the values of the selection graphic element.

Some elements or element attributes cannot be edited why grouped.

Gauge and Button

The gauge and button editors are covered in the section "Complex objects" and "Button objects".

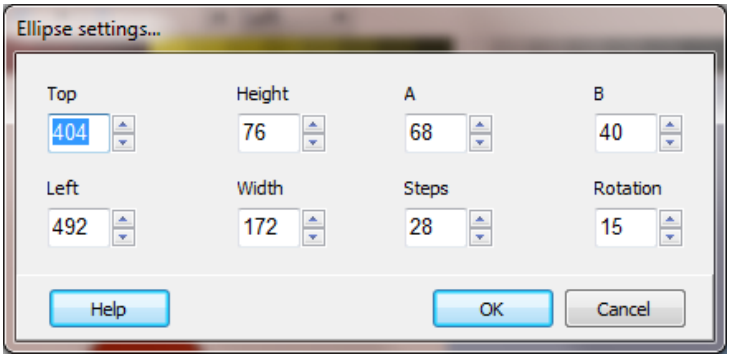
LIBRARY



When one library graphic element has been selected this menu item will be enabled. This will display the library manager window to allow editing of the library elements.

The library feature is discussed later in this section.

Ellipse



Top/Left

Defines the top and left position on the window

Height/Width

Defines the height and width of the element.

A/B

Defines the two end points of the ellipse.

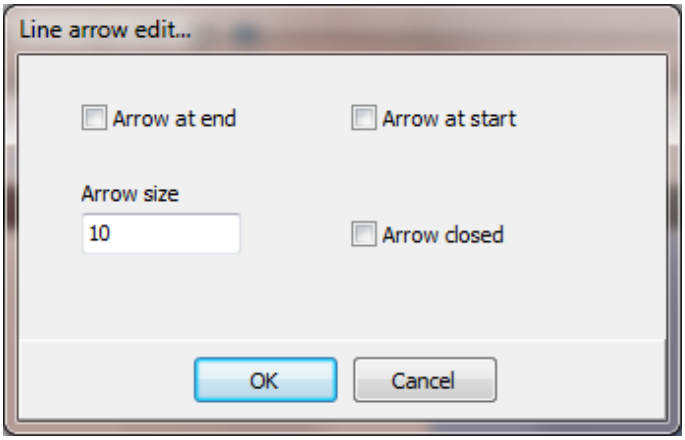
Steps

Defines the number of segments used to draw the ellipse. Larger numbers display a smoother edge to the ellipse.

Rotation

The amount the ellipse is rotated on the center axis.

Line



An arrow can be drawn at the beginning and end of the line. The arrow size can be adjusted. If “Arrow Closed” is enabled a line is drawn to enclose the arrow. The brush attributes are applied to the interior of the arrow.

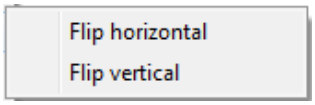
Bitmap

The bitmap editor is covered here.

Pipes

The pipes editor is covered here.

Flip



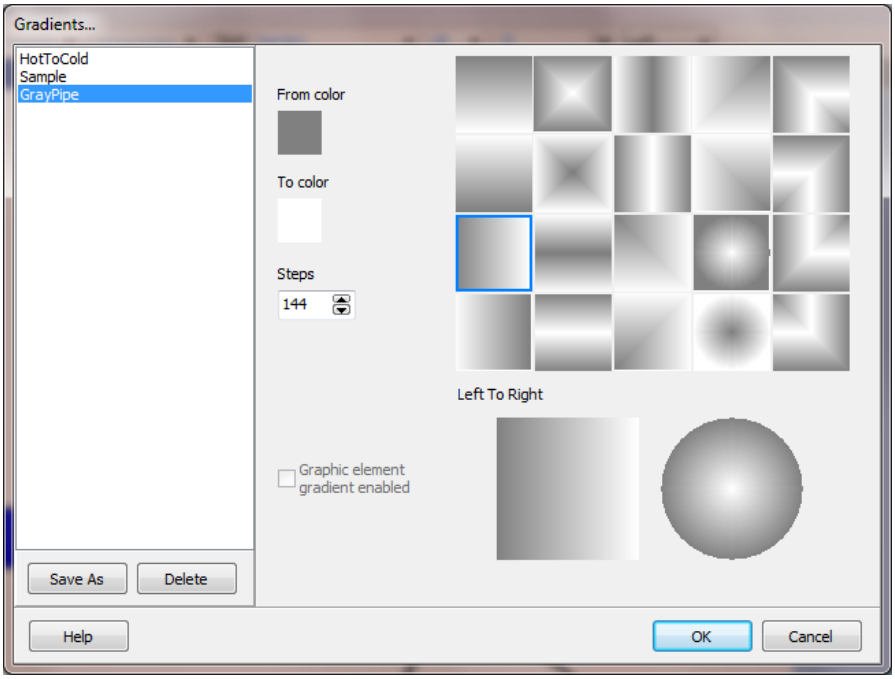
When one or more graphic elements have been selected this menu item will be enabled. The element end points will be flipped vertically or horizontally.

If more than one element is selected the elements will also be flipped in relation to each other.

Not all elements contain end points to be flipped. A flipped circle looks the same after it is flipped as before it was flipped. Selecting and flipping two circles will cause a difference in when comparing the elements to each other.

Library elements do not have end points to flip.

Gradient Fill



Some graphic elements (rectangles, circles and polygons) can be filled with a gradient fill.

From Color

This is the starting color of the fill. It is also the foreground color of the element.

To Color

This is the ending color of the fill. It is also the background color of the element.

Direction Grid

The grip displays the gradient fill selections.

Steps

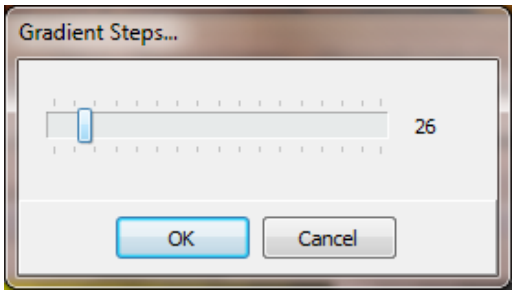
This is the number of steps or bands created to fill the element. Some direction selections double the number of bands to fill the element. Note: Use the lowest number of steps to achieve the desired effect. The lower the steps count the faster the element is drawn.

Graphic element gradient enabled

This must be checked for the element to be displayed with a gradient.
Note: To fill a polygon the polygon must be closed via the polygon menu.

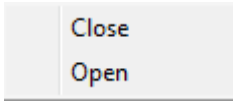
The list box on the left is used to store gradient settings that can be applied to elements. If one of the list elements is highlighted it is the current setting for the graphic element. Select the "Save As" button and supply a name to save the current settings to a file. Select the "Delete" button to delete one of the saved settings. This does not affect any elements.

Gradient fill steps



When one or more arc graphic elements have been selected this menu item will be enabled. This allows for setting the number of steps a gradient fill uses to fill the element. Refer to Gradient Fills for more information.

Polygon open/close



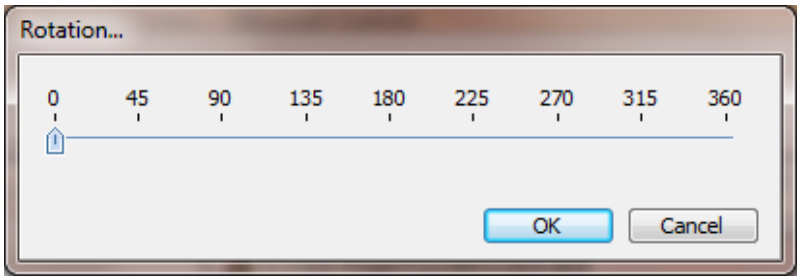
Close:

If the "Close" menu item is selected a line will be drawn from the first line end point, the start point, and the last line end point, and the result will be filled with the brush color and pattern. If after creation the final graphic element has the start and end point connected and need the brush color and pattern to fill the result the "Close" menu item must be selected.

Open:

The opposite of "Close".

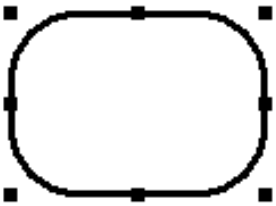
Rotate

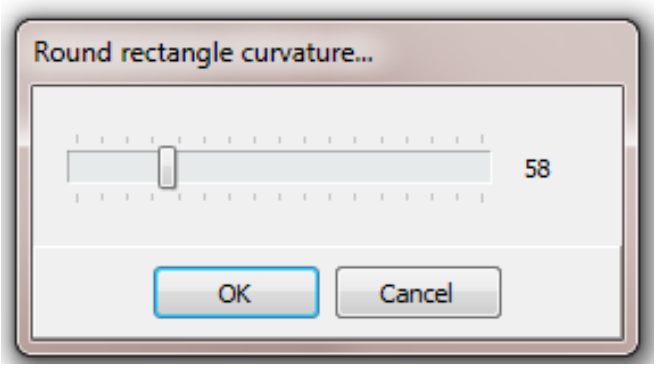


When one or more graphic elements, which can be rotated, are selected, this menu item will be enabled. Text, polygons and bitmaps can be rotated. After the menu item is selected a dialog window will providing for rotation. Move the slider or press an arrow key and the selected elements will rotate.

Note: To provide rotation animation at runtime a copy of the original bitmap is saved. If the size of the bitmap is changed at design time and afterward rotated at design time the size of the element will revert to the original size of the bitmap.

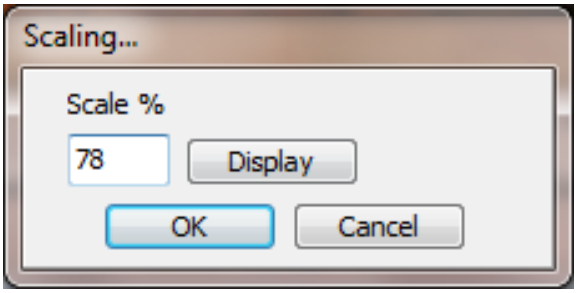
Round rectangle





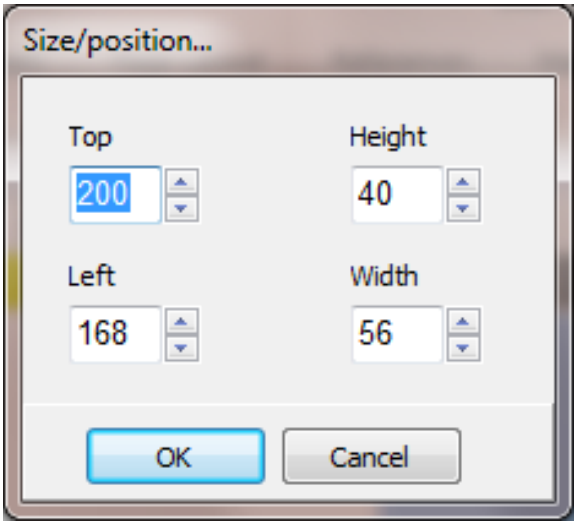
When one or more round rectangle graphic elements have been selected this menu item will be enabled. A dialog will appear to set the value.

Scale

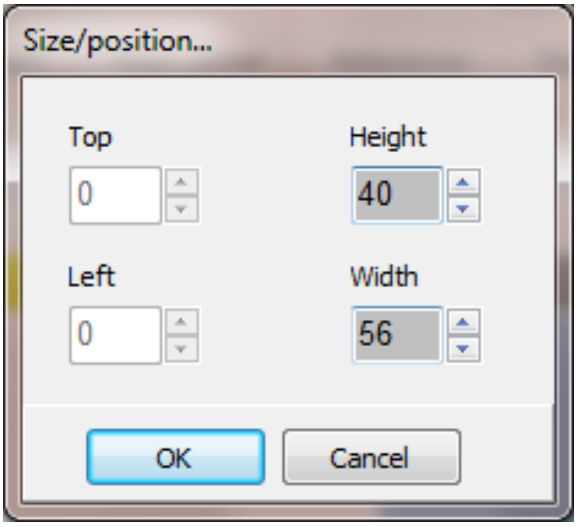


When one or more graphic elements have been selected this menu item will be enabled. A dialog will appear to scale the elements. The scaling is in percent Values greater than 100 increase the size of the elements. Values less than 100 decrease the size of the elements. Selecting "Display" scales the elements, the objects are rendered at the new size and the scale factor is set back to 100. Changing the scale factor will scale the elements using the current size.

Size/Position

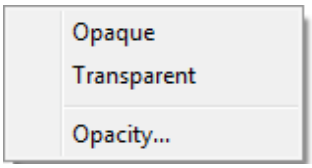


When one graphic element has been selected this menu item will be enabled. A dialog will appear to allow entering the elements position or size.



When more than one graphic element has been selected the top and left cannot be altered. Changing the height or width will change all objects selected.

TRANSPARENCY



When one or more graphic elements have been selected this menu item will be enabled. An element can have a foreground color, a background color, a brush pattern and can be transparent or opaque. These four attributes work together to render the element.

For a non-filled object set the brush style to "Clear".

If the brush style is "Solid" only the foreground color is visible.

If the brush style is not "Clear" or "Solid" then the background color can be visible **IF** the transparent attribute is set to "Opaque".

If the transparent attribute is set "Transparent", the brush will draw the pattern with the foreground color and the background color will not be drawn. (The object will be partially filled with the brush pattern.)

See Opacity below.

Opacity

For objects (rectangles, round rectangles, circles, line polygons, free polygons and, text) that support transparency this is a value between 0 - 255 that sets the opacity.

0 = totally transparent. The background will show through.

255 = totally opaque. The background will not be visible.

Align to grid

The selected objects will be aligned to the grid using the current grid setting. The default setting is configured in the project settings.

Group

The selected objects will be grouped to form a new "group" object. This action cannot be undone. The group can be “ungrouped.

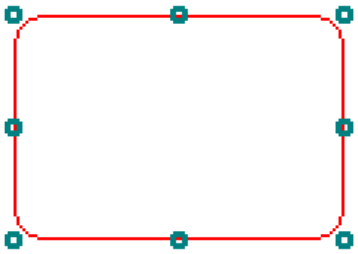
Ungroup

The objects in the selected group object will disconnected from the group and the group object will be deleted. This action cannot be undone.

Lock/Unlock

An object can be locked in place and cannot be moved or resized until the object is unlocked.

1. When an object is locked the 8 selection rectangles that appear when an object is selected will be 8 circles.
2. Locked objects included in a group are unlocked when the group is created. The group can be locked.
3. Objects in a locked group can be resized and moved using the "group editor".
4. Locking only applies during design time. Locking does not apply to objects during runtime.



Move to front/back

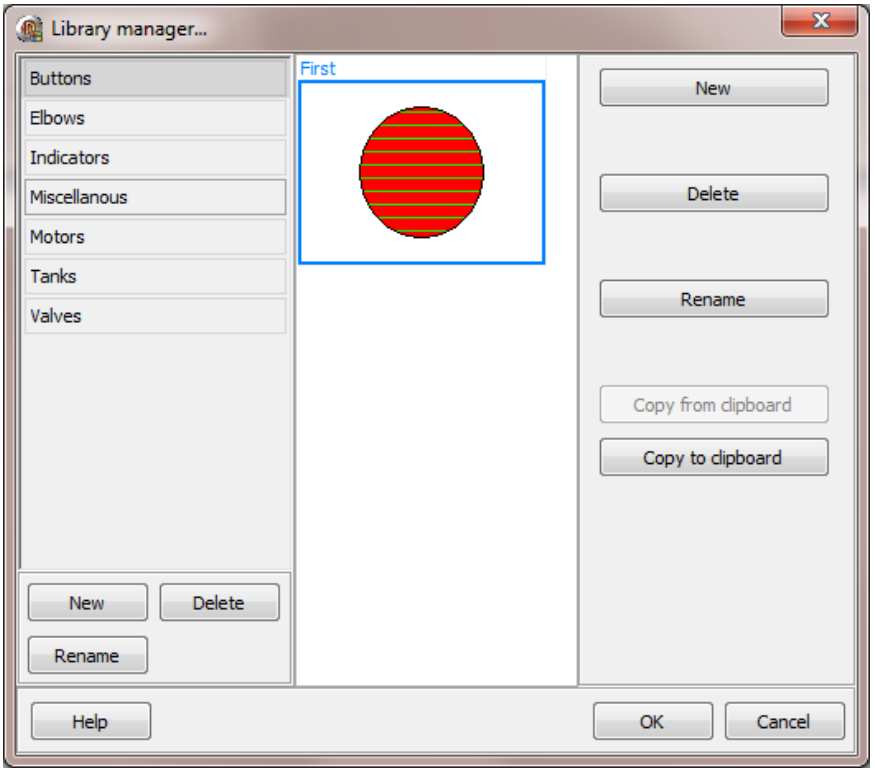
When one or more graphic elements have been selected this menu item will be enabled. The elements are drawn in the order of creation. The most recent element is drawn last and if the objects overlap, it appears to be on top of any other elements.

Selecting "move to front" moves the selected element to the end of the creation (last element created) list and causes it to be drawn last.

Selecting "move to back" moves the selected element to the beginning of the creation (first element created) list and causes it to be drawn first.

Note: Not all graphic elements respond to "layering" for rendering the element and these commands may not cause a change in the visual ordering.

LIBRARY



Library elements are collections of elements. When the library element is referenced in a window it is rendered the same in all uses. Changing the library element changes the image in all uses.

Note: The library only contains HMI graphic elements. To place an image in the library, first paste it into a graphic window. Then continue from 2.

Note: Grouped elements are not supported. Ungroup all elements before copying to the clipboard.

Create a new library element:

1. Create an element(s) or copy existing graphic elements.
2. Copy the HMI element(s) to the clipboard.
3. Select Library/Edit.
4. Select the group, left column, to contain the new library element.
5. Select "New" button and enter a name. The name must be unique for the group. Or select the library element to change.
6. Select the display rectangle in the middle containing the name entered.
7. Select the "Copy from clipboard" button.
8. A scaled version of the contents of the clipboard will be displayed.

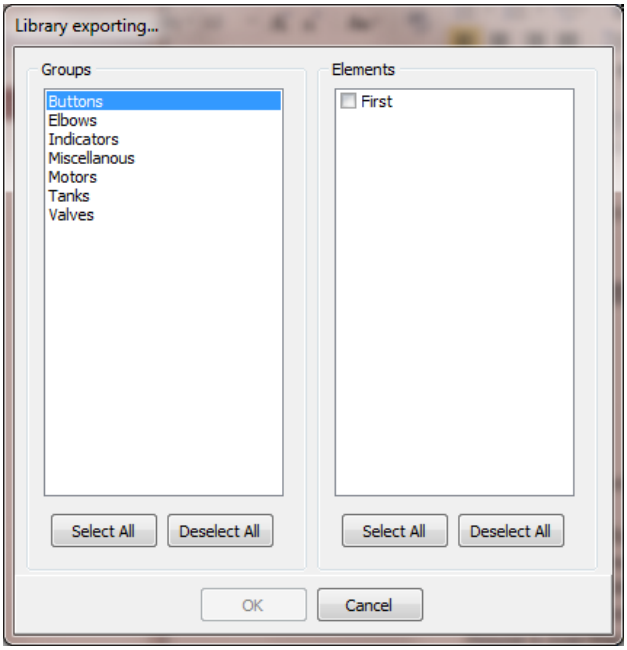
To use an element in a window:

1. Select Library/Edit.
2. Press the left mouse button in the desired element.
3. While holding down the mouse button, drag the mouse to the window. The cursor will change to the "drop" image when the mouse is over the window and it can accept the drop.
4. Release the left mouse button and the library element will appear in the window.

Library element animations are limited.

To edit the contents of an element:
1. Select the element and select the "Copy to clipboard" button.
2. Paste the clipboard into a window.

Library Export

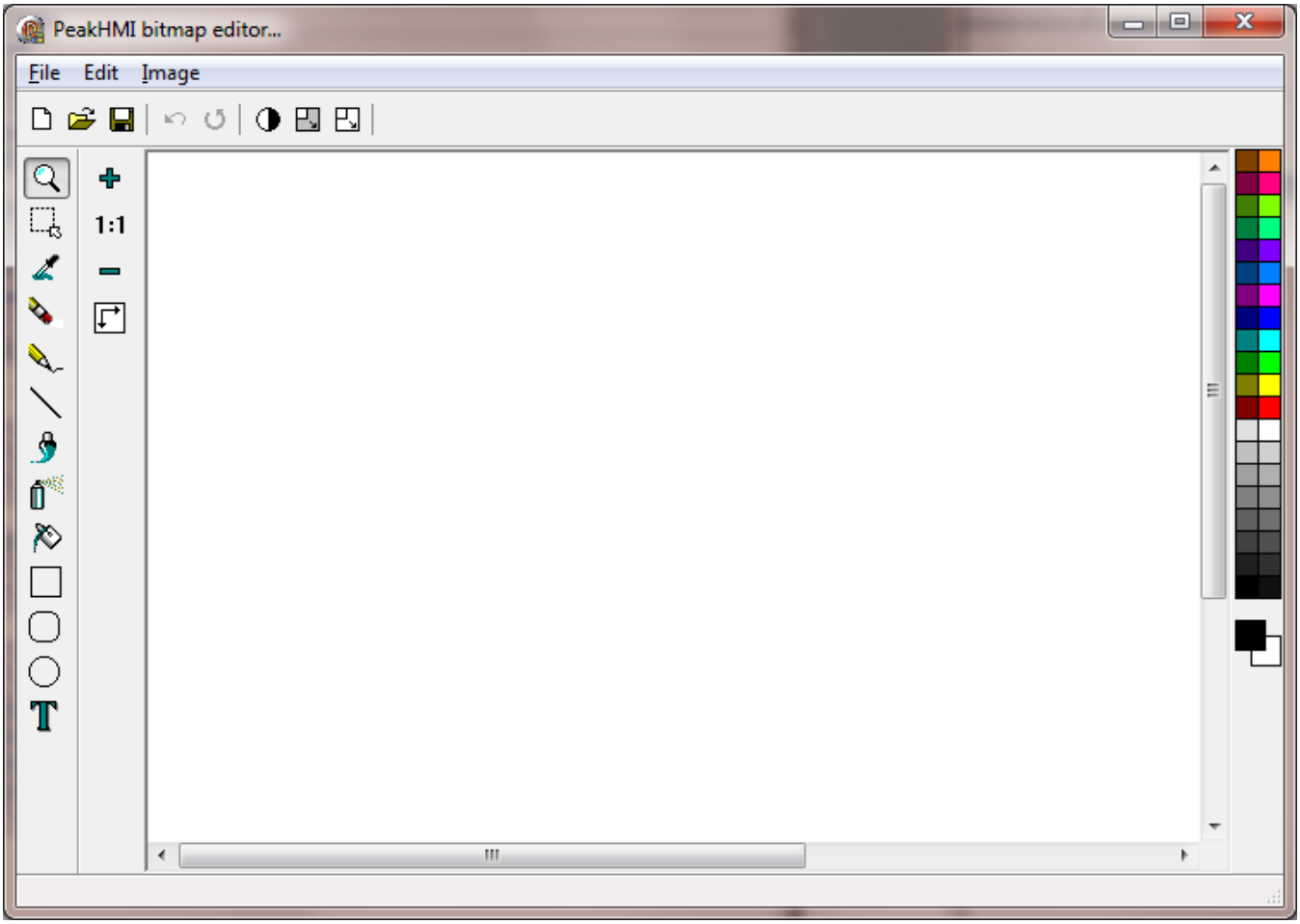


Select the elements to be exported to a file.

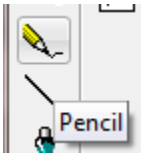
Library Import

Select a file of library elements that were exported. This will import the library elements into the project.

BITMAP EDITOR

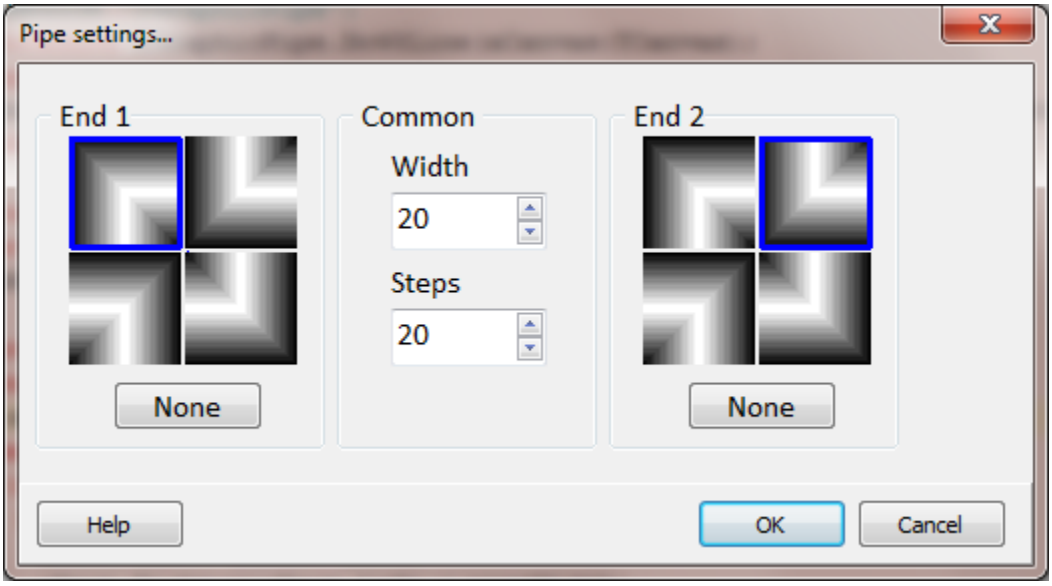


The bitmap editor is a general purpose bitmap editor that provides basic bitmap editing. The tools are on the left side, color selection is on the right side.



Move the mouse over an icon and a “Tool tip” will appear.
Left click a color to set the foreground and right click a color to change the background.

PIPE EDITOR



Each vertical or horizontal pipe can have an end elbow. Diagonal pipes have each end tapered for attaching to a vertical/horizontal pipe to create a 90° angle when connected to another pipe element.

The pen color is the outside color and the foreground color is the inside color.

Width

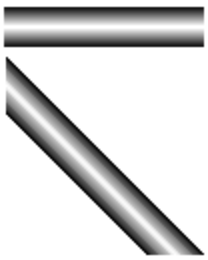
The width of the pipe.

Steps

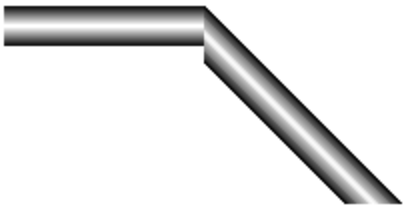
The number of gradient steps. The same value for width and step tends to create a nice effect.

Diagonal pipes

There are several reasons most pipe graphics are all vertical/horizontal; appearance being the main reason.
All measurements are in pixels. Most computer monitors use square pixels. The diagonal length of a square is 1.414213562373095 (square root of 2) times longer than the sides of the square.



Example, a 100 pixel horizontal pipe and a 100 pixel diagonal pipe.
The diagonal pipe looks longer, because it is longer. The width setting for both pipes is 20 but, the diagonal pipe looks slightly wider, because it is wider.

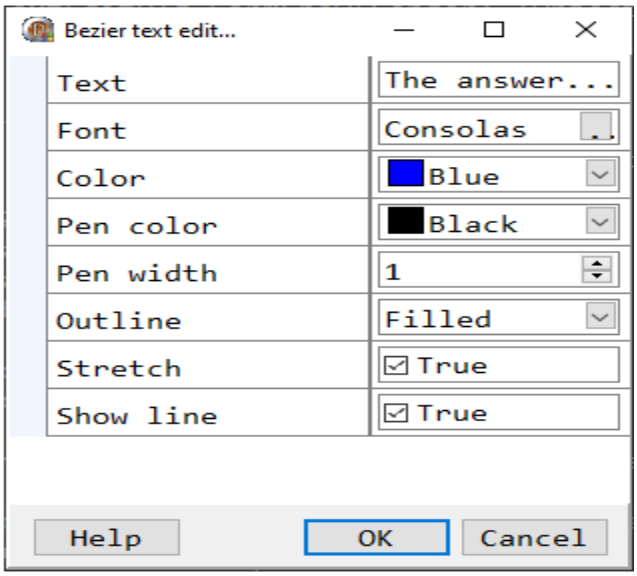


Line up the top edges, to make a connection, and the size difference is very apparent.

To create a matching end, the solution is to make a wider horizontal pipe or a thinner diagonal pipe. The steps setting was set to match the width setting.

A pipe with a 90° pipe end.

BEZIER TEXT EDITOR



Text The text to fit the Bezier line.

Font Font attributes for the text. The text attributes can also be set using the "Brush/text" toolbar.

Color The text color, same as the foreground color.

Pen color The pen color if the "Outline" is not none. Also the color if "Show line" is enabled.

Pen width Same as "pen color" for the pen width.

Outline The text can be:

- **None** Normal appearance of text
- **Clear** The text is rendered with only the pen, along the outside of the character.
- **Filled** The text is normal and the pen is used to draw an outline around the text characters.

Stretch If enabled, the text is rendered from the starting point to the end point. **Note**: Due to font property variations, it might be better to use stretch and adjust the size of the Bezier bounding rectangle to achieve the desired character spacing appearance.

Show line Render the Bezier line using the pen settings.

SCRIPTING

Functions

The example code for the functions below is in Pascal unless an example in Basic is needed for successful use of a function using Basic.

All the name, parameters, results, for Pascal and Basic are the same. The difference is language syntax.

Examples:

Pascal uses ":=" to assign a vale to a variable and Basic uses "=".
Pascal uses a single quote (') to specify a string and Basic uses a double quote (").
Pascal uses a ";" to terminate a line and Basic does not.

```
values:=RV(['tagname 1']);     //Pascal
values=RV(["tagname 1"])       //Basic
```

Pascal does not have an "end if"

```
if x = 0 then beep;            //Pascal
if x = 0 then beep end if      //Basic
```

The Basic scripting engine will accept an "if statement" without an "end if" if the statement is a single line.

```
if x = 0 then beep             //Basic, accepted

if x = 0 then
 beep                          //Basic, must have end if
end if
```

Note: Like Pascal, subroutines in a script must be defined before being used. This example will not compile.

```
Sub B
 X = A
end sub

function A as integer
 return 123
end function
```

This example will compile:

Use a “Forward”, for “A”, before “B” is defined.

```
function A forward

Sub B
 X = A
end sub

function A as integer
 return 123
end function
```

Quick links

Functions
ODBC In/Out

Point Scripts
Result
Warning

Graphic element scripts
Functions

Runtime
Script Globals
Script Storage
Script Structure
Script Timers
Script Examples

The HMI is designed to not require any scripting. That continues to be a design goal as new features are added. Sometimes, scripting must be used when the HMI does not contain a built in function or the task is not something we could predict a user would need. And some tasks are suited for scripting because of the nature of the task.

Pascal and Basic scripting languages are supported. Using scripting should be fairly simple as it is mainly used to issue commands and combine data from different external devices into internal host tagnames. All graphic animations are handled via object configuration. A graphic scripting animation feature for those rare instances it is needed, is provided.

At runtime the HMI has several scripting engines active. General scripts are loaded at runtime start and are compiled to memory (graphic scripts are excluded). Any compile time errors are logged in the "Event Log". Editing the scripts at runtime will not change the scripts in memory. The reload button must be selected or stop and restart runtime to change the "in memory" scripts.

There are three main scripting engines running.

The first engine handles **commands** from the onMouseDown and onMouseUp configurations for objects.

The second engine handles OnOpen, OnDuring, and OnClose for the **runtime** environment and user configured windows. Scripts are added to a queue. If the script is present in the queue and an attempt is made to add it again the attempt is ignored.

WARNING: Calling this script engine from another script or a mouse event and the called script causes a modal window to display, the processing of other scripts for this engine are paused until the called script is complete. The window OnDuring script is called once per second for each open window.

The third engine executes **point scripts**. It executes rapidly and continuously.

The graphic scripting engine is used as needed for graphics. Other scripting engines are used for specific features (e.g. Report generation)

SCRIPT STORAGE

Scripts are stored in the path <project>\Scripts. All scripts for a project must be in this path. Scripts can be in subdirectories without limit on the structure.

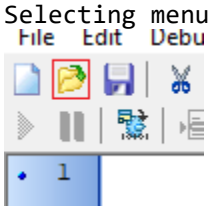
```
Example:
<project>\Scripts          --must be present
<project>\Scripts\Pump1
<project>\Scripts\Pump2
<project>\Scripts\Level 1\North
<project>\Scripts\Level 1\South
```

Scripts can be at the root level <project>\Scripts or any sub level.

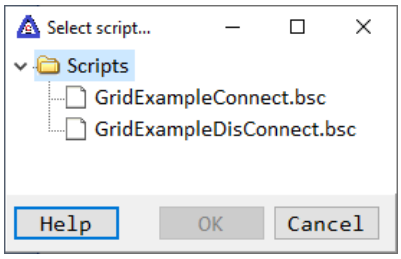
When a script is selected in a configuration screen the relative path of the script selected is saved.

Point scripts and graphic scripts are stored in the point configuration and graphic element.

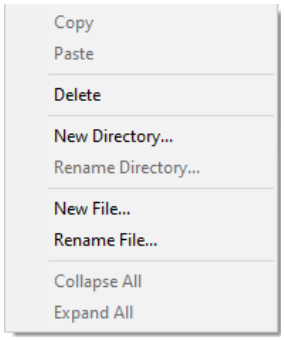
Selecting menu “File/Open” or the file open icon:



Will display the “Select script...” dialog.



Right click the “tree” and the popup menu will appear:



SCRIPT STRUCTURE

See here for “Basic” structure.

Script structure is made of two major blocks:
a) procedure and function declarations and
b) main block.

Both are optional.

There is no need for a main block to be inside begin..end. It could be a single statement. Some examples:

SCRIPT 1:

```
line of script 1
line of script n
```

SCRIPT 2:

```
OpenAWindow;
```

SCRIPT 3:

```
procedure OnMouseXXX;          //this format for graphic scripts when OnMouseXXX callbacks
                               //are used AND graphic scripting animation is required
begin
 CallSomething;
end;

begin
 //area for main code of script
end;
```

SCRIPT 4:

```
begin
 CallSomethingElse;
end;
```

SCRIPT 5:

```
function MyFunction:string;
 begin
  result:='Ok!';
 end;
```

SCRIPT 6:

```
CallSomethingElse;
```

SCRIPT 7:

```
values:=ReadValue(['tag1',5000,'tag2',5000]);
result:=values[0] + values[1];
```

A "result" in the main block is only used in point scripts. For “Basic” use “Return”.

Like in Pascal, statements should be terminated by ";" character. Begin..end blocks are allowed to group statements.

Basic Structure

Script structure is made of two major blocks:
a) “sub” and “function” declarations and
b) main block

Both are optional.

SCRIPT 1:

```
line of script 1
line of script n
```

SCRIPT 2:

```
OpenAWindow
```

SCRIPT 3:

```
sub OnMouseXXX                //this format for graphic scripts when OnMouseXXX callbacks
 CallSomething                //are used AND graphic scripting animation is required
end sub
```

SCRIPT 4:

```
CallSomethingElse
```

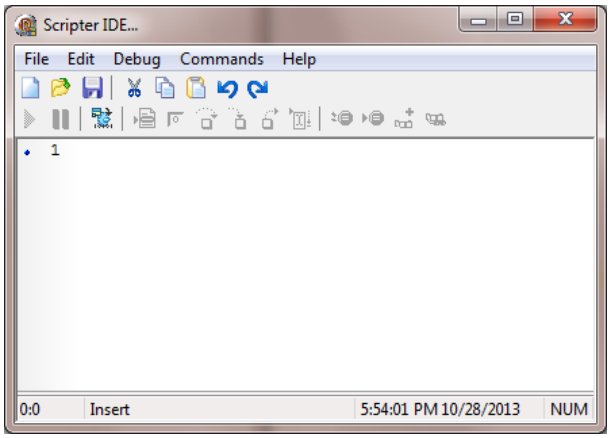
SCRIPT 5:

```
function test
  return  "ABC"
end function

ShowMessage(test)
```

SCRIPTING IDE

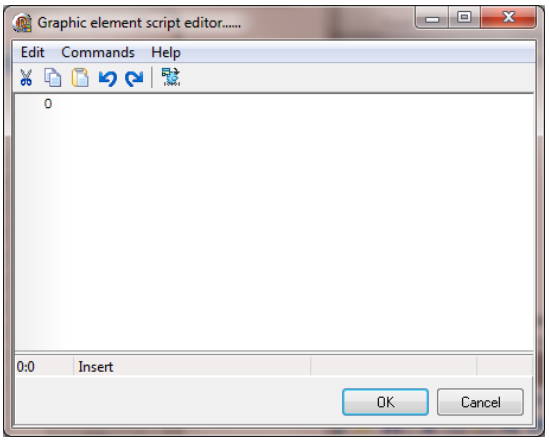
The HMI has one IDE (Integrated development environment) configured differently depending on the type of script opened and if it is at design time or runtime.

This is the full IDE when the script is regular script. The other types of scripts are point scripts and graphic scripts. Moving the mouse over each icon, a tooltip will appear naming the icon.



The “points” looks like the one above but, has fewer commands under the command window, among other slight differences.

This is the IDE when the script is a graphic element. Compared to the one above it has fewer icons and when accessing commands it has fewer commands. It also has some features that are only for graphic scripts.



Point, full scripts and graphic scripts can be edited and debugged at runtime. The script can be changed, the window closed, reopened and the edited script will run.

SCRIPT GLOBALS

Script globals are storage locations for script variables. A script global can contain numbers or strings. Script globals are so named because all scripts can read/write script globals via the GlobalGet and GlobalSet script functions.

Some of the possible uses of a script variable:

1. Need to be referenced in more than one script.
2. Do not need the monitoring functions of host tags.
3. Script globals are retentive between executions of a script.
4. Script globals are retentive between executions of runtime. (If Runtime is stopped via the "Quit" command.)
5. Save the result of a StringGet call.
6. Save the result of an ODBC read command.
7. Source of data for an ODBC write command.

Script globals should not be used to replace local variables that do not need any of the attributes listed.

The section name and item names are case sensitive.

Note: It is good practice to not use a period (.) in the section or item name. Using a period may cause parsing problems in other parts of the HMI.

Script globals can contain a string.

Script globals are referenced with a section and item pair. The section names must be unique. The item names within a section must be unique.

Example:

Pump 1
 StartedCount

Pump 2
 StartedCount

Pump 3
 StartedCount

SCRIPT TIMERS

These timers can be used to execute scripts after a set time has elapsed or the timer fields can be monitored in running scripts and some action taken based on the timer state.

The time base is seconds.

The preset and accumulator limit:
2,147,483,647 seconds = 35,791,394 minutes = 596,523 hours = 24,855 days = 68 years

Timer Fields

Name	Type	Description
TT	Boolean	Timer enabled and not done. (Timer timing)
DN	Boolean	Timer enabled and accumulator >= preset. (Timer done)
EN	Boolean	Timer enabled
ACC	Integer	Accumulator
PRE	Integer	Preset
ARS	Boolean	Timer Auto Restart

Auto restart

When enabled, when the timer is timed out (ACC is greater than/equal to the PRE) the timer begins timing again. The done (DN) field is never set true.

Script

A script to execute when the timer times out. (optional)

Engine

There are several script engines but two, the "Commands" and the "Runtime" (also known as "During") are used for user actions and general purpose scripts (e.g. running a report or calling a custom log command).

Commands This engine is used to process user commands and scripts that might "block" other scripts from executing. "Blocking" occurs when a script calls a function, such as "ShowMessage", and the script halts processing until the user closes the message window. This blocks the current script from running (until the function call returns) and all scripts that might be waiting to be executed from running. This allows flow control in scripts and possibly from script to script.

Runtime This engine is used to process scripts like "While window open" or "During runtime". Blocking calls must **not** be made in the scripts executed by the runtime script engine.

Normally, the "Runtime" engine would be used for scripts executed via a timer event. The "Commands" engine can be used as long as the blocking action is understood.

Use caution when using auto-restart and scripts. If a script is running and blocked (e.g. waiting for user input) the script will be queued once, regardless the number of times the timer completes.

Description

A user field to store information about the timer. (optional)

Timer operation

When the enabled field (EN) becomes true:

1. The timer begins timing
2. The timer timing (TT) field is set true.
3. The accumulator (ACC) begins incrementing.

When the ACC is greater than/equal to the preset (PRE):

1. The DN field is set true.
2. The TT is set false.
3. The ACC stop incrementing.

When the enabled field (EN) becomes false:

1. The ACC is set to zero.
2. The DN is set false.
2. The TT is set false.

SCRIPT FUNCTIONS

This is the main list of script functions. There are more function listed at the end that are not covered. They are rarely used and if help is needed contact support.

These are the HMI specific functions. The functions are listed in alphabetical order. Each function is defined below.

Alarm

"Alarm" is an object with properties and functions. This object is used to fetch information about an active alarm in the master alarm list or one of the group alarm list.

The syntax is alarm.<property or function name>. For example, alarm.AlarmKind returns the alarm type.

The alarm properties and functions can be accessed in the scripting IDE via the menu or typing "alarm." (The word "alarm" followed by a period.) A popup menu will appear listing the properties and functions. Also, placing the cursor after the text "alarm." and pressing the CTRL and SPACE BAR will display the popup menu.

The alarm object holds the information (really a copy) of one alarm. The alarm object is loaded with the data from an alarm via the Load or LoadGroup function.

Example:

```
success:=Alarm.Load(2);
```

This loads the second master alarm list item. Newer alarms are added to the end of the list. The alarm list is zero (0) based.

```
success:=Alarm.LoadGroup(5,3);
```

This loads the third alarm list item from alarm group five. Newer alarms are added to the end of the list. The alarm list is zero (0) based.
The groups are numbered 0 - 5000. Group 0 is the master alarm list.

If the function was able to copy the alarm data from the list, the "success" result will be true; otherwise, the result will be false. Always check the result of the function call.

Example:

```
success:=Alarm.Load(2);
if success then
begin
end;

if Alarm.Load(2) then
begin
end;
```

Properties/functions

Name	Description
Acked	The alarm has been acknowledged.
AckTime	The time the alarm was acknowledged. (0 = not acknowledged)
AlarmArea	See point definitions/alarm settings
AlarmGroup	See point definitions
AlarmKind	0 = Lo Lo or Falling, 1 = Lo or Rising, 2 = Hi, 3 = Hi Hi
AlarmTime	The time the alarm triggered.
ConditionLogic	See point definitions/alarm settings
ConditionStatus	See point definitions/alarm settings
ExceededText	See point definitions/alarm settings
ExceedValue	The alarm setpoint.
IsBlocked	The state of the alarm block. False = not blocked, true = blocked.
IsDigital	True the alarm is a falling or rising alarm.
ListCount	The number of alarms in the specified list of the Load or LoadGroup function. This value is always returned. -1 = the list index is invalid.
NormalStatus	See point definitions
PFS	The point PFS (percent of full scale).
PointDescription	See point definitions
PrimaryArea	See point definitions/alarm settings
Priority	See alarm priority.
QuickHelp	See point definitions/alarm settings
State	0 = alarm is not active, 1 = alarm is active, 2 = alarm is active and has been acknowledged.
Tagname	See point definitions
ValueAtAlarmPoint	The value of the point when the alarm triggered.
Load	Loads the alarm object with an alarm from the master group. See example above.
LoadGroup	Loads the alarm object with an alarm from the specified group. 0 - 5000, 0 = master list, See example above.

Related:
Function list

AcknowledgeCommand

Perform an "Acknowledge" command.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

AcknowledgeCommand;

Related:
Function list
SilenceCommand
SilenceAcknowledgeCommand
AcknowledgePointAlarm

AcknowledgePointAlarm

Perform an "Acknowledge" command on a single point.

Inputs

Variable	Type	Description
Tagname	String	Point tagname to acknowledge
Alarm	Integer	Alarm type to acknowledge

Value	Type
-1	All types (All four types for analog, both types for digital)
0	Lo Lo or Falling
1	Lo or Rising
2	Hi
3	Hi Hi

Outputs

Variable	Type	Description
None		

Note: This will reset the 'Alarm pulse'.

Examples

```
AcknowledgePointAlarm('Pump1Pressure',-1);      // lo, lo lo, hi and hi hi
AcknowledgePointAlarm('Pump1Flow',-1);          // falling and rising
AcknowledgePointAlarm('Pump1Pressure',2);       // hi
```

Related:
Function list
SilenceCommand
SilenceAcknowledgeCommand

Add

This function is used to add all the input values.

Inputs

Variable	Type	Description
UseQuality	Boolean	Use or ignore the quality state of the tagname
Tagname	String	Name of the tag
Item Number	Integer	Data item ID

Outputs

Variable	Type	Description
Value	Float	The inputs are added.

If UseQuality is true and the quality of the tagname is good the value of the item id will be used in the calculation of the function.

If UseQuality is false the value of the item id will be used in the calculation of the function regardless of the quality of the tagname.

Examples
value:=Add(true,['tagname 1', 5000, 'tagname 2', 5000]);

Related:
Function list
AddDigital
AllTrue
AnyTrue
Average

AddDigital

This function returns the count of true or false inputs.

Inputs

Variable	Type	Description
CheckTrue	Boolean	True = Count the number of true inputs False = Count the number of false inputs
UseQuality	Boolean	Use or ignore the quality state of the tagname
Tagname	String	Name of the tag
Item Number	Integer	Data item ID

Outputs

Variable	Type	Description
Value	Integer	The count of true or false tagnames.

If UseQuality is true and the quality of the tagname is good the value of the item id will be used in the calculation of the function.

If UseQuality is false the value of the item id will be used in the calculation of the function regardless of the quality of the tagname.

Examples

```
//the number of false inputs
value:=AddDigital(false, true,['tagname 1', 5007, 'tagname 2', 5007]);

//the number of true inputs
value:=AddDigital(true, true,['tagname 1', 5007, 'tagname 2', 5007]);
```

Related:
Function list
Add
AllTrue
AnyTrue
Average

AllTrue

This function is used to evaluate if all the referenced digital items are true. (AND function)

Inputs

Variable	Type	Description
UseQuality	Boolean	Use or ignore the quality state of the tagname
Tagname	String	Name of the tag
Item Number	Integer	Data item ID

Outputs

Variable	Type	Description
Value	Boolean	True = All data values are true False = At least one data value was not true (false)

If UseQuality is true and the quality of the tagname is good the value of the item id will be used in the calculation of the function.

If UseQuality is false the value of the item id will be used in the calculation of the function regardless of the quality of the tagname.

Note: If UseQuality is true and the quality is bad for all the tagnames the result will be true.

Examples

```
value:=AllTrue(true,['tagname 1', 5007, 'tagname 2', 5007]);
if value then
;
```

Adding a "not" changes the result to - any false (NAND function)

```
value:=not AllTrue(true,['tagname 1', 5007, 'tagname 2', 5007]);
if value then
;
```

Related:
Function list
Add
AddDigital
AnyTrue
Average
DigitalCompare

AnyTrue

This function is used to evaluate if any of the referenced digital items are true. (OR function)

Inputs

Variable	Type	Description
UseQuality	Boolean	Use or ignore the quality state of the tagname
Tagname	String	Name of the tag
Item Number	Integer	Data item ID

Outputs

Variable	Type	Description
Value	Boolean	True = Any data value is true False = All data values are false

If UseQuality is true and the quality of the tagname is good the value of the item id will be used in the calculation of the function.

If UseQuality is false the value of the item id will be used in the calculation of the function regardless of the quality of the tagname.

Note: If UseQuality is true and the quality is bad for all the tagnames the result will be false.

Examples

```
value:=AnyTrue(true,['tagname 1', 5007, 'tagname 2', 5007]);
if value then
;
```

Adding a "not" changes the result to - all false (NOR function)

```
value:=not AnyTrue(true,['tagname 1', 5007, 'tagname 2', 5007]);
if value then
;
```

Related:
Function list
Add
AddDigital
AllTrue
Average
DigitalCompare

Average

This function is used to calculate an average.
There are two "Average" functions. "Average2" is below.

Inputs

Variable	Type	Description
UseQuality	Boolean	Use or ignore the quality state of the tagname
Tagname	String	Name of the tag
Item number	Integer	Data item ID

Outputs

Variable	Type	Description
Value	Float	The inputs are added and value is divided by the number of inputs

If UseQuality is true and the quality of the tagname is good the value of the item id will be used in the calculation of the function.

If UseQuality is false the value of the item id will be used in the calculation of the function regardless of the quality of the tagname.

Examples

value:=Average(true,['tagname 1', 5000, 'tagname 2', 5000]);

Related:
Function list
Add
AddDigital
AllTrue
AnyTrue
Average2

Average2

This function is used to calculate an average. This function provides the option to remove the highest and lowest value.

Inputs

Variable	Type	Description
RemoveOutliers	Boolean	If true, the largest and smallest values are removed.
Values	Array	Array of numeric values

Outputs

Variable	Type	Description
Value	Float	The "Average" (sum off all inputs / count of inputs).

If "RemoveOutliers" is true, at least 3 "Values" must be present. Otherwise at least 2 values must be present.

Examples

value:=Average2(true,[2,4]);
value:=Average2(false,[2,3,4]);
value:=Average2(true,[99,87,85,12,69]);

```
//average cell voltage
values:=RV(['RX_1_GADF_CellV_1','RX_1_GADF_CellV_2',
            'RX_1_GADF_CellV_3','RX_1_GADF_CellV_4']);
value:=Average2(true,values);
```

Note: If the input count is invalid, the result will be 0 (zero).

Related:
Function list
Add
AddDigital
AllTrue
AnyTrue
Average

BACnetNakedWrite

Normal data writes to a BACnet object is via a write command to the point. This function bypasses the normal writing logic. A helper dialog exist in the script editor. (Menu: Edit/Insert/BACnet write...)

Variable	Type	Description
Port name	String	BACNet port name
Object type	String	BACNet object type. See below for allowed types.
Instance	Integer	Object instance
Property name	String	BACNet property name. See below for allowed names.
Array index	Integer	Array index. -1 to ignore the array index.
Write priority	Integer	BACNet write priority
Data type	String	Supported data type. See below for allowed types.
Value	Array of string	The data value to write to the object.instance.property. (A single value or no value, for a null data type, is supported.)

Note: All object types, property names and data types are case sensitive.

Outputs

Variable	Type	Description
Result	Integer	The result of the command. See table below.

Object types

Analog Input	Analog Output	Analog Value
Averaging	Binary Input	Binary Output
Binary Value	Calendar	Life Safety Point
Life Safety Zone	Loop	Multi State Input
Multi State Output	Multi State Value	Schedule
Trend		

Property names

Note: Not all properties are writable.

Acked Transitions	Ack Required	Action
Action Text	Active Text	Active Vt Sessions
Alarm Value	Alarm Values	All
All Writes Successful	Apdu Segment Timeout	Apdu Timeout
Application Software Version	Archive	Bias
Change Of State Count	Change Of State Time	Notification Class
Blank 1	Controlled Variable Reference	Controlled Variable Units
Controlled Variable Value	Cov Increment	Date List
Daylight Savings Status	Deadband	Derivative Constant
Derivative Constant Units	Description	Description Of Halt

Device Address Binding	Device Type	Effective Period
Elapsed Active Time	Error Limit	Event Enable
Event State	Event Type	Exception Schedule
Fault Values	Feedback Value	File Access Method
File Size	File Type	Firmware Revision
High Limit	Inactive Text	In Process
Instance Of	Integral Constant	Integral Constant Units
Issue Confirmed Notifications	Limit Enable	List Of Group Members
List Of Object Property References	List Of Session Keys	Local Date
Local Time	Location	Low Limit
Manipulated Variable Reference	Maximum Output	Max Apdu Length Accepted
Max Info Frames	Max Master	Max Pres Value
Minimum Off Time	Minimum On Time	Minimum Output
Min Pres Value	Model Name	Modification Date
Notify Type	Number Of Apdu Retries	Number Of States
Object Identifier	Object List	Object Name
Object Property Reference	Object Type	Optional
Out Of Service	Output Units	Event Parameters
Polarity	Present Value	Priority
Priority Array	Priority For Writing	Process Identifier
Program Change	Program Location	Program State
Proportional Constant	Proportional Constant Units	Protocol Conformance Class
Protocol Object Types Supported	Protocol Services Supported	Protocol Version
Read Only	Reason For Halt	Recipient
Recipient List	Reliability	Relinquish Default
Required	Resolution	Segmentation Supported
Setpoint	Setpoint Reference	State Text
Status Flags	System Status	Time Delay
Time Of Active Time Reset	Time Of State Count Reset	Time Synchronization Recipients
Units	Update Interval	Utc Offset
Vendor Identifier	Vendor Name	Vt Classes Supported
Weekly Schedule	Attempted Samples	Average Value
Buffer Size	Client Cov Increment	Cov Resubscription Interval
Current Notify Time	Event Time Stamps	Log Buffer
Log Device Object	Log Enable	Log Interval
Maximum Value	Minimum Value	Notification Threshold
Previous Notify Time	Protocol Revision	Records Since Notification
Record Count	Start Time	Stop Time
Stop When Full	Total Record Count	Valid Samples
Window Interval	Window Samples	Maximum Value Timestamp
Minimum Value Timestamp	Variance Value	Active Cov Subscriptions
Backup Failure Timeout	Configuration Files	Database Revision
Direct Reading	Last Restore Time	Maintenance Required

Member Of	Mode	Operation Expected
Setting	Silenced	Tracking Value
Zone Members	Life Safety Alarm Values	Max Segments Accepted
Profile Name		

Data types

Null	Unsigned Integer	Real
Boolean	Character string (ANSI X3.4)	

Result

Value	Description
-1	Pre-processing error
0	No error
1	Port name invalid
2	No such object
3	Instance out of range
4	Property name invalid
5	Array index out of range
6	Write priority out of range
7	Data type not valid
8	Value array error

Example

result:=BACnetNakedWrite('Bac-1','Analog Output',1,'Present Value',-1,2, 'Null',[]);

Related:
Function list

Barcode

The function generates a barcode image from the code value and settings in the script globals.

Inputs

Variable	Type	Description
Barcode type	Integer	1 = Barcode symbol 2 = POSTNET 3 = PDF417 4 = MaxiCode
Script global	String	Script global section name for code value and settings
Print	Boolean	True = print, false = do not print
File	Boolean	True = save to file, false = do not save to file

Outputs

Variable	Type	Description
None		

Examples

```
Barcode(1,'Symbol1', True, False);  //output to printer
Barcode(2,'Symbol1', True, True);   //output to printer and save to file
```

Note:
Only specify the script global section name. Do not include an item name. The items are predefined. Use the new button in the "Barcode" animation to automatically create a section with items or manually enter the items and values. Delete the graphic element after the script global creation if the graphic element is not required.

Related:
Function list

BCDToInteger, IntegerToBCD

These two functions convert BCD to a integer and integer to BCD.

Inputs

Variable	Type	Description
ToConverValue	Integer	The value to convert
ByteCount	Integer	2 byte or 4 byte conversion

Outputs

Variable	Type	Description
Value	Integer	The value as an integer or BCD

Examples

```
value:=BCDToInteger(56, 2);
value:=BCDToInteger(19, 4);

value:=IntegerToBCD(88, 2);
value:=IntegerToBCD (123456, 4);
```

Note:
If the "ByteCount" is not 2 or 4 a zero will be returned.

Related:
Function list

Beep

The computer speaker will play the default "Windows beep" sound.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

```
Beep;
```

Related:
Function list

CameraSaveLoop

IP camera only

This function is used to save the current loop to disk.

Inputs

Variable	Type	Description
Camera Name	String	Name of camera

Outputs

Variable	Type	Description
None		

Example

CameraSaveLoop('IP_Camera_1');

Related:
Function list
StartCameraRecording
StopCameraRecording

CaptureScreen

This function is used to capture the screen to a file.

Inputs

Variable	Type	Description
Filename	String	The complete path and file name to save the screen capture.
DateTime	Boolean	Include the date and time at the bottom of the image.

Outputs

Variable	Type	Description
None		

File type	Extension
Bitmap	BMP
JPEG	JPG
PNG	PNG
Metafile	EMF
Portable Document Format	PDF

Example

```
CaptureScreen('C:\CapturedScreen.jpg', true);
CaptureScreen('C:\CapturedScreen.bmp', false);
```

Notes:
1) If a file with the same name exist it will be replaced with the new capture.
2) If the path does not exist the capture will fail.
3) If the computer has more than one monitor the file name for each monitor will be prefixed with a number and underscore.

Related:
Function list
CaptureWindow
PrintScreen
PrintScreenActiveWindow

CaptureWindow

This function is used to capture a user window or foreground window to a file.

Inputs

Variable	Type	Description
Filename	String	The complete path and file name to save the screen capture
WindowName	String	The "Title" of the user window. If the string is blank the foreground window will be captured
DateTime	Boolean	Include the date and time at the bottom of the image.

Outputs

Variable	Type	Description
None		

File type	Extension
Bitmap	BMP
JPEG	JPG
PNG	PNG
Metafile	EMF
Portable Document Format	PDF

Example

```
CaptureWindow('C:\CapturedForeground.jpg', '', false);      //captures the foreground
                                                            //window to the JPEG file
CaptureWindow('C:\CapturedWindow.bmp', 'Main', true);       //captures the user window
                                                            //titled 'Main' to a bitmap file
```

Notes:
1) If a file with the same name exist it will be replaced with the new capture.
2) If the path does not exist the capture will fail.
3) If a window title is supplied and the window is not open the capture will fail.

Related:
Function list
CaptureScreen
PrintScreen
PrintScreenActiveWindow

CenterInWindow

This command is used to center the graphic element in the window from the graphic script of the graphic element.

Inputs

Variable	Type	Description
OffsetX	Integer	Horizontal offset, negative numbers move to left
OffsetY	Integer	Vertical offset, negative numbers move up

Outputs

Variable	Type	Description
Result	Integer	Error code or 0 for no error.

Error codes

Code	Description
0	No error
-1	Window name missing of invalid (should never happen)
-2	Element width < 1
-3	Element height < 1

Related:
Function list

ClearPortCounters

Clear the counters/watchdog for a port.

Inputs

Variable	Type	Description
Port name	String	The name of the port.

Outputs

Variable	Type	Description
None		

Example

```
ClearPortCounters('Pump_Port_1');
```

Related:
Function list
GetPortCounters

CloseAlarmLogWindow

Close the alarm log window.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

CloseAlarmLogWindow;

Related:
Function list
OpenAlarmLogWindow

CloseAlarmWindow

Close the alarm window.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

CloseAlarmWindow;

Related:
Function list
OpenAlarmWindow

CloseAllUserWindows

Close all user created windows.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

CloseAllUserWindows;

Related:
Function list
OpenWindow
CloseWindow
CloseAllUserWindows2

CloseAllUserWindows2

Closes all user created windows except the window(s) specified.

Inputs

Variable	Type	Description
Exclusions	Array of string	A list of windows to not close when the call is made.

Outputs

Variable	Type	Description
None		

Example

CloseAllUserWindows2(['Pump_Port_1']);

Note:
An empty array is the same as calling "CloseAllUserWindows".

Related:
Function list
OpenWindow
CloseAllUserWindows
CloseWindow

CloseEventWindow

Close the event log window.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

CloseEventWindow;

Related:
Function list
LogEvent
OpenEventWindow

ClosePortDiagnosticWindow

This function is used to close a window monitoring a port's data.

Inputs

Variable	Type	Description
Port name	String	Name of the port to monitor

Outputs

Variable	Type	Description
Value	Boolean	True if the window closed False if the window did not close

Example

```
value:=ClosePortDiagnosticWindow('MODBUS PORT 1');
if value then
begin
//window closed
end;

if ClosePortDiagnosticWindow('MODBUS PORT 1') then
begin
//window closed
end;
```

Related:
Function list
OpenPortDiagnosticWindow

CloseTagMonitorWindow

This function is used to close a window used for monitoring a tag (point).

Inputs

Variable	Type	Description
Tagname	String	Name of the tag

Outputs

Variable	Type	Description
Value	Boolean	True if the window closed False if the window did not close

Example

```
value:=CloseTagMonitorWindow('GreenInkFlow2B');
if value then
begin
//window closed
end;

if CloseTagMonitorWindow ('GreenInkFlow2B') then
begin
//window closed
end;
```

Related:
Function list
OpenTagMonitorWindow

CloseWindow

This function is used to close a user created window by name. If the logged on user does not have access rights to the window it will not close and the function will return false.

Inputs

Variable	Type	Description
Window name	String	Name of window to close

Outputs

Variable	Type	Description
Value	Boolean	True if the window opened False if the window did not close

Example

```
value:=CloseWindow('Pump Station 9');
if value then
begin
//window closed
end;

if CloseWindow('Pump Station 9') then
begin
//window closed
end;
```

Note:
The result of the call is true if the window was sent a "Close" message. Use the IsUserWindowOpen function to verify the window is closed, if needed.

Related:
Function list
CloseAllUserWindows
IsUserWindowOpen
OpenWindow
OpenWindowUserSelect

ColorBlend3

This function will "blend" the "High" and "Middle" or the "Middle" and "Low" colors using the input value and return a color. The input value must be between 100-0. This is similar to the "Blend" animation but, can be used to set a color via scripting.

Inputs

Variable	Type	Description
Value	Float	A value (100-0)
High color	Color	The 100 color
Middle color	Color	The 50 color
Low color	Color	The 0 color

Outputs

Variable	Type	Description
Blended color	Color	The blended color

Example

newColor:=ColorBlend3(67,clGreen,clYellow,clRed);

This example was used to set a pen color on a trend.

```
values:=RV(['SOC_Average']);      //state of charge (average)
newColor:=ColorBlend3(values[0],clGreen,clYellow,clRed);
SetTrendPenColor('Trends','SOCAverage',[1,newColor]);
```

Related:
Function list

ConvertUnits

This function converts an input unit value to another unit value of the same class. For example, inches to feet.

Inputs

Variable	Type	Description
Value	Float	Value to convert
From	Integer	Input value unit type
To	Integer	Output value unit type

Outputs

Variable	Type	Description
Values[0]	Boolean	True if no conversion error False if conversion error
Values[1]	Float	If values[0] = true, the converted value If values[0] = false, the value -1

Note:
Attempting to convert from one class to another class will produce an error. For example, inches to date.

Classes

Area, Distance, Temperature, Volume, Time, Mass

Area			
Index	Units	Index	Units
101	Square Millimeters	109	Square Feet
102	Square Centimeters	110	Square Yards
103	Square Decimeters	111	Square Miles
104	Square Meters	112	Acres
105	Square Decameters	113	Centares
106	Square Hectometers	114	Ares
107	Square Kilometers	115	Hectares
108	Square Inches	116	Square Rods

Distance			
Index	Units	Index	Units
1	Micromicrons	17	Miles
2	Angstroms	18	Nautical Miles
3	Millimicrons	19	Astronomical Units
4	Microns	20	Light Years
5	Millimeters	21	Parsecs
6	Centimeters	22	Cubits
7	Decimeters	23	Fathoms
8	Meters	24	Furlongs
9	Decameters	25	Hands
10	Hectometers	26	Paces
11	Kilometers	27	Rods
12	Megameters	28	Chains
13	Gigameters	29	Links
14	Inches	30	Picas
15	Feet	31	Points
16	Yards		

Temperature	
Index	Units
701	Celsius
702	Kelvin
703	Fahrenheit
704	Rankine
705	Reaumur

Volume			
	Metric		**American Fluid Units**
Index	Units	Index	Units
201	Cubic Millimeters	230	Fluid Gallons
202	Cubic Centimeters	231	Fluid Quarts
203	Cubic Decimeters	232	Fluid Pints
204	Cubic Meters	233	Fluid Cups
205	Cubic Decameters	234	Fluid Gills
206	Cubic Hectometers	235	Fluid Ounces
207	Cubic Kilometers	236	Fluid Tablespoons
208	Cubic Inches	237	Fluid Teaspoons
209	Cubic Feet		**American Dry Units**
210	Cubic Yards	240	Dry Gallons
211	Cubic Miles	241	Dry Quarts
212	Milliliters	242	Dry Pints
213	Centiliters	243	Dry Pecks
214	Deciliters	244	Dry Buckets
215	Liters	245	Dry Bushels
216	Decaliters		**English Imperial Fluid Units**
217	Hectoliters	250	UK Gallons
218	Kiloliters	251	UK Pottle
219	Acre Feet	252	UK Quarts
220	Acre Inches	253	UK Pints
221	Cords	254	UK Gill
222	Cord Feet	255	UK Ounces
223	Decisteres	256	UK Pecks
224	Steres	257	UK Buckets
225	Decasteres	258	UK Bushels

Time	
Index	Units
801	Milliseconds
802	Seconds
803	Minutes
804	Hours
805	Days
806	Weeks
807	Fortnights
808	Months
809	Years
810	Decades
811	Centuries
812	Millennia
813	Date Time
814	Julian Date
815	Modified Julian Date

Mass	
Index	Units
601	Nanograms
602	Micrograms
603	Milligrams
604	Centigrams
605	Decigrams
606	Grams
607	Decagrams
608	Hectograms
609	Kilograms
610	Metric Tons
611	Drams
612	Grains
613	Tons
614	Long Tons
615	Ounces
616	Pounds
617	Stones

Example

```
values:=ConvertUnits(47,14, 6);        //Convert from inches to centimeters
if values[0] then
 good result in values[1]
```

Related:
Function list

CrossReference

This function is used to access the cross reference information of the project. An example is in UserButtonWindow.

Variable	Type	Description
Command	String	Various commands
Parameters	Array of string	Parameters are command specific

Outputs

Variable	Type	Description
Values[0]	Integer	Negative value = command failure 0 = no values to return > 0 = number of strings returned
Values[1]	String	String result 1
Values[n]	String	String result n

Command	Function
Tagnames	Returns all configured tagnames. The point/tagname could be invalid at runtime because of an error. View the "Event log" if a configured tag is not returned.

```
values:=CrossReference('Tagnames', []);                    //return tagnames
```

values[0] contains an error (negative) or the count of strings returned.

```
if (values[0] < 1) then
 ;              //no tagnames returned
```

Screens Returns all screens containing the tagname.

```
values:=CrossReference('Screens', ['<point tagname>']);   //return screens
```

values[0] contains an error (negative) or the count of strings returned.

```
if (values[0] < 1) then
 ;              //no screens returned
```

HasScreen Returns true if the tagname has at least one screen.

```
value:=CrossReference('HasScreen', ['<point tagname>']);
```

value contains true if at least one screen or false if no screen.

```
if value then
 beep;          //tagname has at least one screen
```

Related:
Function list
UserButtonWindow

CustomLogCol

This function appends the values to the log, separated by the "Separator" value.

Inputs

Variable	Type	Description
Log name	String	Custom log name
Values	Array	Values to log

Outputs

Variable	Type	Description
None		

Note:
If the custom log file does not exist an entry will be added to the event log.

Example

```
CustomLogCol('ActionLog',[Now, 33, 'User start pump']);    //"Now" is the current date and time
CustomLogCol('ActionLog',['Start', 19,]);
```

Related:
Function list
CustomLogCopy
CustomLogFlush
CustomLogLineCount
CustomLogLog
CustomLogPoint
CustomLogPoint2
CustomLogSave
CustomLogView

CustomLogCopy

This function saves the log to the file specified.

Inputs

Variable	Type	Description
Log name	String	Custom log name
Filename	String	The full path and file name

Outputs

Variable	Type	Description
Success	Boolean	True if the log was save False if the save failed

Note:
If the custom log file does not exist an entry will be added to the event log.

Example

```
CustomLogCopy('ActionLog', 'C:\ActionLogCopy.txt');
```

Related:
Function list
CustomLogCol
CustomLogFlush
CustomLogLineCount
CustomLogLog
CustomLogPoint
CustomLogPoint2
CustomLogSave
CustomLogView

CustomLogFlush

This function clears the log.

Inputs

Variable	Type	Description
Log name	String	Custom log name

Outputs

Variable	Type	Description
None		

Note:
If the custom log file does not exist an entry will be added to the event log.

Example

```
CustomLogFlush('ActionLog');
```

Related:
Function list
CustomLogCol
CustomLogCopy
CustomLogLineCount
CustomLogLog
CustomLogPoint
CustomLogPoint2
CustomLogSave
CustomLogView

CustomLogLineCount

This function returns the line count of the log.

Inputs

Variable	Type	Description
Log name	String	Custom log name

Outputs

Variable	Type	Description
Count	Integer	Log line count

Note:
If the custom log file does not exist an entry will be added to the event log.

Example

```
count:=CustomLogLineCount('ActionLog');
```

Related:
Function list
CustomLogCol
CustomLogCopy
CustomLogFlush
CustomLogLog
CustomLogPoint
CustomLogPoint2
CustomLogSave
CustomLogView

CustomLogLog

This function is used to append a string to a custom log.

Inputs

Variable	Type	Description
Log name	String	Custom log name
Log string	String	The string to log

Outputs

Variable	Type	Description
None		

Note:
If the custom log file does not exist an entry will be added to the event log.

Example

```
CustomLogLog('ActionLog', 'Door opened');
```

Related:
Function list
CustomLogCol
CustomLogCopy
CustomLogFlush
CustomLogLineCount
CustomLogPoint
CustomLogPoint2
CustomLogSave
CustomLogView

CustomLogPoint

This function appends the values of the specified points to the log, separated by the "Separator" value.

Inputs

Variable	Type	Description
Log name	String	Custom log name
Tagname	String	Name of the tag
Item number	Integer	Data item ID

Outputs

Variable	Type	Description
None		

Notes:
1) If the custom log file does not exist an entry will be added to the event log.
2) The tagname and item must be in pairs.

Example

```
CustomLogPoint('ActionLog',['PmpRunning',5007,'PumpPressure',5000]);
```

Related:
Function list
CustomLogCol
CustomLogCopy
CustomLogFlush
CustomLogLineCount
CustomLogLog
CustomLogPoint2
CustomLogSave
CustomLogView

CustomLogPoint2

This function appends the values of the specified points to the log, separated by the "Separator" value. This function is used (instead of CustomLogPoint) when the first column contents are to be defined in the script. The most common use, log the date and time or just the time into the first column.

Inputs

Variable	Type	Description
Log name	String	Custom log name
Column 1	String	Any text string

These next two variables must be in pairs.

Tagname	String	Name of the tag
Item Number	Integer	Data item ID

Outputs

Variable	Type	Description
None		

Notes:
1) If the custom log file does not exist an entry will be added to the event log.
2) The tagname and item must be in pairs.

Examples

```
CustomLogPoint2('ActionLog', 'some text',  ['PmpRunning',5007,'PumpPressure',5000]);

//this logs the time in the first column
CustomLogPoint2('ActionLog', TimeToStr(Now),  ['PmpRunning',5007,'PumpPressure',5000]);

//this logs the date and time in the first column
CustomLogPoint2('ActionLog', DateTimeToStr(Now),
['PmpRunning',5007,'PumpPressure',5000]);
```

Related:
Function list
CustomLogCol
CustomLogCopy
CustomLogFlush
CustomLogLineCount
CustomLogLog
CustomLogPoint
CustomLogSave
CustomLogView

CustomLogSave

This function saves the custom log to disk.
Custom logs are held in memory until runtime monitoring is stopped or this command is executed.
At runtime monitoring start the file is loaded into memory, if present

Inputs

Variable	Type	Description
Log name	String	Custom log name

Outputs

Variable	Type	Description
None		

Notes:
1) If the custom log file does not exist an entry will be added to the event log.
2) If the custom log is empty the file will not be created or saved.

Example

```
CustomLogSave('ActionLog');
```

Related:
Function list
CustomLogCol
CustomLogCopy
CustomLogFlush
CustomLogLineCount
CustomLogLog
CustomLogPoint
CustomLogView

CustomLogView

This function displays the log in a window. If the log is open in a window, it will be moved to the front.

Inputs

Variable	Type	Description
Log name	String	Custom log name

Outputs

Variable	Type	Description
None		

Note:
If the custom log file does not exist an entry will be added to the event log.

Example

```
CustomLogView('ActionLog');
```

Related:
Function list
CustomLogCol
CustomLogCopy
CustomLogFlush
CustomLogLineCount
CustomLogLog
CustomLogPoint
CustomLogPoint2
CustomLogSave

DBCommand

This function is used to command or integrate an "ODBC Connection" port type.

Inputs

Variable	Type	Description
Port name	String	Port name
Command	String	Command name
Values	Array of variant	Values defined by command

Outputs

Variable	Type	Description
Values	Array of variant	Values defined by command

Note: The following SQL examples might not function correctly with the selected database and might require modification to function correctly. Not all databases support all SQL commands and not all databases use the same syntax. For example: A database allows the "LIKE" operator when using "CREATE TABLE" function and another database indicates a syntax error.

Command	Function
Connection	Sets connection active/inactive

```
values:=DBCommand('ODBC_1', 'Connection', [true]);      //set connection active
values:=DBCommand('ODBC_1', 'Connection', [false]);     //set connection inactive
```

values[0] contains a boolean indicating if the command executed.

```
if values[0] then
 ;              //command success
```

GetStatus Returns status of the connection and query state

```
values:=DBCommand('ODBC_1', 'GetStatus', []);
```

values[0] True if connection is active
values[1] True if query is active

Command	Function
LogString	Logs a string or strings to the diagnostic log.

```
values:=DBCommand(' ODBC_1', 'LogString',['some string']);
values:=DBCommand(' ODBC_1', 'LogString',['some string', 'another string']);
```

values[0] contains a boolean indicating if the command executed.

```
if values[0] then
 ;              //command success
```

Query Sets query active/inactive

```
values:=DBCommand('ODBC_1', 'Query', [true]);    //set query active
values:=DBCommand('ODBC_1', 'Query', [false]);   //set query inactive
```

values[0] contains a boolean indicating if the command executed.

```
if values[0] then
 ;              //command success
```

SQL Sets the query SQL text.

```
values:=DBCommand('ODBC_1', 'SQL', ['Select from * tablename']);
```

values[0] contains a boolean indicating if the command executed.

```
if values[0] then
 ;              //command success
```

Note: Setting the SQL text sets the "Query" state to inactive. Set the "Query" state to active to execute the new SQL text.

SQLEX Performs an SQL command.

```
values:=DBCommand('ODBC_1', 'SQLEX', ['CREATE TABLE Power like Flow']);
```

values[0] contains a boolean indicating if the command executed.

```
if values[0] then
 ;              //command success
```

Related:
DQS
Function list

DBGraphicRead

This function is used to read the text of a database graphic element. The database graphic element connects to the database table and is set, automatically, via the current record index.

Inputs

Variable	Type	Description
Window name	String	Window name containing database graphic element
Name	String	Database graphic element name
Values	Array of variant	Values defined by database graphic element type

Outputs

Variable	Type	Description
Values	Array of variant	Values defined by database graphic type

Text, Edit, Listbox, Drop list, Checkbox and Radio group

```
values:=DBGraphicRead('Main_window', 'Pounds', []);
```

values[0] contains a boolean indicating if the command executed.
This value can be true and the result in values[1] can be empty.

values[1] contains the text.

```
if values[0] then
 textString:=values[1];
```

Memo

For a memo database graphic element the result is the complete line specified.

```
values:=DBGraphicRead('Main_window', 'Pounds', [2]);      //read the second line
```

values[0] contains a boolean indicating if the command executed.
This value can be true and the result in values[1] can be empty.

values[1] contains the complete memo line

```
if values[0] then
 textString:=values[1];
```

Grid

The database grid graphic element is not supported because access to the data is not available and a query would be required to fetch the value of a cell each time the cell content is required. If access to the data is needed, load the data into a recipe and use RecipeGetCell.

Related:
Function list

DeleteFilesAge

This function is used to delete the files in a path that are older than the specified number of days.

Inputs

Variable	Type	Description
PathOption	Integer	Defines the path variable
Options	Array of variant	See below

Outputs

Variable	Type	Description
Error	Integer	Error code

PathOption

1..8 are the paths defined in Log file settings

Value	Directory path
0	Fully qualified directory path
1	Alarm logs
2	Event logs
3	Logger logs
4	DNP logs
5	SNMP logs
6	FTP logs
7	Reports
8	Omni reports

Options

Field	Directory path
1	If PathOption = 0, fully qualified directory path
	If PathOption > 0, ignored, (use empty string ")
2	“Retain” If the file is older than the current date minus the value, the file is deleted. Example: Current date is January 10, this value is 5. All files created on or before January 5 are deleted.

Error codes

When the script function is executed an error code will be returned. The function call creates a thread and file processing is handled outside the main thread.

Code	Description
0	No error
-1	Directory path not found
-2	Path already in queue
-3	Unable to create master object
-4	Thread is terminated
-5	Options count error
-6	"PathOption" field out of range
-7	Unable to resolve "PathOption"
-8	Option field 2 (retain days) less than 1

Examples

```
value:=DeleteFilesAge(0,['C:\CustomLogs', 5]);    //fully qualified path, retain 5
value:=DeleteFilesAge(1,['', 30]);                //Alarm log path in settings, retain 30
value:=DeleteFilesAge(3,['', 10]);                //Logger path in settings, retain 10
```

Related:
Function list

DigitalCompare

This function is used to evaluate the first Boolean that is true from 1 or more tagnames.

Inputs

Variable	Type	Description
UseQuality	Boolean	Use or ignore the quality state of the tagname
Default value	Integer	The value returned if all inputs are false
Tagname	String	Name of the tag
Item number	Integer	Data item ID

Outputs

Variable	Type	Description
Value	Integer	The index of the first true, 0 if a tag has bad quality or the default value.

If UseQuality is true and the quality of any tagname is bad 0 will be returned.

If UseQuality is false the index of the first true Boolean or the default value will be returned.

Examples

```
value:=DigitalCompare(true,999,['tagname 1', 5007, 'tagname 2', 5007]);

case value of
0: bad quality
1: tagname 1 true
2: tagname 2 true
999: all Booleans false
end;
```

Related:
Function list
Add
AddDigital
AllTrue
AnyTrue
Average

DMCPOpenPlotWindow

This function is used to display a plot from a DMCP port.

Inputs

Variable	Type	Description
Port name	String	The name of the port.
Theme name	String	The name of the theme. If blank the default theme will be used.
Plot number (0-7)	Integer	The plot to display when the window opens.
Collect on open	Boolean	If true a command to collect the plot from the controller will be executed when the window is opened. If false the trend will be blank or populated with a previously collected plot.

Outputs

Variable	Type	Description
None		

Notes:
1) Only one plot window per port will be open. If this command is called with a window open for the port, the window will be set to the front window.
2) If the port is configured to use TCP this note in the RMC tools help file is relevant:
Avoid repeatedly closing and re-opening TCP connections. Devices are required to maintain state on TCP connections for two minutes after the connections are closed, and repeatedly opening and closing TCP connections can exhaust resources in the RMC or host device. Instead the connection should be left open by the host device while communicating with the RMC.
The port will use one TCP port for normal data communications and when this window is open another TCP port is opened and used for plot collections and when the window is closed the TCP port is released. Opening and closing this plot window could cause the problem stated in the note above.

Example

```
DMCPOpenPlotWindow('Controller 1', '', 0, true);
DMCPOpenPlotWindow('Controller 99', 'Packer theme', 1, false);
```

Related:
Function list

DMCPSavePlotPictureToFile

This function is used to capture the currently displayed plot to a file.

Inputs

Variable	Type	Description
Port name	String	The name of the port. If the port plot window is not open, the command is terminated.
Path	String	The complete path and file name. The file extension selects the file type.
Width	Integer	The width of the image.
Height	Integer	The height of the image.

Outputs

Variable	Type	Description
None		

File type	Extension
Bitmap	BMP
JPEG	JPG
PNG	PNG
Metafile	EMF
Portable Document Format	PDF

Note:
If a file with the same name/path exist, it will be overwritten.

Example

```
DMCPSavePlotPictureToFile('Controller 1', 'C:\test.bmp',1024,768);
DMCPSavePlotPictureToFile('Controller 1', 'C:\test.jpg',1024,768);
DMCPSavePlotPictureToFile('Controller 1', 'C:\test.png',1024,768);
DMCPSavePlotPictureToFile('Controller 1', 'C:\test.emf',1024,768);
```

Related:
Function list

DNPFreeze

This function is used to send a "freeze" command to a DNP3 outstation.

Inputs

Variable	Type	Description
Port name	String	The name of the port.
Function code	Integer	Function code 7 or 9.
Group	Integer	Counters (20) or Analog inputs (30).
Selection	Array	Empty or start and end index.

Outputs

Variable	Type	Description
Value	Boolean	True if the command was accepted. False if the command was not accepted.

Example

```
value:=DNPFreeze('Pump1', 7, 20, []);              //freeze all counters
value:=DNPFreeze('Pump1', 7, 30, []);              //freeze all analog inputs
value:=DNPFreeze('Pump1', 7, 20, [8 , 24]);        //freeze counters 8 - 24
value:=DNPFreeze('Pump1', 9, 20, []);              //freeze and clear all counters
value:=DNPFreeze('Pump1', 7, 20, [17 , 17]);       //freeze counter 17
```

Related:
Function list

DNPGroup12Write

This function is used to issue a custom group 12, variation 1, command to a DNP3 outstation. For a full description of DNP3 parameters, see the DNP3 specification.

Inputs

Variable	Type	Description
Tagname	String	Name of the point
Item number	Integer	Item number (should be 5007 - Process Variable Digital)
Control code	Byte	Control code, normally: 1: PULSE_ON 2: PULSE_OFF 3: LATCH_ON 4: LATCH_OFF
Count	Byte	Number of times outstation performs operation
On time	Integer	On time in milliseconds
Off time	Integer	Off time in milliseconds

Outputs

Variable	Type	Description
Value	Boolean	True if the command was accepted. False if the command was not accepted.

Example

```
value:=DNPGroup12Write('Pump1',5007, 3,1,0,0); //latch on

//pulse on 2 times, 500 milliseconds on, 12 milliseconds off
value:= DNPGroup12Write ('Pump1', 5007,1,2,500,12);
```

Note:
The write command result status is saved to the point item (Write status (DNP3) - 5127).

Related:
Function list

DNPOSWrite

This function is used to write a value(s) to an internal DNP outstation port group and index.

Inputs

Variable	Type	Description
Port name	String	Port name
Group	Integer	1, 3, 10, 20, 30, 40
Start index	Integer	0-65535
Stop index	Integer	0-65535
Values	Array	Array of values of the same type as the group.

Outputs

Variable	Type	Description
Value	Boolean	True if command was successful False if the command was not successful

Examples

```
value:=DNPOSWrite('Pump1', 1, 0, 0, ['False']);          //group 1, index 0, write false
value:=DNPOSWrite('Pump1', 1, 0, 1, ['False', 'True']); //group 1, index 0-1, write false-true
value:=DNPOSWrite('Pump1', 40, 55, 57, [0,1,2]);        //group 40, index 55-56-57, write 0-1-2
```

Notes:
1) If sequential indexes are to be written, it is much faster to use one command. See example 2 and 3
2) If the index has a point.item configured, the index will be overwritten on the next update.
3) If the group is an output group and the index has a point.item configured, an external write will not be generated.

Related:
Function list

DQS (Double quote string)

This function is used to add two quotes (') to the beginning and end of a string. SQL requires double quotes for some queries/commands. In scripting this can be tedious. This function can replace:

```
someString:= "" + <a string> + "";

s1:= 'SELECT * FROM table1  WHERE User  = ' +  "" + Username + "") ;
s1:= 'SELECT * FROM table1  WHERE User  = ' +  DQS('Username') ;

s1:= 'SELECT * FROM table1  WHERE User  = ' +  DQS(<a variable>) ;
```

Inputs

Variable	Type	Description
AString	String	Some string value

Outputs

Variable	Type	Description
Value	String	The string with double quotes on each end or four quotes if the string is empty.

Examples

```
value:=DQS('Field_1');          //result value is "Field_1"
```

Related:
DBCommand
Function list

EditFieldConfigureRecipe

This function is used to set the recipe configuration for an edit field at runtime. Changes made are only applicable while the window is open. If the window is closed and opened, the settings selected during configuration are applied. If the edit field connection is a point or script global this function has no effect.

Inputs

Variable	Type	Description
Window name	String	The name of the window containing the edit field.
Search	String array	Recipe name, column/field name, row number
Replace	String array	Recipe name, column/field name, row number

Outputs

Variable	Type	Description
Value	Boolean	True if at least one edit field was altered. False if zero edit fields were altered.

The search fields are optional. Search fields are case sensitive.
All three search fields and all three replace fields must be present even if a field is not used.
Two single quotes are used in any unused field location.

Example 1

Window name: Alpha
All edit fields use recipe: Complete
Change all edit fields to use recipe: Half

value:=EditFieldConfigureRecipe ('Alpha', ['', '', ''] , ['Half', '', '']);

No searching applied. Changes all edit fields.

Example 2

Window name: Alpha
All edit fields use column/field name: Large
Change all edit fields to use column/field 'Small'

value:=EditFieldConfigureRecipe('Alpha', ['', '', ''] , ['', 'Small', '']);

No searching applied. Changes all edit fields.

Example 3

Window name: Alpha
Some edit fields use recipe: Complete
Change all edit fields using recipe 'Complete' to use column/field 'Small'.

value:=EditFieldConfigureRecipe('Alpha', ['Complete', '', ''] , ['', 'Small', '']);

Searches for edit fields using recipe 'Complete' and sets the column/field to 'Small'.

Example 4

Window name: Alpha
Some edit fields use column/field: Yellow
Change all edit fields using column/field 'Yellow' to use column/field 'Blue'

value:=EditFieldConfigureRecipe('Alpha', ['', 'Yellow', ''] , ['', 'Blue', '']);

Searches for edit fields using recipe column/field 'Yellow' and sets the column/field to 'Blue'.

Example 5

Window name: Alpha
All edit fields use row 4.
Change all edit fields to use row 5

value:=EditFieldConfigureRecipe('Alpha', ['', '', '',] , ['', '', '5']);

No searching applied. Changes all edit fields.

Example 6

Window name: Alpha
Some edit fields use row 19.
Change all edit fields using row 19 to use row 20

value:=EditFieldConfigureRecipe('Alpha', ['', '', '19'] , ['', '', '20']);

Searches for edit fields using row '19' and sets the row to '20'.

Example 7

Window name: Alpha
Some edit fields use recipe: Complete, column/field: Yellow
Change all edit fields using recipe: Complete, column/field: Yellow to column/field: Red

value:=EditFieldConfigureRecipe('Alpha', ['Complete', 'Yellow', ''] , ['', 'Red', '']);

Searches for edit fields using recipe: Complete, column/field: Yellow and sets the column/field to Red.

Related:
Function list

EmailSendMessage

This function is used to send an Email message without user input.

Note: The user names are configured in the Notification - Email settings.

Inputs

Variable	Type	Description
User name	String	Email user name(s), At least one user name must be present.
Attachments	String	Files to attach to the message.
In message	String	The message to send. It can be blank if the subject is not blank.
In subject	String	The message subject to send. It can be blank if the message is not blank.
Zip filename	String	If the 'Attachments' above is not blank and a file name is supplied the attachments will be zipped and the compressed file will be sent in place of the attachments. Do not include a path, just a file name.

Outputs

Variable	Type	Description
Value	Boolean	True if the message was queued False if the message was not queued

Example

```
value:=EmailUserSendMessage( ['Joe Smith', 'Bob Jones'],          //users to receive email
 ['C:\ScreenPrint.bmp', 'C:\AlarmLog.csv'],                       //files to attach
 'Alarm capture',                                                 //message
 'Fan failure',                                                   //subject field
 ['AlarmCapture']);    //zip the attachments and the zip file name is 'AlarmCapture'

value:=EmailUserSendMessage( ['Bob Jones'],                       //user to receive email
 [],                                                              //no files to attach
 '',                                                              //message is blank
 'Fan failure',                                                   //subject field
 ['']);                                                            //no zip
```

Related:
Function list
EmailUserSendMessage

EmailUserSendMessage

This function is used to send an Email message with user input.

Note: The user names are configured in the Notification - Emaill settings.

Inputs

Variable	Type	Description
User name	String	Email user name(s), if blank all configured Email user names are used.
Attachments	String	Files to attach to the message.
Can change	Boolean	The user can select from the list of user names.
Can edit	Boolean	The user can edit the message/subject before sending.
Show keyboard	Boolean	If enabled, an on screen keyboard is displayed with the window.
In message	String	The message to send. It can be blank and the user will enter the text.
In subject	String	The message subject to send. It can be blank and the user will enter the text.
Left position	Integer	Horizontal position of the window.
Top position	Integer	Vertical position of the window. "Left position" and "Top position" refer to the left and top corner of the window. If both values are -1 the window is centered on the main monitor.
Zip filename	String	If the 'Attachments' above is not blank and a file name is supplied the attachments will be zipped and the compressed file will be sent in place of the attachments. Do not include a path, just a file name.

Outputs

Variable	Type	Description
Value	Boolean	True if the message was queued False if the message was not queued

Example

```
value:=EmailUserSendMessage( ['Joe Smith', 'Bob Jones'],   //users to receive email
 ['C:\ScreenPrint.bmp',
 'C:\AlarmLog.csv'],                //files to attach
True,                               //user can change users
True,                               //user can change message/subject
True,                               //show the on screen keyboard
'',                                 //message is blank, user must enter
'Fan failure',                      //subject field
-1,                                 //left edge of window
-1,                                 //top edge of window
['AlarmCapture']);                  //zip the attachments
value:=EmailUserSendMessage( [],                            //display all users
[],                                                         //no files to attach
True,                               //user can change users
True,                               //user can change message/subject
True,                               //show the on screen keyboard
'',                                 //message is blank, user must enter
'Fan failure',                      //subject field
-1,                                 //left edge of window
-1,                                 //top edge of window
['']);                              //no zip
```

Related:
Function list
EmailSendMessage

ExecuteReport

This function is used to execute the named report.

Inputs

Variable	Type	Description
Report name	String	Name of report to execute

Outputs

Variable	Type	Description
Value	Boolean	True if the report executed False if the report did not execute

Note:
The report engine can return true and the report contain errors. For example: A point.item does not exist. The report will execute, place 'Error' in any cell that does not resolve and return true.

Example

```
value:=ExecuteReport('Pump A');
if value then
begin
//the report executed
end;
```

Related:
Function list

ExecuteScript

This function is used to execute a script. The script is placed in the queue of the engine specified.

Inputs

Variable	Type	Description
Script name	String	Name of the script to execute and the file extension (.psc)
Engine number	Integer	1 = Commands Engine 2 = During Engine

Outputs

Variable	Type	Description
Value	Boolean	The script was queued.

If the engine number is not 1 or 2 the script is not queued and result is set to true.
If the script is already in the queue it will not be added again and the result will be true.

Example

```
value:=ExecuteScript('Start Three Pumps (ABC).psc', 1);
if value then
begin
// the script was queued.
end;

if ExecuteScript('Start Three Pumps (ABC).psc' ,2) then
begin
// the script was queued.
end;
```

Related:
Function list
Script Structure
ExecuteScriptRoutine

ExecuteScriptRoutine

This function is used to queue and execute a procedure in a script. The call is placed in the queue.
Note: This is not a call that returns to the calling script. When used in a script, it places the "call" in the script queue and the calling script continues. The purpose is to queue a procedure in a script file and optional passing parameters.

Inputs

Variable	Type	Description
Script name	String	Name of the script to execute and the file extension (.psc)
Routine name	String	Name of the procedure in script file
Parameters	Array of variant	Value to pass to the routine

Outputs

Variable	Type	Description
Value	Boolean	The script was queued.

If the script is already in the queue it will not be added again and the result will be true.

Example

```
value:=ExecuteScriptRoutine('StartPumps.psc', 'StartCommand',  ['ABC', 12 , 1.0]);

//The called script routine.
procedure StartCommand(tagname, delay, pressure);

//no parameters
value:=ExecuteScriptRoutine('StartPumps.psc', 'StartCommand',  []);

//The called script routine.
procedure StartCommand;
```

Related:
Function list
Script Structure
ExecuteScript

ExportAlarmLog

This function is used export an alarm log from the internal HMI format to a comma separated file (CSV).

Inputs

Variable	Type	Description
Command	Integer	0 = Yesterday's log 1 = Today's log 2 = Use month/day/year field
Month	Integer	Month
Day	Integer	Day
Year	Integer	Year
Destination	String	Destination path. If the path does not exist the export will fail.
Filename	String	If blank, the file name is the same as the log file name except the extension is 'csv'. If not blank, the filename will be used. Any invalid file name characters will cause the export to fail.

Outputs

Variable	Type	Description
Result	Boolean	True if the export completed False if the export failed

Examples

result:=ExportAlarmLog(0,0,0,0,'C:\ExportedLogs\Alarms',''); //exports yesterday's alarm log, the '' are two single quotes
result:=ExportAlarmLog(1,0,0,0,'C:\ExportedLogs\Alarms',''); //exports today's alarm log
result:=ExportAlarmLog(2,3,4,2013,'C:\ExportedLogs\Alarms','SomeName.csv'); //exports the alarm log for March 4, 2013, SomeName.csv is the file name

Notes:

1) If a file exist in the destination path with the same file name, the file will be replaced.
2) If the alarm log does not exist the result will be false.
3) The month, day and year field are ignored if the command is 0 or 1.
4) **WARNING** Depending on the number of entries in the log this command can use a large amount of CPU resources. Contact support if help is needed.

Related:
Function list
ExportEventLog

ExportEventLog

This function is used export an event log from the internal HMI format to a comma separated file (CSV).

Inputs

Variable	Type	Description
Command	Integer	0 = Yesterday's log 1 = Today's log 2 = Use month/day/year field
Month	Integer	Month
Day	Integer	Day
Year	Integer	Year
Destination	String	Destination path. If the path does not exist the export will fail.
Filename	String	If blank, the file name is the same as the log file name except the extension is 'csv'. If not blank, the filename will be used. Any invalid file name characters will cause the export to fail.

Outputs

Variable	Type	Description
Result	Boolean	True if the export completed False if the export failed

Examples

result:=ExportEventLog(0,0,0,0,'C:\ExportedLogs\Events',''); //exports yesterday's event log, the '' are two single quotes

result:=ExportEventLog(1,0,0,0,'C:\ExportedLogs\Events',''); //exports today's event log

result:=ExportEventLog(2,3,4,2013,'C:\ExportedLogs\Events','SomeName.csv'); //exports the event log for March 4, 2013, SomeName.csv is the filename

Notes:

1) If a file exist in the destination path with the same file name, the file will be replaced.
2) If the event log does not exist the result will be false.
3) The month, day and year field are ignored if the command is 0 or 1.
4) **WARNING** Depending on the number of entries in the log this command can use a large amount of CPU resources. Contact support if help is needed.

Related:

Function list
ExportAlarmLog

FetchAlarmReset

The project setting "Enable alarm reset capture" must be enabled for this function to execute properly.

A configured point alarm has three states. Inactive, active or active and acknowledged.
Inactive, the alarm condition is false.
Active, the alarm condition is true.
Active and acknowledged, the alarm condition is true and the alarm state has been acknowledged.
If an alarm transitions to active and auto clear is not true, the active alarm state must be acknowledged before the alarm will transition to inactive, regardless of the monitored value.
If the active alarm condition is acknowledged and the monitored value returns/has returned to a non-alarm state, the alarm condition will transition to inactive (reset).

If the "Enable alarm reset capture" is true the tagname (of the point), alarm condition status, alarm active time, alarm active value, alarm acknowledged time, alarm acknowledged value, alarm reset (transition to inactive) time and alarm reset value are captured. (The last two pairs of values can be the same.)

This data is captured to internal storage.
Calling the "FetchAlarmReset" function process the capture data for access via scripting.
Note:
Calling this function resets the internal storage. New alarm cycles will be captured. It is advised to process all the data, as needed when calling the "FetchAlarmReset".

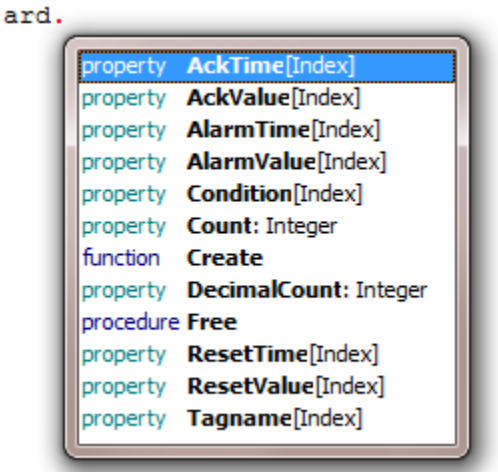
Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
Result	Integer	Count of alarms processed If = -1 an error occurred

Access to the data is via the "ard" (Alarm reset data) helper class. Type ard <period> ard. and the properties list will appear.



Changing the "DecimalCount" property will alter the decimal digits for floating point conversions.

Examples

```
value:=FetchAlarmReset;
if (value < 1) then
 Exit;
```

This example logs all the data to a custom log.

```
value:=FetchAlarmReset;
if (value < 1) then
 Exit;

for i:=0 to value -1 do
 CustomLogCol('AlarmHelperLog',[ard.Tagname[i],
                                ard.Condition[i],
                                ard.AlarmTime[i],
                                ard.AlarmValue[i],
                                ard.AckTime[i],
                                ard.AckValue[i],
                                ard.ResetTime[i],
                                ard.ResetValue[i]]);
```

Related:
Function list
ExportAlarmLog

FileParse

This command provides access to the file parse routine. The routine can parse a comma separated file (CSV), tab delimited file or Excel (XLS). After a read command, the data can be accessed with the "ObjectName" as rows and columns.

Inputs

Variable	Type	Description
Object name	String	Name of the parsed object. The name must be unique for each object.
Command	String	Command to execute
Values	Array of variant	Values defined by command

Outputs

Variable	Type	Description
Values	Array of variant	Values defined by command

Command	Function
ColumnCount	Returns the number of columns.

```
values:=FileParse('AnObjectName', 'ColumnCount', []);

values[0] = error code
values[1] = number of columns

if (values[0] = 0) then
 ;                    //command success
```

ColumnWrite Writes data to one or more columns.

```
values:=FileParse('AnObjectName', 'ColumnWrite', [1,1, 'Blue', 'Green', 'Red']);
                                                  //column, starting row, values

values[0] = error code
if (values[0] = 0) then
 ;                    //command success
```

This command writes all the data to the specified column, starting at the specified row. The number of rows could be extended if, starting row number + count of data values, is greater than the existing row count.

Command	Function
Free	Deletes the parse object.

values:=FileParse('AnObjectName', Free, []);

values[0] = error code
if (values[0] = 0) then
 ; //command success

LastError	Returns the last error code and clears the error code.

values:=FileParse('AnObjectName', 'LastError', []);

values[0] = error code

LastErrorString	Returns the last error code as a string and clears the error code.

values:=FileParse('AnObjectName', 'LastErrorString', []);

values[0] = error code as string

Read	Creates the parse object, if the object exist it is cleared of data and used, reads the file and parses the data. The file extension determines the delimiter used, ".csv" = comma character, ".tab" = tab character, or ".xls" = Excel file. Passing an empty string creates a parse object but no data is loaded.

values:=FileParse('AnObjectName', 'Read', ['C:\SomeFile.csv']); //read the file

values[0] = error code
values[1] = number of rows parsed
values[2] = number of columns parsed

if (values[0] = 0) then
 ; //command success

Command	Function
ReadColumn	Returns a column with each row in an array index.

```
values:=FileParse('AnObjectName', 'ReadColumn', [1,1,3]);
                                        //column, starting row, ending row

values[0] = error code
values[1] = number of rows
values[2] = first row value
values[3] = second row value
...
values[n] = last row value

if (values[0] = 0) then
 ;                      //command success
```

Note: If the starting row or ending row value is zero (0) all rows will be returned.

ReadRow Returns a row with each column in an array index.

```
values:=FileParse('AnObjectName', 'ReadRow', [3,1,4]);
                                        //row, starting column, ending column
values[0] = error code
values[1] = number of columns
values[2] = first column value
values[3] = second column value
...
values[n] = last column value

if (values[0] = 0) then
 ;                      //command success
```

Note: If the starting column or ending column value is zero (0) all columns will be returned.

RowCount Returns the number of rows.

```
values:=FileParse('AnObjectName', 'RowCount', []);

values[0] = error code
values[1] = number of rows

if (values[0] = 0) then
 ;                      //command success
```

Command	Function
RowWrite	Writes data to a row.

```
values:=FileParse('AnObjectName', 'RowWrite', [3,4, 'Blue', 'Green', 'Red']);
                                                    //row, starting column, values

values[0] = error code
if (values[0] = 0) then
 ;                          //command success
```

This command writes all the data to the specified row, starting at the specified column. The number of columns could be extended if, starting column number + count of data values, is greater than the existing column count.

Write Writes the parse object data to disk.

```
values:=FileParse('AnObjectName', 'Write', ['C:\SomeFileName.csv',true]);
                                                    //full path and file name, save as Unicode text
values:=FileParse('AnObjectName', 'Write', ['',false]);
                                                    //save to read file path/name, save as ASCII text

values[0] = error code
if (values[0] = 0) then
 ;                                          //command success
```

Notes:

1) If the file path/name is empty, the file path/name from the read command will be used to save the file.
2) The Unicode flag is applied when saving to CSV or TAB. It is required but ignored when saving to XLS.

Error codes, next page.

Error codes

The first value of the output array (where applicable) is an error code. If the value is not equal to zero, an error occurred with the command and the array will be one element long, containing the error code.

Code	Description
0	No error
1	No object with specified name found
2	Read, directory or file not found
3	Write, directory not found or file create/write failed
4	Row value out of bounds
5	Column value out of bounds
6	Parser object not created
7	Object name blank
8	Command blank
9	Unknown file extension (must be XLS, TAB or CSV)
10	Loading file failed
11	ReadColumn parameter error
12	ReadRow parameter error
13	ColumnWrite parameter error
14	RowWrite parameter error
15	Write, no file name
16	Write, parameter error

Related:
Function list

FofX

This instruction is an F of X function. It is also called a characterizer or FGEN or look-up table.

The number of rows must be 2 - 65535.
Each step of the input column must be greater than the previous step.

All values are floating point.

When the input value is less than or equal to In[0], the output is set to A[0].
When the input value is greater than the last row, the output is set to value of the last row.

Otherwise, when (In[n] < input value) and (input value <= In[n + 1]) we have a match and the output can be calculated.

n = the row

Output = ((A[n+1] - A[n]) / (In[n+1] - In[n])) * (input value - In[n]) + A[n]

Inputs

Variable	**Type**	**Description**
Command	String	F of X operation

Outputs

Variable	**Type**	**Description**
Value	Array	Various

Command	Function
Create	Creates the structure and loads the data table.

```
values:=FofX('Create', [<name>,<file path>]);      //successful create result = 0
if (values[0] < 0) then
 ;                                                 //error
```

The "name" must be unique. If an FofX data table exist with the same name, it will be deleted and the "new" table will be created.

The file path must be a complete file path to a "Comma separated" file, with a ".csv" file extension. The file can be a link (.lnk) file to the actual CSV file. If a link file, the file name must be *<file name>.lnk*.

The file shall contain two columns. Input column and output column.
Each step of the input column must be greater than the previous step.

Command	Function
Execute	Applies the "in value" to the FofX "named table" and returns a result

```
values:=FofX('Execute',[<name>,<in value>]);

if (values[0] < 0) then
  some error                          //if error, values[1] will be zero (0)
else
  values[1] = result of function
```

Error codes

The first value of the output array is an error code. If the value is not equal to zero (0), an error occurred and index 0 (e.g. values[0]) will be the error code. If the result value in index zero (0) is zero (0), the command response will be in index 1 (e.g. values[1]), if the command returns a value.

Code	**Description**
0	No error
-1	Unknown command
-2	Parameter count error
-3	Array name blank
-4	File path blank
-5	File not found
-6	The row count is < 2 or > 65535
-7	The "in" col failed to increase for each row.
-8	Memory create failure
-9	Named FofX does not exist

Related:
Function list

ForceLogOn

This command will force a user to be "logged on". The 'Allow forced logon' for the user must be enabled for this command to succeed. The 'Director' cannot be forced.

Inputs

Variable	Type	Description
User name	String	Exact user name, case sensitive

Outputs

Variable	Type	Description
Value	Boolean	True if successful False if failed

Example

```
value:=ForceLogOn('Operator');
if value then
begin
//a user is logged on
end;

if ForceLogOn('Operator') then
begin
//a user is logged on
end;
```

Related:
Function list

FTPAux

This function is used to perform an FTP command on a file located on the server.

Inputs

Variable	Type	Description
Port name	String	Port name
Command	String	See commands below
Values	Array of variant	Values defined by command

Outputs

Variable	Type	Description
Value	Boolean	Result of command added to queue

Note:
The result can be "no error" and the action fails. Check the event log.

Examples

Command	Function
Delete	Delete a file

```
value:=FTPAux(<FTP port name>,<command>,[<file name>]);

value:=FTPAux('Main_FTP','Delete',['C:\logfile.txt']);
if value then
 ;                    //command placed in queue
```

GetSize Return the size (in bytes) of a file

```
value:=FTPAux(<FTP port name>,<command>,[<file name>,<script global>]);

value:=FTPAux('Main_FTP','GetSize',['12-12-12.txt', 'GetSize.TurbineLog']);
if value then
 ;                    //command placed in queue
```

The file size is placed in the script global when the result is returned from the FTP server. Setting the script global to zero (0) before calling the FTPAux command, might be helpful.

Rename Rename a file

```
value:=FTPAux(<FTP port name>,<command>,[<from file name>,<to file name>]);

value:=FTPAux('Main_FTP','Rename',['12-12-12.txt', '12-12-12-old.txt ']);
if value then
 ;                    //command placed in queue
```

Related:
Function list
FTPGetSetting
FTPSetSetting

FTPGetSetting

This function is used to read an FTP setting from a port. It would mainly be used in scripting for control of the setting but, could be used to display a setting in a window at runtime.

Inputs

Variable	Type	Description
Port name	String	Port name
Setting	String	Setting name

Outputs

Variable	Type	Description
Value	String	String value

The "Setting" is the same string as is shown in the port configuration dialog. Refer to the FTP settings for a description of each attribute. The settings strings are:

Append	Binary mode	Compress
Compress password	Days	Destination file name
Destination path	Include subdirectories	Local port
Logging	Passive mode	Server path
Server port	Source path	Timeout
User name	User password	Script

State - used to start or cancel the transfer

The result is a string.

If the result needs to be displayed in a window, save the string in a global, GlobalSet, and use "script global" animation or "script" animation.

Examples

value:=FTPGetSetting('MainOffice','Days');

Related:
Function list
FTPAux
FTPSetSetting

FTPSetSetting

This function is used to write a setting in an FTP port.

Inputs

Variable	Type	Description
Port name	String	Port name
Setting	String	Setting name
Value	String	The new value for the setting

Outputs

Variable	Type	Description
Value	Boolean	True if setting/value was accepted False if setting/value was not accepted

See FTPGetSetting (page above) for a list of "setting" names.

To initiate a transfer use the word "State" in the setting field and set the value to "True".
value:=FTPSetSetting('MainOffice',**'State'**,True);

To cancel a transfer use the word "State" in the setting field and set the value to "False".
value:=FTPSetSetting('MainOffice',**'State'**,False);

Notes:
1) The result returns true if the setting name exists and the value is not blank. The check for a valid setting value is not performed until the object is commanded to initiate the transfer.
2) When using the "State" setting and the value is "True" the result is "True" if the FTP object is created and the transfer started. If the transfer is in progress the result is "False".
3) When using the "State" setting and the value is "False" the result is "True" if the FTP object is created and the object is commanded to cancel. If the transfer is not in progress the result is "False". If the object is commanded to cancel it might take several moments for the object to stop, close the connection and clean up the transfer.
4) The case must match in all fields. The port and setting. For the value, true and false it must be "True" or "False".

Examples
```
value:=FTPSetSetting('MainOffice','Days',-3);
value:=FTPSetSetting('MainOffice','Logging,False);
if FTPSetSetting('MainOffice','Days',-3) then
;
```

Related:
Function list
FTPAux
FTPGetSetting

GetAlarmGroupCount

This function is used to get the number of active alarms in an alarm group.

Inputs

Variable	Type	Description
Group number	Integer	Alarm group number

Outputs

Variable	Type	Description
Count	Integer	The number of active alarms in the group. -1 if the group does not exist

Example

```
value:=GetAlarmGroupCount(2);
if (value > 0) then //if greater than 0, at least one alarm in the group is active
beep;
```

Related:
Function list

GetPortCounters

This function returns port diagnostic information.

Inputs

Variable	Type	Description
Port name	String	The name of the port

Outputs

Variable	Type	Description
Values	Array of integer	See below

Primary port
Values[0] Integer 0 = watchdog not timed out, 1 = watchdog timed out
Values[1] Integer Watchdog timed out counter.
Values[2] Integer Reads requested
Values[3] Integer Reads completed
Values[4] Integer Writes requested
Values[5] Integer Writes completed
Values[6] Integer N/A
Values[7] Integer N/A
Values[8] Integer N/A
Values[9] Integer N/A

Secondary port
Values[10] Integer 0 = watchdog not timed out, 1 = watchdog timed out
Values[11] Integer Watchdog timed out counter.
Values[12] Integer Reads requested
Values[13] Integer Reads completed
Values[14] Integer Writes requested
Values[15] Integer Writes completed
Values[16] Integer N/A
Values[17] Integer N/A
Values[18] Integer N/A
Values[19] Integer N/A

Example

```
values:=GetPortCounters('Pump_Port_1');
if (values[0] <> 0) then
begin

//the primary port watchdog timed out flag is true
```

Notes:
1) Not all port types/port configurations have all fields.
2) The watchdog timed out flag does not automatically clear. If must be cleared via script command, mouse command or the 'Port Diagnostics' window.

Related:
Function list
ClearPortCounters

GetSystemVariable

This function is used to read a system value.

Inputs

Variable	Type	Description
Item Number	Integer	Data item ID

Outputs

Variable	Type	Description
Values	Array of variant	Values of variables

Item number	Description
1	Current time
2	Current date
3	Active alarm count
4	Day of week
5	User level (logged on)
6	Program path
7	Project name
8	Common application data directory
9	Sounds playing
10	Alarm pulse
11	Logged on user name - HMI
12	Notification email status
13	Notification SMS status
14	Computer name
15	Computer IP address
16	Project path
17	Sound muted
18	Logged on user name – Windows
19	Windows domain name
20	HMI version
21	Monitors
22	Sorted list of user names, level
23	Memory used by the HMI
24	Returns the last user version number value
25	Returns the last user version description
26	Returns the last user version name
27	Simulation state
28	Notification SMS active message count

Examples begin next page.

Current time

The time is returned in 4 values.

values:=GetSystemVariable(1);

values[0] = hour 0..23
values[1] = minutes 0..59
values[2] = seconds 0..59
values[3] = milliseconds 0..999

Current date

The date is returned in 3 values.

values:=GetSystemVariable(2);

values[0] = day 1..31
values[1] = month 1..12
values[2] = year 1..9999

Active alarm count

The count is returned in 1 value.

values:=GetSystemVariable(3);

values[0] = number of active alarms 0...n

Day of week

The value is returned in 1 value.

values:=GetSystemVariable(4);

values[0] = The day of the week. (1..7)

1=Sunday 2=Monday...

User level

The logged on user level. If a user is not logged on the return value is -1.

values:=GetSystemVariable(5);

values[0] = 0 - 65535 logged on user level
values[0] = -1 a user is not logged on

Program path

This returns the path of the runtime program.

values:=GetSystemVariable(6);

values[0] = 'C:\Program Files\Everest' //example

Project name

This returns the name of the current project.

values:=GetSystemVariable(7);

values[0] = 'Station 3' //example

Common application data directory

This returns the 'Common Application Data Directory' path for the runtime program. The OS determines the directory path.

values:=GetSystemVariable(8);

values[0] = 'C:\ProgramData\<Application name>\' //example Windows 7

Sounds playing

This returns the number of sounds queued to play.

values:=GetSystemVariable(9);

values[0] = 1 //one sound is in the queue

Alarm pulse

When a new alarm condition becomes true an internal flag is set and remains set until an 'Acknowledge' command is issued. The flag will be set true on each new alarm condition.

values:=GetSystemVariable(10);

values[0] = 1 //an alarm condition became true
values[0] = 0 //an alarm condition became true and was acknowledged or an alarm condition has not become true.

Logged on user name

The HMI logged on user name. If a user is not logged on the return value is '' (empty).

```
values:=GetSystemVariable(11);
if (values[0] = 'Ray') then
beep;
```

or to ignore case:

```
if (UpperCase(values[0]) = UpperCase('Ray')) then
beep;
```

Notification email status

This holds the last action/status for the email logic of the notification logic. The value can change very quickly and does not change until another step is performed.

Value Description
0 No Action (no action has been performed)
1 Email sent
2 Send complete
5 SMPT open command

These only apply to messages that do not require an acknowledgment.
3 Send success
4 Send failure

Errors
-1 SMTP abort from not ready after 10 seconds

-2 POP3Checking abort from not ready after 10 seconds

Notification SMS status

This holds the last action/status for the SMS logic of the notification logic. The value can change very quickly and does not change until another step is performed.

Value Description
0 No Action (no action has been performed)
1 SMS sent
2 Initialization complete
5 Message read complete (Only applies when acknowledgement is required)

These only apply to messages that do not require an acknowledgment.
3 Send success
4 Send failure

Errors
-1 Watchdog timeout

Computer name

This returns the name of the computer.

```
values:=GetSystemVariable(14);

if (values[0] = 'Operations 101') then
 beep;
```

or to ignore case:

```
if (UpperCase(values[0]) = UpperCase('Operations 101')) then
 beep;
```

Note: If the intent is to display the computer name or computer IP address on a graphic screen, the most efficient method is to populate a script global when monitoring starts or when the graphic screen is opened. Reading these two values in a script that executes over and over (graphic script for example) is not efficient.

Computer IP address

This returns all the IP addresses of the computer

```
values:=GetSystemVariable(15);
```

values[0] contains the count of IP addresses.
values[1] through values[n] contain the IP address, as a string.

```
//example
values[0] = 2                          //number of IP addresses returned
values[1] = '192.168.1.1'              //address 1
values[2] = '10.0.0.5' //address 2
```

In the scripting examples is an example of how to collect and store all the IP addresses for the computer.

Project path

This returns the path of the project.

```
values:=GetSystemVariable(16);

values[0] = 'C:\Project_1'             //example
```

Sound muted

This returns true if sound is muted or false if sound is unmuted.

```
values[0]:=GetSystemVariable(17);
if values[0] then
 beep;                                 //is muted
```

See Mute

Logged on user name

The MS Windows logged on user name. If a user is not logged on the return value is '' (empty).

```
values:=GetSystemVariable(18);
if (values[0] = 'Ray') then
beep;
```

or to ignore case:

```
if (UpperCase(values[0]) = UpperCase('Ray')) then
 beep;
```

Windows domain name

The MS Windows domain name.

```
values:=GetSystemVariable(19);
if (values[0] = 'Line 1') then
beep;
```

or to ignore case:

```
if (UpperCase(values[0]) = UpperCase('Line 1')) then
 beep;
```

Note: Noted during testing: if the computer is not on a domain the computer name is returned.

HMI version

The HMI version (same as Configure.exe file version) .

```
values:=GetSystemVariable(20);
if (values[0] = '10.0.8.0') then
 beep;
```

Note: If the intent is to display the version on a graphic screen, the most efficient method is to populate a script global when monitoring starts or when the graphic screen is opened. Reading this value in a script that executes over and over (graphic script for example) is not efficient.

Monitors

This returns the monitor count and information about each monitor.

```
values:=GetSystemVariable(21);
```

values[0]	= count of monitors.
values[1]	= first monitor number

Each monitor returns six properties.

values[x]	= monitor number
values[x + 1]	= left
values[x + 2]	= top
values[x + 3]	= width
values[x + 4]	= height
values[x + 5]	= True if primary monitor

Sorted list of user names

This returns a sorted list of user names and user levels. The sorted user names are first in the list followed by user levels.

Format
User count
User X
User Y
User Z
...
User X level
User Y level
User Z level

values:=GetSystemVariable(22);

values[0]	= count of users
values[1]	= first user
values[n]	= n user
...	
values[x]	= user name 1 user level
values[x + 1]	= user name 2 user level
values[n]	= user name n user level

Example

```
values:=GetSystemVariable(22);  //get sorted user name list
s1:='';                         //empty string
count:=values[0];               //how many user names
for i:= 1 to count do
 begin
  s1:=s1 + values[i];
  if (i <  count) then
   s1:= s1 + #13;               //add carriage return if more.
 end;

value:=GlobalSet('User','names',s1); //save to script global
```

Memory used by the HMI

This is the actually memory used/in use. It is not the amount of memory reserved by the application

```
values:=GetSystemVariable(23);

if (values[0] > 1073741824) then   // 1,073,741,824 is 1 gigabyte
 Beep;
```

Returns the last user version value/description/name

```
values:=GetSystemVariable(24);      //version number
ge.text:=values[0];

values:=GetSystemVariable(25);      //version description
ge.text:=values[0];

values:=GetSystemVariable(26);      //version name
ge.text:=values[0];
```

Simulation

This returns the true if at least one port is in simulation.

```
value:=GetSystemVariable(27);
```

Notification SMS pending queue count

This is the pending send message queue count, see SMSPurgeSendQueue.

```
value:=GetSystemVariable(28);
```

Related:
Function list

GetUserInputBoolean

This function is used to allow the user to select one of three buttons.

Inputs

Variable	Type	Description
Prompt	String	User prompt
Top button	String	Text in top button
Middle button	String	Text in middle button
Bottom button	String	Text in bottom button

Outputs

Variable	Type	Description
Value	Integer	The value of the button selected 1 = Top button 2 = Middle Button 3 = Bottom

If the user closes the window without making a selection, the value 3 is returned.
The buttons will wrap the text as needed.

Example

```
value:=GetUserInputBoolean('Start/Stop Pump…', 'Start', 'Stop', 'Cancel');

case value of
1: place start pump command; //top button
2: place stop pump command; //middle button
3: nothing; //bottom button
end;
```

OR

```
if value = 1 then
 place start pump code
else if value = 2 then
 place stop pump code
```

Related:
Function list
GetUserInputBoolean2
GetUserInputBoolean4
GetUserInputInteger
GetUserInputInteger64
GetUserInputFloat

GetUserInputBoolean2

This function is used to allow the user to select one of two buttons.

Inputs

Variable	Type	Description
Prompt	String	User prompt
Top button	String	Text in top button
Bottom button	String	Text in bottom button

Outputs

Variable	Type	Description
Value	Integer	The value of the button selected 1 = Top button 2 = Bottom button

If the user closes the window without making a selection, the value 2 is returned.
The buttons will wrap the text as needed.

Example

```
value:=GetUserInputBoolean2('Start/Stop Pump…', 'Start', 'Stop');

case value of
1: place start pump command; //top button
2: place stop pump command; //bottom button
end;

OR

if value = 1 then
place start pump code
else if value = 2 then
place stop pump code
```

Related:

Function list	GetUserInputBoolean4	GetUserInputInteger64
GetUserInputBoolean	GetUserInputInteger	GetUserInputFloat

GetUserInputBoolean4

This function is used to allow the user to select one of four buttons.

Inputs

Variable	Type	Description
Prompt	String	User prompt
Top button	String	Text in top button
Middle top button	String	Text in middle top button
Middle bottom button	String	Text in middle bottom button
Bottom button	String	Text in bottom button

Outputs

Variable	Type	Description
Value	Integer	The value of the button selected 1 = Top button 2 = Middle Top Button 3 = Middle Bottom Button 4 = Bottom

If the user closes the window without making a selection, the value 4 is returned.
The buttons will wrap the text as needed.

Example

```
value:=GetUserInputBoolean4('Start/Stop/Jog Pump…', 'Start', 'Stop', 'Jog', 'Cancel');

case value of
1: place start pump command; //top button
2: place stop pump command; //middle top button
3: place jog pump command; //middle bottom button
4: nothing; //bottom button
end;
```

OR

```
if value = 1 then
 place start pump code
else if value = 2 then
 place stop pump code
else if value = 3 then
 place jog pump code
```

Related:
Function list
GetUserInputBoolean
GetUserInputBoolean2
GetUserInputInteger
GetUserInputInteger64
GetUserInputFloat

GetUserInputDate

This function is used to allow the user to select a date.

Inputs

Variable	Type	Description
Prompt	String	A string to prompt the user for a date
Accept button	String	Text in the "Accept" button
Cancel button	String	Text in the "Cancel" button
Minus days	Integer	The number of previous days allowed (0 - 32768)
Plus days	Integer	The number of future days allowed (0 - 32768)

Outputs

Variable	Type	Description
Value[0]	Integer	Day
Value[1]	Integer	Month
Value[2]	Integer	Year

Minus days
To limit the day selection to X days before the current day. For example, to allow the user to select any day prior to the day the function is called enter 32768 for this field. To limit the day selection to 7 days prior to the day the function is called enter 7 for this field.

Plus days
To limit the day selection to X days after the current day. For example, to allow the user to select any day after the day the function is called enter 32768 for this field. To limit the day selection to 7 days after the day the function is called enter 7 for this field.

Example

```
//select a date in the last 89+ years and the next 89+ years
values:=GetUserInputDate('Select a date…', 'Accept', 'Cancel',32768 ,32768);
if values[0] <> 0 then
begin
//the OK button was selected
end;

//select today and the previous seven days
values:=GetUserInputDate('Select a date…', 'Accept', 'Cancel',7,0);
if values[0] <> 0 then
begin
//the OK button was selected
end;
```

Related:
Function list
GetUserInputTime
SetWindowDate

GetUserInputFloat

This function is used to allow the user to enter a floating point value. The value the user enters is range checked when the user presses the accept button. If the value is not in range the window will not close.

Inputs

Variable	Type	Description
Prompt	String	A string to prompt the user for a value
Accept button	String	Text in the "Accept" button
Cancel button	String	Text in the "Cancel" button
Low limit	Integer	The lowest value the user can enter
High limit	Integer	The highest value the user can enter

Outputs

Variable	Type	Description
Value[0]	Boolean	True if the user pressed the accept button and the value is in range False if the user pressed the cancel button.
Value[1]	Float	The value the user entered.

Example

```
values:=GetUserInputFloat('Enter new setpoint…', 'Accept', 'Cancel',22.5,75.5);
if values[0] then
begin
WriteValue(['PumpAPressSetpoint',5000, values[1]]);
end;
```

Related:
Function list
GetUserInputInteger
GetUserInputInteger64
GetUserInputBoolean
GetUserInputBoolean2
GetUserInputBoolean4

GetUserInputInteger

This function is used to allow the user to enter an integer value. The value the user enters is range checked when the user presses the accept button. If the value is not in range the window will not close.

Inputs

Variable	Type	Description
Prompt	String	A string to prompt the user for a value
Accept button	String	Text in the "Accept" button
Cancel button	String	Text in the "Cancel" button
Low limit	Integer	The lowest value the user can enter
High limit	Integer	The highest value the user can enter

Outputs

Variable	Type	Description
Value[0]	Boolean	True if the user pressed the accept button and the value is in range False if the user pressed the cancel button.
Value[1]	Integer	The value the user entered.

Example

```
values:=GetUserInputInteger('Enter new setpoint…', 'Accept', 'Cancel',0,100);
if values[0] then
begin
WriteValue(['PumpAPressSetpoint',5000, values[1]]);
end;
```

Related:
Function list
GetUserInputBoolean
GetUserInputBoolean2
GetUserInputBoolean4
GetUserInputInteger64
GetUserInputFloat

GetUserInputInteger64

This function is used to allow the user to enter a 64 bit signed integer value. The value the user enters is range checked when the user presses the accept button. If the value is not in range the window will not close.
Range: -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

Inputs

Variable	Type	Description
Prompt	String	A string to prompt the user for a value
Accept button	String	Text in the "Accept" button
Cancel button	String	Text in the "Cancel" button
Low limit	Integer	The lowest value the user can enter
High limit	Integer	The highest value the user can enter

Outputs

Variable	Type	Description
Value[0]	Boolean	True if the user pressed the accept button and the value is in range False if the user pressed the cancel button.
Value[1]	Integer	The value the user entered.

Example

```
values:=GetUserInputInteger64('Enter new setpoint…', 'Accept', 'Cancel',0,100);
if values[0] then
begin
 WriteValue(['PumpAPressSetpoint',5000, values[1]]);
end;
```

Related:
Function list
GetUserInputBoolean
GetUserInputBoolean2
GetUserInputBoolean4
GetUserInputInteger
GetUserInputFloat

GetUserInputString

This function is used to allow the user to enter a string.

Inputs

Variable	Type	Description
Prompt	String	A string to prompt the user for a value
Input	String	The string that will be placed in the edit field when the dialog is displayed (optional)
Accept button	String	Text in the "Accept" button
Cancel button	String	Text in the "Cancel" button
Virtual keyboard	Boolean	If true the popup keyboard will be displayed
Maximum length	Integer	Limits the length of the string (0 = unlimited)
Section	String	Section name (optional)
Item	String	Item name (optional)

Outputs

Variable	Type	Description
Values[0]	Boolean	True if the user pressed the accept button and the string is not empty False if the user pressed the cancel button.
Values[1]	String	The value the user entered.

The section and name refer to globals. If used, both fields must be valid.
If the user selected the cancel button value[1] will be empty.

Note: In a port configuration, a string read might have been configured to copy the string to an optional script global. Writing a value to the script global does not write the value to the PLC. GetUserInputString has an optional parameter to save the string to a script global. Configuring the function to save the string to the script global does not write the string to the PLC. "StringSet" must be used to write a string to a PLC. See the second example

Examples

```
values:=GetUserInputString('Enter string…', '','Accept', 'Cancel',true,0,'','');
if values[0] then
 LogEvent(values[1]);

values:=GetUserInputString('Enter string…', '','Accept', 'Cancel',true,0,'','');
if values[0] then                    //did the user select the OK button
 StringSet('Port X', '40000', '0', values[1]);

value:=GlobalGet('Pmp1','Message');
values:=GetUserInputString('Enter string…', value, 'Accept',
'Cancel',false,64,'Pmp1','Message');
```

Related:
Function list
GetUserInputStringMask

GetUserInputStringMask

This function is used to allow the user to enter a string with an optional mask character. The input is limited to a single line of text. A common use for this would be to allow a user to enter a password without the password displayed in the text field.

Inputs

Variable	Type	Description
Prompt	String	A string to prompt the user for a value
Input	String	The string that will be placed in the edit field when the dialog is displayed (optional)
Accept button	String	Text in the "Accept" button
Cancel button	String	Text in the "Cancel" button
Virtual keyboard	Boolean	If true the popup keyboard will be displayed
Maximum length	Integer	Limits the length of the string (0 = unlimited)
Section	String	Section name (optional)
Item	String	Item name (optional)
Mask character	String	Character to display in edit field (if blank, the text will not be masked)

Outputs

Variable	Type	Description
Values[0]	Boolean	True if the user pressed the accept button and the string is not empty

False if the user pressed the cancel button.

Values[1]	String	The value the user entered.

The section and name refer to globals. If used, both fields must be valid.
If the user selected the cancel button value[1] will be empty.

Note: See the note above about writing a string to a PLC.

Example

```
values:=GetUserInputStringMask('Enter string…', '','Accept', 'Cancel',true,0,'','*');
if values[0] then
LogEvent(values[1]);

value:=GlobalGet('Pmp1','Message');
values:=GetUserInputStringMask('Enter string…', value, 'Accept',
'Cancel',false,64,'Pmp1','Message','*');
```

Related:
Function list
GetUserInputString

GetUserInputTime

This function is used to allow the user to enter a time value.

Inputs

Variable	Type	Description
Prompt	String	A string to prompt the user for a value
Accept button	String	Text in the "Accept" button
Cancel button	String	Text in the "Cancel" button
Hour	Integer	0-23
Minute	Integer	0-59
Second	Integer	0-59
Millisecond	Integer	0-999

Outputs

Variable	Type	Description
Values[0]	Boolean	True if the user pressed the accept button and the time is valid False if the user pressed the cancel button.
Values[1]	Integer	Hour
Values[2]	Integer	Minute
Values[3]	Integer	Second
Values[4]	Integer	Millisecond

Example

```
values:=GetUserInputTime('Enter time…', 'Accept', 'Cancel',0,0,0,0);
if values[0] then
 theTime:=IntToStr(values[1]) + ':' + IntToStr(values[2]) + ':' + IntToStr(values[3]);
```

Related:
Function list
GetUserInputDate

GlobalClearSection

This function is used to clear all the names and values from a section.

Inputs

Variable	Type	Description
Section	String	Section name

Outputs

Variable	Type	Description
None		

Example

```
GlobalClearSection('Pump1');
```

Related:
Function list
Script Globals
GlobalGetSection
GlobalGetSectionCount
GlobalGetSectionItemNames
GlobalSave
GlobalSet

GlobalGet

This function is used to read the value of a script global.

Inputs

Variable	Type	Description
Section	String	Section name
Item	String	Item name

Outputs

Variable	Type	Description
Value	Variant	Value of requested section:item

If the <section:item> does not exist the result is '0'.

Example

value:=GlobalGet('Pump1','StartedCount');

Related:
Function list
Script Globals
GlobalClearSection
GlobalGetSection
GlobalGetSectionCount
GlobalGetSectionItemNames
GlobalSave
GlobalSet

GlobalGetSection

This function is used to read all item values of a script global section.

Inputs

Variable	Type	Description
Section	String	Section name
Sorted	Boolean	See below

Outputs

Variable	Type	Description
Values	Array of variant	Values of variables

The result is an array of values. The first element of the array contains the count of items. Example:

values[0] = 2
values[1] = 'Red'
values[2] = 'Blue'

If the <section> does not exist, the result is a single element array with a value of '-1' in the first element.

If the <section> is empty, the result is a single element array with a value of '0' in the first element.

Sorted

If "sorted" is true the section item names are sorted and the item values returned.
If "sorted" is false the item values are returned in the order stored in the section.

Example

values:=GlobalGetSection('Pump1');

Related:
Function list
Script Globals
GlobalClearSection
GlobalGet
GlobalGetSectionCount
GlobalGetSectionItemNames
GlobalSave
GlobalSet

GlobalGetSectionItemNames

This function is used to read all item names of a script global section.

Inputs

Variable	Type	Description
Section	String	Section name
Sorted	Boolean	See below

Outputs

Variable	Type	Description
Values	Array of variant	Values of variables

The result is an array of values. The first element of the array contains the count of items. Example:

values[0] = 2
values[1] = 'Dark'
values[2] = 'Light'

If the <section> does not exist, the result is a single element array with a value of '-1' in the first element.

If the <section> is empty, the result is a single element array with a value of '0' in the first element.

Sorted

If "sorted" is true the section item names are sorted and returned.
If "sorted" is false the section item names are returned in the order stored in the section.

Example

values:=GlobalGetSectionItemNames('Pump1');

Related:
Function list
Script Globals
GlobalClearSection
GlobalGet
GlobalGetSection
GlobalGetSectionCount
GlobalSave
GlobalSet

GlobalGetSectionCount

This function is used to read the count of items in script global section.

Inputs

Variable	Type	Description
Section	String	Section name

Outputs

Variable	Type	Description
Value	Integer	Item count

If the <section> does not exist, the result is '-1'.
If the <section> is empty, the result is '0'.

Example

value:=GlobalGetSectionCount('Pump1');

Related:
Function list
Script Globals
GlobalClearSection
GlobalGet
GlobalGetSection
GlobalGetSectionItemNames
GlobalSave
GlobalSet

GlobalSave

This function is used to save the script globals to disk.
The HMI saves the script globals to disk when runtime monitoring is stopped.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

```
GlobalSave;
```

Related:
Function list
Script Globals
GlobalClearSection
GlobalGet
GlobalGetSection
GlobalGetSectionCount
GlobalGetSectionItemNames
GlobalSet

GlobalSet

This function is used to set the value of a script global.

Inputs

Variable	Type	Description
Section	String	Section name
Item	String	Item name
Value	Variant	New value

Outputs

Variable	Type	Description
Value	Boolean	Command was successful

Example

value:=GlobalSet('Pump1','StartedCount',34); //saves the value 34
value:=GlobalSet('ButtonActive',MainWindow','Stop Button'); //save the text string Stop Button

Related:
Function list
Script Globals
GlobalClearSection
GlobalGet
GlobalGetSection
GlobalGetSectionCount
GlobalGetSectionItemNames

GPPCommand (General purpose port command)

This function is used to control/read/write/monitor a "general purpose" communication port type.

Inputs

Variable	**Type**	**Description**
Port name	String	Port name
Command	String	Command name
Values	Array of variant	Values defined by command

Outputs

Variable	**Type**	**Description**
Values	Array of variant	Values defined by command

Command	Function
Active	Opens/closes the port

```
values:=GPPCommand('Pump1','Active',[true]);      //opens the port
values:=GPPCommand('Pump1','Active',[false]);     //closes the port
```

values[0] contains a boolean indicating if the command executed.

```
if values[0] then
 ;              //command success
```

ClearInCount Clears the input buffer bytes received count

```
values:=GPPCommand('Pump1', 'ClearInCount',[]);
```

values[0] contains a boolean indicating if the command executed.

```
if values[0] then
 ;              //command success
```

GetStatus Returns status of several port properties

```
values:=GPPCommand('Pump1', 'GetStatus',[]);
```

values[0] True if port is open, false if port is closed
values[1] Byte count in receive buffer

Command	Function
LogString	Logs a string or strings to the diagnostic log.

```
values:=GPPCommand('Pump1', 'LogString',['some string']);
values:=GPPCommand('Pump1', 'LogString',['some string', 'another string']);
```

values[0] contains a boolean indicating if the command executed.

```
if values[0] then
 ;                //command success
```

PeekBytes Returns all the bytes from the input buffer but does not alter the input buffer byte count

```
values:=GPPCommand('Pump1', ' PeekBytes',[]);
```

values[0] contains the number of bytes returned, could be 0
values[1] contains an array of bytes (see example at ReadBytes)

Note: If the input buffer byte count is needed and not the bytes, the 'GetStatus' command is more efficient.

ReadBytes Returns all the bytes from the input buffer and sets the input buffer byte count to zero (0)

```
values:=GPPCommand('Pump1', ' ReadBytes',[]);
```

values[0] contains the number of bytes returned, could be 0
values[1] contains an array of bytes

Example

```
var
 inBuffer,inArray:variant;
 registerValue:integer;
begin
  inBuffer:=GPPCommand('GP_1','ReadBytes',[]);     //read the bytes
  count:=inBuffer[0];
  inArray:=inBuffer[1];          //Get the bytes to an array, easier to work with.
  registerValue:=inArray[4] + (inArray[3] * 256);
end;
```

Command	Function
SetDiagnostic	This provides setting one or more fields of the port diagnostic properties. A '#' (hash tag) is used to ignore a field. Each call must include a value for all fields. Some values take actions.

Value	Action
'#'	Ignore
'-'	Decrement field (only for 1..5)
'+'	Increment (only for 1..5)

Field order	Description
1	Reads requested
2	Reads completed
3	Writes requested
4	Writes completed
5	Watchdog count
6	Watchdog timed out (boolean)
7	Error (string)

Examples

```
//set the watchdog timed out true, ignore all others
values:=GPPCommand('Pump1', 'SetDiagnostic',[ '#', '#' , '#' , '#' , '#' ,true, '#']);
if values[0] then
 ;        //command success

//increment reads requested, ignore all others
values:=GPPCommand('Pump1', 'SetDiagnostic',['+', '#' , '#' , '#' , '#' , '#', '#']);

//increment reads completed, ignore all others
values:=GPPCommand('Pump1', 'SetDiagnostic',[' #', '+' , '#' , '#' , '#' , '#', '#']);

//set error string, ignore all others
values:=GPPCommand('Pump1', 'SetDiagnostic',[' #', '#' , '#' , '#' , '#' , '#', 'No
response']);

//set watchdog timeout and increment watchdog counter, ignore all others
values:=GPPCommand('Pump1', ' SetDiagnostic',[' #', '#' , '#' , '#' , '+' ,'true', '#']);
```

Command	Function
WritesBytes	Write the bytes out the port

values:=GPPCommand('Pump1', 'WriteBytes',[1,87,0,19,56]);

buf:=[1,87,0,19,56];
values:=GPPCommand('Pump1', ' WriteBytes', buf);

values[0] contains the number of bytes written, could be 0

Example:

To write an ASCII string with a carriage return and linefeed.

```
var
 s1:string;
 r:variant;
 cnt,i:integer;
begin
//if the port state is not configured to be active when runtime
//starts, the first and last line open and close the port.
//otherwise both can be removed.

 values:=GPPCommand('tcp1_', 'Active',[true]); //open port

 s1:='Test' + chr($0D) + chr($0A);     //string to transmit
 cnt:=length(s1);                      //get the string length

 //create a byte array to hold the bytes.
 r:=VarArrayCreate([0,cnt-1],$0011);    //varByte
 for i := 1 to cnt do                   //copy the characters as bytes
   r[i-1]:=ord(s1[i]);

 values:=GPPCommand('tcp1_','WritesBytes',r);    //send the bytes
 values:=GPPCommand('tcp1_', 'Active',[false]);  //close port
end;
```

Related:
Function list

Int64UnixDTMS

This function decodes an int64 (64 bit signed integer) to date and time values. The int64 must be encoded as an UNIX time with microseconds. This conforms to the Allen Bradley LINT time data type.

Inputs

Variable	Type	Description
<a value>	Int64	An int64 Unix encoded time (with microseconds).

Outputs

Variable	Type	Description
Values	Array of integer	See below

Values[0]	year
Values[1]	month
Values[2]	day
Values[3]	hour
Values[4]	minute
Values[5]	second
Values[6]	millisecond
Values[7]	microsecond

Example

value:= Int64ToUnixDTMS (1428779032120789);

Related:
Function list

ISO8601

This returns a date and or time, ISO8601 formatted string.

Inputs

Variable	Type	Description
DateTime	TDateTime	A "date/time"
Date	Boolean	Include date
Time	Boolean	Include time
FormatForFilename	Boolean	The result will replace ":" and "," with "-".

Outputs

Variable	Type	Description
Value	String	The date and or time as a ISO8601 formatted string.

Examples

Assume "Now" is March 26, 2024 10:11:12:456 -5 UTC

```
value:=ISO8601(Now,true,false,false);      //2024-03-26
value:=ISO8601(Now,true,true,true);        //2024-03-26T10-11-12-456-05-00
value:=ISO8601(Now,false,true,false);      //10-11-12-456-05-00
value:=ISO8601(Now,true,true,false);       //2024-03-26T10:11:12:456-05:00
```

Notes:

1. If "DateTime" is invalid a runtime error will occur and an empty string returned.
2. If "Date" and "Time" are false an empty string is returned.

Related:
Function list

IsUserWindowOpen

This function returns true if a user window with the input name is open.

Inputs

Variable	Type	Description
Window name	String	Name of window to check if open

Outputs

Variable	Type	Description
Value	Boolean	True if the window is open False if the window is not open or a window with the input name is not configured

Examples

```
value:=IsUserWindowOpen('Pump Station 9');
if value then
begin
//window is open
end;

if IsUserWindowOpen('Pump Station 9') then
begin
//window is open
end;

if not IsUserWindowOpen('Pump Station 9') then
begin
OpenWindow('Pump Station 9'); //window is not opened or not configured, so issue open
command
end;
```

Related:
Function list
CloseWindow
OpenWindowEx
OpenWindowUserSelect

JSONOutToString

This function process the file specified and returns a JSON string

Inputs

Variable	Type	Description
File name	String	A file name in the <project>\JSONOut path.
Options	Array	Array of options.

Outputs

Variable	Type	Description
Array	Error	0 for no error, and a negative number for error.
	String	The result of processing all data collection commands in the file.

Options

Reload

The first call to JSONOutToString(<file name>,[]) loads the file and parses the text to validate all the commands are formatted correctly and are accessing existing data. If the validation fails an error code is returned (See error codes below). Subsequent calls to JSONOutToString(<file name>,[]) will return the same error code. One solution is to quit runtime, correct the issue with the file and restart runtime. Another solution is to correct the issue with the file and call JSONOutToString(<file name>,['Reload']). This will clear the error code and execute the first time call validation. **Always check the error code value** before accessing the JOSN result. See note 2 below.

Example

```
values:=JSONOutToString('Pump_1',[]);
if (values[0] = 0) then
 values[1]                          //result string if no error
```

Notes:

1) The file must be in the project directory fixed path <project name>\JSONOut, include only the directory path inside the fixed path. Example: "C:\Lift house\JSONOut\pumps\Pump_1". Blue is the fixed path and red is the portion to use in the call: value:=JSONOutToString('pumps\Pump_1',[]); The "Insert/JSONOut file name" menu can be used to a select file.

2) **Always check the error code value** before attempting to use the JSON result. If an error is present the JSON string portion of the result will not be present. Attempting to access values[1] when values[0] is not zero (0), will cause a script failure.

Error codes

The first value of the output array is an error code. If the value is not equal to zero, an error occurred and the result array will be one element long, containing the error code. If the result value in the first array element is 0, the JSON string will be in the next (second) element. See the example, above.

Code	Description
0	No error
-1	File name empty
-2	File not found
-3	Unable to create runtime object
-4	File is empty
-5	Initialize, unable to process line, see "Event log"
-6	Initialize, tagname not found, see "Event log"
-7	Initialize, point ID invalid, see "Event log"
-8	Initialize, tagname not found, see "Event log"
-9	Point not found
-10	Initialize, Script global not found, see "Event log"
-11	Initialize, user array count less than 1, see "Event log"
-12	Initialize, unknown user array, see "Event log"
-13	Initialize, failed to create variant array, see "Event log"
-14	Array not found
-15	Array read failed
-200...-299	See UArray error codes. -0 ..-99 is the code.

Related:
Function list

JSONToHostPoints

This function parses a JSON string and places the values in the host points defined by the script global items. The script global item and JSON item (full path) must match.

Inputs

Variable	Type	Description
Script global	String	A script global section name
JSON string	String	A JSON string

Outputs

Variable	Type	Description
None		

Example

JSONToHostPoints('Power loads', <JSON string>);

Related:
Function list
JSONToScriptGlobal

JSONToScriptGlobal

This function parses a JSON string and places the values in the script global items that match the JSON item names. The script global item and JSON item (full path) must match.

Inputs

Variable	Type	Description
Script global	String	A script global section name
JSON string	String	A JSON string

Outputs

Variable	Type	Description
None		

Example

JSONToScriptGlobal('Power loads',<JSON string>);

Related:
Function list
JSONToHostPoints

KillAProcess

This function attempts to terminate an external process (program).
The function attempts to send the application a close message. If the close message fails the function attempts to terminate the process. Also, see KillAProcess2.

Inputs

Variable	Type	Description
Window title	String	The application main window title (not case sensitive)
Time to wait	Integer	Milliseconds to wait for response (0-10000), see Notes

Outputs

Variable	Type	Description
Result	Boolean	True = a window title match was found False = a window was not found

Example

result:=KillAProcess('Notepad',100);

Notes:

1) A value of -1 and the function will skip the attempt to close the application and issue a terminate command.
2) Use the lowest timeout value possible to prevent processing delays.

Related:
Function list
KillAProcess2
LaunchApplication

KillAProcess2

This function attempts to terminate an external process (program) using the application <name>.exe. **Note:** This will kill all processes with the name entered. For example, two files opened with "Notepad.exe", will be two processes. Calling this function with "Notepad.exe" will terminate both processes.

Inputs

Variable	Type	Description
Application name	String	The application program name <name>.exe.

Outputs

Variable	Type	Description
Result	Integer	Returns the number of processes terminated.

Example

result:=KillAProcess2('Notepad.exe');

Related:
Function list
KillAProcess
LaunchApplication

LaunchApplication

This function is used to launch an application, control panel application or access a file with an application.

Inputs

Variable	Type	Description
File	String	The file, application or control panel to display
Operation	String	Action to be performed
Parameter[0]	String	Parameter passed to the application
Parameter[1]	String	Parameter passed to the application
Parameter[2]	String	Parameter passed to the application
Parameter[3]	String	Parameter passed to the application
Parameter[4]	String	Parameter passed to the application

Outputs

Variable	Type	Description
None		

File
The file variable will need to be the complete path if it references a file or application. Put a " at the beginning and the end of the string. If it is a control panel application, it does not need the full path.

Some application paths, e.g. MS Paint, Calculator, are known to MS Windows and the complete path is not required.

Operation

Edit	Launches an editor and opens the document for editing.
Open	Launches an application. If this file is not an executable file, its associated application is launched.
Print	Prints the document file.

Parameter[x] (optional)
Values that need to be passed to the application. If it is a string with spaces, place a " at the beginning and end of the string.

Note: The quote symbols are double quotes. It is not two single quotes.
" quotation mark, or double quote
' apostrophe, or [single] quote

The scripting engine needs the single quote to delimit a string. For this function when passing a string it would need to be:
'" <some string> "' (no space between the quote-double quote and the double quote-quote)

quote double quote <some string> double quote quote

Examples

```
//Launch the calculator
LaunchApplication('calc.exe', 'open',[]);

//Launch MS paint
LaunchApplication('mspaint.exe', 'open',[]);

//Open a word document
LaunchApplication('"C:\Text files\Start pump instructions.doc"', 'edit',[]);
```

Related:
Function list
KillAProcess
KillAProcess2

LoadRecipe

This function is used to command a recipe to write values to the destination points when the recipe is configured with the ingredients in each row and the recipe is a column. If the recipe is configured for each row as a recipe and the ingredients are in the column, use LoadRecipe2.

Inputs

Variable	Type	Description
Recipe name	String	Name of an XLS/ODBC recipe (HMI recipe name)
Column	String	The column of the spreadsheet/table. XLS: The field can be a number or letter. 1 = A, 2 = B, etc. or A, B, C, D, etc. ODBC: The field must be a string that is the name of the column.

Outputs

Variable	Type	Description
None		

Examples

XLS:

```
LoadRecipe ('Butter Cookies', 'B');          //column B
LoadRecipe ('Fuel mixture', 47);             //column AU
```

ODBC:

```
LoadRecipe ('Butter Cookies', 'LBS'); //column LBS
LoadRecipe ('Fuel mixture', 'Large');        //column Large
```

Related:
Function list
RecipeReloadSheet

LoadRecipe2

This function is used to command a recipe to write values to the destination points when the recipe is configured with the ingredients in each column and the recipe is a row. If the recipe is configured for each row as an ingredient and the recipes are in the rows, use LoadRecipe.

Inputs

Variable	Type	Description
Recipe name	String	Name of an XLS/ODBC recipe (HMI recipe name)
Row	String	The row/name of the recipe in the spreadsheet/table. XLS: The field can be a number or letter. 1 = A, 2 = B, etc. or A, B, C, D, etc. ODBC: The recipe name must be in the first field/column of the table. **Note:** Do not use the first column for a point when using this command. The first column is the recipe name.

Outputs

Variable	Type	Description
None		

Examples

XLS:

```
LoadRecipe2 ('Butter Cookies', 'B');        //row B
LoadRecipe2 ('Fuel mixture', 47);           //row AU
```

ODBC:

```
LoadRecipe2 ('Butter Cookies', 'LBS');      //row LBS
LoadRecipe2 ('Fuel mixture', 'Large');      //row Large
```

Related:
Function list
RecipeReloadSheet

LogEvent

Adds an entry to the event log file with the current time.

Inputs

Variable	Type	Description
Event string	String	String to add to event log file

Outputs

Variable	Type	Description
None		

Example

LogEvent ('A text string');

The maximum string length is 100 characters.

Related:
Function list
CloseEventWindow
OpenEventWindow

LoggerGetMaxMinMean

This function returns the maximum, minimum and mean value for a data logger point.

Inputs

Variable	Type	Description
Tagname	String	The point tagname
Item number	Integer	The point item number, usually 5000
Day	Integer	Day (see below)
Month	Integer	Month
Year	Integer	Year

Outputs

Variable	Type	Description
Values	Array of float	Success, maximum, minimum and mean Success will be 0, less will be failure -1 = no data found

If the "Day" is 0 (zero) the current day/date is used.
If the "Day" is negative (< 0) the current day minus the value is used.

Day value	Day
0	Current day, today, etc.
-1	Yesterday
-2	Day before yesterday
-n	Previous days

If the day is not 0 or a negative value, the day, month and year must be valid.

Examples

```
values:=LoggerGetMaxMinMean('Pressure',5000, 0,0,0);     //today
if (values[0] > -1) then
 begin
  maxValue:=values[1];
  minValue:=values[2];
  meanValue:=values[3];
 end;

values:=LoggerGetMaxMinMean('Pressure',5000, -1,0,0);    //yesterday
values:=LoggerGetMaxMinMean('Pressure',5000, -2,0,0);    //day before yesterday
values:=LoggerGetMaxMinMean('Pressure',5000, 10,2,2018);        //10 February 2018
```

Related:
Function list

LogOn

The "log on" dialog will appear.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
Value	Boolean	True if a user is logged on. False if a user is not logged on.

Note:
This function returns immediately after being called. The result/value is the "current state" of a user being logged on or a user not being logged on.

Example

```
value:=LogOn;
if value then
begin
//a user is logged on
end;

if LogOn then
begin
//a user is logged on
end;
```

Related:
Function list
LogOff
ForceLogOn

LogOn2

A specialized logon, with a log off button, will appear. The password is limited to numbers; a ten key pad is provided in the dialog.

Inputs

Variable	Type	Description
Names	String array	String array of user names. The names must be all or a subset of the names in the user configuration. The case must match.
Options	Variable	Array of options. (see below)

Outputs

Variable	Type	Description
None		

Notes:

1) Unlike the "LogOn" function, this function is blocking, blocks the script from continuing until the dialog is dismissed by user action. i.e. successful log on, cancel selection, etc.
2) The popup keyboard option does not apply to this logon dialog.
3) Because this script is called when a user is not logged in, it must be placed in a graphic element script. If a user is always logged in, the script can be in the script directory and called as needed.

Examples

LogOn2(['Rich', 'Andrea'], []);

LogOn2(['Rich', 'Andrea'], ['buttonOpacity=64']);

Option properties

All options use the item name/pair format. e.g. WindowLeft=100

Option	Default	Description
WindowLeft	-1	Set the window "top" position *1
WindowTop	-1	Set the window "left" position *1
ClickColor	Grey	The color used when a keypad button is pressed.
ButtonOpacity	128	The opacity used when a keypad button is pressed.
WindowColor	Dark grey	The window color.
ButtonFontColor	Black	The user name button font color.
ButtonFontSize	16	The user name button font size.

ButtonHeight	64	The user name button height.
ButtonFont	Consolas	The user name button font name.

Notes:

1) When both properties are minus one (-1) the dialog will be centered on the main screen.

Related:

Function list
LogOff
LogOn
ForceLogOn

LogOff

This function logs off the current user. A dialog window will appear with two buttons.

Inputs

Variable	Type	Description
Prompt	String	Window Title
Accept button	String	Accept Button Text
Cancel button	String	Cancel Button Text
Silent	Boolean	If true the dialog window will not appear.

Outputs

Variable	Type	Description
Value	Boolean	True if the user was logged off or silent. False if the user was not logged off.

If a user is not logged on when this command is issue the command will return true without showing the dialog window.

```
value:=LogOff('Log off', 'Yes', 'No', false);
if value then
begin
//the user was logged off
end;

if LogOff('Log off', 'Yes', 'No',true) then
begin
//the user was logged off
end;
```

Related:
Function list
LogOn

MousePosition

This function is used to get the mouse pointer position.

Inputs

Variable	Type	Description
Local client	Boolean	If false the mouse coordinates will be in global coordinates. If true the mouse coordinates will be in client coordinates.

Outputs

Variable	Type	Description
Position	Array of variant	[0] - horizontal position [1] - vertical position

Examples

```
pt:=MousePosition(false);
//returns the global position of the mouse pointer

pt:=MousePosition(true);
//returns the client position of the mouse pointer

pt:=MousePosition(false);
value:=OpenWindowEx('Pump Station 9',pt[0],pt[1],true);
//opens the window with center of the window at the mouse pointer

pt:=MousePosition(false);
value:=OpenWindowEx('Pump Station 9',pt[0],pt[1],false);
//opens the window with the left and top at the mouse pointer
```

Related:
Function list
MousePositionSet
OpenWindowEx

MousePositionSet

This function is used to set the mouse pointer position.

Inputs

Variable	Type	Description
X	Integer	Horizontal position
Y	Integer	Vertical position
Local client	Boolean	If false the mouse coordinates will be in global coordinates. If true the mouse coordinates will be in client coordinates.

Outputs

Variable	Type	Description
Result	Boolean	True if the function (OS) was called

Examples

value:=MousePositionSet(10,10,false);
//sets the position of the mouse pointer in screen coordinates

value:=MousePositionSet(10,10,true);
//sets the position of the mouse pointer in client (window) coordinates

Related:
Function list
MousePosition
OpenWindowEx

MQTTV5Publish

This function is used to publish data to the server outside the normal point or string objects.

Inputs

Variable	Type	Description
Port name	String	MQTT version 5 port name
Topic name	String	Topic name in the server
Data	String	Data to send to server
QoS	Integer	0,1 or 2
Retain	Boolean	Replace any existing "retained" value
Properties	Array of string	String(s) of properties to attach Format: <property ID>=<value> Example: '3=123'
User properties	Array of string	String(s) of properties to attach Format: <name>=<value> Example: 'TheName'='XYZ'

Outputs

Variable	Type
Result	Integer

Result	Description	Result	Description
0	Success	-1	Missing topic name
-2	QoS not 0, 1, or 2	-3	Invalid property
-4	Invalid port	-5	Port not MQTT Version 5
-6	Port write error	-7	Create error

Examples

```
value:=MQTTV5Publish('Pmp_1','Lever', 'Inc', 0, False, ['2=XYZ'], ['User=Ops', 'Shift=2']]);
value:=MQTTV5Publish('Pmp_1','Lever', '123.456', 0, False, [], []);        //no properties
```

Notes:

1) Supported properties: Message Expiry Interval (2), Content type (3), Response topic (8), Correlation data (9). Any other property value will cancel the publish. (ID in decimal.)
2) All properties can be present once.
3) The server may limit the QoS level.
4) The server may not support "retain".
5) Invalid publish values may cause the server to disconnect.

Related:
Function list

Mute

When state is "true", function silences all playing sounds and disables all sounds from playing until the command is called with "state" set to false.

Inputs

Variable	Type	Description
State	Boolean	True = silence all sounds playing and disable all sounds from being queued for play False = enable sounds to be queued for play
LogIt	Boolean	Add entry to the event log on change of state

Outputs

Variable	Type	Description
None		

Example

```
Mute(true, true);          //mute sound
Mute(false, true);         //unmute sound
```

Related:
Function list
PlaySound
PlaySound2

Navigate

This function is the portal to an object that maintains a list of window names that have been opened using a "Navigate" function. This object provides for a "Previous/Next" logic system. This type of system is most often seen on a single window interface but, can be used in multi-window systems.

Notes:

1) Windows not opened with a navigate command will not be incorporated into the list.
2) All commands and parameters are case sensitive, where used.
3) All commands that open a window, close the current opened window.

Inputs

Variable	Type	Description
Properties	Array	An array consisting of a command and any other needed values for the command.

Outputs

Variable	Type	Description
Various	Variant	The result of the command.

Command	Parameter 1	Result	Description
Can next		Boolean	True when the current window is not the last in list
Can previous		Boolean	True when the current window is not the first in the list
Clear		Boolean	Clear the list of all names except the current window
Count		Integer	Returns the count of names in the list
Current window		String	Returns the current window name
First		Boolean	Opens the first window in the list
Get next		String	Returns the next window in the list
Get previous		String	Returns the previous window in the list
Last		Boolean	Opens the last window in the list
List		Array of string	Returns the count and all window names in an array
Next		Boolean	Opens the next window in the list
Open	Window name	Boolean	Opens a window and adds window name to the list
Previous		Boolean	Opens the previous window in the list
Set limit	Limit value	Boolean	Sets the list maximum size (4-256), default is 32

Examples

Command	Function
Can next	Returns true if the current window is not the last window in the list. Can be used to change the appearance of a button when a "Next" command will be successful.

```
value:=Navigate(['Can next']);
if value  then
 ;                              //Next command will function
```

Can previous	Returns true if the current window is not the first window in the list. Can be used to change the appearance of a button when a "Previous" command will be successful.

```
value:=Navigate(['Can previous']);
if value  then
 ;                              //Previous command will function
```

Clear	Clears the list of all windows except the current window.

```
value:=Navigate(['Clear']);
```

Count	Returns the count of window names in the list.

```
value:=Navigate(['Count']);
```

Current window	Returns the name of the current window.

```
value:=Navigate(['Current window']);
```

First	Opens the first window in the list and closes the current window.

```
value:=Navigate(['First']);
```

Get next	Returns the next window name in the list from the current window.

```
value:=Navigate(['Get next']);        //if the result is empty ('') there is not a "next"
                                      //window.
```

Get previous	Returns the previous window name in the list from the current window.

```
value:=Navigate(['Get previous']);    //if the result is empty ('') there is not a "previous"
                                      //window.
```

Last Opens the last window in the list and closes the current window.

```
value:=Navigate(['Last']);
```

List Returns the count and window names in the list.

```
values:=Navigate(['List']);
```

```
values[0]    = count of window names returned
values[1]    = first window name in the list
values[n]    = window name in the list up to the count(values[0]) value
```

Next Opens the next window in the list and closes the current window.

```
value:=Navigate(['Next']);
```

Open Opens the named window, adds the name to the list and closes the current window.

```
value:=Navigate(['Open', '<a window name>']);
```

Previous Opens the previous window in the list and closes the current window.

```
value:=Navigate(['Previous']);
```

Set limit Sets the maximum size of the list. When the limit is reached, the lower half of the list is deleted. The value must be 4 – 256. The default is 32.

```
value:=Navigate(['Set limit',32]);
```

Related:
Function list
OpenWindow
OpenWindowEx
OpenWindowUserSelect

MemoCommand

This function is used to send a command to a memo element on a window.

Inputs

Variable	Type	Description
Window name	String	The name of the window that contains the memo element.
Memo name	String	The name of the memo element.
Command	Integer	Command
Command fields	Array of Variant	See commands below

Outputs

Variable	Type	Description
None		

Commands

Name	Value	Description
Open	1	Open a file. The command field contains the full path of the file to open.
Save	2	Save the current memo contents to the file. If the command field is blank, a save is performed. The memo contents are saved to the file used to load the memo. If the command field is not blank, it contains the full path and file name to save the file. This is a "Save as" command.
Append	3	Append the string to the end of the memo text.
Clear	4	Clear the memo contents.

Examples

```
//load the file into the memo
MemoCommand('Mixer 1', 'Main', 1, ['C:\Instructions\Peanut.txt']);

MemoCommand('Mixer 1', 'Main', 2, []);                    //saves the memo contents

//saves the memo contents to the file specified
MemoCommand('Mixer 1', 'Main', 2, ['C:\Instructions\PeanutChange.txt']);
```

```
//append the text to the end of the memo
MemoCommand('Mixer 1', 'Main', 3, ['First detected!']);

//clear the memo contents
MemoCommand('Mixer 1', 'Main', 4, []);
```

Related:
Function list

ODBCAddToInList

This function is used to add a read command to the 'in' list.

Inputs

Variable	Type	Description
Port name	String	Port name
List	Integer	0 = point list, 1 = script globals

List = 0 (point list)
Point name String Name of the point
Item number String Item number of the point to receive the data

List = 1 (script globals)
Section name String Script global section name
Item name String Script global item name receive the data

Column	String	Column (field) in the database table
Row	String	Row in the database table

Outputs

Variable	Type	Description
Value	Boolean	True if addition was accepted False if the addition was not accepted

Note:
This function does not check for the presence of another command in the list with the same configuration.

Examples

value:=ODBCAddToInList('Recipe1',0,'Flour',5000, 'Ingredients', '2');

value:=ODBCAddToInList('Recipe2',1,'Flour','PoundsNeeded', "Ingredients', '2');

Related:
Function list
ODBCAddToOutList
ODBCClearInList
ODBCIssueRead

ODBCAddToOutList

This function is used to add a write command to the 'out' list.

Inputs

Variable	Type	Description
Port name	String	Port name
List	Integer	0 = point list, 1 = script globals

List = 0 (point list)
Point name String Name of the point
Item number String Item number of the point for data source

List = 1 (script globals)
Section name String Script global section name
Item name String Script global item name for data source

Column	String	Column (field) in the database table
Row	String	Row in the database table

Outputs

Variable	Type	Description
Value	Boolean	True if addition was accepted False if the addition was not accepted

Note:
This function does not check for the presence of another command in the list with the same configuration.

Examples

value:=ODBCAddToOutList('Recipe1',0,'Flour',5000);

value:=ODBCAddToOutList('Recipe2',1,'Flour','PoundsNeeded');

Related:
Function list
ODBCAddToInList
ODBCClearOutList
ODBCIssueWrite

ODBCClearInList

This function is used to clear the 'in' list of all commands.

Inputs

Variable	Type	Description
Port name	String	Port name
List	Integer	0 = point list, 1 = script globals

Outputs

Variable	Type	Description
Value	Boolean	True if the list was cleared False if the list was not cleared

Example

value:=ODBCClearInList('Recipe2', 0);

Related:
Function list
ODBCClearOutList

ODBCClearOutList

This function is used to clear the 'out' list of all commands.

Inputs

Variable	Type	Description
Port name	String	Port name
List	Integer	0 = point list, 1 = script globals

Outputs

Variable	Type	Description
Value	Boolean	True if the list was cleared False if the list was not cleared

Example

value:=ODBCClearOutList('Recipe3', 1);

Related:
Function list
ODBCClearInList

ODBCIssueRead

This function is used to issue all the commands in the 'in' list.

Inputs

Variable	Type	Description
Port name	String	Port name

Outputs

Variable	Type	Description
Value	Boolean	True if the command was accepted False if the command was not accepted

The port executes the command in a thread. Check the 'command active' value to determine when all the reads have executed.

Example

value:=ODBCIssueRead('Recipe3');

Related:
Function list
ODBCIssueWrite

ODBCIssueWrite

This function is used to issue all the commands in the 'out' list.

Inputs

Variable	Type	Description
Port name	String	Port name

Outputs

Variable	Type	Description
Value	Boolean	True if the command was accepted False if the command was not accepted

The port executes the command in a thread. Check the 'command active' value to determine when all the writes have executed.

Example

value:=ODBCIssueWrite('Cook4');

Related:
Function list
ODBCIssueRead

ODBCDatalogger

This function is used to add a record to the database. The refresh rate must be zero (0) for this command to be successful.

Inputs

Variable	Type	Description
Port name	String	Port name

Outputs

Variable	Type	Description
Value	Boolean	True if addition was accepted False if the addition was not accepted

Examples

```
value:=ODBCDatalogger('Air Temp Log');
if value then
beep; //success
```

Related:
Function list
ODBCDatalogger2
ODBCDataLoggerDelete
ODBCDataLoggerSetRefresh
ODBCDataLoggerPause

ODBCDatalogger2

This function is used to add a record to the database.

1) The refresh rate must be zero (0) for this command to be successful.
2) The string **ODBCLoggingControlExternal=1** must be in the connection parameters.
3) The number of values passed must match the count of fields defined.
4) This connection will attempt to reconnect after a failed first attempt or a lost connection after a successful connection. To enable reconnection logic, add a string **ODBCLoggingReconnectMills=5000** in the connection parameters. The value (e.g. 5000) is the count milliseconds after the connection fails to attempt a reconnection and cannot be less than 1000. A value less than 1000 disables the logic.

Inputs

Variable	Type	Description
Port name	String	Port name
Values	Array of string	Values to write to database

Outputs

Variable	Type	Description
Value	Boolean	True if addition was accepted False if the addition was not accepted

Examples

```
value:=ODBCDatalogger2('<ODBC port name>',['<some value>', ... ]);
if value then
beep; //success

value:=ODBCDatalogger2('Air Temp Log', [Now,'123.4', 'Low', 'Run']);
```

Related:
Function list
ODBCDatalogger
ODBCDataLoggerDelete
ODBCDataLoggerSetRefresh
ODBCDataLoggerPause

ODBCDataLoggerDelete

This function is used to delete all records in the table.

Inputs

Variable	Type	Description
Port name	String	Port name
Use Delete	Boolean	True = Use "Delete" command False = Use "Truncate" command (faster)

Outputs

Variable	Type	Description
Value	Boolean	True if the command was accepted False if the command was not accepted

Notes:

1) This command uses the SQL "Truncate" command to delete all the records in the table. This command is much faster than the "Delete" command. Not all databases support the "Truncate" command. If the database used does not support the "Truncate" command, set "Use delete" to true and the "Delete" command will be used.
2) Excel does not support the "Truncate" or "Delete" command.

Example

```
value:=ODBCDataloggerDelete('Air Temp Log', False);
if value then
 beep;                                  //success
```

Related:
Function list
ODBCDatalogger
ODBCDatalogger2
ODBCDataLoggerSetRefresh

ODBCDataLoggerPause

This function is used to pause data logging. This only applies to a data logger that was configured with a refresh rate of one hundred (100) or greater.

Inputs

Variable	Type	Description
Port name	String	Port name
State	Boolean	True = pause False = resume

Outputs

Variable	Type	Description
Value	Boolean	True if the command was accepted False if the command was not accepted

Example

```
value:=ODBCDataloggerPause('Air Temp Log',True);
if value then
beep; //success
```

Related:
Function list
ODBCDatalogger
ODBCDatalogger2
ODBCDataLoggerDelete
ODBCDataLoggerSetRefresh

ODBCDataLoggerSetRefresh

This function is used to modify the refresh rate of the data logger. This only applies to a data logger that was configured with a refresh rate of one hundred (100) or greater.

Inputs

Variable	Type	Description
Port name	String	Port name
New rate	Integer	New refresh rate

Outputs

Variable	Type	Description
Value	Boolean	True if new rate was accepted False if the new rate was not accepted

Example

```
value:=ODBCDataloggerSetRefresh('Air Temp Log',5000);
if value then
beep; //success
```

Related:
Function list
ODBCDatalogger
ODBCDataLoggerDelete
ODBCDataLoggerPause

ODBCSetTableName

This function is used to set the name of the table.

Inputs

Variable	Type	Description
Port name	String	Port name
Table name	String	Table name

Outputs

Variable	Type	Description
Value	Boolean	True if the function succeeded False if the function did not succeed

Example

```
value:=ODBCSetTableName('Recipe2','Large Cookie');
if ODBCSetTableName('Recipe2','Large Cookie') then
begin
//success
end;
```

Related:
Function list

OmniRetrieveReport

This function is used to command the selected port to retrieve a report from the flow computer. The named report is added to an internal queue. The reports are retrieved in the order added to the queue.

Inputs

Variable	Type	Description
Port name	String	Name of port
Report name	String	Name of report to retrieve
Display	Boolean	Open the report for viewing when retrieval complete

Outputs

Variable	Type	Description
Value	Boolean	True if the report was queued False if the report did not queue

Report names

Name	Extension				
Current snapshot	.snr	Current status	.sts	Current product file	.prd
Historical alarm	.alr	Historical audit trail	.adt	Historical last local snapshot	.snl
Batch last	.b01	Batch 2nd last	.b02	Batch 3rd last	.b03
Batch 4th last	.b04	Batch 5th last	.b05	Batch 6th last	.b06
Batch 7th last	.b07	Batch 8th last	.b08	Daily last	.d01
Daily 2nd last	.d02	Daily 3rd last	.d03	Daily 4th last	.d04
Daily 5th last	.d05	Daily 6th last	.d06	Daily 7th last	.d07
Daily 8th last	.d08	Prover last	.p01	Prover 2nd last	.p02
Prover 3rd last	.p03	Prover 4th last	.p04	Prover 5th last	.p05
Prover 6th last	.p06	Prover 7th last	.p07	Prover 8th last	.p08
Archive 701	.701	Archive 702	.702	Archive 703	.703
Archive 704	.704	Archive 705	.705	Archive 706	.706
Archive 707	.707	Archive 708	.708	Archive 709	.709
Archive 710	.710	Alarm 711	.711	Audit 712	.712

Notes:
1) A blank report name aborts any current report retrieval and empties the queue.
2) The port name and report name are case sensitive.
3) If the named report is in the queue, 'true' is returned and the report is NOT added to the queue.
4) The demo version is limited to the 'Current snapshot' and 'Batch last'.

Example
value:=OmniRetrieveReport('Flow1','Current snapshot',false);

Related:
Function list
OmniViewReport

OmniViewReport

Open the Omni calendar report viewing window.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

OmniViewReport;

Related:
Function list
OmniRetrieveReport

OnAlarmEvent

The function is called when an alarm is added to or removed from the active alarm list. The script is specified in the project settings. The script should only contain the one procedure.

Inputs

Variable	Type	Description
Tagname	String	Point tagname
State	Boolean	True = alarm added to list False = alarm removed from list
Condition	String	Alarm condition status
Digital	Boolean	Point type, digital or analog
AlarmTime	TDateTime	The alarm active time or reset time. (see state)
Value	Float	The point value at alarm or reset time.

Outputs

None

Example

Pascal

```
procedure OnAlarmEvent(tagname, state, condition, digital, alarmTime, value);
 begin
   CustomLogCol ('AlarmEvent',[tagname, state, condition, digital, DateTimeToStr(alarmTime),
                                    value]);
 end;
```

Basic

```
function OnAlarmEvent(tagname, state, condition, digital, alarmTime, value)
      CustomLogCol ("AlarmEvent",[tagname, state, condition, digital,
      DateTimeToStr(alarmTime), value])
 end function
```

Note: If the "Event log" has an entry "Script: <script name> missing "OnAlarmEvent" function.", the script specified in "Project settings/On alarm script", must be corrected and "Runtime" restarted for "OnAlarmEvent" to function.

Related:
Function list

OnAlarmPanelCellDraw

The function is only called when the cell of an alarm panel is rendered. The script should be fast and only used when other configuration properties do not provide the needed feature. This callback should be in a standalone script for faster processing.

Below is a sample script.

```
//this is an example of the OnAlarmPanelCellDraw (override) function.
//the script should limit its actions (be fast).

function OnAlarmPanelCellDraw(args):variant;
var
 r:variant;
 cellBGColor:TColor;
begin
 result:=null;      //default, no result = draw as configured

//the args contain:
//args[0]           count values X, accessing beyond X is an error
//args[1]           tagname
//args[2]           state
//args[3]           alarm kind
//args[4]           alarm group
//args[5]           alarm time
//args[6]           priority
//args[7]           cell text

//this example, the cell background color is set from the alarm group
value
 case args[4] of
  1:    cellBGColor:=clRed;
  2:    cellBGColor:=clYellow;
  3:    cellBGColor:=clGreen;
  else  Exit;                 //no change, exit
 end;
//the color can be number. See SetWindowColor for partial list of
colors.

//now that we have a color, send the result back
//each property to alter is a pair, property name/property value

 r:=VarArrayCreate([0,2],varVariant);
 r[0]:='bgColor';    //cell property name
 r[1]:=cellBGColor;    //color
```

```
 result:=r;     //return the array
end;            //end of function

//possible properties and examples
//property names are case sensitive

//background color
 r:=VarArrayCreate([0,2],varVariant);
 r[0]:='bgColor';    //cell property name
 r[1]:=cellBGColor;    //color

//text color
 r:=VarArrayCreate([0,2],varVariant);
 r[0]:='textColor';    //cell property name
 r[1]:=clWhite;    //color

//cell text
 r:=VarArrayCreate([0,2],varVariant);
 r[0]:='cellText';    //cell property name
 r[1]:='STAGE 4';    //text

//changing the cell color and text
 r:=VarArrayCreate([0,4],varVariant);
 r[0]:='textColor';    //cell property name
 r[1]:=clRed;        //color
 r[2]:='cellText';    //cell property name
 r[3]:='RUN FAST';     //text
```

Related:
Function list

OnCalculatorButtonClick

The function is only called via a Calculator graphic element script when the left mouse button is clicked in a button.

Inputs

Variable	**Type**	**Description**
Button number	Integer	The button number, see below
Value	Float	The calculator displayed value

Outputs

Variable	**Type**	**Description**
Result	Boolean	True = halt processing False = continue processing

Button numbers

Key	Value	Key	Value	Key	Value
0	0	Addition	10	Clear entry	20
1	1	Subtraction	11		
2	2	Multiplication	12		
3	3	Division	13	Extra key 1	23
4	4	+/-	14	Extra key 2	24
5	5	Period	15	Extra key 3	25
6	6	Clear	16	Extra key 4	26
7	7	Equal	17	Extra key 5	27
8	8	Percent	18		
9	9	Backspace	19		

Example

```
//in this example extra key 1 is used to square the value.
function OnCalculatorButtonClick(ID: integer; var value: extended):boolean;
begin
 result:=false;                          //continue processing
 if (ID = 23) then                       //extra key 1
  value:=sqr(value);                     //square the value
end;
```

Related:
Function list

OnRecipeClick

The function is only called via a Recipe grid element script when the left mouse button is pressed in a grid cell. The "args" contain data that can be used in the script and are defined in the example.

Below is a sample script.

```
function OnRecipeClick(args):variant;    //args is array of variant
var
 r:variant;
begin
 result:=null;      //default, no result = cancel, set first

//the args contain:
//args[0]           count values, will be 4
//args[1]           current grid cell contents
//args[2]           grid column
//args[3]           grid row
//args[4]           script global hive name

//col and row 0 are set to read only, no need to check

//ask the use for a value
 values:=GetUserInputInteger('Enter value', 'Accept', 'Cancel',0,100);
 if values[0] then
  begin    //the entered value is in range and the user selected OK
   r:=VarArrayCreate([0,2],varVariant);
   r[0]:=true;        //success
   r[1]:=values[1];   //return the value to the grid
   result:=r;
  end;
end;
```

Related:
Function list
OnRecipeValidate

OnRecipeValidate

The function is only called via a Recipe grid element script when the user ends a cell edit. The "args" contain data that can be used in the script and are defined in the example.

Below is a sample script.

```
function OnRecipeValidate(args):variant;   //args is array of variant
var
 r:variant;
 checkOK:boolean;
begin
 result:=null;         //default, no result = cancel, set first

//the args contain:
//args[0]           count values, will be 5
//args[1]           current grid cell contents
//args[2]           grid column
//args[3]           grid row
//args[4]           script global hive name
//args[5]           user entered value

//perform any needed checks on the data and set checkOK state.
//for example checkOK:=(args[5] > -1); //value zero or greater

//col and row 0 are set to ready only, no need to check

 if VarIsNull(args[5]) then    //safety
  Exit;

//if the new value is not a number, scram.
 if not IsNumeric(args[5]) then
  Exit;

 checkOK:=not ((args[5] < 0) or (args[5] > 100));//0 - 100 accepted

 if checkOK then
  begin               //if the data is good, return true.
   r:=VarArrayCreate([0,1],varVariant);
   r[0]:=true;        //success
  end
 else
  begin     //if the data is bad, return false and some options.
   r:=VarArrayCreate([0,4],varVariant);
```

```
   r[0]:=false;              //data entered not valid
   r[1]:='Out of range';     //empty r[1]:=''; do not log event
   r[2]:='Data error';       //empty r[2]:=''; no error message
   r[3]:='Value out of range';//empty r[3]:=''; no error message
                        //for the error message dialog to be displayed
  end;                  //r[2] and r[3] must not be blank(empty)

 result:=r;      //valid or invalid, return the array
end;
```

Related:

Function list
OnRecipeClick

OnTreeviewClick

The function is only called via a Treeview graphic element script when the left mouse button clicks an item in the treeview.

Inputs

Variable	Type	Description
Caption	String	Item caption
TagID	String	User defined value
ItemID	String	Fixed item ID

Outputs

Variable	Type	Description
None		

Example

```
//in this example the tag ID was used to pass the window name to the script.
procedure OnTreeviewClick(caption, tag, fixedID: string);
begin
 if (tag = '') then
  Exit;
 value:=Navigate(['Open', tag]);
end;
```

Related:
Function list
Navigate

OpenAlarmLogWindow

Open the alarm log window.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

OpenAlarmLogWindow;

Related:
Function list
CloseAlarmLogWindow
OpenAlarmLogFilterWindow

OpenAlarmLogFilterWindow

Opens the "Alarm log (filtered)" window.

Inputs

Variable	Type	Description
ApplyFilter	Boolean	Applies the saved filter settings when opened.

Outputs

Variable	Type	Description
None		

Example

```
OpenAlarmLogFilterWindow(false);
```

Related:
Function list
OpenAlarmLogWindow

OpenAlarmWindow

Open the alarm window.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

OpenAlarmWindow;

Related:
Function list
CloseAlarmWindow

OpenBrowserWindow

This function is used to open a browser window. If the logged on user does not have access rights to the window, it will not open and the function will return false. If the URL is blank the URL in the 'Configuration/Browser' dialog is used.

Inputs

Variable	Type	Description
URL	String	Any allowed string

Outputs

Variable	Type	Description
Value	Boolean	True if the window opened False if the window did not open

Warning: The URL can browse the current computer's files, settings, etc.

Example

```
value:=OpenBrowser('http://www.<some web site>/');
if value then
begin
//browser opened
end;

if OpenBrowser('C:\') then
begin
//browser opened
end;
```

Related:
Function list

OpenDriveStatusWindow

Open the drive status window.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

```
OpenDriveStatusWindow;
```

Related:
Function list

OpenEventWindow

Open the event log window.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

OpenEventWindow;

Related:
Function list
CloseEventWindow
LogEvent

OpenPortDiagnosticWindow

This function is used to open a window and display a port's data.

Inputs

Variable	Type	Description
Port name	String	Name of the port to monitor

Outputs

Variable	Type	Description
Value	Boolean	True if the window opened False if the window did not open

Examples

```
value:=OpenPortDiagnosticWindow('MODBUS PORT 1');
if value then
begin
//window opened
end;

if OpenPortDiagnosticWindow ('MODBUS PORT 1') then
begin
//window opened
end;
```

Related:
Function list
ClosePortDiagnosticWindow

OpenScriptMonitorWindow

Opens the scripting monitor window. Opens the same window as the runtime panel "Scripts" button.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

OpenScriptMonitorWindow;

Related:
Function list

OpenTagMonitorWindow

This function is used to open a window and display a tag's data or open the "All points" monitor window.

Inputs

Variable	Type	Description
Tagname	String	Name of the tag An empty tagname will open an "All points monitor" window.

Outputs

Variable	Type	Description
Value	Boolean	True if the window opened False if the window did not open

Examples

```
value:=OpenTagMonitorWindow('GreenInkFlow2B');
if value then
begin
//window opened
end;

if OpenTagMonitorWindow ('GreenInkFlow2B') then
begin
//window opened
end;

OpenTagMonitorWindow ('');          //opens "All points monitor" window
```

Related:
Function list
CloseTagMonitorWindow

OpenWindow

This function is used to open a user created window by name. If the logged on user does not have access rights to the window it will not open and the function will return false.

Inputs

Variable	Type	Description
Window name	String	Name of window to open

Outputs

Variable	Type	Description
Value	Boolean	True if the window opened False if the window did not open

Warning: Do not use/call "OpenWindow/OpenWindowEx" while opening another window. For example: Window "A" is opened via a script or another path. In window "A" an "On window open" calls "OpenWindow('B') an error could occur causing the HMI to freeze. If two windows need to be opened in order. Use one script and call OpenWindow('A'); OpenWindow('B');.

Examples

```
value:=OpenWindow('Pump Station 9');
if value then
begin
//window opened
end;

if OpenWindow('Pump Station 9')then
begin
//window opened
end;
```

Related:
Function list
CloseWindow
IsUserWindowOpen
Navigate
OpenWindowEx
OpenWindowUserSelect

OpenWindowEx

This function is used to open a user created window by name. If the logged on user does not have access rights to the window it will not open and the function will return false. The window is positioned at the supplied coordinates.

Inputs

Variable	Type	Description
Window name	String	Name of window to open
Horizontal position	Integer	Left or horizontal
Vertical position	Integer	Top or vertical
Center	Boolean	Center the window on the horizontal and vertical coordinates
Only once	Boolean	If a window with the same name is open bring it to the front if true. If false create a new window with the same name.

Outputs

Variable	Type	Description
Value	Boolean	True if the window opened False if the window did not open

Warning: The position variables can contain coordinates that are not visible (off screen).

Warning: Do not use/call "OpenWindowEx/OpenWindow" while opening another window. For example: Window "A" is opened via a script or another path. In window "A" an "On window open" calls "OpenWindowEx('B') an error could occur causing the HMI to freeze. If two windows need to be opened in order. Use one script and call OpenWindowEx('A'); OpenWindowEx('B');.

Examples

```
value:=OpenWindowEx('Pump Station 9',10,10,false);
if value then
begin
//window is opened with the top and left corner of the window at 10,10
end;

if OpenWindow('Pump Station 9',500,500,true)then
begin
//window is opened and the center of the window is at 500,500
end;
```

Related:

OpenWindowUserSelect

This function is used to open a user created window the logged on user has access rights to open. A list selection will appear and if the user selects a window and selects the OK button the window will open and the function will return true. Otherwise the function returns false.

Inputs

Variable	Type	Description
Window name	String	Name of window to open

Outputs

Variable	Type	Description
Value	Boolean	True if the window opened False if the window did not open

Examples

```
value:=OpenWindowUserSelect('Open window…');
if value then
begin
//window opened
end;

if OpenWindowUserSelect('Open window…') then
begin
//window opened
end;
```

Related:
Function list
OpenWindow
CloseWindow

PlaySound

This function is used to add a sound to the play list. The sound continues to play until the silence command is issued. The name is case sensitive.

Inputs

Variable	Type	Description
Sound name	String	Name of the sound to play

Outputs

Variable	Type	Description
Value	Boolean	True if the sound was added to the queue False if the was not added to the queue

Examples

```
value:=PlaySound('UserStartedPump');
if value then
begin
//sound added
end;

if PlaySound ('UserStartedPump") then
begin
//sound added
end;
```

Related:
Function list
Mute
PlaySound2
SilenceCommand
SilenceAcknowledgeCommand

PlaySound2

This function is used to add a sound to the play list. The name is case sensitive.

Inputs

Variable	Type	Description
Sound name	String	Name of the sound to play.
Count	Integer	The number of times to play the sound. -1 The sound will play until the "Silence" command is issued. 1 - 500,000 The sound will play X times.
Insert	Boolean	Insert sound after current sound playing.

Note: If "Insert" is true the sound will be inserted in the queue so it is the next sound played. If "Insert" is false the sound will be added to the end of the sound queue. Example: The sound queue contains sounds A B C D E F G. Sound B is playing. PlaySound2 is called with sound name X and insert is true. The sound queue will contain A B X C D E F G. If insert is false the sound queue will contain A B C D E F G X.

Outputs

Variable	Type	Description
Value	Boolean	True if the sound was added to the queue False if the was not added to the queue

Examples

```
value:=PlaySound2('UserStartedPump', -1, false); //play the sound until the silence command
if value then
begin
//sound added
end;

if PlaySound2('UserStartedPump'', 2, true) then //play the sound 2 times
begin
//sound added
end;
```

Related:
Function list
Mute
PlaySound
SilenceCommand
SilenceAcknowledgeCommand

PointExist

This function test if a point exist at runtime via the point tagname. **Note:** A point may exist during configuration and be deleted when runtime starts (still exist during configuration). Check the event log for the reason a point might not exist at runtime.

Inputs

Variable	Type	Description
Tagname	String	The tagname of the point to check

Outputs

Variable	Type	Description
Exist	Boolean	True if the point exist at runtime False if the point does not exist at runtime

Example

```
value:=PointExist('Pump1PressureLow'); //tagname of point to test for existence
if value then
begin
 //point exist
end;
```

Related:
Function list

PrintScreen

Performs a "Print Screen".

PrintScreenActiveWindow prints only the active window, not the full monitor(s) screen.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

PrintScreen;

The same action as pressing the "Print Screen" key.

PrintScreenActiveWindow;

The same action as pressing the "Print Screen" key with the 'ALT' key held down.

Related:
Function list
CaptureScreen
CaptureWindow

PTZ (pan-tilt-zoom)

This function is used to change or read a camera PTZ. Not all cameras support all actions.

Inputs

Variable	**Type**	**Description**
Port name	String	Port name
Command	String	PTZ command
Values	Array of variant	Values defined by command

Outputs

Variable	**Type**	**Description**
Values	Array of variant	Values defined by command

Commands

Absolute	ContinuousMove	Fetch	GetDateTime
Home	Preset	Relative	Status
Stop	Tour		

Command	Function
Absolute	PTZ to an "absolute" position.

The command supports two movements and two optional speed variables.
The command must have at least one sub command.
The order of sub commands defines the value order.

Sub command	**Description**
PT	Pan/Tilt
Z	Zoom
OA	Pan/Tilt speed (optional)
OB	Zoom speed (optional)

```
values:=PTZ('ParkingLotA', 'Absolute', ['PT', -0.5, 0.2]);          //pan/tilt, no zoom
values:=PTZ('ParkingLotA', 'Absolute', ['PTOA', 0.5, -0.2, 0.4]);   //pan/tilt with speed

values:=PTZ('ParkingLotA', 'Absolute', ['Z', 0.8]);                 //zoom, no pan/tilt
values:=PTZ('ParkingLotA', 'Absolute', ['ZOB', -0.8, 0.2]);   //zoom with speed, no pan/tilt

values:=PTZ('ParkingLotA', 'Absolute', ['PTZ', 0.5, 0.2, -0.03]);   //pan/tilt zoom, no speed

//pan/tilt and zoom, pan/tilt speed, no zoom speed
values:=PTZ('ParkingLotA', 'Absolute', ['PTZOA', 0.5, 0.2, 0.03, 0.4]);
```

```
//pan/tilt and zoom, pan/tilt speed, no zoom speed
values:=PTZ('ParkingLotA', 'Absolute', ['PTZOAOB', 0.5, 0.2, 0.03, 0.4, 0.07]);

values[0] = error code
values[1] = number of values (will be zero)

if (values[0] = 0) then
 ;                          //command issued
```

Back to commands

Command	Function
ContinuousMove	PTZ move continuously

The command supports two movements, a timeout and optional zoom and zoom speed variable.
The command must have at least one command.
The order of sub commands defines the value order.

Sub command	**Description**
PT	Pan/Tilt/Timeout
Z	Zoom (optional)
OB	Zoom speed (optional)

Notes:

1) Pan and tilt values are the velocity and can be negative for reverse direction.
2) The command must have at least the pan, tilt and timeout as the first three values, 0 is allow for each one. All zeros will act like a "Stop" command.
3) Timeout seconds is a floating point value. e.g. 1.0 = 1 second, 0.5 = ½ second.

```
values:=PTZ('Lot2', 'ContinuousMove', ['PT', -0.5, 0.5, 0.5]);        //pan/tilt, ½ second timeout
values:=PTZ('Lot2', 'ContinuousMove', ['PTZ', 1, 1, 10.0, 0.75]);     //pan/tilt,10 second timeout
                                                                      zoom
values:=PTZ('Lot2', 'ContinuousMove', ['PTZ', 1, 0.38, 10.0, 0.75]);  //pan/tilt,10 second
                                                                      timeout zoom
values:=PTZ('Lot2', 'ContinuousMove', ['PTZOB', 0.22, 1, 3.0, 0.5, 1.0]);//pan/tilt, 3 second
                                                                      timeout, zoom, zoom speed

values[0] = error code
values[1] = number of values (will be zero)

if (values[0] = 0) then
 ;                          //command issued
```

Back to commands

Command	Function
Relative	PTZ to a "relative" position.

The command supports two movements and two optional speed variables.
The command must have at least one command.
The order of sub commands defines the value order.

Sub command	**Description**
PT	Pan/Tilt
Z	Zoom
OA	Pan/Tilt speed (optional)
OB	Zoom speed (optional)

```
values:=PTZ('ParkingLotA', 'Relative', ['PT', -0.5, 0.2]);          //pan/tilt, no zoom
values:=PTZ('ParkingLotA', 'Relative', ['PTOA', 0.5, -0.2, 0.4]);   //pan/tilt with speed

values:=PTZ('ParkingLotA', 'Relative', ['Z', 0.8]);                 //zoom, no pan/tilt
values:=PTZ('ParkingLotA', 'Relative', ['ZOB', -0.8, 0.2]);   //zoom with speed, no pan/tilt

values:=PTZ('ParkingLotA', 'Relative', ['PTZ', 0.5, 0.2, -0.03]);   //pan/tilt zoom, no speed

//pan/tilt and zoom, pan/tilt speed, no zoom speed
values:=PTZ('ParkingLotA', 'Relative', ['PTZOA', 0.5, 0.2, 0.03, 0.4]);

//pan/tilt and zoom, pan/tilt speed, no zoom speed
values:=PTZ('ParkingLotA', 'Relative', ['PTZOAOB', 0.5, 0.2, 0.03, 0.4, 0.07]);

values[0] = error code
values[1] = number values (will be zero)

if (values[0] = 0) then
 ;                                  //command issued
```

Note: To only pan or tilt use zero (0) for the axis to be unaffected.

Back to commands

Command	Function
Home	PTZ to a "home position Speed (optional)

```
values:=PTZ('ParkingLotA', 'Home', []);            //move to home position
values:=PTZ('ParkingLotA', 'Home', [1.0]);         //move to home position, with speed
```

values[0] = error code
values[1] = number of values (will be zero)

```
if (values[0] = 0) then
 ;                                  //command issued
```

Back to commands

Command	Function
Stop	Stop PTZ movement Pan/tilt, zoom, each can be true or false

```
values:=PTZ('ParkingLotA', 'Stop', [true, true]);      //stop all (pan/tilt, zoom)
```

values[0] = error code
values[1] = number of values (will be zero)

```
if (values[0] = 0) then
 ;                                  //command issued
```

Back to commands

Command	Function
GetDateTime	Fetch date time from camera

Note: This command is called when runtime starts and "generally" is not required.

```
values:=PTZ('ParkingLotA', 'GetDateTime', []);
```

values[0] = error code
values[1] = number of values (will be zero)

```
if (values[0] = 0) then
 ;                                  //command issued
```

Back to commands

Command	Function
Fetch	Poll the device for PTZ status

Note: Use "Status" command to access returned values.

values:=PTZ('ParkingLotA', 'Fetch', []);

Sub command	Description
S	Script to call

The name of a script to queue (command script engine) when the PTZ data is returned from the camera.

values:=PTZ('ParkingLotA', 'Fetch', ['S', 'FetchReturned']);

values[0] = error code
values[1] = number of values (will be zero)

```
if (values[0] = 0) then
 ;                                    //command issued
```

Back to commands

Command	Function
Status	Status value of device

Note: Values are result of last fetch command.

values:=PTZ('ParkingLotA', 'Status', []);

```
if (values[0] = 0) then
 ;                                    //command success
```

values[0] = error code
values[1] = number of values (will be 9)
values[2] = pan position
values[3] = tilt position
values[4] = zoom position
values[5] = pan/tilt move status (string)
values[6] = zoom status (string)
values[7] = UTC time of last successful "Fetch", string is cleared when "Fetch" request is transmitted
values[8] = last error (string)
values[9] = send queue depth
values[10] = clock offset (in seconds)

Back to commands

Command	Function
Preset	PTZ to preset position/zoom

Note: Not all cameras support the speed parameter. If using the speed parameter and the camera does not move, remove the speed parameter and try again.

Sub command	Description
P	Preset name
PS	Preset name and speed

```
values:=PTZ('ParkingLotA', 'Preset', ['P',2]);          //preset #2
values:=PTZ('ParkingLotA', 'Preset', ['PS', 2, 1.5]);   //preset #2, speed
```

values[0] = error code
values[1] = number of values (will be zero)

```
if (values[0] = 0) then
 ;                                  //command issued
```

Back to commands

Error codes

The first value of the output array (where applicable) is an error code. If the value is not equal to zero, an error occurred.

Code	Description
0	No error
-1	Port name missing or invalid
-2	Command missing
-3	Unknown command
-4	PTZ not active for port
-5	Command added to send queue
-6	Parameter count invalid
-7	Absolute/relative parameter invalid
-8	Out of memory
-9	Pan value out of range
-10	Tilt value out of range
-11	Zoom value out of range
-12	Fetch script name missing or script not found
-13	Time value conversion failed

Related:
Function list

PortPreReadEvent

This function is called from a port driver before a read is issued. See here for a list of port types that support this function.

Inputs

Variable	Type	Description
Port name	String	Port name
Index	Integer	Configured read index
Primary	Boolean	True if primary port driver

Outputs

Variable	Type	Description
Value	Integer	0 = Process read 1 = Skip read

Example

```
function PortPreReadEvent(portName, readIndex, isPrimary):boolean;
begin
 //return 0 to issue read, return 1 to skip read
 result:=0;
end;
```

Notes:

1) The script should be very short and fast. Otherwise overall processing may be degraded.
2) If the "read" issue is skipped (result = 1) and the read delay timer is greater than 0, the read delay timer will be enabled and the next read will be issued when the timer completes. If the read delay timer is zero (0) the next configured read will be processed. This function will be called again with the next configured read index.
3) When the port type is MODBUS Master TCP/IP (Single Socket) the port name is the "slave" port name, not the "master" port name.

Related:
PortSetParameter
Function list

PortSetParameter

This function is used set port parameters.

Communication port types supporting this function

Enron TCP/IP	Enron RTU over TCP/IP
MODBUS Master TCP/IP	
MODBUS Master RTU over TCP/IP	MODBUS Master TCP/IP (Single Socket)
Siemens S7-200 TCP	Siemens S7-300/400/1200 (TCP)
MODBUS master serial ports *2	

Notes:

1) "WriteListCount" command is supported by all communication ports.
2) MODBUS master serial ports only.

Inputs

Variable	Type	Description
Port name	String	Port name
Command	String	Command
Value	Array of variant	Parameters

Outputs

Variable	Type	Description
Value	Boolean	True if command executed

Command	Function
ReadDelayTime	Sets the read delay timer Parameter 1, delay time in milliseconds Parameter 2, force a read request

```
value:=PortSetParameter('Port_1', 'ReadDelayTime', [1000, true]);
value:=PortSetParameter('Port_1', 'ReadDelayTime', [0, false]);
```

Note: The "next read index" is set to the beginning of the port configured reads list.

DisconnectThreshold	Sets the threshold to close connection between read/write request Parameter 1, threshold (0 or greater than 1000)

```
value:=PortSetParameter('Port_1', 'DisconnectThreshold', [1000]);
value:=PortSetParameter('Port_1', 'DisconnectThreshold', [60000]);
```

Note: Some external devices will drop or close the connection if the HMI does not send read/write request after X amount of time. This parameter defines the value compared against the read delay time value, to close the connection. For example. The read delay time is set to 300000 (5 minutes). Setting this value, to any value > 1000 and less than 300000, will configure the port to close the connection after each successful read/write command and reopen for the next read/write command.

Command	Function
PortPreReadEvent	Sets the script to call before issuing a read Parameter 1, script name

value:=PortSetParameter('Port_1', 'PortPreReadEvent', ['Port_1_Read']);

Command	Function
WriteListCount	Sets the maximum writes the write list queue can contain before new writes are rejected for exceeding the limit. Parameter 1, maximum list size Range 1 - 8192

value:=PortSetParameter('Port_1', 'WriteListCount', [1024]); //default is 1024

Note: An event is placed in the event log when an attempt to add a write to the port write list fails because the list limit is exceeded. If that error is occurring, the most likely cause is a scripting error. Contact support if assistance is required.

Command	Function
ResetConnection	Closes and reopens OS connection. Parameter 1, 1 = primary, 2 = secondary Parameter 2, future

value:=PortSetParameter('Port_1', 'ResetConnection', [1,0]); //primary
value:=PortSetParameter('Port_1', 'ResetConnection', [2,0]); //secondary

Note: The MODBUS master serial ports are the only ports supporting this call. If this call is needed for other serial port types, contact support.

Related:
PortPreReadEvent
Function list

QueryPerformanceCounter

This function returns the value of the performance counter, a high resolution timer stamp that can be used for time-interval measurements.

Inputs

Variable	Type	Description
Milliseconds	Boolean	True = result is milliseconds False = result is microseconds

None

Outputs

None

Example

value:=QueryPerformanceCounter;

Related:
Function list

QuitRuntime

This function will stop runtime monitoring and quit the runtime program.

Inputs

Variable	Type	Description
Configuration	Boolean	True = launch configuration False = do not launch configuration

Outputs

None

Example

QuitRuntime(False);

Note:
If the software license is "Runtime" only, the configuration variable is ignored.

Related:
Function list

ReadValue

This function is used to read a value or values from the runtime database. The RV function might be more suitable if reading the default point items. The RV function is faster.

Inputs

Variable	Type	Description
Tagname	String	Name of the point
Item number	Integer	Data item ID

Outputs

Variable	Type	Description
Values	Array of variant	Values of tagname/item id

The values can all be of the same type (i.e. Boolean, integer) or they can be of any type as defined by the item number. Each read consists of a tagname and item number pair. If either one is missing all the values returned will be invalid.

The items numbers most used would be:

5000 Process Variable Analog (float)	5007 Process Variable Digital (digital)
5008 Percent of Full Scale (float)	5009 Analog Host Pointer Index (float)

Examples

```
values:=ReadValue(['tagname 1', 5007]);
if values[0] then
;
```

If more than one value needs to be acquired it is faster to combined all reads into one function call. It will be faster than executing multiple read function calls.

```
values:=ReadValue(['tagname 1', 5007, 'tagname 2', 5007]);
if values[0] and values[1] then
;

values:=ReadValue(['tagname 1', 5007, 'tagname 2', 5000]);
if values[0] and (values[1] > 50) then
;
```

Related:
Function list
RV
WriteValue
WriteValuePulse

RecipeDuplicate

This function is used to duplicate a row or column in a recipe; to create a new recipe. Using this function to add an ingredient to a recipe will add the column/row but, the new ingredient will not have a point configured to write ingredient the value.

Inputs

Variable	Type	Description
Recipe name	String	Name of a recipe
Axis	String	Column or Row
Source	String	The source column or row
Destination	String	The destination column or row XLS: A blank will create a new column or row after the last column/row. ODBC: If the destination name is not found a new record or field will be created.
Save	Boolean	XLS: If true, the sheet will be saved to disk. ODBC: N/A but, must be included in script function call. ODBC saves all changes.
Options	Array of string	See below

Outputs

Variable	Type	Description
Value	Boolean	The function executed

Options

When the recipe type is ODBC and the command is to duplicate a field (column) the HMI uses the SQL "ALTER TABLE" command. This command needs the new field data type. A text type field is best. For example, MS Access: 'Text(60)', 'LongText'

Examples

XLS:

```
value:=RecipeDuplicate('Cookies', 'Row', '5', '10',true, []);  //Row 5 to row 10, save
value:=RecipeDuplicate('Cookies', 'Row', '6', '', false, []);   //Row 6, add row at end, do not save
value:=RecipeDuplicate('Cookies', 'Column', 'C', 'F', true, []);  //Column C to Column F, save
```

ODBC:

```
value:=RecipeDuplicate('Cookies', 'Row', 'Bigbatch', 'Bigbatch2',true, []);
                                        //copy Bigbatch to Bigbatch2, N/A
value:=RecipeDuplicate('Cookies', 'Column', 'Milk', 'AlmondMilk',true, ['TEXT(32)']);
                                        //field Milk to field AlmondMilk, N/A
```

Notes:

1) During testing with MS Access, if the table was open in MS Access and the function was executed to add a field, the function failed with MS Access reporting the table could not be locked.
2) During testing with MS Access the table was not updated when a new record was created. "Refresh All" or F5 had to be selected.
3) During testing with MS Access setting the "Options/Client settings/Refresh interval" lower than the default would update the table for duplicates faster. See #2 for new record updates.
4) For ODBC, attempting to duplicate a record or field with an existing name will fail.

Related:

Function list
RecipeGetCell
RecipeSetCell

RecipeGetCell

This function is used to read the contents of a recipe cell.

Inputs

Variable	Type	Description
Recipe name	String	Name of a recipe.
Column	String	The column/field name of the spreadsheet/table. XLS: The field can be a number or letter. 1 = A, 2 = B, etc. or A, B, C, D, etc. ODBC: The field must be a string that is the name of the column.
Row	Integer	The row/record of the spreadsheet/table. The row begins at 1 (one) and increases.

Outputs

Variable	Type	Description
Value	String	The value of the cell.

Examples

XLS:

```
value:=RecipeGetCell('Butter Cookies', 'B', 3);          //column B, row 3
value:=RecipeGetCell ('Fuel mixture', 47, 55);           //column AU, row 55
```

ODBC:

```
value:=RecipeGetCell('Butter Cookies', 'LBS', 3);        //column LBS, record 3
value:=RecipeGetCell ('Fuel mixture', 'Oil', 55);        //column Oil, record 55
```

Related:
Function list
RecipeSetCell

RecipeReloadSheet

This function is used to reload a sheet/table into HMI memory. The sheets/tables are loaded when runtime monitoring is started. If changes are made to the sheet/table data (externally), monitoring must be stopped and restarted **or** this function called for the recipe to reload the disk/table data to memory.

Note: If changes are made to the sheet in Excel the data needs to be reloaded; be sure to save the sheet in Excel before calling this function. The data is loaded from the disk file.

Inputs

Variable	Type	Description
Recipe name	String	Name of a recipe

Outputs

Variable	Type	Description
None		

Examples

```
RecipeReloadSheet('Butter Cookies');
RecipeReloadSheet('Fuel mixture');
```

Related:
Function list
LoadRecipe
LoadRecipe2

RecipeSaveSheet

This function is used to save the "in memory" recipe data to disk. This function is used only for XLS recipes. The sheets are loaded when runtime monitoring is started. If changes are made to the internal data this command will save the sheet to disk.

Notes:
1) If changes are made to the sheet in Excel and this function is called, the changes will be lost.
2) ODBC connections write to the database when the RecipeSetCell or RecipeDuplicate function is executed.

Warning: This function overwrites the existing file for XLS and the database table field/record for ODBC.

Inputs

Variable	Type	Description
Recipe name	String	Name of a recipe

Outputs

Variable	Type	Description
None		

Examples

RecipeSaveSheet ('Butter Cookies');
RecipeSaveSheet ('Fuel mixture');

Related:
Function list
RecipeReloadSheet

RecipeSetCell

This function is used to set the contents of a recipe cell.

Inputs

Variable	Type	Description
Recipe name	String	Name of a recipe.
Column	String	The column/field name of the spreadsheet/table. XLS: The field can be a number or letter. 1 = A, 2 = B, etc. or A, B, C, D, etc. ODBC: The field must be a string that is the name of the column.
Row	Integer	The row/record of the spreadsheet/table. The row begins at 1 (one) and increases.
Value	String	The value to place in the cell.

Outputs

Variable	Type	Description
Value	Boolean	True = The cell was set. False = The cell was not set.

Examples

XLS:
value:=RecipeSetCell('Butter Cookies', 2 , 3,1.5); //column B, row 3, value 1.5
value:=RecipeSetCell ('Fuel mixture', 'AC', 55,88); //column AC, row 55, value 88

ODBC:
value:=RecipeSetCell('Butter Cookies', ' LBS' , 3,1.5); //column LBS, record 3, value 1.5
value:=RecipeSetCell ('Fuel mixture', 'Oil', 55,88); //column Oil, record 55, value 88

Note:
The column and row must exist in the loaded recipe. This function will not add a column or row. See RecipeDuplicate to add a record/row or column/field.

Related:
Function list
RecipeGetCell

ReportSetCell

This function is used to set the value of a cell in a report when the report is generated at runtime. This function is only used in a script that is called from the report generator. If this function is called outside a report generation it is ignored.

This function completely replaces the text in the cell. The text format and color are unchanged.

Inputs

Variable	Type	Description
Column	String	Column index (A, B, C, etc.)
Row	Integer	Row index (1, 2, 3, etc.)
Value	String	New cell value

Outputs

Variable	Type	Description
None		

Examples

```
ReportSetCell('A', 4, '123.456');
ReportSetCell('D', 10, 'Limit reached');

ReportSetCell('D', 10, FloatToStr(values[0])); //set a cell to a floating point number
ReportSetCell('D', 11, IntToStr(values[1])); //set a cell to an integer
```

Related:
Function list
ReportSetCellColor

ReportSetCellColor

This function is used to set the color of a cell in a report when the report is generated at runtime. This function is only used in a script that is called from the report generator. If this function is called outside a report generation it is ignored.

Inputs

Variable	Type	Description
Column	String	Column index (A,B,C, etc.)
Row	Integer	Row index (1,2,3, etc.)
Text or background	Boolean	False = set the text color True = set background color
Color	Integer	New color

Outputs

Variable	Type	Description
None		

Examples

ReportSetCellColor('A', 4, False, 255); //sets the text color
ReportSetCellColor('D', 10, True, 15678); //sets the background color

Related:
Function list
ReportSetCell

ReportSetFileName

This function is used to set the report file name. This function is only used in a script that is called from the report generator. If this function is called outside a report generation it is ignored.

Inputs

Variable	Type	Description
Filename	String	The new file name. Do not include the file extension.

Outputs

Variable	Type	Description
None		

Note:
If the file name contains a path, the alternate path will also be set.

Examples

```
ReportSetFileName('NewFileName');
ReportSetFileName('C:\Reports\Generated\NewFileName');
```

Related:
Function list

RestartShutdownComputer

This function is used to turn off or restart the computer.

Inputs

Variable	Type	Description
Command	String	Shutdown or Restart

Outputs

Variable	Type	Description
Result	Boolean	True if the command was accepted by the OS False if the command was not accepted. If false an entry will be added to the event log.

Example

This is in a button script.

```
procedure OnMouseUp;
begin                          //ask the user
 value:=GetUserInputBoolean2('Shutdown the computer...', 'Yes', 'No');
 if (value = 1) then
  value:=RestartShutdownComputer('Shutdown');
end;
```

This script is in a linked script and does not ask the user.

```
value:=RestartShutdownComputer('Shutdown');
```

To restart:

```
value:=RestartShutdownComputer('Restart');
```

Related:
Function list

Result / Return

This function is used in point scripts to return a value to the process variable of the point containing the script. It is ignored in other scripts. **Note:** For "Basic" scripts use "Return".

Inputs

None

Outputs

None

Warning

When a point script is executed a result can be returned to the process variable of the point using two methods. Use only one method. Using both methods waste processing time and does not produce a different result.

The tagname of the point containing the script is "tag3".

The following examples read the process variable of two inputs, sum the values and returns the result.

Example 1

```
values:=ReadValue(['tag1', 5000, 'tag2', 5000]);
result:=values[0] + values[1];
```

Basic example

```
values=ReadValue(["tag1", 5000, "tag2", 5000])
return values[0] + values[1]
```

Example 2

```
values:=ReadValue(['tag1', 5000, 'tag2', 5000]);
WriteValue(['tag 3', 5000, values[0] + values[1]]);
```

Example 1 executes **faster** than example 2.

Related:
Function list
ReadValue
WriteValue

RV

This function is used to read a point value or values from the runtime database using the default point item number. Use ReadValue to read any point item value.

Default item numbers

Type	Item number	Description
Analog	5000	Process variable analog (float)
Digital	5007	Process variable digital (digital)

Inputs

Variable	Type	Description
Tagname	String(s)	Name of the point

Outputs

Variable	Type	Description
Values	Array of variant	Values of tagname/item id

The values can all be of the same type (i.e. Boolean, integer) or can be of any type as defined by the point type default item number.

Examples

```
values:=RV(['tagname 1']);
if values[0] then
;
```

If more than one value is to be read, it is faster to combined all reads into one function call. It will be faster than executing multiple read function calls.

```
values:=RV(['tagname 1', 'tagname 2']);
if values[0] and values[1] then
;
```

```
values:=RV(['tagname 1', 'tagname 2']);
if values[0] and (values[1] > 50) then
;
```

Related:
Function list
ReadValue
WriteValue
WriteValuePulse
WV

Scale

This function scales a value from an input range to an output range.

Inputs

Variable	Type	Description
InputValue	Float	Value to scale
InputRangeLow	Float	Input range low
InputRangeHigh	Float	Input range high
OutputRangeLow	Float	Output range low
OutputRangeHigh	Float	Output range high

Outputs

Variable	Type	Description
Value	Float	The scaled value

Note:
If the input range is zero the output value will be zero.

Example

```
value:=Scale(50, 0,100,0,500);      //the result is 250
value:=Scale(50, 0,100,0,1);        //the result is 0.5
```

Related:
Function list

SchedulerEdit

This function display the scheduler dialog.

Inputs

Variable	Type	Description
AlterRuntimeSchedule	Boolean	True = implement schedule changes
SaveToDisk	Boolean	True = save all changes to disk

Outputs

None

Note:
If "AlterRuntimeSchedule" is true, when the edit window is closed, all schedules will be reloaded/restarted. Any schedule with the "Runtime start action enabled" will be processed.

Example

```
SchedulerEdit(True, True);          //alter running schedules and save changes to disk
```

Related:
Function list
SchedulerSetParameter
SchedulerOpenMonitorWindow
SchedulerSetState

SchedulerOpenMonitorWindow

This function opens the scheduler monitor window.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

SchedulerOpenMonitorWindow;

Related:
Function list
SchedulerEdit
SchedulerSetState
SchedulerSetParameter

SchedulerSetParameter

This function sets the value of a schedule parameter.

Inputs

Variable	Type	Description
Schedule name	String	The name of the schedule.
Parameter name	String	The name of the parameter.
Value	Variant	The new value for the parameter. The value must be a number or true/false.

Outputs

Variable	Type	Description
Result	Integer	0 (zero) True if success Negative number for failure

Example

result:=SchedulerSetParameter('Everyday','Room temp',72);

Related:
Function list
SchedulerEdit
SchedulerOpenMonitorWindow
SchedulerSetState

SchedulerSetState

This function sets the enabled state of a schedule.

Inputs

Variable	Type	Description
Schedule name	String	The name of the schedule.
Value	Boolean	True sets the schedule enabled False sets the schedule disabled.

Outputs

Variable	Type	Description
Result	Integer	0 (zero) True if success Negative number for failure

Example

result:=SchedulerSetState('Everyday',False);

Related:
Function list
SchedulerEdit
SchedulerOpenMonitorWindow
SchedulerSetParameter

ScreenSaverSuspend

This function commands the OS to suspend the screen saver.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Notes:

1) This command suspends the screen saver if the OS initiated the screen save mode. This command will not suspend the screen saver if the screen saver program, the <screen saver name>.scr program is launched by the user.
2) If the screen saver is configured for "On resume, display logon screen" the screen saver will end and the user must logon.

Example

ScreenSaverSuspend;

Related:
Function list

SendKeys

This function is used to send keystrokes to the system.

Inputs

Variable	Type	Description
Modifier keys	String	This variable can be empty or contain 1-3 of the modifier keys.
Keys	String	The keys to send to the system. This variable must contain at least one character.

Outputs

Variable	Type	Description
None		

Modifier keys
Shift +
CTRL ^
ALT %

Examples

```
SendKeys('%','P');          //Send ALT P to the system
SendKeys('','P');           //Send P to the system
SendKeys('^%','P');         //Send CTRL + ALT P to the system
SendKeys('','Test');        //Sends 'Test' to the system
SendKeys('',chr(173));      // '' is two single quotes, toggles the mute
                               command to the OS
SendKeys('',chr(120));      //modifier, key code, F9 $78 = 120

Basic
SendKeys("","x")            'modifier, key code, F9 = $78 = 120 = x
```

Related:
Function list

SetAlarmBlocks

This function is used to block/unblock the HMI from executing the logic for a configured alarm.

Alarms are enabled/disabled, at runtime start, as determined by project configuration.
Alarm logic can be blocked/unblocked at runtime to temporarily disable and then enable the alarm logic processing for a point.
Alarm blocks are designed to be a temporary solution, when for example, a transmitter fails and the configured alarm needs to be 'blocked' so as not to provide false information.
The alarm configuration should be altered if the 'block' is used as a permanent solution.

Inputs

Variable	Type	Description
Tagname	String	Tagname
Blocks	Boolean	Block/Unblock
Check	Boolean	True = process alarm check False = do not process alarm check False should only be used at runtime start, on runtime start script, to alter the alarm block before processing. If the change is permanent, change the configuration.
ForceOff	Boolean	True = If the alarm is active it will be cleared False = No change

Notes:
1) ForceOff would be used, for example, when the transmitter fails and generates an alarm, the only method to clear the alarm is to restore the transmitter signal to "in range" and "Acknowledge" the alarm and then the alarm can return to normal. When the block command is true, the ForceOff attribute is true and the alarm is active it will be cleared.
2) Digital alarms are detected by a change of state. For example, if the input is true, the block is removed from the rising alarm and the alarm is checked it will not generate an alarm because the check logic does not detect a change of state.
3) An entry is created in the event log for each call to SetAlarmBlocks.

Outputs

Variable	Type	Description
None		

The tagname variable is an array of tagnames. The array must contain one of more tagnames. The points referenced by the tagnames must all be of the same base type, analog or digital, per command.

The enables variable is an array of Booleans. It must contain two or four Boolean values.

Analog points

SetAlarmBlocks(['tagname 1'],[True,True,True,True] ,True,True); // lolo, lo hi, hihi

To set the alarm blocks for more than one point, all to the same state:

SetAlarmBlocks(['tagname 1','tagname 2','tagname 3','tagname 4','tagname 5'], [False,True,True,False] ,True,True); // lolo false , lo true, hi true, hihi false

Digital points

SetAlarmBlocks (['tagname A'],[True,True] ,True,True); //falling, rising

To set the alarm blocks for more than one point, all to the same state:

SetAlarmBlocks (['tagname A','tagname B','tagname C','tagname D','tagname E'], [False,True] ,True,True); //falling false , rising true

Related:
Function list
SetAlarmDelays
SetAlarmSetpoints

SetAlarmDelays

This function is used to set the time delay of a point alarm.

Inputs

Variable	Type	Description
Tagnames	String	Tagname
Times	Integer	Alarm delay value

Outputs

Variable	Type	Description
Value	Boolean	True = An error occurred (could be partial success) False = Success

The "Tagnames" variable is an array of tagnames. The array must contain one or more tagnames. All tagnames must be of the same type.

Notes:
1) The "Times" array must contain two (2) or four (4) elements. The element order is Falling, Rising or Lo Lo, Lo, Hi, and Hi Hi.
2) To not alter a delay time set the value to minus 1 (-1).
3) Calling this function while a point alarm is timing does not alter, the current count. Changing the delay value only alters the start value. This call does not alter a running count.
4) Changes are not saved and are lost when runtime ends.

Examples

```
value:=SetAlarmDelays(['PumpTemperature'], [-1,10]);
//only alter rising alarm delay

value:= SetAlarmDelays(['PumpPressueHigh'], [5, 5, 5, 5]);
//change all delays, Lo Lo, Lo, Hi, and Hi Hi
```

Related:
Function list
SetAlarmBlocks
SetAlarmSetpoints

SetAlarmEnables

Deprecated This command will be removed in a future release. Do not use for new projects. Use SetAlarmBlocks for blocking alarm processing for a point alarm.

This function is used to enable or disable the alarm configuration for points. At runtime start the point alarm must have been enabled.

Inputs

Variable	Type	Description
Tagname	String	Tagname
Enables	Boolean	Enable/Disable

Outputs

Variable	Type	Description
None		

The tagname variable is an array of tagnames. The array must contain one of more tagnames. The points referenced by the tagnames must all be of the same base type, analog or digital, per command.

The enables variable is an array of Booleans. It must contain two or four Boolean values.

Analog points

SetAlarmEnables(['tagname 1'],[True,True,True,True]); // lolo, lo hi, hihi

To set the alarm enables for more than one point, all to the same state:

SetAlarmEnables(['tagname 1','tagname 2','tagname 3','tagname 4','tagname 5'], [False,True,True,False]); // lolo false , lo true, hi true, hihi false

Digital points

SetAlarmEnables(['tagname A'],[True,True]); //falling, rising

To set the alarm enables for more than one point, all to the same state:

SetAlarmEnables(['tagname A','tagname B','tagname C','tagname D','tagname E'], [False,True]); //falling false , rising true

Related:
Function list

SetAlarmSetpoints

This function is used to set the setpoint of a point alarm.

Inputs

Variable	Type	Description
Tagnames	String	Tagname
Type	Integer	0 = value 1 = tagname
Setpoints	Variant Array	Setpoint value 0 = value e.g. 1.6, 0.3, 99, -56.1 1 = tagname e.g. PressureSP, FlowSP
Check	Boolean	True = process alarm check False = do not process alarm check False should only be used at runtime start, on runtime start script, to alter the alarm setpoint before processing. If the change is permanent, change the configuration.

Outputs

Variable	Type	Description
Value	Integer	If the value is zero (0) the function executed without error. If the value is less than zero (0) the function failed. See error codes below.

The "Tagnames" variable is an array of tagnames. The array must contain one or more tagnames. All tagnames must be of the analog type.
The "Type" variable defines what type the "Setpoints" variable contains.
If the type is zero (0) the setpoints are numerical values.
If the type is one (1) the setpoints are tagnames. The HMI will use the process variable analog (item 5000) of the specified tagname as the setpoint.

Notes:
1) The "Setpoints" array must contain four (4) elements. The element order is Lo Lo, Lo, Hi, and Hi Hi.
2) To not alter a setpoint leave the element blank, put two single quotes in place of the value or tagname.

Examples

```
value:=SetAlarmSetpoints(['PumpPressue1'], 0, [20, 40, 80, 95], True);
//Lo Lo, Lo, Hi, and Hi Hi

value:=SetAlarmSetpoints(['PumpPressue1'], 0, ['', '', 65, 88], True);
//do not change Lo Lo or Lo, set Hi, and Hi Hi
```

value:=SetAlarmSetpoints(['PumpPressue1'], 1, ['PressLoLo', 'PressLo', '', ''] , True);
//Lo Lo and Lo, collect value from tagname PressLoLo and PressLo, do not change Hi and Hi Hi

Notes:

1) When a setpoint field is empty, put two single quotes.
2) When a setpoint field is a number, do not use quotes around the value.
3) When a setpoint field is a tagname, surround the tagname with single quotes.
4) The setpoint field types must all match. Mixing of values and tagnames is not permitted, the function will fail.
5) The setpoint field must contain at least one valid element, value or tagname.
6) The setpoint filed must contain 4 elements.
7) If any tagname is invalid, the function will fail.

Error codes:

-1 The tagname field contained an unrecognized tagname.
-2 The tagname field is not an analog or analog host type point.
-3 The tagname field was empty.

-10 Type field not zero (0) or one (1).

-20 Setpoints field does not contain 4 elements.
-21 Setpoints field contains at least one unrecognized tagname.
-22 Setpoints field contains at least one point that is not analog. (Only if Type = 1)
-23 Setpoints field contains at least one value that is not a string. (Only if Type = 1)

-30 A setpoint value conversion error.

Related:
Function list
SetAlarmBlocks
SetAlarmDelays

SetCell

This command is used to set the cell contents of a "Digital grid", "Analog grid" or "Dynamic grid", from the graphic script of the graphic element.

Inputs

Variable	Type	Description
Column	Integer	Grid column, 0 or 1.
Row	Integer	Grid row
Property ID	Integer	Property of the cell
Value	Variant	Cell value

Outputs

Variable	Type	Description
None		

Property ID

Value	Digital grid	Analog grid	Dynamic grid
1	State (any value <> 0 will be true)	String value	
2	Text value	Text value	Cell text
3			Background color
4			Font name
5			Font style *1
6			Font color
7			Font size
8			Horizontal alignment *2
9			Vertical alignment *3

Notes

1) If the string is empty the font style is not applied. The string must contain all desired font styles for the cell. Each style is separated by a comma ','.
   Example: bold 'fsBold'
   Example: bold and underline 'fsBold,fsUnderline'
2) Horizontal values are Left, Center or Right.
3) Vertical values are Top, Center or Bottom.

Examples

```
ge.SetCell(1,1, 1, 0);                    //column, row, property ID, value
ge.SetCell(0,1, 2, 'Some text');          //column, row, property ID, value
ge.SetCell(0,1, 5, 'fsBold,fsItalic');    //column, row, property ID, value
ge.SetCell(0,1, 8, 'Left');               //column, row, property ID, value
ge.SetCell(0,1, 9, 'Center');             //column, row, property ID, value
```

Related:
Function list

SetMAStationConfigurationName

This command is used to set an MA station configuration to a configuration via script control. Call this command for each MA station in a window before the window is opened. (If a change is desired.)

Inputs

Variable	Type	Description
Station name	String	Name of an MA station
Configuration name	String	Name of an MA station configuration

Outputs

Variable	Type	Description
None		

Example

```
SetMAStationConfigurationName('Station #1', 'Loop #1');
```

Related:
Function list

SetNotificationUserEmailAddress

This function is used to change the Email address of a user configured for Email notifications.

Inputs

Variable	Type	Description
User name	String	User name (optional)
User email address	String	Current Email address (optional)
New email address	String	Email address to replace the existing address

Outputs

Variable	Type	Description
Value	Boolean	True if the Email address was replaced False if the Email address was not replaced

The 'User name' or 'User Email address' must not be blank. One field must be used.
Changes made at runtime are not saved in the project.
The new Email address is checked for validity, as much as can be done. If the new address fails, an entry will be placed in the event log and the existing user address will not be changed.
The 'User name' and 'User Email address' are case sensitive.

Example

Use this format to replace the user Email address using the existing Email address.
value:=SetNotificationUserEmailAddress('', 'user1@testsuite.com', 'user1@homeoffice.com');

Use this format to replace the user Email address using the user name.
value:=SetNotificationUserEmailAddress('Joe Smith', '', 'user1@homeoffice.com');

Possible Email address errors.

Address is blank
User name too long
Invalid Character
No domain name
Too many @ symbols
User name ends with a period
Domain name ends with a period
Domain name has two periods
Sub domain name has an invalid character
Sub domain name invalid end character
Sub domain name is too short
Missing @ symbol
Domain too long
No user name
Missing . symbol
User name starts with a period
Domain name starts with a period
User name has two periods

Related:
Function list

SetNotificationUserSMSNumber

This function is used to change the telephone number of a user configured for SMS notifications.

Inputs

Variable	Type	Description
User name	String	User name (optional)
User telephone number	String	Current telephone number (optional)
New user telephone number	String	Telephone number to replace the existing telephone number

Outputs

Variable	Type	Description
Value	Boolean	True if the number address was replaced False if the number address was not replaced

The 'User name' or 'User telephone number' must not be blank. One field must be used.
Changes made at runtime are not saved in the project.
The 'User name' is case sensitive.
The 'User telephone number' must exactly match.

Example

Use this format to replace the user telephone number using the existing telephone number.
value:=SetNotificationUserSMSNumber('', '012345678', '987654321');

Use this format to replace the user telephone number using the user name.
value:=SetNotificationUserSMSNumber('Joe Smith', '', '987654321');

Related:
Function list

SetPortReadEnable

This function is used to enable or disable a read a port is configured to initiate. This could be used for example to read a value or set of values only when a certain screen is viewed.

Inputs

Variable	Type	Description
Port name	String	Name of port
Read index	Integer	Number of the read
Enable	Boolean	Desired state of the read

If the enable is true the read will be active and issued as determined by the port. If enabled is false the read will not be active and the port will not process it. If the requested read enable is false and it is the only active read the command will fail.

Outputs

Variable	Type	Description
Value	Boolean	True if the requested state was set

Example

```
value:=SetPortReadEnable('Chiller Port 1',1,true);
if value then
begin
//read state was set -- enabled
end;

if SetPortReadEnable('Chiller Port 1',1,false) then
begin
//read state was set -- disabled
end;
```

Related:
Function list

SetSystemClock

This function is used to set the data and time on the computer.

Note:
For Windows 7 and above, the logged on user must be the administrator or have administrator privileges and UAC must be disabled.

Inputs

Variable	Type	Description
Year	Integer	1601 - 30827
Month	Integer	1 - 12
Day	Integer	1 - 31
Hour	Integer	0 - 23
Minutes	Integer	0 - 59
Second	Integer	0 - 59
Milliseconds	Integer	0 - 999

Outputs

Variable	Type	Description
Success	Boolean	True for success, false for failure

Example

```
success:=SetSystemClock(2014, 2, 25, 13, 22, 30 ,0);
```

Related:
Function list

SetSystemWindowPosition

This function is used to define the position of the next system window opened. It must be called before each system window is opened or the window will be opened in the default position.

Note: If needed, use the graphic configuration "Window/settings: Show mouse position" feature to determine precise coordinates.

Inputs

Variable	Type	Description
X	Integer	X (horizontal) coordinate for left side of window
Y	Integer	Y (vertical) coordinate for top of window

Outputs

Variable	Type	Description
None		

Example

```
SetSystemWindowPosition (0, 0);    //left, top
```

System windows that can use this function

EmailUserSendMessage *
GetUserInputBoolean
GetUserInputBoolean2
GeUserInputBoolean4
GetUserInputDate
GetUserInputFloat
GetUserInputInteger
GetUserInputString
GetUserInputStringMask
GetUserInputTime
LogOff
LogOn
OmniViewReport
OpenWindowUserSelect
SchedulerOpenMonitorWindow
SMSUserSendMessage *
ViewDateRangeTrendHistory **
GetUserInputTime

*This function overrides the values configured in the script command.
**This function overrides the form last saved position logic on form display.

Related:
Function list

SetTaskState

This function is used to set the enabled state of a task.

Inputs

Variable	Type	Description
Task name	String	Task name
Value	Boolean	True to enable, false to disable

Outputs

Variable	Type	Description
Value	Boolean	True if state was accepted False if state was not accepted

Examples

```
value:=SetTaskState('FTP Task 1',false);

if SetTaskState('FTP Task 1',false) then
begin
//success
end;
```

Related:
Function list
TaskExecute
TaskScheduleEdit

SetTrendPen

This function is used to change the point.item of one or more pens on a trend. The point.item must have been configured for data logging for native trends. ODBC trends use the ODBC connection and field names. The window containing the named trend must be open.
This function call is not for static trend. Use "SetTrendStaticPen" for static trends.

Inputs

Variable	Type	Description
Window name	String	Window name
Trend name	String	Trend name

The next variable consists of three parts for each pen to modify.

Pen number	Integer	Pen number (1-32)
Tagname	String	Point tagname/field name (if blank, the pen will be deleted) For ODBC trends, set to the field name
Item number	Integer	Item number of point, set to 0 (zero) for ODBC trends

Outputs

Variable	Type	Description
Result	Integer	0 (zero) True if success Negative number for failure

Notes:
1) It is **best** to set all the pens with one script command than using one command for each pen. The trend reloads all pen data on each command.
2) If the trend is not showing the current date, this command will reset the trend to the current date.

Examples

This sets pen 1 of trend "Flow_Trend" in window "Main_Window" to point.item "TankFlow.5000".

```
SetTrendPen('Main_Window','Flow_Trend',[1,'TankFlow',5000]);
```

This sets pen 1 and 2.

```
SetTrendPen('Main_Window','Flow_Trend',[1,'TankFlowUpper',5000,
2,'TankFlowLower',5000]);
```

This sets pen 1, 2 and 8.

```
SetTrendPen('Main_Window','Flow_Trend',[1,'TankFlowUpper',5000,
2,'TankFlowMiddle',5000, 8,'TankFlowLower',5000]);
```

This sets pen 1, clears pen 2 and sets pen 8.

```
SetTrendPen('Main_Window','Flow_Trend',[1,'TankFlowUpper',5000,
2,'',0, 8,'TankFlowLower',5000]); // '' is two single quotes
```

Related:
Function list

SetTrendPenColor

This function is used to change the color of a trend pen(s). This function call is not for static trends.

Inputs

Variable	Type	Description
Window name	String	Window name
Trend name	String	Trend name

The next variable consists of two parts for each pen to modify.

Pen number	Integer	Pen number (1-32)
Pen color	Color	New pen color

Outputs

Variable	Type	Description
None		

Examples

This sets pen 1 of trend "Flow_Trend" in window "Main_Window" to "Red".

SetTrendPenColor('Main_Window','Flow_Trend',[1,clRed]);

This sets pen 1 to green and pen 2 to yellow

SetTrendPenColor('Main_Window','Flow_Trend',[1,clGreen,2,clYellow]);

Related:
Function list

SetTrendStaticPen

This function is used to change the point.item of one or more pens on a trend (static). The point.item must have been configured for data logging for native trends. ODBC trends use the ODBC connection and field names. The window containing the named trend must be open. This function call is not for live trends. Use "SetTrendPen" for live trends.

Inputs

Variable	Type	Description
Window name	String	Window name
Trend name	String	Trend name
Start time	String	Trend starting time *2
Start date	String	Trend starting date *2
End time	String	Trend ending time *2
End date	String	Trend ending date *2
Zoom to fit	Integer	Zoom the axis 0 = X and Y axis 1 = X axis 2 = Y axis Any other value and the axis are not altered.

The next variable consists of three parts for each pen to modify.

Pen number	Integer	Pen number (1-8)
Tagname	String	Point tagname/field name (if blank, the pen will be deleted) For ODBC trends, set to the field name
Item number	Integer	Item number of point, set to 0 (zero) for ODBC trends

Outputs

Variable	Type	Description
Result	Integer	0 (zero) True if success Negative number for failure

Notes:

1) It is **best** to set all the pens with one script command than using one command for each pen. The trend reloads all pen data on each command.
2) For ODBC trend (static) this could be an increasing value. See "ODBC X-Axis Type." If the X-Axis Type is "number" or "UNIX time", place the beginning range value in "Start time" and the ending range value in "End time."

Examples

This sets pen 1 of trend "Flow_Trend" in window "Main_Window" to point.item "TankFlow.5000".

SetTrendStaticPen('Main_Window', 'Flow_Trend',<start time>, <start date>, <end time>, <end date>, true, [1,'TankFlow',5000]);

This sets pen 1 and 2.

SetTrendStaticPen('Main_Window','Flow_Trend', <start time>, <start date>, <end time>, <end date>, true, [1,'TankFlowUpper',5000,2,'TankFlowLower',5000]);

This sets pen 1, 2 and 8.

SetTrendStaticPen('Main_Window','Flow_Trend', <start time>, <start date>, <end time>, <end date>, true, [1,'TankFlowUpper',5000, 2,'TankFlowMiddle',5000, 8,'TankFlowLower',5000]);

This sets pen 1, clears pen 2 and sets pen 8.

SetTrendStaticPen('Main_Window','Flow_Trend', <start time>, <start date>, <end time>, <end date>, true, [1,'TankFlowUpper',5000,2,'',0, 8,'TankFlowLower',5000]);
// '' is two single quotes

Related:
Function list

SetTrendPenMini

This function is used to change the point.item of a "Mini single pen trend".

Inputs

Variable	Type	Description
Window name	String	The name of the window
Mini trend name	String	The name of the mini single pen trend.
Tagname	String	Point tagname (must not be blank)
Item number	Integer	Item number of point

Outputs

Variable	Type	Description
Result	Integer	0 (zero) True if success Negative number for failure

Example

This sets the pen of mini trend "Flow_Trend" in window "Main_Window" to point.item "TankFlow.5000".

```
SetTrendPenMini('Main_Window','Flow_Trend','TankFlow',5000);
```

Related:
Function list

SetWindowColor

This function is used to set the color of the window.

Inputs

Variable	Type	Description
Window name	String	The name of the window
Foreground	Integer	Foreground color
Background	Integer	Background color
Brush	Integer	Brush style

Outputs

Variable	Type	Description
None		

Foreground/Background
This is the color selection. If the brush style is 'Solid' or 'Clear' the background color has not effect. The color selection must be in the range of 2,147,483,646 .. 2,147,483,647 ($7FFFFFFF-1 .. $7FFFFFFF). Some common colors are listed below. Use the 'color picker' menu item in the script editor if required.

Brush
The brush must be one of the following values:

Solid = 0	Clear = 1
Horizontal = 2	Vertical = 3
FDiagonal = 4	BDiagonal = 5
Cross = 6	DiagCross = 7

Example

```
//setting window color to red
SetWindowColor('Main_Window',255,255,0);
```

Related:
Function list

Color	Value	Color	Value	Color	Value
Black	0	WebAntiqueWhite	14150650	WebTomato	4678655
Maroon	128	WebWheat	11788021	WebCrimson	3937500
Green	32768	WebAliceBlue	16775408	WebBrown	2763429
Olive	32896	WebGhostWhite	16775416	WebChocolate	1993170
Navy	8388608	WebLavender	16443110	WebSandyBrown	6333684
Purple	8388736	WebSeashell	15660543	WebLightSalmon	8036607
Teal	8421376	WebLightYellow	14745599	WebLightCoral	8421616
Gray	8421504	WebPapayaWhip	14020607	WebOrange	42495
Silver	12632256	WebNavajoWhite	11394815	WebOrangeRed	17919
Red	255	WebMoccasin	11920639	WebFirebrick	2237106

Lime	65280	WebBurlywood	8894686	WebSaddleBrown	1262987
Yellow	65535	WebAzure	16777200	WebSienna	2970272
Blue	16711680	WebMintcream	16449525	WebPeru	4163021
Fuchsia	16711935	WebHoneydew	15794160	WebDarkSalmon	8034025
Aqua	16776960	WebLinen	15134970	WebRosyBrown	9408444
LtGray	12632256	WebLemonChiffon	13499135	WebPaleGoldenrod	11200750
DkGray	8421504	WebBlanchedAlmond	13495295	WebLightGoldenrodYellow	13826810
White	16777215	WebBisque	12903679	WebOlive	32896
MoneyGreen	12639424	WebPeachPuff	12180223	WebForestGreen	2263842
SkyBlue	15780518	WebTan	9221330	WebGreenYellow	3145645
Cream	15793151	WebYellow	65535	WebChartreuse	65407
MedGray	10789024	WebDarkOrange	36095	WebLightGreen	9498256
WebSnow	16448255	WebRed	255	WebAquamarine	13959039
WebFloralWhite	15792895	WebDarkRed	139	WebSeaGreen	5737262
WebLavenderBlush	16118015	WebMaroon	128	WebGoldenRod	2139610
WebOldLace	15136253	WebIndianRed	6053069	WebKhaki	9234160
WebIvory	15794175	WebSalmon	7504122	WebOliveDrab	2330219
WebCornSilk	14481663	WebCoral	5275647	WebGreen	32768
WebBeige	14480885	WebGold	55295	WebYellowGreen	3329434
WebLawnGreen	64636	WebDarkTurquoise	13749760	WebWhite	16777215
WebPaleGreen	10025880	WebCadetBlue	10526303	WebLightgrey	13882323
WebMediumAquamarine	11193702	WebDarkCyan	9145088	WebGray	8421504
WebMediumSeaGreen	7451452	WebTeal	8421376	WebSteelBlue	11829830
WebDarkGoldenRod	755384	WebDeepskyBlue	16760576	WebSlateBlue	13458026
WebDarkKhaki	7059389	WebDodgerBlue	16748574	WebSlateGray	9470064
WebDarkOliveGreen	3107669	WebBlue	16711680	WebWhiteSmoke	16119285
WebDarkgreen	25600	WebNavy	8388608	WebSilver	12632256
WebLimeGreen	3329330	WebDarkViolet	13828244	WebDimGray	6908265
WebLime	65280	WebDarkOrchid	13382297	WebMistyRose	14804223
WebSpringGreen	8388352	WebMagenta	16711935	WebDarkSlateBlue	9125192
WebMediumSpringGreen	10156544	WebFuchsia	16711935	WebDarkSlategray	5197615
WebDarkSeaGreen	9419919	WebDarkMagenta	9109643	WebGainsboro	14474460
WebLightSeaGreen	11186720	WebMediumVioletRed	8721863	WebDarkGray	11119017
WebPaleTurquoise	15658671	WebPaleVioletRed	9662683		
WebLightCyan	16777184	WebBlueViolet	14822282		
WebLightBlue	15128749	WebMediumOrchid	13850042		
WebLightSkyBlue	16436871	WebMediumPurple	14381203		
WebCornFlowerBlue	15570276	WebPurple	8388736		
WebDarkBlue	9109504	WebDeepPink	9639167		
WebIndigo	8519755	WebLightPink	12695295		
WebMediumTurquoise	13422920	WebViolet	15631086		
WebTurquoise	13688896	WebOrchid	14053594		
WebCyan	16776960	WebPlum	14524637		
WebAqua	16776960	WebThistle	14204888		
WebPowderBlue	15130800	WebHotPink	11823615		
WebSkyBlue	15453831	WebPink	13353215		
WebRoyalBlue	14772545	WebLightSteelBlue	14599344		
WebMediumBlue	13434880	WebMediumSlateBlue	15624315		
WebMidnightBlue	7346457	WebLightSlateGray	10061943		

SetWindowDate

This function is used to change any elements on a window that display data based on a selected date. The "Trend" element and "Round Trend Chart" use the date.

Another method to view past days trends is using the ViewTrendHistory mouse command.

Inputs

Variable	Type	Description
Window name	String	The name of the window.
Day	Integer	Day
Month	Integer	Month
Year	Integer	Year

If the date is not a valid date, the command is terminated. If all date fields are "0" then the current date is used. (The date the computer has for the current date).

Example

```
//setting the date to 2 August 2007
SetWindowDate('Tank A Level', 2, 8,2007);
```

Related:
Function list
GetUserInputDate

SilenceAcknowledgeCommand

Perform a "Silence" command and then an "Acknowledge" command.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

SilenceAcknowledgeCommand;

Related:
Function list
AcknowledgeCommand
SilenceCommand
AcknowledgePointAlarm

SilenceCommand

Perform a "Silence”. The silence command stops all sound playing. The sound queue is cleared.

Inputs

Variable	Type	Description
None		

Outputs

Variable	Type	Description
None		

Example

SilenceCommand;

Related:
Function list
AcknowledgeCommand
SilenceAcknowledgeCommand

Simulate

This function requires the "Simulation" feature to be enabled. If the feature is not enable, the function returns an error code (-4).

Inputs

Variable	Type	Description
Action	String	Specifies action/command
Parameters	Array of variant	Command parameters

Outputs

Variable	Type	Description
Values	Array of variant	Values of variables

Example

values:=Simulate('Status',['Fire panel 6A2']);

Commands

Status	Watchdog	Camera

Command	Function
Status	Returns if port is in simulation.

values:=Simulate('Status',['Fire panel 6A2']); //port name

values[0] = error code
values[1] = 0 = not in simulation, 1 = in simulation

Command	Function
Watchdog	Sets the port watchdog state and point quality state.

values:=Simulate('Watchdog',['Fire panel 6A2',true]);
//port name, watchdog state (true/false)

values[0] = error code

If the watchdog state is set, it will remain in the state set until altered with this command. If the watchdog is set true the point quality is set "bad". If the watchdog state is set false, the point quality is set "good".

Command	Function
Camera	Sets the image link to render in the graphic element.

File types

PNG	Portable Network Graphics
JPG/JPEG	Joint Photographic Experts Group
BMP	Bitmap Image File
ICO	Icon
SVG	Scaled Vector Graphic (Static)

```
values:=Simulate('Camera',['Turbine', 'C:\Images\Cam_1.jpg']);
                                //port(camera) name, full path to image
```

Note: To remove the link, pass an empty string for the file path. e.g.

```
values:=Simulate('Camera',['Turbine', '']);          //port name, empty string
```

values[0] = error code

Error codes

The first value of the output array (where applicable) is an error code. If the value is not equal to zero, an error occurred.

Code	**Description**
0	No error
-1	Port name missing or invalid
-2	Command missing
-3	Unknown command
-4	Global simulation not active
-5	Port is not in simulation
-6	Parameter count error (too many or too few)
-7	File not found

Related:
Function list

SMSPurgeSendQueue

This function is used to remove pending (not sent) SMS messages from the SMS notifications queue.

Inputs

Variable	Type	Description
Future	Array	Future, pass in empty array

Outputs

Variable	Type	Description
Value	Boolean	True if the command was accepted False if the command was not accepted (Send queue was empty)

Notes:

1) The command, if accepted, sets a flag to allow any active sending of a message to complete and is processed before another message is sent.
2) An event will be added to the event log when the command purges the send queue.
3) The action also clears all messages waiting for an acknowledgement, if any are configured.
4) The action removes all message in the queue from SMSSendMessage or SMSUserSendMessage commands.

Examples

Clears the "send" queue.

```
value:=SMSPurgeSendQueue([]);
```

Related:
Function list

SMSSendMessage

This function is used to send an SMS message.

Inputs

Variable	Type	Description
User name	String	SMS user name(s)
Message	String	SMS message

Outputs

Variable	Type	Description
Value	Boolean	True if the message was queued False if the message was not queued

Examples

Sends the message to one SMS user.

value:=SMSSendMessage(['Joe Smith'] , 'Tank 1A overflow');

Sends the message to several SMS users.

value:=SMSSendMessage(['Joe Smith' , ' Bill Jones' , 'Operations'] , 'Tank 1A overflow');

Related:
Function list
SMSUserSendMessage

SMSUserSendMessage

This function is used to send an SMS message with user input.

Inputs

Variable	Type	Description
User name	String	SMS user name(s), if blank all configured SMS user names are used.
Can change	Boolean	The user can select from the list of user names.
Can edit	Boolean	The user can edit the SMS message before sending.
Show keyboard	Boolean	If enabled, an on screen keyboard is displayed with the window.
In SMS	String	The SMS message to send. It can be blank and the user will enter the text.
Left position	Integer	Horizontal position of the window.
Top position	Integer	Vertical position of the window.

"Left position" and "Top position" refer to the left and top corner of the window. If both values are -1 the window is centered on the main monitor.

Outputs

Variable	Type	Description
Value	Boolean	True if the message was queued False if the message was not queued

Examples

```
value:=SMSSendMessage(['Joe Smith'] , True, True, True, '', -1, -1);
value:=SMSSendMessage(['Joe Smith' , ' Bill Jones' , 'Operations'] , true, true, true, 'Tank 1A overflow' , 100 ,20);
value:=SMSSendMessage([] , True, True, True, 'Tank 1A overflow' , -1 ,-1);
value:=SMSSendMessage([] , True, True, False, 'Tank 1A overflow' , -1 ,-1); //no keyboard
value:=SMSSendMessage([] , True, False, True, 'Tank 1A overflow' , -1 ,-1); //user cannot edit
```

Related:
Function list
SMSSendMessage

StartCameraRecording

This function is used to start camera recording if recording is enabled. This does not affect loop recording if enabled.

Inputs

Variable	Type	Description
Camera name	String	Name of camera

Outputs

Variable	Type	Description
None		

Example

```
StartCameraRecording('IP_Camera_1');
```

Related:
Function list
StopCameraRecording
CameraSaveLoop

StopCameraRecording

This function is used to stop camera recording if recording is enabled. This does not affect loop recording if enabled.

Inputs

Variable	Type	Description
Camera name	String	Name of camera

Outputs

Variable	Type	Description
None		

Example

```
StopCameraRecording('IP_Camera_1');
```

Related:
Function list
StartCameraRecording
CameraSaveLoop

StringGet

This function is used to read a string from a port. A valid read must be configured and enabled for the register range containing the string.

Inputs

Variable	Type	Description
Port name	String	Port name
Address	String	Register address or tagname
Format	Integer	String format in registers
Count	Integer	Number of characters to return

Outputs

Variable	Type	Description
Value	String	String value

The address is the register address. This value must be a valid address for the type of port configured. For Ethernet/IP it is the tagname of a string type in the PLC.

The format is the byte structure of the registers and might not be applicable for the port type. For registers that are single byte the format is ignored.

Format	Description	
0)	Default	
1)	2 characters per word, character order high, low – default	ABCD
2)	2 characters per word, character order low, high	BADC
3)	1 character per word, character in high byte	
4)	1 character per word, character in low byte	
5)	4 characters per longword, character order	ABCD
6)	4 characters per longword, character order	DCBA

PLCs with byte based memory (Siemens) are always 0. One character per byte in order of memory address.

For AB Logix the format and count parameters are ignored. The string returned from the external device is returned in the result.

For AB PLC5 DF1/PCCC the format and count parameters are ignored. The string returned from the external device is returned in the result.

If the result needs to be displayed in a window, save the string in a global (GlobalSet) and use the "Script Global" animation or use a "Graphic script" animation, if applicable.

Notes:
1) If the start address plus the count is greater than the table length, the results will be unpredictable. If this is the case, segment the StringGet call into 2 or more reads and add the strings together after collection from the tables.
2) For K Sequence (Automation Direct/Koyo), the format should be 2.
3) For SAIA, the format should be 5.
4) For GE SRTP, our testing was with format 2.
5) Count is ignored for protocol that provide support for strings.

Examples

```
value:=StringGet('Pump1','400040',0,14);

//string end past end of table read, string length 20 characters
value1:=StringGet('Pump1','400040',0,14);
value2:=StringGet('Pump1','400014',0,6);
totalString:=value1 + value2;
```

MODBUS Example

Register	Value	Format 0/1	Format 2	Format 3	Format 4
400050	16706	AB	BA	A	B
400051	17220	CD	DC	C	D
400052	23129	ZY	YZ	Z	Y
400053	22615	XW	WX	X	W

```
value:=StringGet('MPort1','400050',0,8); result would be value = ABCDZYXW
value:=StringGet('MPort1','400050',2,8); result would be value = BADCYZWX
value:=StringGet('MPort1','400050',3,4); result would be value = ACZX
value:=StringGet('MPort1','400050',4,4); result would be value = BDYW
```

Related:
Function list
StringGetNullCheck
StringSet

StringGetNullCheck

This function sets the property options for "StringGet" null validation checks. "StringGet" is a "no checks" function that converts the bytes returned from a communication port without verifying if the bytes (characters) are NULL. For example, a returned byte of decimal 0 (zero) is converted to ASCII NULL. A NULL character is not permitted in many string instances because it is interpreted as an end of string character. The NULL checks are disabled until one or both of the properties are set. Normally this function is called in the "On runtime start" script. This function can be called at any time to enable/disable/alter the NULL checks.

Inputs

Variable	Type	Description
Command	Integer	0 = read, 1 = write
Properties	Array	Various

Outputs

Variable	Type	Description
Value	Array	Various

Properties are enabled or disabled. 'E' = enabled, 'D' = disabled.

Property 1
The first property checks for a NULL byte in the first character of the string and if the byte is NULL replaces the complete string with the specified character.

Example: ['EBlank', 'D']	If the first character is NULL the string 'Blank' is returned from "StringGet".
Example: ['E0', 'D']	If the first character is NULL the string '0' is returned from "StringGet".
Example: ['E', 'D']	If the first character is NULL the string '' (an empty string) is returned from "StringGet".
Example: ['D', 'D']	The first character check is not performed.

Property 2
The second property checks for a NULL byte in each character of the string and if the byte is NULL, the byte with the specified character(s).

Example: ['D', 'E-']	If a character is NULL the character is replaced with '-'.
Example: ['D', 'E']	If a character is NULL the character is deleted.
Example: ['D', 'E ']	If a character is NULL the character is replaced with a space character.
Example: ['D', 'E12']	If a character is NULL the character is replaced with a '12'.

Example: ['D', 'D'] The second character check is not performed.

Command	Function
0	Return the check property values

values:=StringGetNullCheck(0,[]);
values[0] = True if command accepted, false if error detected, boolean
values[1] = Count of values returned, after this value (0 if values[0] = false)
values[2] = First null byte check enabled, a boolean
values[3] = First null byte check replacement value, a string
values[4] = Second null byte check enabled, a boolean
values[5] = Second null byte check replacement value, a string

Command	Function
1	Set the check property values

values:=StringGetNullCheck(1,['D', 'D']); //disable both checks
values:=StringGetNullCheck(1,['E0', 'D']);
values[0] = True if command accepted, false if error detected, boolean
values[1] = Count of values returned, integer (after this value)

Note:
If the first property check (NULL in first byte) is enabled and detects a NULL in the first character position of the string, the string is replaced with the specified value and the second check, if enabled, is not performed.

Related:
Function list
StringGet

StringSet

This function is used to write a string to a port.

Inputs

Variable	Type	Description
Port name	String	Port name
Address	String	Register address or tagname
Format	Integer	String format in registers
String	String	The string to write

Outputs

Variable	Type	Description
Value	Boolean	True if added to write queue False if not added to write queue

The address is the register address. This value must be a valid address for the type of port configured. For Ethernet/IP it is the tagname of a string type in the PLC.

The format is the byte structure of the registers and may not be applicable for the port type. For registers that are single byte the format is ignored.

Format	Description	
0)	Default	
1)	2 characters per word, character order high, low – default	ABCD
2)	2 characters per word, character order low, high	BADC
3)	1 character per word, character in high byte	
4)	1 character per word, character in low byte	
5)	4 characters per longword, character order	ABCD
6)	4 characters per longword, character order	DCBA

PLCs with byte based memory (Siemens) are always 0. One character per byte in order of memory address.

Notes:

1) The maximum string length is determined by the PLC/protocol.

MODBUS, Mitsubishi Q	240 characters
Siemens	222 characters
PLC5	82 characters
Saia	128 characters (32 registers), format 5 or 6, only
GE SRTP	70 characters (Tested with format 2)

2) For word/longword based registers, if the string length is not even, a zero will be written for the unused byte(s).
3) BACnet/IP write priority is 0.

Examples

value:=StringSet('Pump1','400040',0,'Fire');
value:=StringSet('Pump1','401234',2,'Raise');

MODBUS Example

value:=StringSet('Pump1','400050',<format>,'ABCDZYXW');

Register	Format 0/1	Format 2	Format 3	Format 4
400050	AB	BA	A0	0B
400051	CD	DC	B0	0A
400052	ZY	YZ	C0	0D
400053	XW	WX	D0	0C
400054	N/A	N/A	Z0	0Y
400055	N/A	N/A	Y0	0Z
400056	N/A	N/A	X0	0W
400057	N/A	N/A	W0	0X

Related:
Function list
StringGet

TaskExecute

This function will execute the specified task.

Inputs

Variable	Type	Description
Task name	String	The name of the task to execute.
CalculateNextTime	Boolean	Determine next time.

Outputs

Result	Boolean	True if the task executed.

None

Note:

1) The "CalculateNextTime" input is **required** but, only applies to tasks that are timed based (Daily, Weekly, Monthly or Hourly). If the value is true, the next execution date and/or time will be calculated.
2) The command will execute the task regardless of the task "enabled" state.

Example

```
value:=TaskExecute('SendFile', ['True']);    //task name
```

Related:
Function list
SetTaskState
TaskScheduleEdit

TaskScheduleEdit

This function display the task scheduler dialog.

Inputs

Variable	Type	Description
AlterRuntimeTask	Boolean	True = implement task changes
SaveToDisk	Boolean	True = save all changes to disk

Outputs

None

Note:
If "AlterRuntimeTask" is true, when the edit window is closed, all tasks will be reloaded/restarted. Any task with the "When runtime starts" trigger will not be executed.

Example

```
TaskScheduleEdit(True, True);        //alter running task and save changes to disk
```

Related:
Function list
SetTaskState
TaskExecute

TimerGet

This function is used to read the value of a script timer field.

Inputs

Variable	Type	Description
Index	Integer	Timer number
Field	String	Field of the timer

Outputs

Variable	Type	Description
Value	Variant	Value of requested field

Field Values

Name	Type	Description
TT	Boolean	Timer enabled and not done. (Timer timing)
DN	Boolean	Timer enabled and accumulator >= preset. (Timer done)
EN	Boolean	Timer enabled
ACC	Integer	Accumulator
PRE	Integer	Preset
ARS	Boolean	Timer Auto Restart

Examples

```
value:=TimerGet(7,'TT'); //returns a Boolean
value:=TimerGet(3,'ACC'); //returns an integer
```

Related:
Function list
Script Timers
TimerSet

TimerSet

This function is used to set the value of a script timer field.

Inputs

Variable	Type	Description
Index	Integer	Timer number
Field	String	Field of the timer
Value	Variant	New field value

Outputs

Variable	Type	Description
Value	Boolean	Command was successful

Field Values

Name	Type	Description
EN	Boolean	Timer enabled
ACC	Integer	Accumulator
PRE	Integer	Preset
ARS	Boolean	Auto Restart

Note: Changing the ACC or PRE after the timer has started has no effect on the active timing.

Examples

```
//enable timer number 7
value:=TimerSet(7,'EN',True);

//set the preset of timer 10 to 60 seconds
value:=TimerSet(10,'PRE',60);
```

Related:
Function list
Script Timers
TimerGet

TotalizerGetValue

This function returns a value from a totalizer.

Inputs

Variable	Type	Description
Name	String	Totalizer name
Date	Integer	Totalized date (see table below)

Outputs

Variable	Type	Description
Values	Variant array	First index is the boolean result True if totalizer located otherwise false If false array has only one element: values[0]

Date	Value	Result index
-1	Previous day total	values[1]
0	Today total (current running total)	values[1]
1-31	Date of month total	values[1]
32	Total of all 31 dates	values[1]
33	All days of the month	Values[1-31]

Examples

```
values:=TotalizerGetValue(['OilFlowA',0]);    //return today's total
if values[0] then
 oilTotal:=values[1];

values:=TotalizerGetValue(['OilFlowA',33]);  //return total for all days
if values[0] then
 begin
  day1:=values[1];
  ...
  day31:=values[31];
 end;
```

Related:
Function list
Totalizers
TotalizerSetValue
TotalizerShowValues

TotalizerSetValue

This function sets a totalizer value.

Inputs

Variable	**Type**	**Description**
Name	String	Totalizer name
Date	Integer	Totalized date (see table below)
Values	Array of float	Values (see table below)

Outputs

Variable	**Type**	**Description**
Value	Boolean	True if totalizer located False if totalizer not located

Date	**Value**	**Values count**	**Values**
-1	Previous day	1	Value to set [value]
0	Today total	1	Value to set [value]
1-31	Date of month	1	Value to set [value]
32	Set all 31 dates to value	1	Value to set [value]
33	Set all 31 dates to values	31	[value 1, value 2, value N, value 31]

Examples

```
values:=TotalizerSetValue('OilFlowA',0, [0]);          //set today's total to 0
values:=TotalizerSetValue('OilFlowA',-1, [22]);        //set yesterday's total to 22
values:=TotalizerSetValue('OilFlowA',15, [9]);         //set the 15th total to 9
values:=TotalizerSetValue('OilFlowA',32, [0]);         //set all totals to 0
values:=TotalizerSetValue('OilFlowA',33, [0,0,0,0,0,0,0,1,1,1,1,1,1,1,2,2,2,2,2,2,2,
                                          3,3,3,3,3,3,3,4,4,4]);    //set dates to values
```

Related:
Function list
TotalizerGetValue
TotalizerShowValues

TotalizerShowValues

This function views the 31 values of a totalizer in a horizontal trend.

Inputs

Variable	Type	Description
Modal	Boolean	If true the window will be modal. (User must close window to continue)
Names	Array of string	Totalizer names (No totalizer names will open an empty trend window.)

Outputs

Variable	Type	Description
None		

Example

```
TotalizerShowValues(True, ['OilFlowA', 'OilFlowB']);        //view two totalizer values
```

Related:
Function list
TotalizerGetValue
TotalizerSetValue

Treeview

This function collapses/expands a Treeview graphic element.

Inputs

Variable	Type	Description
Properties	Array	An array consisting of a command and any other needed values for the command.

Outputs

Variable	Type	Description
None		

Both commands search for a window name matching the parameter 1 value.
If a window is found a search for a treeview graphic element with an item ID matching the parameter 2 value, in the first item of the treeview. If search success the treeview items will be collapsed.

Command	Parameter 1	Parameter 2	Description
Collapse	Window name	Item ID	Collapse treeview
Expand	Window name	Item ID	Expand treeview

Note: If more than one treeview graphic elements has the same item ID for the first leaf in the treeview, the command will only execute on the first treeview found.

Examples

Command	Function
Collapse	Collapses the treeview.

```
Collapse(['P1Main', 7]);
```

Expand	Expands the treeview.

```
Expand(['P1Main', 7]);
```

Related:
Function list
OnTreeviewClick

TrendAddBand

This function adds a band to the trend. The window containing the named trend must be open.

Inputs

Variable	Type	Description
Window name	String	The name of the window
Trend name	String	The name of the trend
Y1 value	Float	The 'Y1' value
Y2 value	Float	The 'Y2' value
Color	Integer	Pen color
Style	Integer	Fill style

Outputs

Variable	Type	Description
Result	Integer	0 (zero) True if success Negative number for failure

Color

Refer to SetWindowColor for a list of common colors. The script editor has an "Insert color" command.

Style

Refer to SetWindowColor for a list of fill styles.

Example

result:=TrendAddBand('Main_Window','Flow_Trend',20,80,255,7); //Y1,Y2 axis value, color, fill style

Related:
Function list
TrendAddLimit
TrendRemoveLimits

TrendAddLimit

This function adds a fixed indicator to the trend. The window containing the named trend must be open. For the horizontal trend, the indicator is a fixed horizontal line and for the round trend chart, the indicator is a fixed circle.

Inputs

Variable	Type	Description
Window name	String	The name of the window
Trend name	String	The name of the trend
Y value	Float	The "Y" value
Color	Integer	Line color
Width	Integer	Line width (pen)
Style	Integer	Pen style

Outputs

Variable	Type	Description
Result	Integer	0 (zero) True if success Negative number for failure

Color

Refer to SetWindowColor for a list of common colors. The script editor has an "Insert color" command.

Width

The pen width is a value from 1 to 64.

Notes:
1) If the pen width is greater than one (> 1) the style is solid. Pen styles only apply to a pen width of one (1).
2) For the round trend chart the "Y" value must be in the trend theme range.

Style
The style must be one of the following values:

Solid	= 0	Dash	= 1	Dot	= 2
DashDot	= 3	DashDotDot	= 4		

Example

This adds a solid red line with a width of three pixels at the Y axis point of fifty.
result:=TrendAddLimit('Main_Window','Flow_Trend',50,255,3,0); //Y axis value, color, pen width, pen style

Related:
Function list
TrendAddBand
TrendRemoveLimits

TrendRemoveLimits

This function removes all limits and bands from a trend. The window containing the named trend must be open.

Inputs

Variable	Type	Description
Window name	String	The name of the window
Trend name	String	The name of the trend

Outputs

Variable	Type	Description
Result	Integer	0 (zero) True if success Negative number for failure

Example

```
result:=TrendRemoveLimits('Main_Window','Flow_Trend');
```

Related:
Function list
TrendAddLimit
TrendAddBand

TrendSavePlotPictureToFile

This function saves the plot image to a file. The window containing the named trend must be open.

Inputs

Variable	Type	Description
Window name	String	The name of the window
Trend name	String	The name of the trend
Path	String	The complete path and file name. The file extension selects the file type.
Zoom to fit	Integer	Zoom before picture capture. 0 = X and Y axis 1 = X axis 2 = Y axis

Outputs

Variable	Type	Description
None		

File type	Extension
Bitmap	BMP
JPEG	JPG
PNG	PNG
Metafile	EMF
Portable Document Format	PDF

Notes:
1. If a file with the same name/path exist, it will be overwritten.
2. Zoom is ignored for the round trend chart.

Examples

TrendSavePlotPictureToFile('Flow Trend Window', 'Amps', 'C:\test.bmp',0);
TrendSavePlotPictureToFile('Flow Trend Window', 'Volts', 'C:\test.jpg',1);
TrendSavePlotPictureToFile('Flow Trend Window', 'GPM', 'C:\test.png',2);
TrendSavePlotPictureToFile('Flow Trend Window', 'Flow', 'C:\test.emf',3); //do not alter zoom amount

Related:
Function list

UArray

This function is used to read/write/manipulate/etc. arrays. The "Commands" menu item in the script editor can assist in formatting commands.

The format is:

values:=UArray('<array name>', '<command>', [<dimension>], [<options>]);

The result is an array and always contains at least two elements.

values[0] The result of the command.
0 is success, less than zero (< 0) is an error code.

values[1] The number of remaining elements in the result array.
This value can be 0.

values[n] is command dependent.

Note:
Loading/saving/copying a large number of elements with one command might cause a temporary HMI slowdown, especially strings.

Commands

Average	Clear	Copy	Count
Delete	ErrorToString	Fill	First
Insert	Last	LastError	Load
Max	Min	Monitor	NonZero
Port	Read	Replace	Save
Stack	Sum	Write	

Average Returns the average (mean).

Option 1

-1 From specified dimension index to the end of the array

> 1 From specified dimension index to "count" elements

Examples:

```
values:=UArray('Pump1', 'Average', [5,27,0], [-1]);
values:=UArray('Pump2', 'Average', [0], [5]);
```

Notes:

1) If no error, the average result will be in the second index (e.g. values[2]).
2) When the values are summed the internal value could roll-over.
3) The result is a floating point value. If the array type is integer (64 bit) the result type is integer (64 bit). If the array type is unsigned integer (64 bit) the result type is unsigned integer (64 bit).

Clear Clears the array contents.

No options

Example:

```
values:=UArray('Pump1', 'Clear', [], []);
```

Copy Copies an array or portion of an array.
The source and destination array can be the same.

The array name field is the source array.
The dimension field is the source array dimension start.

<u>Option 1</u>
Array name Destination array name

<u>Option 2</u>
Dimension Destination array dimension start

<u>Option 3</u>
Count Number of elements to copy

Examples:
values:=UArray('Pump1', 'Copy', [5,27,0], ['Pump1',[6,27,0], 50]);
values:=UArray('Pump1', 'Copy', [-1], ['Pump2',[-1],-1]); //copy complete array

Notes:
1) The source and destination array must be the same data type.
2) If the source array dimension field is "-1" the beginning of the array will be used. The same applies to the destination array dimension field.
3) If the count is -1, the count will be length of the source array.

Count Returns the array element count.

No options

Example:
value:=UArray('Pump1', 'Count', [], []);

If no error, the count will be in the second index (e.g. values[2]).

Delete Delete removes the specified elements and moves "up" the elements after the deleted elements. The count of array elements is not altered.

Option 1
The number of elements to delete.
> 0 From specified dimension index to "count" elements

Examples:
values:=UArray('Pump1', 'Delete', [5], [-1]); //from dimension 5 to end
values:=UArray('Vat1', 'Delete', [2,2,3], [4]); //beginning at dimension 2,2,3, delete 5 elements

Notes:
1) Deleting beyond the end of an array dimension will succeed if the delete is not past the end of the array storage.
2) To delete from X (dimension position) to the end of the array use the fill command.

ErrorToString Returns the error code description.

Any valid array name can be used.
The error code is passed in the dimension property.

Example:
values:=UArray('Pump1', 'ErrorToString', [0], [-5]);
If no error, the description will be in the second index (e.g. values[2]).

Insert Insert X values into the array at the starting dimension and move elements "down" after the inserted elements. The count of array elements is not altered.

Option 1
[array] Elements to insert

Examples:
values:=UArray('Pump1', 'Insert', [5], [77,123]);
values:=UArray('Vat1', 'Insert', [2,2,3], ['Full', 'Low', 'Empty']);

Notes:
1) For each element inserted the element contents on the end of the array are lost.
2) Inserting beyond the end of an array dimension will succeed if the insert is not past the end of the array storage.

Fill Fills the array with a value.

Option 1
The value is the number of elements to fill.
-1 From specified dimension index to the end of the array
> 0 From specified dimension index to "count" elements

Option 2
The value to write to the elements(s). For a string the value must be in quotes.

Examples:
```
values:=UArray('Pump1', 'Fill', [0,4], [-1,123]);        //fill the complete array with 123
values:=UArray('Vat1', 'Fill', [4,1], [-1, 'Some text');  //fill the complete array with 'Some text'
```

Notes:
1) Filling beyond the end of an array dimension will succeed if the fill is not past the end of the array storage.
2) The dimensions count must match the array configuration.
3) The data value must be in range for the array data type or an error might occur.

First/Last Searches the array.
First searches from the specified dimension with increasing indexes. (beginning to end)
Last searches from the specified dimension with decreasing indexes. (end to beginning) Note: For the "Last" search command, if the dimension variable is the beginning of the array the search begins at the end of the array.

Option 1 (for all data types)
The value is the number of elements to check for comparison.
-1 From specified dimension index to the end or beginning of the array
> 0 From specified dimension index to "count" elements

Option 2 (for all data types)
The value for comparison.

Option 3
Floating point type

A symbol for the search type.
'>' The first occurrence greater than option 2.
'< ' The first occurrence less than option 2.

'=' The first occurrence equal to option 2.
'<>' The first occurrence not equal to option 2.

Examples:
values:=UArray('Pump1', 'First', [0], [-1, 12.34, '>']); //first occurrence greater than 12.34
values:=UArray('Pump1', 'Last', [99], [-1, 78, '']); //first occurrence (reverse) equal to 78

Note:
Floating point "=" (equal) and "<>" (not equal) use the "difference between" calculation for equality testing.

String type

The search for strings is an equality search.

Character case comparison
True The character case must match, case sensitive.
False The character case is ignored, case insensitive.

Examples:
values:=UArray('Pump1', 'First', [0], [-1, 'Full', True]);//first occurrence of 'Full', match case
values:=UArray('Pump1', 'Last', [99], [-1 ,'A', False]);//first occurrence (reverse) 'A', ignore case

All other data types

A symbol for the search type.
'>' The first occurrence greater than option 2.
'<' The first occurrence less than option 2.
'=' The first occurrence equal to option 2.
'<>' The first occurrence not equal to option 2.

Examples:
values:=UArray('Pump1', 'First', [0], [-1, 12, '>']); //first occurrence greater than 12
values:=UArray('Pump1', 'Last', [99], [-1, 78, '']); //first occurrence (reverse) equal to 78

Result
If no error, the format is:
values[0] 0 (zero)
values[1] count elements to follow
values[2] 1-3 dimension length
values[3..n] Z, Y, X
values[n+1] value match of comparison

Example, single dimension array:
values[0] = 0, no error
values[1] = 3, elements to follow
values[2] = 1, dimension count
values[3] = 32, X
values[4] = value match of comparison

Example, two dimension array:
values[0] = 0, no error
values[1] = 4, elements to follow
values[2] = 2, dimension count
values[3] = 1, Y
values[4] = 32, X
values[5] = value match of comparison

If a comparison match is not found:
values[0] = -12, First/last no match found
values[1] = 0, no elements follow

LastError Returns the last error code. 0 = no error
The error code is set to 0 after this call.

No options

Example:
```
value:=UArray('Pump1', 'LastError', [], []);
```

If no error, the count will be in the second index (e.g. values[2])).

Load Load the specified CSV (comma separated values) file data into the array.

Option 1

<file name> The file name can be a fully qualified path or only a file name. If the string is only a file name, the project path and the file name will be used to locate the file.

Examples:

```
values:=UArray('Pump1', 'Load', [0], ['<some file name>']);          //project path will be used
values:=UArray('Pump2', 'Load', [0], [<' full path and file name'>]); //full path
```

Result

values[0] = 0, code error
values[1] = 2, elements to follow
values[2] = 0, lines loaded from file
values[3] = 0, line with error

If an error is detected on a line the load function stops and the offending line number is returned in the result.

Notes:

1) Each line must be formatted as [<dimension>],<value>. e.g. [1,2,3],77.3
2) The file can contain a portion of the array elements. Only those lines with a valid dimension and data value will be imported.
3) For string data type arrays the value can be blank. For all other array types a value must be specified.

Max Returns the maximum value in the array.

Option 1

-1 From specified dimension index to the end of the array
> 1 From specified dimension index to "count" elements

Examples:

```
values:=UArray('Pump1', 'Max', [5,27,0], [24]);
values:=UArray('Pump2', 'Max ', [0], [-1]);
```

If no error, the maximum value will be in the second index (e.g. values[2]).

Note:

The result data type will be the same data type as the array type.

Min Returns the minimum value in the array.

Option 1

-1 From specified dimension index to the end of the array

> 1 From specified dimension index to "count" elements

Examples:

```
values:=UArray('Pump1', 'Min', [5,27,0], [24]);
values:=UArray('Pump2', 'Min ', [0], [-1]);
```

If no error, the minimum value will be in the second index (e.g. values[2]).

Note:

The result data type will be the same data type as the array type.

Monitor Opens the array monitor window for the specified array.

No options

Example:

```
values:=UArray('Pump1', 'Monitor', [], []);
```

Note:

Multiple monitor windows of the same array can be opened.

NonZero Returns the count of elements that are not zero (<> 0).

For boolean, returns the count of true elements.
For strings, returns the count of non-blank elements.

Option 1

-1 From specified dimension index to the end of the array

> 1 From specified dimension index to "count" elements

Examples:

```
values:=UArray('Pump1', 'NonZero', [5,27,0], [24]);
values:=UArray('Pump2', ' NonZero', [0], [-1]);
```

If no error, the count will be in the second index (e.g. values[2]).

Note:

The result data type is longword.

Port This command is used for several operations related to arrays and communication ports. The HMI supports many varied communication protocols. Check the communication port information for application of an array "port" command.

Commands	**Description**
Link	Connects a port read command/result to an array.
Write	Writes array elements to the port/device.

MODBUS masters
AB DF1 SLC/Micrologix
AB Logix

Notes:
1) The result data type is integer.
2) Command supported in unlimited license.

Read Reads values from the array.

Option 1
The read option value is the number of elements to read.

-1	From specified dimension index to the end of the array
> 0	From specified dimension index to "count" elements

Examples:

```
values:=UArray('Pump1', 'Read', [0], [-1]);  //read the complete array
values:=UArray('Vat1', 'Read', [2,2,3], [4]); //read 4 elements beginning at dimension 2,2,3
```

If no error, the first value will be in the second index (e.g. values[2]).

Notes:
1) Reading beyond the end of an array dimension will succeed if the read is not past the end of the array storage.
2) The dimensions count must match the array configuration.

Replace	Searches for a matching value and replaces it with the specified value.

Option 1
The option value is the number of elements to search from the start dimension.

-1	From specified dimension index to the end of the array
> 0	From specified dimension index to "count" elements

Option 2

Value	The value to match.

Option 3

Value	The value to replace when match found.

Option 4

True/False	If true, stop searching on the first match/replace. If false, continuing searching until option 1 value.

Option 5 (Only for searching string type arrays)

True/False	If true, match case. If false, case is ignored.

Examples:

```
values:=UArray('Pump1', 'Replace', [0], [-1, 22, 0, False]);
values:=UArray('Pump2', 'Replace', [2,3,6], [7, 'Empty', 'Full', True, False]);
```

Result

values[0]	= 0, code error
values[1]	= 1, elements to follow
values[2]	= 0, number of matches found

Note:
Floating point matches use the "difference between" calculation for equality testing.

Save	Save the array to a CSV (comma separated values) file.
<u>Option 1</u> <file name>	The file name can be a fully qualified path or only a file name. If the string is only a file name, the project path and the file name will be used to locate the file.

Examples:

```
values:=UArray('Pump1', 'Save', [0,0,0], ['<some file name>']);      //project path will be used
values:=UArray('Pump2', 'Save', [0], [<' full path and file name'>]); //full path
```

<u>Result</u>

values[0] = 0, code error
values[1] = 1, elements to follow
values[2] = 0, lines saved to file

Note:
Each line will be formatted as [<dimension>],<value>. e.g. [1,2,3],77.3

Stack Implements a "stack".

Any array can be used as a stack due to the data structure/storage of array data. A "stack" is sometimes referred to as a LIFO (Last-in, first-out).

Command	**Description**
Stack	Sets the stack index to 0
Pop	Returns the last data value pushed on the stack and decrements the index
Push	Places the data value on the stack and increments the index
Count	Returns the number of stacked values

Examples:

```
values:=UArray('Pump1', 'Stack', [0], ['Stack']);        //sets stack pointer index to -1

values:=UArray('Pump1', 'Stack', [0], ['Pop']);          //returns the last value pushed
values[0]  //error code or 0 for no error
values[1]  //count elements returned, 0 if error otherwise 1
values[2]  //data if values[0] = 0 (no error)

values:=UArray('Pump1', 'Stack', [0], ['Push',<some value>]);          //place data on stack
values[0]  //error code or 0 for no error
values[1]  //count elements returned, 0 if error otherwise 1
values[2]  //new stack count if  values[0] = 0 (no error)

values:=UArray('Pump1', 'Stack', [0], ['Count']);        //returns the stack depth
values[0]  //error code or 0 for no error
values[1]  //count elements returned, 0 if error otherwise 1
values[2]  //count of values on stack if values[0] = 0 (no error)
```

Sum Returns the sum (addition).

Option 1

-1 From specified dimension index to the end of the array

> 1 From specified dimension index to "count" elements

Examples:

values:=UArray('Pump1', 'Sum', [5,27,0], [-1]);

values:=UArray('Pump2', 'Sum ', [0], [5]);

If no error, the sum result will be in the second index (e.g. values[2]).

Notes:

1) When the values are summed the internal value could roll-over.
2) The result is a floating point for floating point array type. If the array type is unsigned integer (64 bit) or longword the result type is unsigned integer (64 bit) otherwise, the result is signed integer (64 bit).

Write Writes a value or values to the array.

Option 1

The write option value(s) is the data to write to the array element(s)

1..n From 1 to N data values

Examples:

values:=UArray('Pump1', 'Write', [0], [77]); //write one value

values:=UArray('Vat1', 'Write', [2,2,4], ['High', 'Very good']); //write 2 strings beginning at dimension 2,2,4

Notes:

1) Writing beyond the end of an array dimension will succeed if the write is not past the end of the array storage.
2) The dimensions count must match the array configuration.
3) If a failure is detected (e.g. failed data type conversion), writing is halted and the failed write value index is logged to the event log.

Error codes

Value	Description
0	No error
-1	Unknown command/sub command
-2	Invalid array dimension
-3	Read/write/delete beyond array storage
-4	Array name not found
-5	Option 1 invalid or missing
-6	Option 2 invalid or missing
-7	Option 3 invalid or missing
-8	Option 4 invalid or missing
-9	Unknown write error (normally a conversion error, wrong data type)
-10	Bad array type (operation not possible for array data type)
-11	Math operation failed
-12	First/last no match found
-13	File name missing
-14	File does not exist
-15	Load error
-16	File is empty
-17	Stack bounds error
-18	Option missing
-19	Copy failed
-20	Function not supported
-21	Port name unknown
-22	Port read count odd
-23	Write count too large
-24	Illegal address or bad address for array type
-25	Read count too large
-26	Failed to create needed memory
-27	Failed to create fetch message
-28	Link already exists
-29	Option 5 invalid or missing
-30	Option 6 invalid or missing
-31	Port write buffer full
-32	License limited

UnixTime

This function is used to convert an HMI date/time to/from a Unix time (seconds since January 1, 1970).

Inputs

Variable	Type	Description
Command	Integer	0 = Unix to HMI 1 = HMI to Unix
Time	Longword Float	0 (Unix time) 1 (HMI time)

Outputs

Variable	Type	Description
Value	Variant	Longword = HMI to Unix Float = Unix to HMI

Examples

```
value:= UnixTime(0, 1582664936);   //2020-02-25 21:08:56

value:= UnixTime(1,Now);           //current date/time to Unix time
```

Related:
Function list
CrossReference

UserButtonWindow

This function is used to display a window with buttons. The window calls a script function when a button is "clicked" and the window can close or stay open. The window can be modal or non-modal.

The buttons are automatically ordered in the window based on the configured properties.

The user button window properties and functions can be accessed in the scripting IDE by typing "btnWind." (The word "btnWind" followed by a period.) Or if the "btnWind" exists, place the caret after the "." and press the space bar.

Window properties

Name	Data type	Default	Description
allClose	Boolean	True	Window will close on any button click
autoSize	Boolean	True	Button size determined by window size
cancelBtnCaption	String		If not blank, cancel button in last position
clickedBtnTag	Longword	0	Set when button clicked
orientationColumns	Boolean	False	Button order. False = column, true = row
windowBgColor	Color	Black	Window color
windowCaption	String	Button window...	Title, blank = no title bar
windowfullScreen	Boolean	False	If true, window opens maximized
windowHeight	Integer	600	Window height
windowHideCloseX	Boolean	False	True = close X in title bar disabled
windowIsModal	Boolean	False	Window modal state
windowLeft	Integer	0	Window left position *4
windowMarginBottom	Integer	3	Bottom border
windowMarginLeft	Integer	3	Left border
windowMarginRight	Integer	3	Right border
windowMarginTop	Integer	3	Top border
windowTop	Integer	0	Window top position *4
windowWidth	Integer	800	Window width

Window functions (see notes below)

Name	Data type	Default	Description
Add	String	<button caption>	Call to add a button to the window
AddBtns	Array/string	Captions/command	Adds a list of buttons to the window
AddClose	String	<button caption>	Call to add a button to the close list *1,2
Clear			Removes all buttons *6
CreateWindow	Strings	Script & function name	Create the window *3
Initialize			Clears button lists, set properties to default

When the "Add" or "AddBtns" function is called the current global button properties (color, font color, font size, etc.) are applied to the created button.

Change a global property, call the function and the new button will have the new global property setting.

After the button is created use the button property to alter the button property.
Example: btnWind.ButtonColor[<index>]:=clRed; //index begins at zero

Button properties

Name	Data type	Default	Description	Global /Per button
buttontag	Longword	0	User defined	No/Yes
color	Color	clBtnFace	Button color	Yes / Yes
fontColor	Color	clBtnText	Button caption color	Yes / Yes
fontName	String	Calibri	Font family	Yes / Yes
fontSize	Integer	12	Font size	Yes / Yes
fontStyles	Set	[]	Font style	Yes / Yes
height	Integer	40	Button height	Yes / No
marginBottom	Integer	3	Bottom border	Yes / No
marginLeft	Integer	3	Left border	Yes / No
marginRight	Integer	3	Right border	Yes / No
marginTop	Integer	3	Top border	Yes / No
width	Integer	120	Button width	Yes / No

Notes:

1) If “allClose” is true it is not required to add button(s) to this close list.
2) The caption must be identical to the string used in “Add”.
3) Script and function name for button click callback. See example below.
4) Set either property to -1 and the window will center on the main screen.
5) The command scripting engine loads callbacks from memory. Script changes, to the callback functions, in the scripting IDE are not in memory until the script is saved and reloaded. For easier troubleshooting, call the callback function in the same script passing test parameters. Contact support if assistance is needed.
6) Use this function when configuring for a new window, with existing window setting, to clear all the buttons from the list. Use “Initialize” to reset all properties to default and clear the buttons from the list.

Add

This function adds a single button to the button list and returns the button’s index in the button list.

AddBtns

This function adds the array of captions to the window. If the command is “HasScreens” each caption (assumed to be a tagname) is tested for usage on at least one graphic screen and if true the button is added to the window. The result is the count of buttons added via the call.
Note: The first array index **must** be the count of captions in the array.

Format: value:=btwWind.AddBtns([<array of strings>] ,'<Command>'); //first string must be count value

Only the captions (tagnames) used on a graphic screen are added to the window.

```
value:=btwWind.AddBtns([2,'Pmp_1','Pmp_2'] ,'HasScreens');
```

All the captions are added to the window.

```
value:=btwWind.AddBtns([2, 'Pmp_1','Pmp_2'] ,'');
```

Example:

This complete script example would be in a script file, at the root of the project/script directory, named: **CRefScreen.psc.**

```
//the main function is the last section

//-------------------------------------------------------------
//when a tag button is selected this function is called
procedure TagSelected(caption,cancelCaption:string);
begin
 if (caption = '') then
  Exit;
 if (caption = cancelCaption) then
```

```
  Exit;

//get screens containing tagname
 values:=CrossReference('Screens',[caption]); //caption is the tagname
 if (values[0] < 1) then
  begin
   Beep;
   Exit;
  end;

 btnWind.Clear;                                //remove all buttons
 btnWind.windowCaption:='Select a graphic screen...';    //title bar

 for i:= 1 to values[0] do
  index:=btnWind.Add(values[i]);

 btnWind.cancelBtnCaption:='Cancel';        //add cancel button
 btnWind.CreateWindow('CRefScreen.psc','ScreenSelected');
end;
//-------------------------------------------------------

//-------------------------------------------------------
//when a screen button is selected this function is called
procedure ScreenSelected(caption,cancelCaption:string);
begin
 if (caption = '') then
  Exit;
 if (caption = cancelCaption) then
  Exit;

 OpenWindow(caption);
end;
//-------------------------------------------------------

//-------------------------------------------------------
//main function
var
 i:integer;
begin
 values:=CrossReference('Tagnames',[]);   //get all the tags
 if (values[0] < 1) then
  begin
   Beep;
   Exit;
  end;

 btnWind.Initialize;           //set to defaults
 btnWind.windowTop:=-1;        //center on main screen
 btnWind.windowCaption:='';    //no title bar

 for i:= 1 to values[0] do
```

```
  btnWind.Add(values[i]);

 btnWind.cancelBtnCaption:='Cancel';         //add cancel button
 btnWind.CreateWindow('CRefScreen.psc','TagSelected');
end;
//-------------------------------------------------------
```

Related:
Function list
CrossReference

UserChangePassword

This function is used to change the logged on user's password.

Inputs

Variable	Type	Description
Log result	Boolean	Log if the password was changed

Outputs

None

Example

UserChangePassword(True);

Related:
Function list

UserConfigurationEditor

This function is used to edit "User" configurations. **Note:** The logged on user must be configured to allow configuration changes.

Inputs

Variable	Type	Description
Log result	Boolean	Log if the "OK" or "Cancel" button was selected.

Outputs

None

Example

UserConfigurationEditor(True);

Related:
Function list

Valve2Input

This function is used to calculate one of four states from two inputs. The most common use would be to determine the "state" of a valve. This function generates an integer result based on the values of two digital inputs

Inputs

Variable	Type	Description
Tagname 1	String	Name of the tag
Item number 1	Integer	Data item ID
Tagname 2	String	Name of the tag
Item number 2	Integer	Data item ID

Outputs

Variable	Type	Description
Value	Integer	Result of truth table

The input values must be Boolean as defined by the item number. All inputs must be present.

Truth table

Input 1	Input 2	Result
True	True	1
False	False	2
False	True	3
True	False	4

Example

value:=Valve2Input('valveOpened', 5007,'valveClosed', 5007);

In this example:

If value = 1 the opened and closed inputs are both true and a valve fault condition.
If value = 2 the opened and closed inputs are both false and a valve travel condition.
If value = 3 the opened input is false and the closed input is true and a valve closed condition.
If value = 4 the opened input is true and the closed input is false and a valve opened condition.

Related:
Function list

ViewDateRangeTrendHistory

This function is used to display a trend in a window for a range of dates. This command is only for the horizontal trend.

Inputs

Variable	Type	Description
Show modal	Boolean	If true the window will be modal.
Start date	TDateTime	The starting trend date. If zero (0), the start date will be the current date minus 30 days.
End date	TDateTime	The ending trend date. If zero (0), the end date will be the current date minus 1 day.
Tagname	String	Tagname of data point
Item number	Integer	Item of data point
Trend settings file	String	The optional trend settings file name. (The path is the project, this is only a file name).

Outputs

Variable	Type	Description
None		

Notes:
1) Use this command with caution. If the amount of data to load, ((data logging frequency X number of days) X number of data points) is large the program/computer may use up all available memory.
2) Displaying the current day can give partial results. The data logger has a buffer and does not write the data to disk until required. It is advised not to use this feature to display the data of a point for the current day. Use a trend graphic element.

Examples

Display the trend for two analog points.

ViewDateRangeTrendHistory(True, 0, 0, ['Ch_1', 5000, 'Ch_2', 5000], '');

Display the trend for two analog points and one digital.

ViewDateRangeTrendHistory(True,0,0,['Ch_1',5000,'DM1',5000,'V1_FOpen',5007],'');

Example Pascal
A script that allows a user to select the start date and end date and displays three points for the selected days. The script can be called, for example, from a button with the On Mouse Up command using the "Queue script" command.

```
var
 startDT,endDT,aDT:TDateTime;
begin

//ask the user for the start date
 startValues:=GetUserInputDate('Select the start date…', 'Accept', 'Cancel',32768 ,0);
 if (startValues[0] = 0) then    //if zero the user selected cancel
  Exit;

//ask the user for the end date
 endValues:=GetUserInputDate('Select the end date…', 'Accept', 'Cancel',32768 ,0);
 if (endValues[0] = 0) then    //if zero the user selected cancel
  Exit;

 startDT:=EncodeDate(startValues[2],startValues[1],startValues[0]);    //year, month, day
 endDT:=EncodeDate(endValues[2],endValues[1],endValues[0]);        //year, month, day

 aDT:=Date;
 if (aDT = startDT) or (aDT = endDT) then
  begin
   ShowMessage('The end and/or start date is today and is not allowed. The date has been ' +
              'adjusted to the day before today.');
   if (aDT = startDT) then
    startDT:=IncDay(startDT,-1);
   if (aDT = endDT) then
    endDT:=IncDay(endDT,-1);
  end;

 if (endDT < startDT) then
  begin
   ShowMessage('The end date is before the start date.');
   Exit;
  end;

//if here we are good to go
 ViewDateRangeTrendHistory(True,startDT,endDT,['Ch_1',5000,'DM1',5000,'V1_FOpen',5007],'');

end;
```

Example Basic
A script that allows a user to select the number of days from yesterday to view. The script can be called, for example, from a button with the On Mouse Up command using the "Queue script" command.

```
Dim startDT,endDT,values

'ask how many days back, from yesterday
values=GetUserInputInteger("Enter previous days…", "Accept",
"Cancel",2,365)
if (values[0] <> 0) then   'zero the user selected cancel
 endDT=IncDay(Now,-1)
 startDT=IncDay(endDT,-values[1])
 'if here we are good to go
 ViewDateRangeTrendHistory(True,startDT,endDT,["SOC_Average",5000],"")
end if
```

Related:
Function list

WindowPosition

This function provides a method to get, set or center an open window position.

Inputs

Variable	Type	Description
Command	String	Get, Set or Center
Window name	String	Name of window to get or set
Properties	Array	Left, top, width, height

Outputs

Variable	Type	Description
Array(5)	Variant	The first element is always a boolean indicating success or failure of the command.

Examples

```
values:=WindowPosition('Get', <window name>, 0,0,0,0);
//the four 0 properties are not used and any value can be used

if values[0] then  //if true the window is open and the four array values are valid
 begin
  // values[1] = left
  // values[2] = top
  // values[3] = width
  // values[4] = height
 end;

values:=WindowPosition('Set', <window name>, 10,20,-1,500);
//10 = left
//20 = top
//-1 = width, if the value is -1 the width is not set, any other value and the width is set
//500 = height, if the value is -1 the height is not set, any other value and the height is set
if values[0] then
 ;                    //success
```

```
values:=WindowPosition('Center', '<window name to move>', '<window name to center on>'
,0,0,0);
if values[0] then
 ;                    //success
```

Related:

Function list

WindowRR (Window Round Robin)

This function provides a method to automatically open/close an ordered list of windows. A script global hive is used for configuration and status information. **Note:** The "Pulse" command must be called at regular intervals for the logic to function. A simple example project has been posted on the web.

Inputs

Variable	Type	Description
Script global	String	Script global section name
Command	Array	An array of command information

Outputs

Variable	Type	Description
None		

Command	Description
Pulse	Execute internal logic *1
Start	Start the rotation logic at the first window
Stop	Stop all rotation logic
Pause	Pause the rotation logic at the current window
Resume	Resume the rotation logic at the last window
ForceRotation	Open next window and close current window
PreviousRotation	Open previous window and close current window

Script global items	Description
DefaultCount	The count time for the window to be open *2
WindowName0..N	The name of the window. (WindowName511 max)
WindowTime0..N	(Optional) Configured window open time *2
ActiveWindowName	The name of the open window in the list
WindowIndex	The N value of the open window
RemaindingCount	The count until the next window is opened
RotationState	Inactive (0), Active (1), Paused (2)
ErrorString	If an error is detected, it will be stored here *3

Notes:

1. The pulse command must be called at a regular interval. The simplest method is to create a script timer and a single line script containing the pulse command. e.g. WindowRR('<script global section>',['Pulse']);

   The pulse command decrements the counter, when needed. If the window open time is set to five (5) and the pulse command is called once per second the window cycle time will be about five seconds. If the pulse command is called every two (2) seconds the cycle time will be about ten (10) seconds and so on.

2. If a WindowTimeN is not defined for a window the DefaultCount will be used. If the DefaultCount is not defined, a count of thirty (30) will be used.

3. The error string will be overwritten for each error. To clear the error string use The "GlobalSet" function. e.g. value:=GlobalSet('<section name>','ErrorString','');

Examples

```
WindowRR('<section name>',['Start']);
WindowRR('<section name>',['Resume']);
WindowRR('<section name>',['PreviousRotation']);
```

Related:
Function list

WriteMultiple

This function is used to write multiple values, in one command, to an external device. This command does not write to the internal database.

Inputs

Variable	Type	Description
Port name	String	Port name
Address	String	Starting address
Data type	Integer	The data type 0 = 32 bit float 1 = 32 bit signed integer 2 = 32 bit unsigned integer
Values Array	Array of values to write	

Outputs

Variable	Type	Description
None		

Note: The only communications port supporting this command is the "Delta Motion Control" (DMCP). Contact support if another communications port is needed. Not all communication protocols provide a function to write multiple locations in one command.

The number/count/limit of "Values" that can be written in one command is defined by the communication protocol. If the count exceeds that limit the write will not occur.

<u>DMCP</u>
TCP/IP = 1024
UDP/IP = 256

The values must all be of the same type. Values are converted to match the "Data type".

Examples of automatic conversion

#1
WriteMultiple('Arm 1', '58.26', 1, [1, 56 ,-89 , 58.7]); //Data type = 1 (signed integer), start address 58.26, values

Value 1, 56 and -89 will be unchanged. The value 58.7, a float, will be converted to 59.

#2
WriteMultiple('Arm 1', '58.26', 2, [1, 56, -89, 58.2]); //Data type = 2 (unsigned integer), start address 58.26, values

Value 1 and 56 will be unchanged. The value -89, a signed integer, will be converted to 89 and the value 58.2, a float, will be converted to 58.

Examples

WriteMultiple('Arm 1', '58.26', 0, [1.56,-23.76, 5.0]);
WriteMultiple('Arm 1', '58.26', 1, [1,-23, 5]);
WriteMultiple('Arm 1', '58.26', 2, [1,23, 5]);

Related:
Function list
WriteValue
WriteValuePulse

WriteValue

This function is used to write a value or values to the runtime database. The WV function might be more suitable if reading the default point items. The WV function is faster.

Inputs

Variable	Type	Description
Tagname	String	Tagname to modify
Item number	Integer	Item to modify
Item value	Variant	Value to assign to tagname/ID

Outputs

Variable	Type	Description
None		

The values can all be of the same type (i.e. Boolean, integer) or they can be of any type as defined by the item number.

Scaling
If the tag type is analog and the "scaled enabled" is true the value is interpreted as in "engineering units" and the value is scaled to "instrument range" before output to the connected device. If the "scaled enabled" is not true the value is interpreted as in "instrument range" and the value is not converted before it is output to the connected device.

Read/Write
If the access rights is read/write the value is assigned to the internal stored value location and any other processing is preformed just as if the value had come from the connected devices. The value is then transmitted to the connected device. The next time the connected device supplies a new value the internal stored value is replaced. The time between the "WriteValue" command and the next refresh from the external device may be very short or very long.

The items numbers most used would be:

Item	Name
5000	Process Variable Analog (float)
5007	Process Variable Digital (digital)
5008	Percent of Full Scale (float)
5009	Analog Host Pointer Index (float)

Examples

WriteValue(['tagname 1', 5007, 1]);

If a need to write more than one value, combine the writes into one write. It will be faster than executing multiple writes.

WriteValue (['tagname 1', 5007, 1, 'tagname 2', 5000, 25]);

Related:
Function list
ReadValue
RV
Result
WV
WriteValuePulse

WriteValuePulse

This function is used to write a value to the runtime database and then write the inverted value after a defined time has elapsed.

Inputs

Variable	Type	Description
Tagname	String	Tagname to modify
Item number	Integer	Item to modify
Initial value	Boolean	Value to assign to tagname/ID
Time delay	Integer	Time delay in seconds

Outputs

Variable	Type	Description
None		

Read/Write
If the access rights is read/write the initial value is assigned to the internal stored value location and any other processing is preformed just as if the value had come from the connected devices.

The initial value is then transmitted to the connected device.

The next time the connected device supplies a new value the internal stored value is replaced. The time between the "WriteValue" command and the next refresh from the external device may be very short or very long.

After the time has elapsed, if the access rights is read/write the inverted initial value is assigned to the internal stored value location and any other processing is preformed just as if the value had come from the connected devices.

The inverted initial value is then transmitted to the connected device.

The next time the connected device supplies a new value the internal stored value is replaced. The time between the second "WriteValue" command and the next refresh from the external device may be very short or very long.

Write
If the access rights is write, the initial value is transmitted to the external device. After the elapsed time the inverted initial value is written to the external device.

The items numbers most used would be:

5007 Process Variable Digital (digital)

Example

WriteValuePulse('tagname 1', 5007, 1,10);

Related:
Function list
ReadValue
Result
WriteValue

WV

This function is used to write a point value or values to the runtime database using the default point item number. Use WriteValue to write any other point item (that is writable) value

Default item numbers

Type	Item number	Description
Analog	5000	Process variable analog (float)
Digital	5007	Process variable digital (digital)

Inputs

Variable	Type	Description
Tagname	String	Tagname to modify
Item value	Variant	Value to assign to point (default value)

Outputs

Variable	Type	Description
None		

The values can all be of the same type (i.e. Boolean, integer) or mixed types.

Scaling
If the tag type is analog and the "scaled enabled" is true the value is interpreted as in "engineering units" and the value is scaled to "instrument range" before output to the connected device. If the "scaled enabled" is not true the value is interpreted as in "instrument range" and the value is not converted before it is output to the connected device.

Read/Write
If the access rights is read/write the value is assigned to the internal stored value location and any other processing is preformed just as if the value had come from the connected devices. The value is then transmitted to the connected device. The next time the connected device supplies a new value the internal stored value is replaced. The time between the "WriteValue" command and the next refresh from the external device may be very short or very long.

Examples

```
WV(['tagname 1', 1]);
```

If a need to write more than one value, combine the writes into one write. It will be faster than executing multiple writes.

```
WV(['tagname 1', 1, 'tagname 2', 25]);
```

Related:
Function list
ReadValue
Result
RV
WriteValue
WriteValuePulse

In addition to the above functions are the standard language functions. For help on any of these functions please contact support.

Abs
AnsiCompareStr
AnsiCompareText
AnsiLowerCase
AnsiUpperCase
Append
ArcTan
Assigned
AssignFile
Beep
Chdir
Chr
CloseFile
CompareStr
CompareText
Copy
Cos
CreateDir
CreateFile
CreateOleObject
Date
DateTimeToStr
DateToStr
DayOfWeek
Dec
DecodeDate
DecodeTime
Delete
DeleteFile
DirectoryExists
EncodeDate
EncodeDateTime
EncodeTime
EOF
Exp
FileCopy
FileExists
FilePos
FileRename
FileSize
FloatToStr
Format
FormatDateTime
FormatFloat
Frac
GetActiveOleObject
High
Inc
IncDay
IncHour
IncMilliseconds
IncMonth
IncSecond
InputQuery
Insert
Int
IntToHex
IntToStr
IsLeapYear
IsNumeric
IsValidIdent
Length
Ln
Low
LowerCase
Now
Odd
Ord

Pos
Raise
Random
RandomRange
ReadLn
Reset
Rewrite
Round
Scripter
SetLength
SetOf
ShowMessage
Sin
Sqr
Sqrt
StrToDate
StrToDateTime
StrToFloat
StrToInt
StrToIntDef
StrToInt64
StrToUInt64
StrToTime
Time
TimeToStr
Trim
TrimLeft
TrimRight
Trunc
UpperCase
VarArrayCreate
VarArrayHighBound
VarArrayLowBound
VarIsNull
VarToStr
Write
WriteLn

POINT SCRIPTS

WARNING

The scripts in points should only call the following commands:

GetSystemVariable
Result
ReadValue
WriteValue
WriteValuePulse

WriteValue and WriteValuePulse should not be used to write to points in external devices from a point script. The only points written to in a point script must be internal points in the HMI-Host Port.

Calling the other script commands or writing to external points will most likely result in unstable operation or complete failure.

The runtime script editor provides for editing and debugging scripts.

WARNING

Be aware! When executing/running/debugging scripts in the runtime editor, it may conflict with in memory script results.

When compiling a script at runtime the program writes to disk any changes made to the script. If testing/debugging make copies of the script before making changes and make changes to the copy. Once satisfied, copy the changes to original script as needed.

Each example is separated by a dashed (----------) line.

This was an analog host point script. The user had a tank level gauge that reported a range of 0 - 111 inches. The output was in gallons.

```
const
cRadius = 50.0;
cRadius2 = cRadius * cRadius;
cTankFactor = 0.0135;

begin
// replace the ExternalTankLevel with the tagname from the real tank level tag.
values:=ReadValue(['ExternalTankLevel', 5000]);
result:=values[0] * cRadius2 * cTankFactor; //tank level in inches * 50 * 50 * 0.0135
end;
```

The user wanted to toggle a bit. The 'ToggleBoolean' mouse command is a better choice if the command is via the user clicking the mouse in a graphic element.

```
values:=ReadValue(['C100', 5007]);
if values[0] then
WriteValue(['C100', 5007, 0]) // it is on, turn it off
else
WriteValue(['C100', 5007, 1]); // it is off, turn it on
```

The input word from a MODBUS register is 0 - 65535. We need that as a "small integer" -32768 - 32767. (After this was published we added support for small integers to the point definition.)

```
values:=ReadValue(['Power Factor', 5000]);
//(raw in - raw low) * ((eu high - eu low) / (raw high - raw low)) + eu low
```

```
result:= values[0] * ((32767- -32768) / 65535) + -32768;
```

This script was executed via a task command when a user logged in. It opened a different window for each user.

```
values:=GetSystemVariable(11); //get the user name

if (values[0] = '' ) then //no one logged in
Exit;

//Uppercase just to prevent possible case typos
case UpperCase(values[0]) of
UpperCase('Bob'):
begin
CloseAllUserWindows;
OpenWindow('Main');
end;
UpperCase('Sue'):
begin
CloseAllUserWindows;
OpenWindow('Money');
end;
UpperCase('Frank'):
begin
CloseAllUserWindows;
OpenWindow('Taste');
end;
UpperCase('Joe'):
begin
CloseAllUserWindows;
OpenWindow('Sales');
end;
end;
```

This is a script to append some values to a text file.

```
const
fileName = 'C:\test.txt';
var
FFile:TextFile;
value1,value2,value3,finalString:string;
begin
AssignFile(FFile,fileName); //get a file handle
Append(FFile); //open the file, the file must exist

//the strings to write are stored in script globals. Either populate the script globals
//earlier in this script or some other script or method.
value1:=GlobalGet('fileWrite','string1') + ',';
value2:=GlobalGet('fileWrite','string2') + ',';
value3:=GlobalGet('fileWrite','string3');

finalString:=value1 + value2 + value3;

WriteLn(FFile,finalString); //write the string
FFile.Free;                 //closes the file and releases the memory
end;
```

The input value was in GPM and it was needed as GPS (gallons per second).
This is a point script.

```
values:=GetSystemVariable(1);
currentSecond:=values[2];
lastSecond:=GlobalGet('Flow','lastSecond');

if (currentSecond = lastSecond) then //we have a new second
Exit; //no so leave, nothing to do
```

```
value:=GlobalSet('Flow','lastSecond',currentSecond); //save it

values:=ReadValue(['GPM',5000]); //get the gpm
if (values[0] > 0) then //if the gpm is > 0 then do the calc
begin
overflow:=GlobalGet('Flow','overflow'); //from last time
totalClicks:=GlobalGet('Flow','total'); //total 100 gal clicks

gps:=values[0] / 60; //the gallons per second
overflow:=overflow + gps; //add to the existing overflow

if (overflow >= 100) then //less than 100 nothing to do
begin
remainder:=overflow mod 100; //how much to carry over
toClick:=trunc(overflow / 100); //how many clicks this pulse
totalClicks:=totalClicks + toClick; //the total 100x clicks

if (totalClicks > 1000000) then
totalClicks:=0;

value:=GlobalSet('Flow','overflow',remainder); //save for next run
end
else
value:=GlobalSet('Flow','overflow',overflow); //save for next run

value:=GlobalSet('Flow','total',totalClicks); //save the total clicks
result:=totalClicks;
end;
```

This script was in the startup script to clear the values.

```
value:=GlobalSet('Flow','overflow','0'); //from last run, clear it
value:=GlobalGet('Flow','total'); //load last total
WriteValue(['GPMScaledTo100clicks',5000,value]); //write to point
```

This was a script to read the value of a 'knob' graphic element when it changed and set 3 boolean values in the PLC.

```
values:=ReadValue(['KnobValue',5000]);
case values[0] of
0:
begin
WriteValue(['Off point',5007,1]); //off - on
WriteValue(['On point',5007,0]); //on - off
WriteValue(['Auto point',5007,0]); //auto - off
end;
1:
begin
WriteValue(['Off point',5007,0]); //off - off
WriteValue(['On point',5007,1]); //on - on
WriteValue(['Auto point',5007,0]); //auto - off
end;
2:
begin
WriteValue(['Off point',5007,0]); //off - off
WriteValue(['On point',5007,0]); //on - off
WriteValue(['Auto point',5007,1]); //auto - on
end;
end;
```

This example collects the IP addresses of the computer and stores the result in a script global.

```
var
i:integer;
s1:string;
begin
values:=GetSystemVariable(15); //get the IP addresses
if (values[0] = 0) then //might not have any IP addresses
 begin
  GlobalSet('IP_Addresses','All','No IP addresses');
  Exit;
 end;

//if here we have at least one address
s1:=''
for i:= 1 to values[0] do
 begin
  s1:= s1 + values[i];                    //add the next IP address to the string
  if (i < values[0]) then        //if we have more add a carriage return
  s1:= s1 + #13;
 end;
GlobalSet('IP_Addresses','All',s1); //save the string
end;
```

This example asks a user for a string value, pads the string to be 20 characters long (if length less than 20 characters) and saves the value in a script global. The padded character is a “space” character.

```
function PadString20(s1:string):string;
const
 pads = '                    ';   //20 space characters
var
 i:integer;
begin
 result:=s1;                  //default is return the string unaltered
 i:=length(s1);               //how long is the string
 if (i < 20) then             //less than 20, then pad the end
  result:=s1 + Copy(pads,1,20 - i); //append x spaces to the end of the string
end;

begin
 values:=GetUserInputString('Enter string...','', 'Accept', 'Cancel',true,0,'','');
 if not values[0] then
  Exit;              //user selected cancel if here
 value:=GlobalSet('test1','answer',PadString20(values[1]));
end;
```

This example is for a “jog” type button. “On Mouse Down” in the button (or graphic element), a command is sent to the external device and “On Mouse Up” the command is cleared. If the user drags the mouse pointer off the button, the “On Mouse Up” event is not fired. This is the normal action of buttons.

This script sends the “On Mouse Up” command if the button is released with the mouse pointer over the button or the mouse pointer is moved outside the button bounds. This script is in button/graphic element.

```
function IsMouseInButton:integer;
var
 x,y:integer;
begin
 x:=ge.MouseXX;
 if (x < 0) or (x > ge.Width) then
  result:=false
 else
  begin
   y:=ge.MouseYY;
   if (y < 0) or (y > ge.Height) then
    result:=false;
   else
    result:=true;
  end;
end;

procedure OnMouseDown;
begin
 WV(['MouseDownFlag',true]);
//set jog on
end;

procedure OnMouseUp;
begin
 WV(['MouseDownFlag',false]);
//set jog off
end;
```

```
begin
 values:=RV(['MouseDownFlag']);
 if values[0] then   //mouse down or was down
  begin
   if not IsMouseInButton then
    OnMouseUp;
  end;
end;
```

GRAPHIC ELEMENT SCRIPTS

Function list

Notes:
1) If the hide or on/off animation is enabled and the condition to hide the graphic element is true, the script will not execute.
2) If the script has a compile error or a fatal error during execution (i.e. divide by zero), the graphic element foreground color will be set to the "Quality bad color". This does not apply to OnMouseDown/OnMouseUp procedures.
3) The script animation is the last animation processed before the graphic element is rendered. For example, if an animation sets the foreground color to red and the script sets the foreground color to blue, the graphic element will be rendered with a foreground color of blue. Note 1 above is the exception.

Graphic script structure

The script structure as defined in SCRIPT STRUCTURE applies. This description covers how it applies to graphic scripts and scripts with mouse action processing. A graphic script can have any number of functions/procedures. It can also have procedures for mouse commands and other mouse actions.

OnMouseDown	When the left mouse button is pressed in the graphic element. Note: Not all graphic elements support graphic scripting mouse commands.
OnMouseUp	When the left mouse button is pressed in the graphic element and released while the mouse pointer is still above the graphic element.
OnAccept	When the "Accept" button is pressed in an "Edit Field" that has focus.
OnCancel	When the "Cancel" button is pressed in an "Edit Field" that has focus.
OnTreeviewClick	When the left mouse button is pressed in a treeview graphic element item.
OnCalculatorButtonClick	When the left mouse button is clicked in a calculator button.
OnRecipeClick	When the left mouse button is clicked in a recipe grid cell.
OnRecipeValidate	When the user ends editing a recipe grid cell.

If one of the procedures is used, the following script structure must be followed.

Case 1, no OnMouseXX commands.
The structure is the same as defined in SCRIPT STRUCTURE.

Case 2, only mouse down command.

```
procedure OnMouseDown;
begin
  some code
end;
```

Case 3, mouse down and mouse up command.

```
procedure OnMouseDown;
begin
 some code
end;

procedure OnMouseUp;
begin
 some code
end;
```

Case 4, mouse down, mouse up command and some graphic display code.

```
procedure OnMouseDown;
begin
  some code
end;

procedure OnMouseUp;
begin
 some code
end;

begin
 Graphic display code here.
end;
```

OnMouseDown/OnMouseUp

The “MouseResult” property can be used to alter the flow of mouse clicks.
Before the **OnMouseDown** procedure is called the “MouseResult” is set to 3 (Normal success). Other mouse down actions will not execute. (i.e. mouse down command) The OnMouseUp procedure will be called, if required.

If the “MouseResult” is set to 1 (cancel) in the OnMouseDown procedure the mouse click processing will stop and the normal mouse up command process will not occur.

If the “MouseResult” is set to 2 (continue search) in the OnMouseDown procedure the mouse down processing will continue searching for a graphic element to handle the mouse down event. The OnMouseUp procedure will not be called if another graphic element is found to process the mouse down event. The allows for layering graphic elements.

If the “MouseResult” is set to 4, continue mouse down actions. (i.e. mouse commands, if configured will execute)

Any value other than zero to four (0-4) will default to 3.

The required user level is set in the “Graphic element script editor...”, “Edit/User level...” menu item.

The graphic element script engine uses a subset of the scripting engine commands for graphic processing. The OnMouseDown/OnMouseUp procedures can utilize all the script commands.

Add
AddDigital
AllTrue
AnyTrue
Average
DigitalCompare
GetPortCounters
GetSystemVariable
GlobalGet
GlobalSet
IsUserWindowOpen
MousePosition
ReadValue
StringGet
TimerGet
Valve2Input

The graphic element (ge) fields are exposed via the ge.<property name>.

The fields are the same that are automatically modified via the animations. Not all graphic elements utilize all fields. For example, the text fields of the 'ge' do not apply to a plain rectangle.

These are the field/property names.

BackgroundColor	Bottom	BrushStyle	Column
EndAngle	FontName	FontSize	FontStyle
ForegroundColor	Height	Hidden	HintAlignment
HintBorderColor	HintBorderWidth	HintColor	HintDisabled
HintFontColor	HintFontName	HintFontSize	HintFontStyle
HintHeight	HintLeft	HintMarginBottom	HintMarginLeft
HintMarginRight	HintMarginTop	HintShape	HintText
HintTop	HintTranparent	HintVisible	HintWidth
Left	MouseButton*3	MouseResult	MouseX
MouseXX*2	MouseY	MouseYY*2	Opacity
PenColor	PenStyle	PenWidth	Right
Rotation	Row	StartAngle	Text
TextAlignment	Top	Value*4	ValueMax
ValueMin	Width	WindowName	

Notes:

1. ValueMax and ValueMin are used for gauge ranges. **Note:** Changing these values at runtime may cause the gauge to appear distorted because other design time settings control display features of the gauge.

2. MouseX and MouseY are in screen coordinates and are updated on “mouse down” in the graphic element. MouseXX/MouseYY are in local coordinates (graphic element).

3. 1 = left button, 2 = middle button, 3 = right button, 0 = unknown

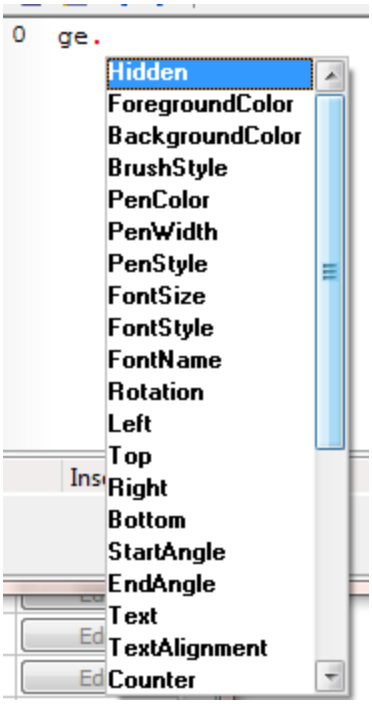
4. Sets the value of a “gauge” element.

The fields are accessed using ge.<property name>

For example:

```
ge.Visible:=false;                    //hide the graphic element
ge.ForegroundColor:=clBlue;           //set the foreground color to blue
ge.ForegroundColor:=125479;           //set the foreground color to a custom
color
```

Type “ge.” in the script editor and a properties selection list will appear.



Colors

In the script editor the predefined colors can be accessed by the menu "Edit/Insert color..." or by typing “cl” and then press “CTRL” + “space bar”. A pop up menu will appear.

Pen styles

The pen styles are psSolid, psDash, psDot, psDashDot, psDashDotDot and psClear. Per MS windows, if the pen width is greater than 1, the pen style is psSolid.

Brush styles

The brush styles are: bsSolid, bsClear, bsHorizontal, bsVertical, bsFDiagonal, bsBDiagonal, bsCross and bsDiagCross.

Text alignment

The text alignments are: DT_LEFT, DT_CENTER and DT_RIGHT.

Text style

The text styles are: fsBold, fsItalic, fsUnderline and fsStrikeOut.

User values

Every graphic element, using scripting, has an 'object'. This object provides several features. One feature is user storage that persist while the graphic element is in use. The object contains 32, of each, Booleans, integers (-2147483648...2147483647), singles (4 byte float, 1.5e-45 ... 3.4e+38) and strings. If the index is out of bounds, the result will be false for boolean, zero for integer and single and a blank string for string.

The values are accessed via "GetXXX" and "SetXXX" functions.

GetUserBoolean, GetUserInteger, GetUserSingle and GetUserString
SetUserBoolean, SetUserInteger, SetUserSingle and SetUserString

```
value:=ge.GetUserBoolean(0);      //0-31 Booleans
value:=ge.GetUserInteger(31);     //0-31 integers
value:=ge.GetUserSingle(23);      //0-31 singles
value:=ge.GetUserString(5);       //0-31 strings

ge.SetUserBoolean(8,true);        //the first value is the index, the second is the value to store
ge.SetUserInteger(7,12);          //the first value is the index, the second is the value to store
ge.SetUserSingle(3,12.34);        //the first value is the index, the second is the value to store
ge.SetUserString(11,'test');      //the first value is the index, the second is the value to store
```

Other fields/properties/functions

Counter

This field is set to 0 when the graphic element script is executed the first time the graphic element is created and is incremented each time the script is executed. The field will rollover to 1 after the value 2,147,483,647 is reached. When the graphic element is destroyed (when the window is closed) the counter will restart at 0 when the script executes the first time. Warning: This property can be altered in the script.

Example

```
if (ge.Counter = 0) then
begin
 //first script execution when the graphic element is created
```

```
end;
```

Examples

This example changes the foreground color of a graphic element based on the value of two digital inputs.

```
value:=Valve2Input('ValveOpen',5007,'ValveClosed',5007);
case value of
1: ge.ForegroundColor:=clGray;          //fault
2: ge.ForegroundColor:=clYellow;        //travel
3: ge.ForegroundColor:=clGreen;         //closed
4: ge.ForegroundColor:=clRed;           //open
end;
```

Treeview

This function sets the value of the treeview leaf caption or tag.

Command	Parameter 1	Parameter 2	Parameter 3	Description
Caption	Command	Fixed ID	Value	Sets the caption of the item with a matching item ID to the parameter 3 value.
Tag	Command	Fixed ID	Value	Sets the tag ID of the item with a matching item ID to the parameter 3 value.

Example

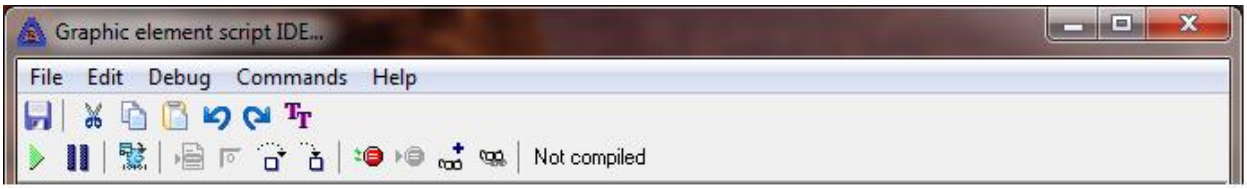
```
ge.Treeview('Caption', 7, 'New caption');

ge.Treeview('Tag', 7, 'P2Main');
```

RUNTIME GRAPHIC ELEMENT SCRIPT EDITING

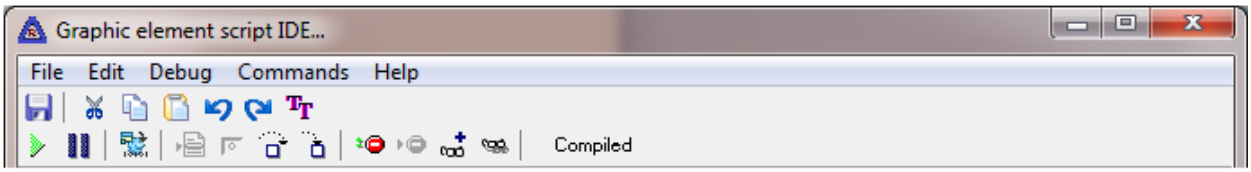
Function list

Pressing and holding the "Control key" (CTRL) while pressing the mouse left or middle button (CTRL + left button or CTRL + middle button) on a graphic element, with scripting enabled, will display the runtime graphic script editor window. The window provides editing, running and debugging the graphic element script. The window provides a standalone script engine for debugging.

When the window is first opened it appears as:



When the runtime editor is opened the graphic element script engine stops executing the script. The script is loaded into the window and is not complied as the "Not complied" label indicates. To run the script, compile the script and if no errors, the label will change to "Complied."



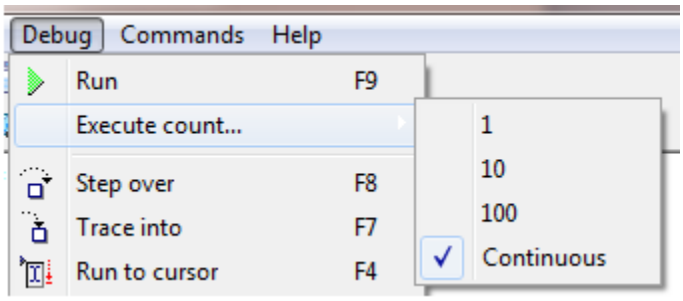
Now, select the "green run" arrow and the script will execute the next time the window is updated, should not be more than ½ second after the button is pressed.

Notes:

1) The script execution must be synchronized with the window drawing logic and execute when the window is drawn, twice per second.
2) If the script stops, because of an error, breakpoint, etc., the graphic element will be rendered as it appears at design time.
3) Mouse commands are normally processed via the command script engine. The graphics script engine is for rendering the graphics and the command script engine process commands. This allows the window to update while the command script is processing. To allow debugging/editing mouse commands the mouse commands are processed with the editor window script engine. Mouse commands are put in an internal queue and processed when the window update logic executes the script.

4) If changes are made to the script, selecting the “Save” command or “saving” when closing the window, will save the changes to the running instance **AND** to the configuration file.
5) When the editing window is closed, the graphic element script is reloaded in the owning window and compiled for normal execution. Any script error will be logged to the event log.

Execute count

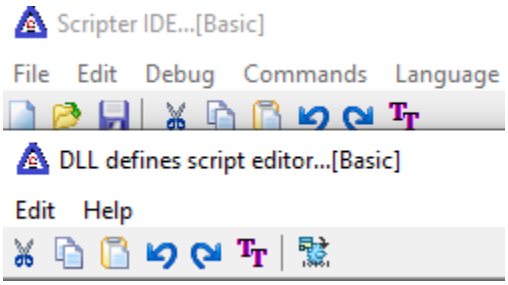


To aid in debugging the script can be configured to update a fixed number of times and then stop or execute until stopped. The default is “Continuous.”

DLL ACCESS

Warning: Access to any external DLL may cause the HMI program to fail or degrade program operation.

For access to an external DLL in a script the "Enable DLL access" property **must** be enabled. To edit the DLL defines script select the "Edit/DLL defines..." menu item in the script editor. When the IDE is in "Basic" the "Basic" DLL script can be edited and the same for "Pascal".



The script can only contain declarations. **Do not** include any executable code in the defines file.

The DLL defines script is not required. The DLL defines can be included in the script. It is more efficient to define the DLL in the define script.

Example of a DLL function define.

Pascal

```
function MessageBoxW(hwnd: pointer; text, caption: string; msgtype:
                     integer): integer; stdcall; external 'User32.dll'
                     name 'MessageBoxW';
```

Basic

```
function MessageBoxW lib "User32.dll" alias "MessageBoxW" stdcall
                     (hwnd as pointer, text as string, caption as
                     string, msgtype as integer) as integer
```

Parameter	Description
MessageBox	Function name to be used in script.
hwnd	A pointer to a window handle. (Can be 0 or nil) *1
text	The message text.
caption	The window caption.
msgType	The type of message window. *1
integer	This is the result of the MessageBox API call.
stdcall	The DLL function call type.*2
external	Defines the next parameter as the DLL file name *3
name	Function name in the DLL.

Notes:

1. See MS windows documentation for MessageBoxW API call for more information.
2. The DLL function call type can be stdcall, register, pascal, cdecl or safecall.
3. The DLL file must be in the same directory as the HMI program or in the "Windows" DLL search path.

Example 1

Calling the above DLL function when the DLL define is in the DLL define script.

Pascal

```
MessageBoxW(null, 'The code executed.','Code status message...',0);
```

Basic

```
MessageBoxW(0, "The code executed.","Code status message...",0)
```

Example 2

Calling the above DLL function when the DLL define is **not** in the DLL define script.

Pascal

```
function MessageBoxW(hwnd: pointer; text, caption: string; msgtype:
                    integer): integer; stdcall; external 'User32.dll';

begin
 MessageBoxW(null, 'The code executed.','Code status message...',0);
end;
```

Basic

```
function MessageBoxW lib "User32.dll" alias "MessageBoxW" stdcall
                    (hwnd as pointer, text as string, caption as
                    string,msgtype as integer) as integer

MessageBoxW(0, "The code executed.","Code status message...",0)
```

Script Draw Object (SDO)

The script draw object (SDO) helper is used to create static graphic elements in an open window. The graphic elements can be created, changed or deleted via scripting. The graphic elements are deleted when the window is closed.

The SDO is a helper object. It is called from the script to manage the script created draw objects of all windows.

Properties (see notes below)

Name	Data type	Default	Description
BackgroundColor	Color	Black	Background color when brush style is not solid.
Bottom	Integer	0	Bottom of the bounding rectangle, in pixels
BrushStyle	Brush style	bsSolid	Applies a brush style to the graphic element
Count	Integer	0 (Read only)	Count of script graphic elements in window
Curvature	Integer	0	Curvature of round rectangle object corners
FontName	String	Arial	
EndAngle	Integer	0	End angle for arc graphic element
FontCharacterSet	Character set	Default for OS	
FontSize	Integer	8	
FontStyle	Text style	No style	
ForegroundColor	Color	Black	Foreground graphic element color
Height	Integer	0 (Read only)	Height of the graphic element, in pixels
ID	String	(Read only)	Current graphic element loaded.
LastError	String	(Read only)	If not empty, error message of last operation
Left	Integer	0	Left of bounding rectangle, in pixels
LibraryGroup	String		Used when referencing a library object
LibraryName	String		Used when referencing a library object
Opacity	Integer	255	0 to 255, see also TransparentState
PenColor	Color	Black	
PenStyle	Pen style	psSolid	
PenWidth	Integer	0	
PointCount	Integer	0	Polygon line X/Y (vertices) point count
PolyOpen	Boolean	True	If false, the polygon line is automatically closed
pX	Array of integer		The X axis for each polygon vertex
pY	Array of integer		The Y axis for each polygon vertex
Right	Integer	0	Right of bounding rectangle, in pixels
Rotation	Integer	0	N/A at this time
ScriptName	String		Script name called on mouse up in element
ScriptFunction	String		Script function called on mouse up in element
Shape	String	Circle	See below for supported shapes
StartAngle	Integer	0	Start angle for arc graphic element
Text	String		Text value for text graphic elements

TextAlignment	Text alignment	Left	
Top	Integer	0	Top of bounding rectangle, in pixels
TransparentState	Integer	OPAQUE	OPAQUE or TRANSPARENT
Width	Integer	0 (Read only)	Width of graphic element, in pixels
WindowName	String		Applicable open window name

Notes:

1) Not all properties apply to all graphic element types.
2) Default for strings is an empty string.
3) The window specified in "Window name" must be open.

Shapes

Supported shapes: **Rectangle**, **Line**, **Circle**, **Text**, **Round rectangle**, **Arc**, **Polygon line** and **Library** graphics.

Functions

Delete

Delete a script draw graphic element by id reference.
Example

```
sdo.WindowName:= 'First Window';
sdo.Delete('circle1');
```

DeleteAll

Delete all script draw graphic elements in a window.
Example

```
sdo.WindowName:= 'First Window';
sdo.DeleteAll;
```

DeleteType

Delete all script draw graphic elements in a window of the selected type
Example

```
sdo.WindowName:= 'First Window';
sdo.DeleteType('Circle');
```

GetIDS

Return the count of script draw graphic elements and the ids.
Example

```
sdo.WindowName:='First SDO';
values:=sdo.GetIDS;
s1:= '';
if (values[0] > 0) then
 for i:= 1 to values[0] do
  s1:=s1 + values[i] + #13#10;          //add carriage return and linefeed
//s1 now contains all the ids
```

Load

Loads an existing script draw graphic element properties into the SDO helper.
Example

```
sdo.WindowName:='First SDO';
values:=sdo.Load('circle1');
```

Make

Creates a script draw graphic element using the properties in the SDO helper or updates and existing graphic element.

Create a circle

```
sdo.WindowName:='First SDO';
sdo.Shape:= 'Circle';
if not sdo.Make('circle1') then
 ShowMessage(sdo.LastError);
```

Creates a polygon line

```
sdo.WindowName:='First SDO';
sdo.Shape:= 'Polygon line';
sdo.PointCount:=4;
sdo.pX[0]:=10;          sdo.pY[0]:=10;
sdo.pX[1]:=40;          sdo.pY[1]:=10;
sdo.pX[2]:=40;          sdo.pY[2]:=40;
sdo.pX[3]:=10;          sdo.pY[3]:=40;
if not sdo.Make('OnlyPolygonLine') then
 ShowMessage(sdo.LastError);
```

Update text color of text element

```
sdo.WindowName:='First SDO';
sdo.Load('TroubleTxt1’);      //load all element settings
sdo.ForegroundColor:=clWhite;
sdo.Make('TroubleTxt1’)
```

SetDefault

Sets all the SDO helper properties to the default state.
Example

```
sdo.SetDefault;
```

ScriptName/ ScriptFunction

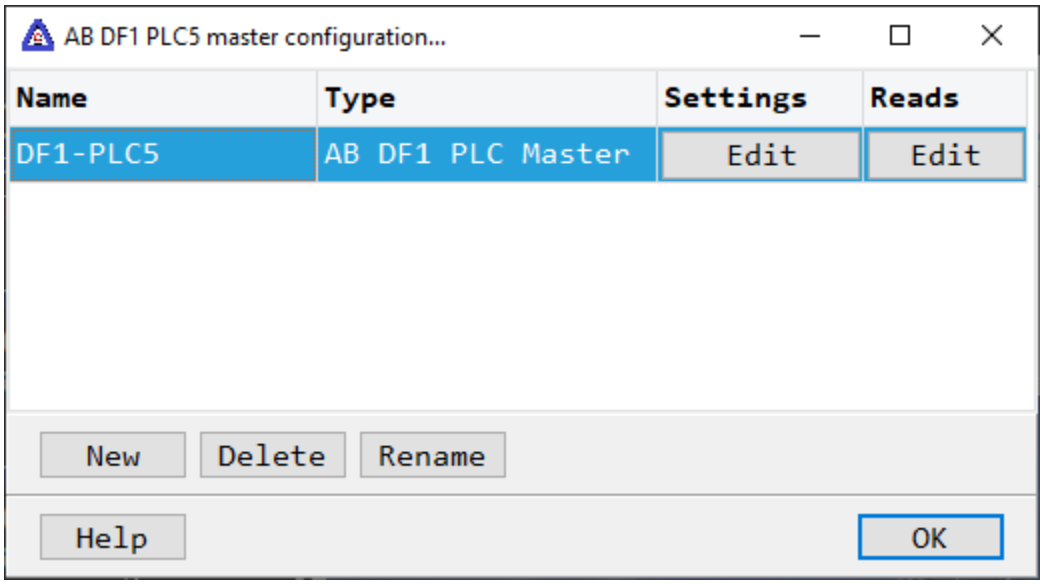
Script file name and function, in the script, to call when the mouse button is released on the graphic element.

Example:

```
sdo.ScriptName:= 'SDOMouse.psc';
sdo.ScriptFunction= 'MouseUp';

procedure MouseUp(id:string); //graphic element ID is parameter
begin
  ShowMessage(id);
end;
```

AB DF1 PLC5

Each AB DF1 master object is listed in the window.

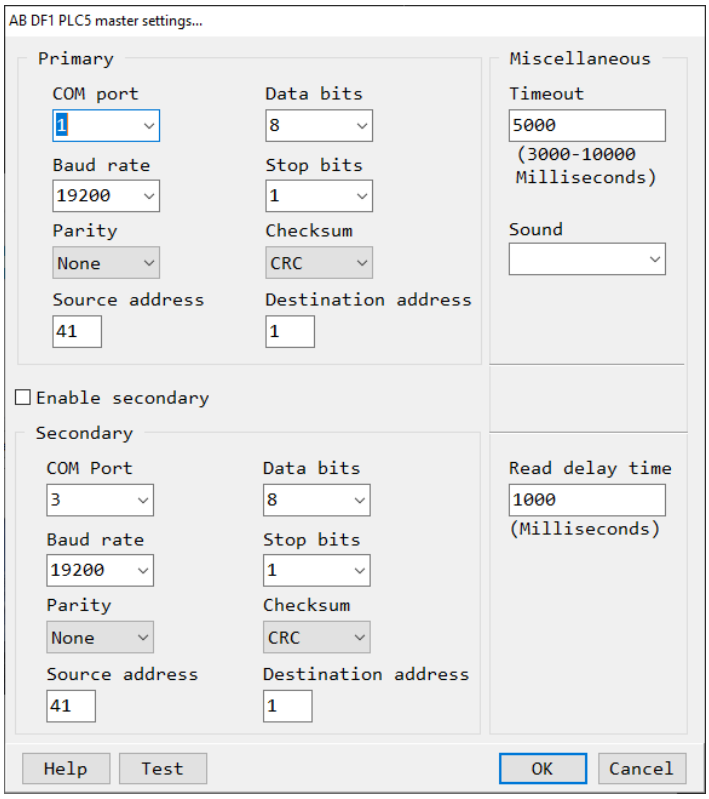


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an AB DF1 master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Source and destination address

These values are in octal. Check the device documentation for allowable values. For example: Using a KF2 to connect to a PLC, the source address must match the switch settings on the KF2. If the KF2 address switches are set to 51 (octal) enter 41 (decimal).

NOTE: If the receiving module has the "ignore duplicate messages" option enabled, and the number of reads is one (1) or the number of reads is greater than one (1) but only one (1) read is enabled the message will be ignored and data collection will fail. Disabling the "ignore duplicate messages" option will prevent messages from being ignored under both conditions described. If that is not possible, create a dummy read, for example N7:0 with a count of 1. A dummy read is a read with a different address than the single read. Verify the dummy read is enabled.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

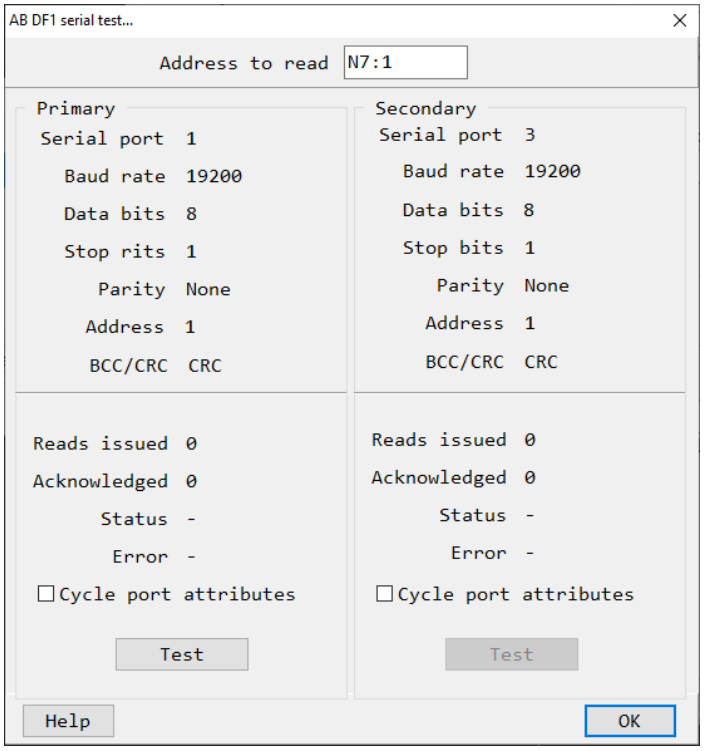
When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Test button

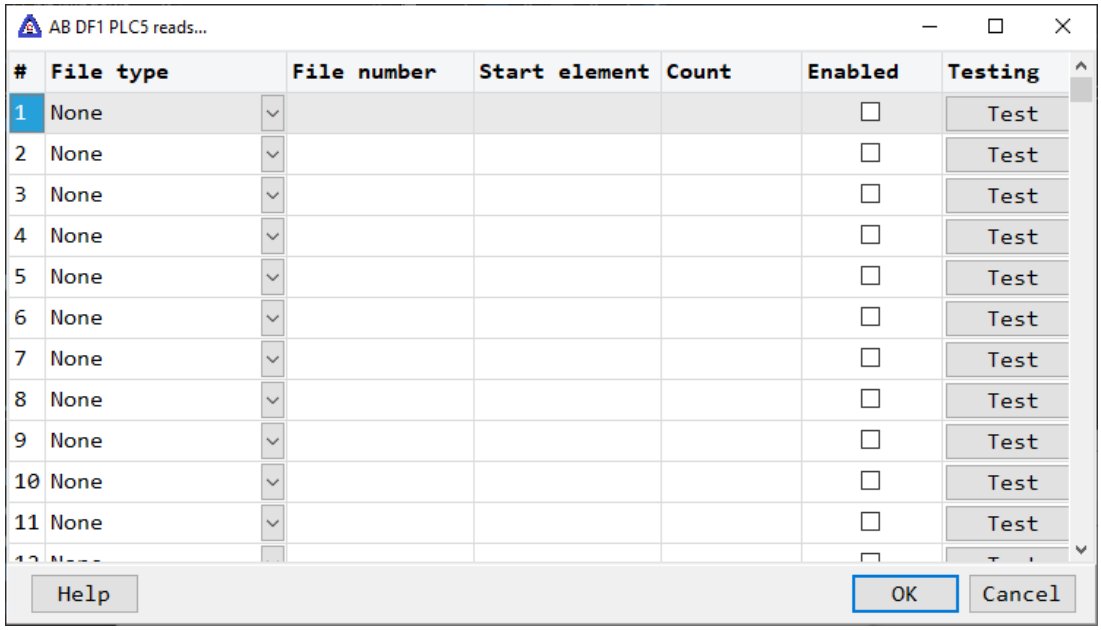


When the test button is selected the program will attempt to read one element of data from the device at the address entered.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the AB defined addressing for access to the data files in the PLC. Please refer to the AB documentation.

The DF1 “Duplicate Message Detection” logic may be enabled in the communication device. If more than one read command is enabled the detection logic will not cause a problem. If only one read command is enabled the “Duplicate Message Detection” logic must be disabled. (Or create another read command.)

Example address:

Common

N7:10, N7:451/0, N7:33/15
T4:152/TT,T4:152/EN,T4:152.ACC
B3:45/5
B3/567

I:001/00, I:010/15
O:001/01, O:012/17

Note: Input and output references are in octal for point addressing. The input and output reference must contain the complete address. The value after the colon must be three digits and any bit references must be two digits. Example I:XXX or O:XXX I:001, O:005, I:001/01, O:005/17, I:022/01, O:019/17, I:167/00, O:177/17

Legal file numbers are 3 to 999. File number 0 (zero) is for outputs, file number 1 (one) is for inputs and file number 2 is for status. If the file type is input, output, or status the file number is set to the correct value when the 'OK' button is selected.

File type

Example: Integer (N), Input (I), Binary (B)

File

Legal file numbers are 3 to 999. File number 0 (zero) is for outputs, file number 1 (one) is for inputs and file number 2 is for status. If the file type is input, output, or status the file number is set to the correct value when the 'OK' button is selected.

Start element

This is the starting element for the file type to read. The different file types have different element counts. Example: Integer (N) has a one word element; Timer (T) has a three word element.

Count

This is the number of elements to read. The file type determines the maximum number of elements that can be read in a single command.

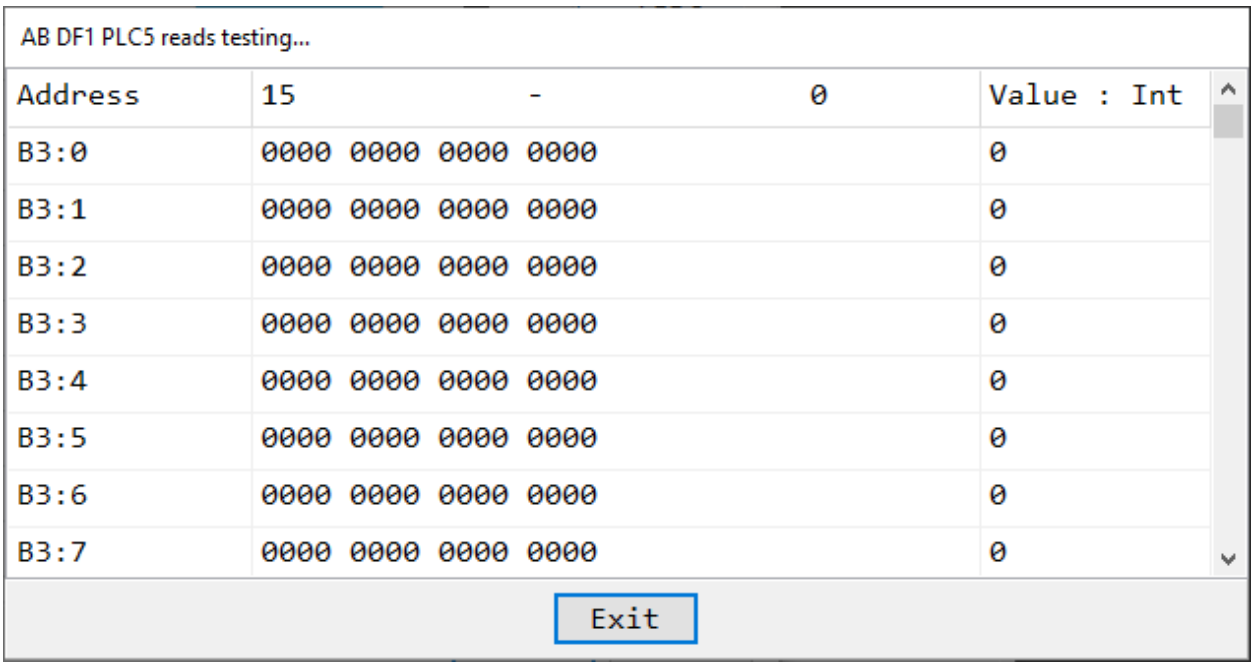
The maximum is 122 words (244 bytes) of data.

Enabled

The read requests are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

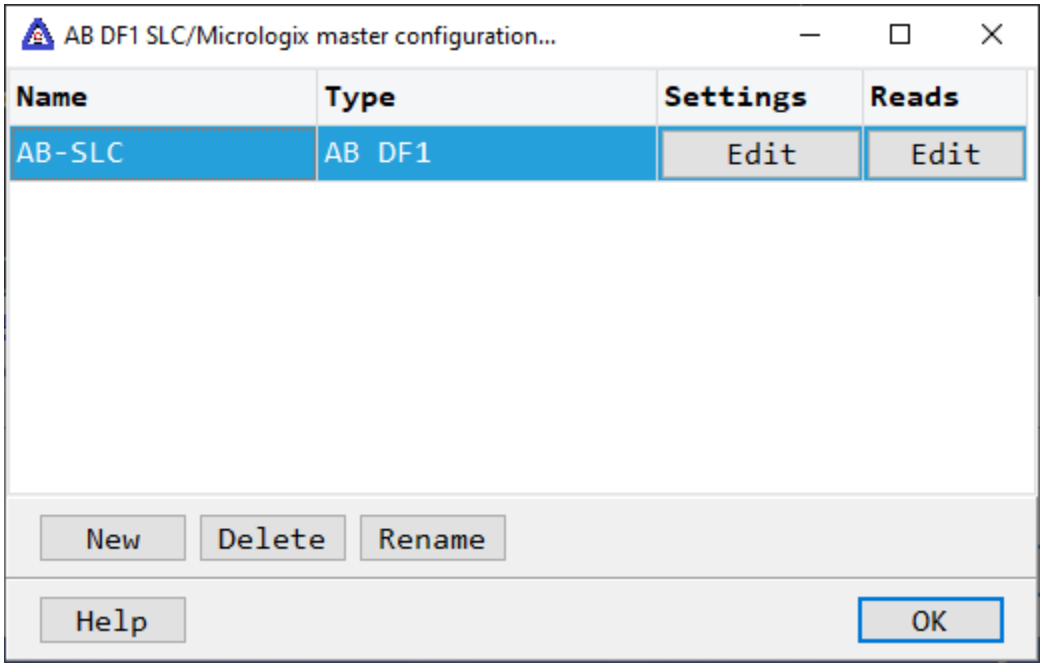
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. If the error string contains 'STS' or 'EXT STS' the text that follows is the error code, in hexadecimal, returned from the external device. The HMI uses a function code $0F. PCCC, assuming the port test succeeded, if "Forward open success..." is displayed and does not change, close the testing window and reattempt the test.

Error messages

No file type selected	A file type must be selected.
Element out of range	Start element must be: 0-999
Count exceeds limit	The number of elements to read is too high for the file type.
Element start + count exceeds limit	Element start + count greater than file end.
Count < 1	Must read at least one element
File number out of range	The file number must be: 0-999

AB DF1 SLC/MICROLOGIX

Each AB DF1 master object is listed in the window.

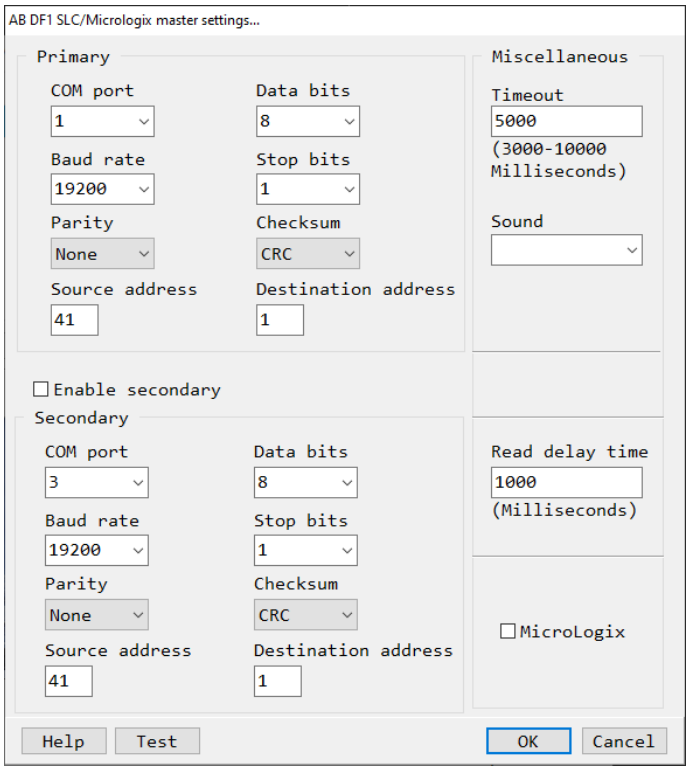


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an AB DF1 master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Source and destination address

These values are in octal. Check the device documentation for allowable values. For example: Using a KF2 to connect to a PLC, the source address must match the switch settings on the KF2. If the KF2 address switches are set to 51 (octal) enter 41 (decimal).

NOTE: If the receiving module has the "ignore duplicate messages" option enabled, and the number of reads is one (1) or the number of reads is greater than one (1) but only one (1) read is enabled the message will be ignored and data collection will fail. Disabling the "ignore duplicate messages" option will prevent messages from being ignored under both conditions described. If that is not possible, create a dummy read, for example N7:0 with a count of 1. A dummy read is a read with a different address than the single read. Verify the dummy read is enabled.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

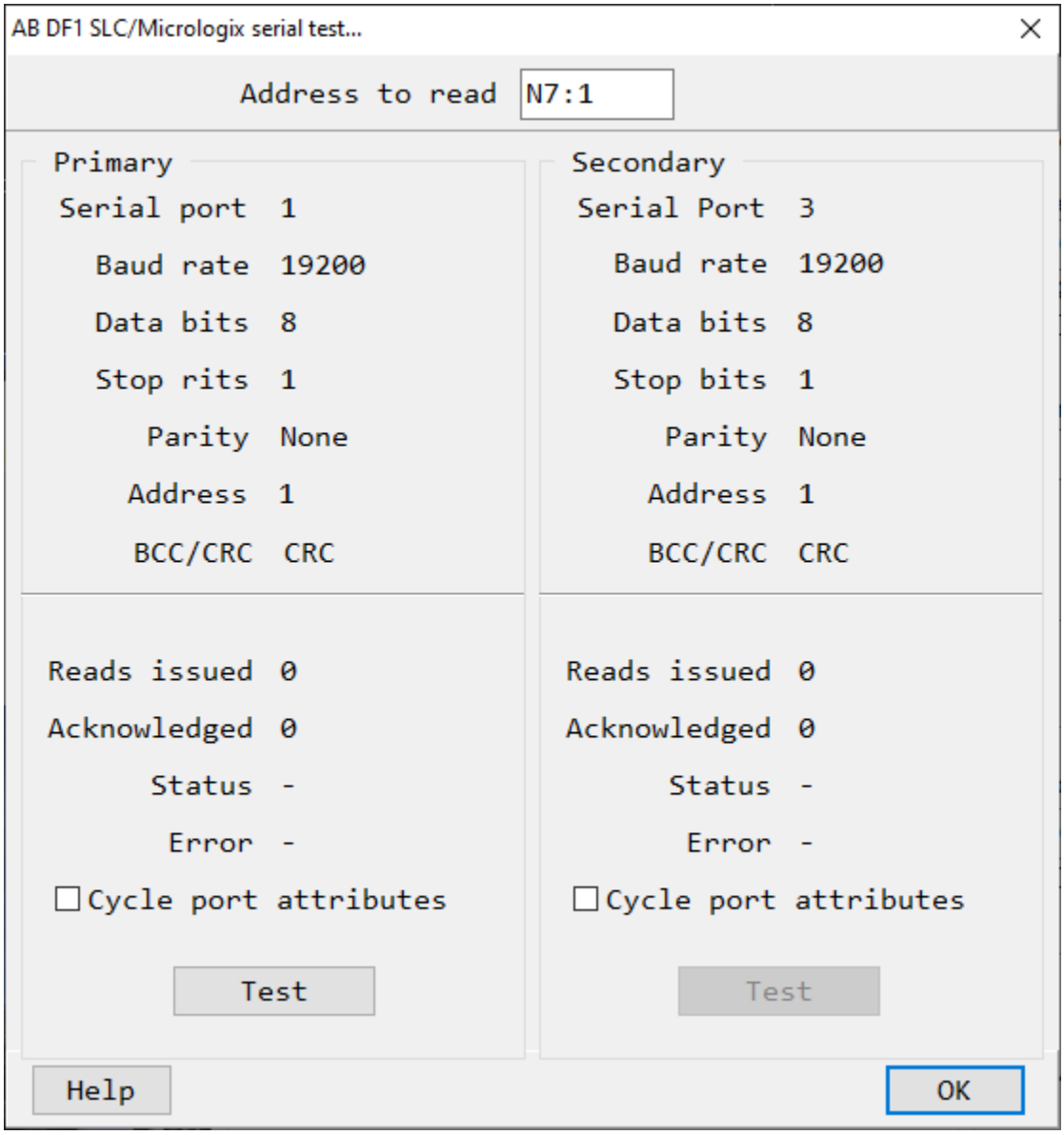
A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Test button



When the test button is selected the program will attempt to read one element of data from the device at the address entered.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads

AB DF1 SLC/Micrologix reads...

#	File type	File/Slot #	Start element	Count	Enabled	Testing
1	Binary	10	0	22	☑	Test
2	None				☐	Test
3	None				☐	Test
4	None				☐	Test
5	None				☐	Test
6	None				☐	Test
7	None				☐	Test
8	None				☐	Test
9	None				☐	Test
10	None				☐	Test
11	None				☐	Test
12	None				☐	Test

Help OK Cancel

The address ranges shown may or may not be present in the slave device.

The HMI uses the AB defined addressing for access to the data files in the PLC. Please refer to the AB documentation.

The DF1 “Duplicate Message Detection” logic may be enabled in the communication device. If more than one read command is enabled the detection logic will not cause a problem. If only one read command is enabled the “Duplicate Message Detection” logic must be disabled. (Or create another read command.)

Example address:

Common

N7:10, N7:451/0, N7:33/15
T4:152/TT,T4:152/EN,T4:152.ACC
B3:45/5
B3/567

I:1.0/00, I:1.1/12, I:4.4/15
O:2.0/01, O:7.0/16, O:8.1/31

Fixed SLC, SLC 5/01 and 5/02 do not support digital input and output data table access. The format of the address must be used.

I:X.X/XX
O:X.X/XX

File Type : Slot . Word / Bit

Note: When configuring reads if the file type is input or output be aware that setting the count greater than the number of words the card supports, may or may not return data

(a successful read). In some cases the PLC will return the data for the selected card (slot) and additional cards (slots) up to the count value.

Example:
Slot 1 1746-IA16 1 word
Slot 4 1747-KE 45 words
Slot 5 1746-NI8 8 words
Slot 6 1746-IA16 1 word

A read
Slot = 1
Start element = 0
Count = 45

For this read the PLC will return 45 words of data. The data past the first word I:1.0 is for other cards.

For the HMI point source address and PLC addressing to match, configure one read for each populated slot (if data is needed) and set the count to the number of words the card utilizes.

File type

Example: Integer (N), Input (I), Binary (B)

File/Slot

Legal file numbers are 3 to 255. For inputs and outputs the file number is the slot number.

Start element

This is the starting element for the file type to read. The different file types have different element counts. Example: Integer (N) has a one word element; Timer (T) has a three word element.

Count

This is the number of elements to read. The file type determines the maximum number of elements that can be read in a single command.

For SLC 5/01 or 5/02 = 82 bytes (41 words).
For SLC 5/03 or 5/04 = 236 bytes (118 words).

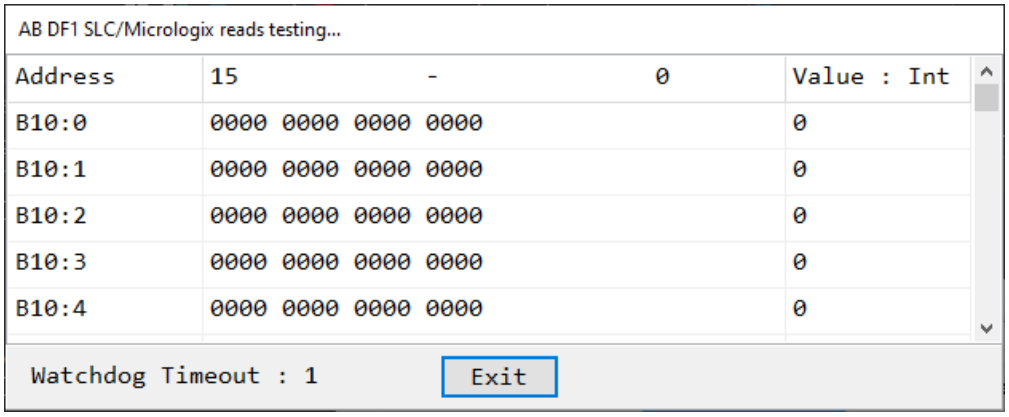
Example: Timer (T) 3 words per element. Maximum count is 40.
Float (F) 2 words per element. Maximum count is 61.

Enabled

The read requests are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. If the error string contains 'STS' or 'EXT STS' the text that follows is the error code, in hexadecimal, returned from the external device. The HMI uses a function code $0F. PCCC, assuming the port test succeeded, if "Forward open success..." is displayed and does not change, close the testing window and reattempt the test.

Error messages

No file type selected	A file type must be selected.
Element out of range	Start element must be: 0-255
Count exceeds limit	The number of elements to read is too high for the file type.
Element start + count exceeds limit	Element start + count greater than file end.
Count < 1	Must read at least one element
File number out of range	The file number must be: 0-255
Invalid file type	The file number does not match the assigned file type.

AB DF1 SLC/Micrologix Master arrays

Commands

Link

The link command "links" a read configuration to an array. Each time the requested data is returned from the external device the data is written to the specified array.

Supported types	Boolean, small integer, word and float (double)
Syntax	values:=UArray('<array name>', 'Port', [<starting dimension index>], ['Link', '<port name>', <read index>]);
Result	values[0] = 0 (zero) or error code values[1] = 0
Example	values:=UArray('Turbine1Pressures', 'Port', [0], ['Link','Turbine1', 2]);

Notes:

1) Only one array can be "linked" with a read. The last called "link" command for a port/index is used.
2) The array must be sized to contain all the data from the read request. The array can be larger than the request.
3) More than one read request can be linked to an array.
4) The file type must be compatible with the array data type.

Write

The write command "writes" array elements to the port device. The data type of the array determines how the data is written. The write is added to the port write queue and is processed according to the port read/write logic.

Boolean	The "count" must be in increments of 16.

122 words of data can be written. For floats, 61 float values can be written.

Syntax	values:=UArray('<array name>', 'Port', [<starting dimension index>], ['Write', '<port name>', '<beginning holding address register>', <count array elements to write>]);
Result	values[0] = 0 (zero) or error code values[1] = 0
Example	values:=UArray('Turbine1Limits', 'Port', [2], ['Write','Turbine1', 'N7:12', 4]);

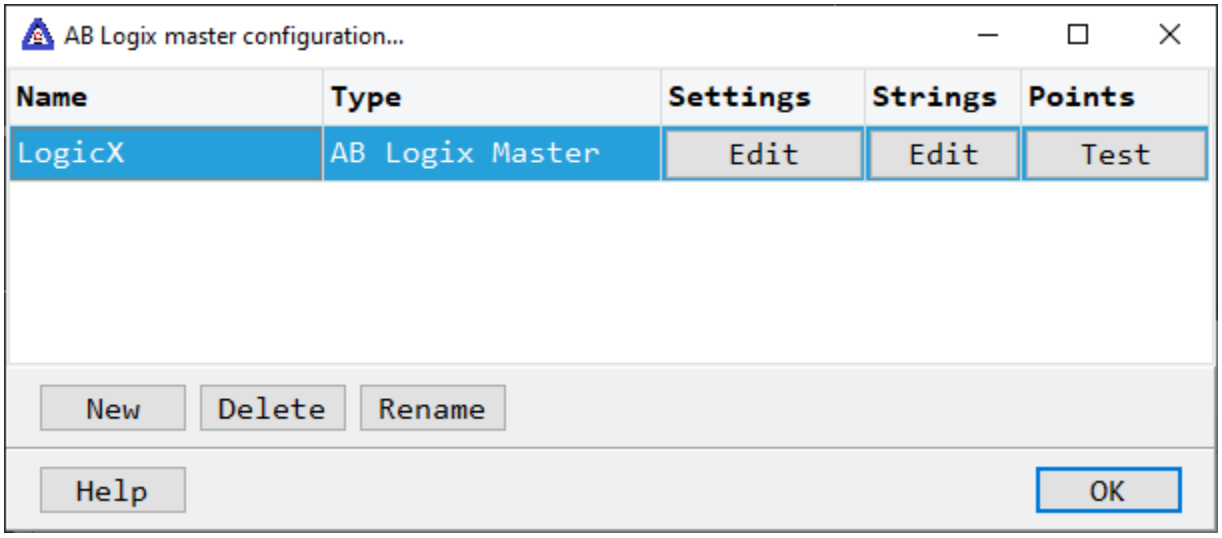
Notes:

1) For double type (8 byte) writes, the float value is converted to a single (4 byte) floating point value. Values outside the range or precision of a single float (32-bit floating point, IEEE-754 standard format) number will be lost.
2) The array must be sized to provide all the data specified in the write request. The array can be larger than the request.
3) The file type must be compatible with the array data type..

AB Logix (Original buffer)

See here for AB Logix (Large buffer)

Each AB Logix master (Original buffer) object is listed in the window.

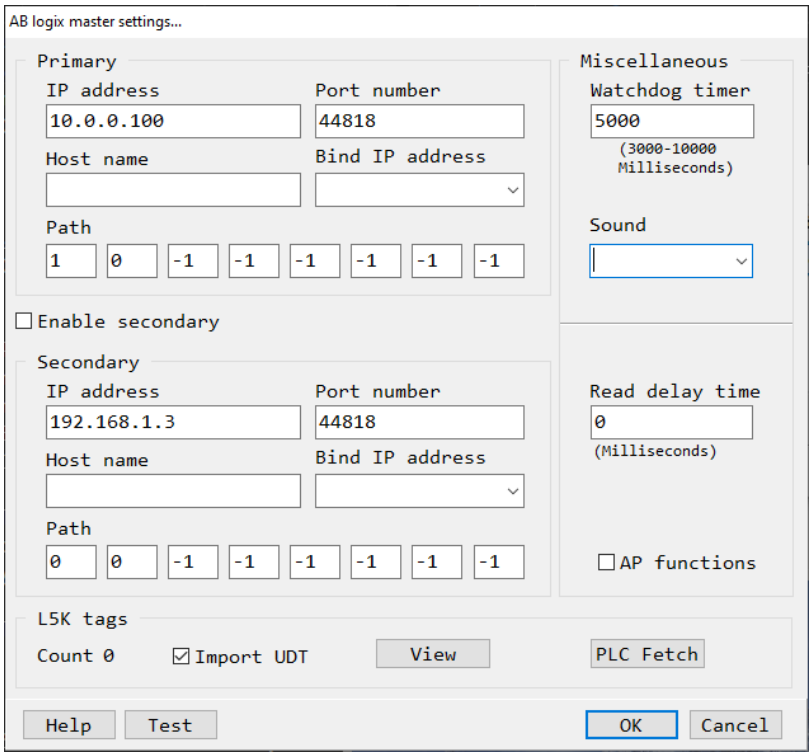


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an AB Logix master object select the "Delete" button.

Settings



The TCP port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address

This field specifics the IP address of an interface device on the computer, used to “bind” the communications path. Leave the field blank to allow the OS to select the device.

Path

This is the path to the endpoint (controller). A "-1" value in the path is ignored. The most common path will be 1 - 0. That is from the Ethernet/IP card to the CPU (slot 0). If the CPU was in slot 1 the path would be 1 - 1. All the non-used path fields must be -1. If the path is to a CPU in another rack the path would be longer. Please refer to AB documentation for proper path settings.

Example

Rack 1

Slot 0 ControlNet card
Slot 1 CPU

Slot 2 Ethernet/IP card

Rack 2

Slot 0 ControlNet card
Slot 1 CPU

To access the CPU in rack 2 the path would be: 1 - 0 - 2 - 9 - 1 - 1

1 = Ethernet/IP
0 = ControlNet card in Rack 1 slot 0
2 = Channel 1 of ControlNet card
9 = Rack 2
1 = ControlNet card in Rack 2 slot 0
1 = CPU Rack 2 Slot 1

Enable secondary

The "Enable secondary" checkbox provides for a second path to be used as a hot backup. When in run mode the primary configuration is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary configuration. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary configuration, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary configuration an entry will be made in the event log and operations will return to the primary port.

If the computer contains two network interface cards (NIC) the program will use the first one detected for the primary and the second one detected for the secondary. If only one NIC is detected the program will use it for the primary and the secondary.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:
The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

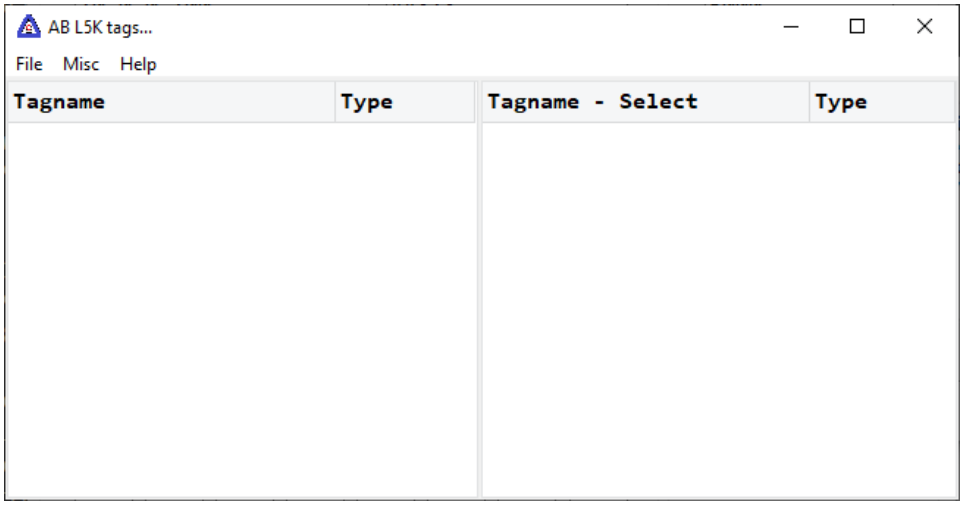
PLC Fetch

The program will attempt to connect to the PLC, collect the controller tags, program tags, data structures, etc. and display the results. Depending on connection speed, number of tags, etc., a window may appear and close automatically when the collection is complete. If a successful collection the "Create points" window will appear.

L5k tags

The HMI can attempt to parse an ".L5K" file for controller tags and program tags. The parsed list can be used for copying and pasting the tag names when creating/editing points. Refer to the source address section for point address formats when used with Logix controllers.

View

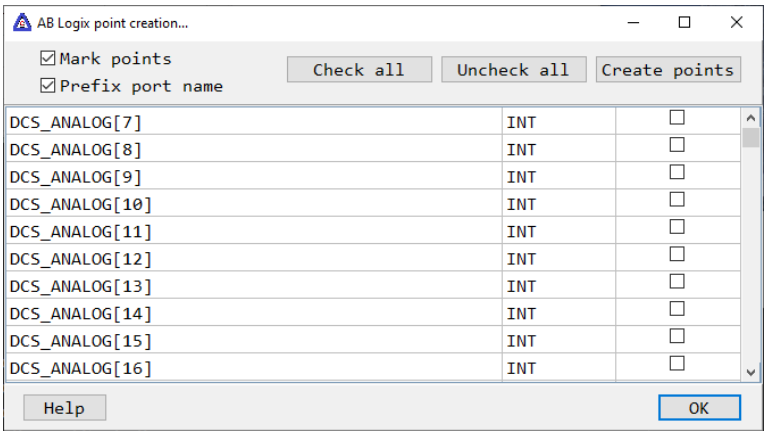


Import UDT

When enabled the program will attempt to import the 'User Data Type' structures and save them in the project path to a file named 'ABCustomTypes.ini'.

Notes:
1. The system overhead time slice (SOTS) may need to be adjusted for optimum operations.
2. If communication performance is degraded due to a **very** large number of tags, consider creating two or more ports and dividing the tags between the ports. The number of ports (open connections) the PLC allows is determined by the PLC.

Create points



The list of tags from the L5K file or PLC fetch, expanded to show all the tagname is displayed with a checkbox on each line that can be used to create an HMI point. Select the desired checkboxes and then select the "Create points" button to create the points.

Mark points

The imported points will be marked for display. When the point is viewed in the "Point configuration" window the point text color will be "red" and "bold" until the point is displayed in the "Point configuration" window.

Prefix port name

If enabled the port name will be prefixed to the tagname and item.
<port name>.tagname, <port name>.tagname.item
West_Turbine.PressureLo, West_Turbine.Pressure.HiHi

Create points

Enable the checkbox to create the point when the "Create Points" button is selected. The space bar can be used to toggle the checkbox.

Notes:
1. Points with existing tagnames will not be overwritten.
2. AI and AO data type is set to 4 byte float.
3. Point “access rights” is set to 'Read/Write.'
4. Selecting the “Check all” button could lead to many points that are not required and a slowdown of data collection. For example, a timer has several boolean values (Timer timing, enabled, done, etc.) If only the “Done” boolean value is required, reading the other boolean values should not be configured. Only create points for data values that are required.

Filter file

A “filter” can be used to remove points from the list. The filter file must be placed in the project directory and the file name is formatted <port name>_CreateFilter.txt
Example: Crusher71_CreateFilter.txt.

The file can utilize three sections, each section delineated with a header marker.
1) The header marker must be formatted as below.
2) Only one header maker of each header type is processed.
3) The :::Include::: header is ignored if the :::Exclude::: header is not present.
4) See the examples below.
5) All comparisons are case sensitive.
6) Excluded tagnames, the cell background will be shaded and the text with have a strikethrough.

:::Exclude:::

All tagnames are scanned and if the tagname contains the text, the tagname is marked as “excluded”.

:::Include:::

After the “exclude” scan, all the excluded tagnames are scanned and if the tagname contains the text, the tagname’ s “excluded” mark is removed.

:::Exact:::

All tagnames are scanned and if the tagname “exactly” matches, it is marked “excluded”.

Example the PLC has a program (task) named “Mixer” and it has many tags that are not needed in the HMI. But, it does have a few tags that are needed. A filter file formatted:

```
:::Exclude:::
Mixer|
:::Include:::
Mixer|FeedPressure
Mixer|Speed

That would exclude all “Mixer|” tagnames except the tagnames in
the :::Include::: section.
```

Example A large array of controller tags are present and only one tag in the array is needed in the HMI.

```
:::Exclude:::
rocks[
:::Include:::
rocks[23]

That would exclude all “rock” array tagnames except the tagnames
in the :::Include::: section.
```

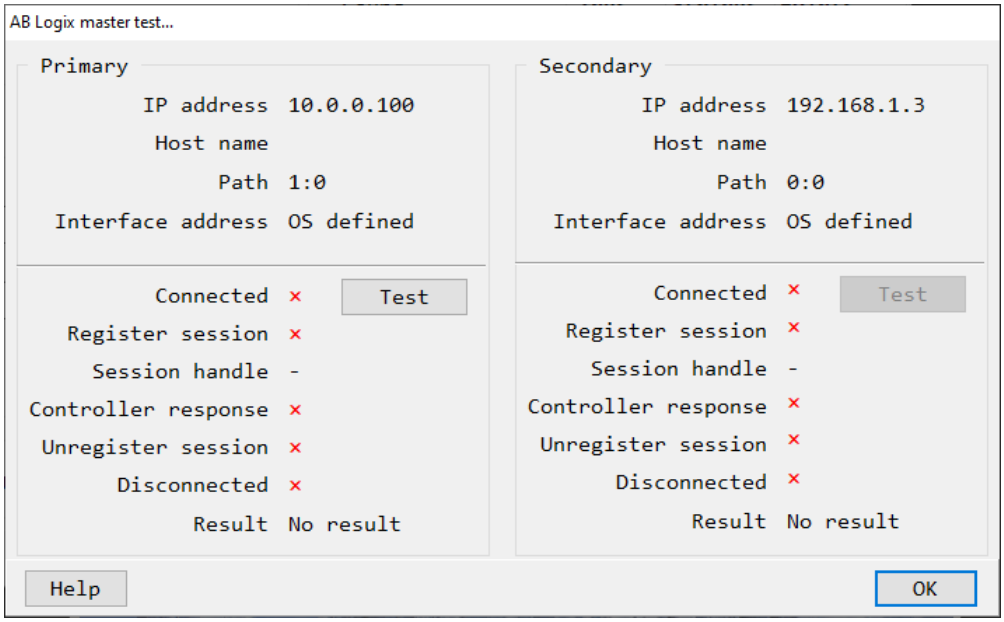
Example A program (task) tags are not needed in the HMI.

```
:::Exclude:::
AggerateCounter|

That would exclude all “AggerateCounter|” tagnames.
```

Example The following tagnames are not needed in the HMI.

```
:::Exact:::
AggerateCounter|Count_1
AggerateCounter|Count_2
AggerateCounter|Count_3
Rocks[23]
Mixer|FeedPressure
Mixer|Speed

All the tagnames exactly matching would be marked excluded.
```

Test button

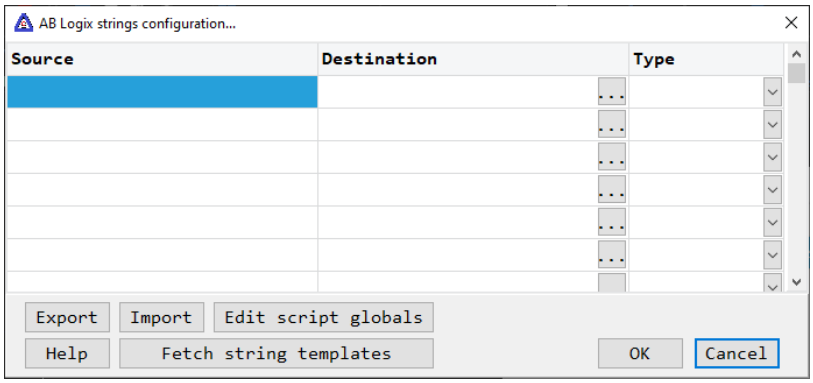


When the test button is selected the program will attempt to connect to the ENET card, register a session and get the attributes of the Logix controller.

Connected:	Connection to ENET card
Register Session:	Request a session handle
Session Handle:	Session handle value
Controller Response:	Command to the controller
Unregister Session:	Release the session handle
Disconnected:	Disconnect from the ENET card
Result:	Test result
Failed:	Could not connect to the ENET card

Partial Success: Connected to the ENET card, failed Logix controller response
Success: Connected to ENET card, successful Logix controller response
A partial success result is normally due to an incorrect path setting or the controller is missing.

Strings



Strings are not regular points. They do not have alarms, printing, etc. Strings may be displayed in any window via the "Script Global" animation. Any other actions, parsing, comparing, etc. must be done in scripts. The default string length type is “STRING” or “ASCII”. To support UDT (user defined type) string types the string template must be collected from the PLC. See “Fetch string templates”

Source

This is the tagname in the PLC. It must be a valid tagname with a datatype "string".

Destination (optional)

If desired, select a script global location and the string will be copied to the location when the string value is returned from the external device. If this option is used or not the string may still be accessed via the "StringGet" and "StringSet" script commands. If this is configured and access to the string, in a script, is desired, use the "GlobalGet" or "StringGet" to copy the string. The format of the destination is: <section>.<name>.
Note: Writing to the script global does not write the value to the PLC. "StringSet" must be used to write a string to the PLC.

Type

The string type (size). “Default” is predefined 82 character string type defined in the PLC.

Export

This is used to export the configuration to an Excel worksheet.

Import

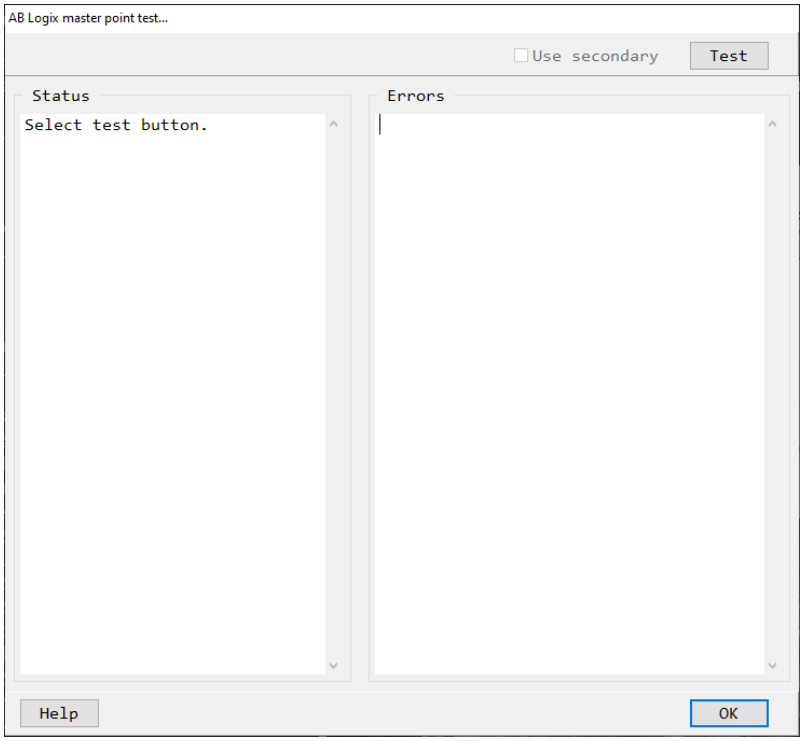
This is used to import string configurations from an Excel worksheet. The contents of the grid are completely replaced by the contents of the file.

Fetch string templates

Note: Any changes to a string UDT after fetching the templates **requires** the fetch operation to be repeated. Use this button to collect UDT string information from the controller. The data collection can take a long time depending on the count of UDTs, the network speed, PLC loading, etc.. The maximum string size supported is 400 characters.

After the string UDT data is collected from the PLC, a checkbox list will appear allowing for the selection of UDT, by name, for the HMI to support . **Note:** It is possible for UDTs to appear in the list that are not string types.

Points



This feature is used to test if a point or string, with the source address entered, exist in the external device.

Use Secondary

If enabled the secondary port settings will be used.

Test

Selecting this button will execute the test. All the configured points and strings will be tested. The test is very fast.

Note: Right click and copy the contents of either text area.

Status

This area is to display general information regarding the test and testing progress as well as displaying each point/string success. Each point/string tagname will be displayed with a 'success' suffix if the source address is in the external device.

Example:
Connecting...Connected
Session registered
CPU test successful
Point testing begins...
Alarm_Active...
Close_Discharge_Valve_CRT... success
CRT_Acknowledge... success

The first four lines are general status updates. The first point 'Alarm_Active', failed and the reason will be displayed in the 'Error' area. (See below) The remaining lines each indicate success.

Error

This area is to display the error that was generated for a source address. Several types of error are possible. Each point error has three lines. Examples:

Tagname: Alarm_Active
Source address: Alarm_Active1
Error: General Status Error:1E

Tagname: Discharge_Value_Close_Timer.ACC
Source address: 1Discharge_Value_Close_Timer.ACC
Error: General Status Error:1E

Tagname: N7[0]
Source address: N7[0]
Error: Point is digital, returned data type is not digital

Tagname: DeviceProfileType.FirstMask
Source address: AP:DeviceProfileType.FirstMask
Error: Path segment error

Path segment error can have several causes. The “External access” property of the tag could be set to “None”. The tag could be a “local” variable of an “Add-On” instruction.

The 'General Status Error:1E' is the most common when a source address does not exist in the external device.

The other error in the example is the point type does not match the data type of the source address. In the above example the point type is digital and the data for the source address is not digital. In this case it was a two byte integer.
Each string type has two lines. Example:

Source address: string1
Error: General Status Error:1E

The string data type is 'structure'. The structure data type covers many different collections of data. If the data type returned is one of the atomic data types, i.e. Boolean, real, etc. an error will be flagged.

Source address

The HMI uses controller tagnames to access controller data. The HMI supports: boolean, small integer (byte), integer, double integer, floats and strings. Arrays and structures (timers, control, etc.) are also supported at the structure field level.

There are two special characters. The colon (:) and the period (.). The colon is used to indicate a program name and the period is used to indicate a field name.

Format

The source address must match the controller tagname. If the data type is a simple type then the source address and controller tagname must match exactly.
Examples of possible source address.

Tagname	Source address	Data type
Door_Open	Door_Open	Boolean
Door_Closed	Door_Closed	Boolean
Water_Press	Water_Press	Real
Water_Temp	Water_Temp	Integer
Tank_Level	Tank_Level	String
Open_Time	Open_Time.ACC	Timer (accumulator)
Open_Done	Open_Done.DN	Timer (done)

For access to array elements add the array bounds.

Door_Open	Door_Open[0]	Array[8] of Boolean
Door_Closed	Door_Closed[2]	Array[8] of Boolean

Boolean arrays are grouped into 32 bits. The array index specifies the bit group.
Door_Open[0] is bit 0 to 31
Door_Open[1] is bit 32 to 63
Door_Open[2] is bit 64 to 95

Use a period (.) to specify the bit in the group.
Door_Open[0].0 is group 0 bit 0
Door_Open[0].31 is group 0 bit 31
Door_Open[5].0 is group 5 bit 0
Door_Open[44].2 is group 44 bit 2

Note: When writing a boolean of a boolean array, for example Water_Temp[9].7, the last returned value of the complete 32 bit group, is used to mask the bit and set it true or false.

Water_Press	Water_Press[3]	Array[8] of Real
Water_Temp	Water_Temp[7]	Array[8] of Integer
Open_Time	Open_Time[0].ACC	Array[8] Timer (accumulator)
Open_Done	Open_Done[0].DN	Array[8] Timer (done)

For multi-dimensional arrays.

Door_Open	Door_Open[0,1,2]	Array[8,8,8] of Boolean
Water_Press	Water_Press[3,2]	Array[8,4] of Real

If the tag is in a program, prefix the program name with a colon
<programName:>tagname.

First:Door_Open	Door_Open[0]	Array[8] of Boolean
First:Door_Closed	Door_Closed[2]	Array[8] of Boolean
First:Water_Press	Water_Press[3]	Array[8] of Real
First:Water_Temp	Water_Temp[7]	Array[8] of Integer

For access to I/O cards

From Logix5000 Data Access (Rockwell Automation Publication 1756-PM020C-EN-P - June 2012)

However, do not access complete UDT Tags that contain nested system structures. (For example, Module-Defined, Predefined, or Add-On Defined.)

...

Predefined, Module-Defined, and Add-On-Defined structure tags have a more complex set of rules than UDTs, and have a greater potential to change in the future. Do *not*

access complete structure tags of these types, or complete UDTs with nested tags of these types.

....

Instead, access atomic members of these tags ... using either *one* of the methods that follow.

- Create an alias of the atomic member and access the alias instead of the structure.
- Create an atomic tag or UDT structure tag with an atomic member, and then have the user program copy the data to and from the tag or atomic member. Access the new tag or atomic member instead.

AB Logix arrays

Commands

Link

The link command adds a "read" message to the continuous read queue and links the result to an array. Each time the requested data is returned from the external device the data is written to the specified array. The fetched array data type, from the external device, must match the data type of the HMI array.

Each link command "read count" is limited to the device/protocol buffer length.
Original buffer, 484 bytes.
Large buffer, specified in the port configuration.

Supported types ***

Data type (AB)	HMI array type	Controller byte count
BOOL	Boolean	1 See notes
DINT (32 bit signed integer)	Integer	4
INT (16 bit signed integer)	Small integer	2
LINT (64 bit signed integer)	Integer (64 bit)	~~8~~
~~LREAL (64 bit, floating point number)~~	~~Float~~	~~1 See notes~~
REAL (32 bit floating point number)	Float	4 See notes
SINT (8 bit signed integer)	Short integer	1
STRING	String	Defined by string type
~~UDINT (32 bit unsigned integer)~~	~~Long word See notes~~	~~4~~
~~UINT (16 bit unsigned integer)~~	~~Word~~	~~2~~
~~ULINT (64 bit unsigned integer)~~	~~Unsigned integer (64 bit)~~	~~8~~
~~USINT (8 bit unsigned integer)~~	~~Byte~~	~~1~~

*** Our controller only supported BOOL, DINT, INT, REAL, SINT and STRING. Our testing was limited to the those types. Contact support if reading/writing is not as expected. Support for the other types should be soon.

Syntax

values:=UArray('<HMI array name>', 'Port', [<destination dimension>], ['Link', '<HMI port name>', '<source tag name/dimension>', <count elements to read>]);

Result

values[0] = 0 (zero) or error code
values[1] = 0

Example

```
values:=UArray('Turbine1Pressures',          //HMI array name
               'Port',                       //command
               [0],                          //destination array dimension
               ['Link',                      //sub command
                'Turbine1',                  //AB Logix port name
                'Pressures[0]', //controller tag name and starting dimension
                42]);                        //elements to read (count)
                                             see notes
```

Notes:

1) The HMI array must be sized to contain all the data from the read request. The array can be larger than the requested count.
2) More than one "port link" can be linked to an array.
3) A link command only needs to be added once. The read will be added to a repeating fetch queue.
4) **Booleans:** in the controller, are stored in 32 bit increments. A boolean array, size 512, is stored as 16 consecutive, 32 bit values. Divide the number of boolean values to read by 32, to determine the count value.
5) **String:** Specify the string structure via the name collected when fetching the string types.
   Examples:
   values:=UArray('Turbine1Pressures', 'Port', [0], ['Link', 'Turbine1', 'Pressures[0]', 42, **'String20'**]);

   values:=UArray('Turbine1Pressures', 'Port', [0], ['Link', 'Turbine1', 'Pressures[0]', 19, **'STRING'**]);
6) The HMI attempts to validate the size of arrays, request configuration, buffer sizes, etc. but, some array access configuration data does not provide enough information for a definitive answer. For example, string lengths are not known until the string is returned from the controller. Care must be exercised, when configuring the "Port/Link" command, to ensure success.
7) An error in the port diagnostics
   a. containing ": 2105" indicates attempting to read past the end of an array.
   b. containing ": 06" indicates attempting to read more data than the buffer can hold. Break the data into 2 or more links/reads.
   c. containing ": 04" indicates a controller tag name error.
8) **Floats (REAL, LREAL):** An option field is required to specify the float size, 32 (REAL) or 64 (LREAL).
   Examples:
   values:=UArray('Turbine1Pressures', 'Port', [0], ['Link', 'Turbine1', 'Pressures[0]', 42, **32**]);
   values:=UArray('Turbine1Pressures', 'Port', [0], ['Link', 'Turbine1', 'Pressures[0]', 42, **64**]);

Write

The write command "writes" array elements to the port device. The data type of the array determines how the data is written. The write is added to the port write queue and is processed according to the port read/write logic.

Syntax

values:=UArray('<HMI array name>', 'Port', [<source dimension>], ['Write', '<HMI port name>', '<destination tag name/dimension>', <count elements to write>]);

Result

values[0] = 0 (zero) or error code
values[1] = 0

Example

```
values:=UArray('Turbine1Pressures',     //HMI array name
               'Port',                  //command
               [0],                     //source array dimension
               ['Write',                //sub command
                'Turbine1',             //AB Logix port name
                'Pressures[0]',         // tag name and starting dimension
                42]);                   //elements to write (count)
```

String

Each string in the controller is assumed to be defined as a length field (DINT) followed by a character array (SINT). The write command must include the name of a fetched string type.
Example:
values:=UArray('Pmp1', 'Port', [0], ['Write', 'TB1', 'SAP[0]', 42, **'String20'**]);

The write command must look up the correct "type code" and size for the string. Be sure to "fetch" the string templates at design time. Without a successful fetch operation and saving the project, the command will fail.

Floats (REAL, LREAL)

An option field is required to specify the float size, 32 (REAL) or 64 (REAL).
Examples:
values:=UArray('Turbine1SP', 'Port', [0], ['Link', 'Turbine1', 'SAP[0]', 42, **32**]);
values:=UArray('Turbine1SP', 'Port', [0], ['Link', 'Turbine1', 'SAP[0]', 42, **64**]);

Notes:

1) A "Write error" in the port diagnostics would create an event log entry.
   a. An entry containing "Codes : CD 00 FF 01 07 21" indicates the tag type used does not match the controller tag type.
   b. An entry containing "Codes : CD 00 FF 01 05 21" indicates attempting to write past the end of the array.

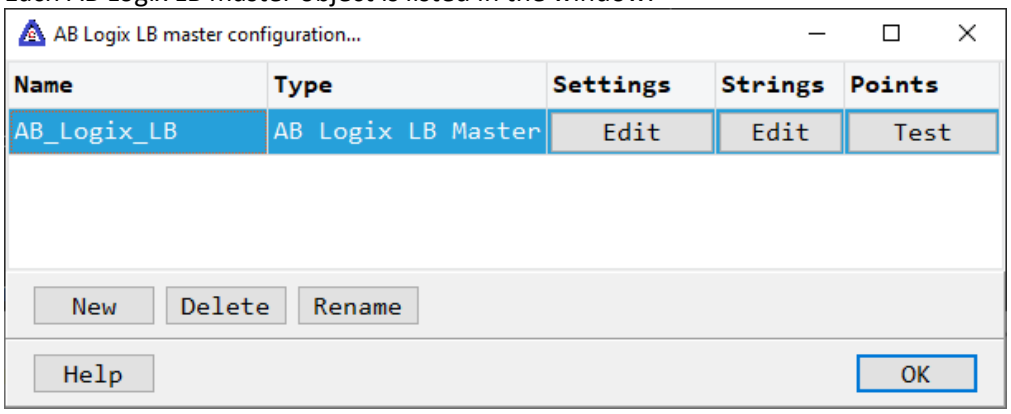
2) **Booleans** in the controller, are stored in 32 bit increments. A boolean array, size 512, is stored as 16 consecutive, 32 bit values. The write count is the number of booleans to write to the controller. The HMI will pack the booleans into 32 bit groups. The destination array starting dimension is the 32 bit word boundary. I.e. CogArray[0] = boolean 1, CogArray[1] = boolean 32, CogArray[2] = boolean 64. The controller only allows writes on the 32 bit boundary.

AB Logix (Large buffer)

See here for AB Logix (Original buffer)

Note: The “AB Logix Large buffer” driver uses the “Large Forward Open” service. The device must support the service. If the device does not support the service, use the AB Logix (Original buffer) driver. Use the “Port test” feature to verify the device is present and supports the service.

Each AB Logix LB master object is listed in the window.

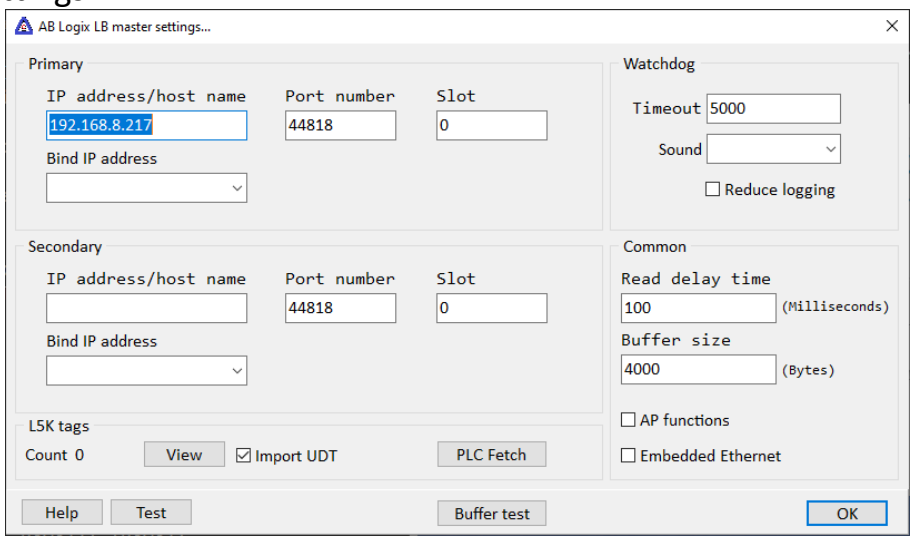


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an AB Logix LB master object select the "Delete" button.

Settings



The TCP port has a primary configuration and if the secondary IP address/host name is not blank, a secondary configuration. Select the configuration attributes as needed.

Slot

The rack/chassis slot of the controller. For CompactLogix the slot must be “0” (zero).

Bind IP address See here.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Reduced logging

When enabled and a watchdog timeout occurs, a single entry will be placed in the event log and the sound will be queued (if configured). When the device responds, after a watchdog timeout, a single entry will be placed in the event log. The watchdog counter will continue to count each time the watchdog timer expires.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests. A value of 0 (zero) disabled the read delay timer.

Operation:
The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out, it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is placed in the event log.

Buffer size

The buffer size is adjustable. The default of 4000 bytes is normally suitable. Use the "Buffer test" as needed.

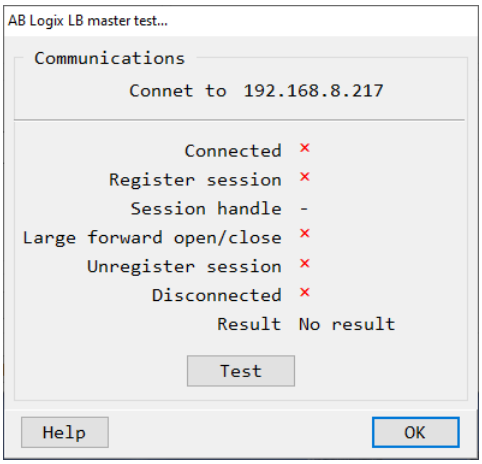
AP functions

See analog functions.

Embedded Ethernet

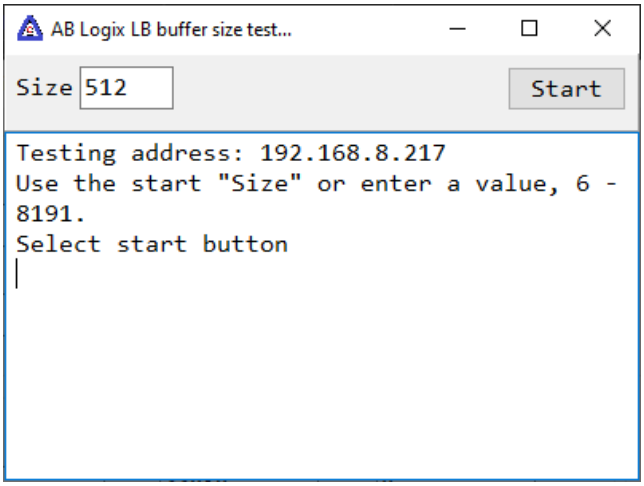
If using the Ethernet port on a controller, enable this property.

Test button



When the test button is selected the program will attempt to connect to the device and create a connection using the buffer size configured.

Buffer test



The external device's maximum buffer size can be tested using the "Buffer Test". The maximum size we found during testing was 4003 bytes. The default buffer size is 4000 bytes.

PLC Fetch

The program will attempt to connect to the PLC, collect the controller tags, program tags, data structures, etc. and display the results. Depending on connection speed, number of tags, etc., a window may appear and will close automatically when the collection is complete. If a successful collection, the "Create points" window will appear.

The point creation option is the same as the AB Logix (Original buffer).

L5K Tags

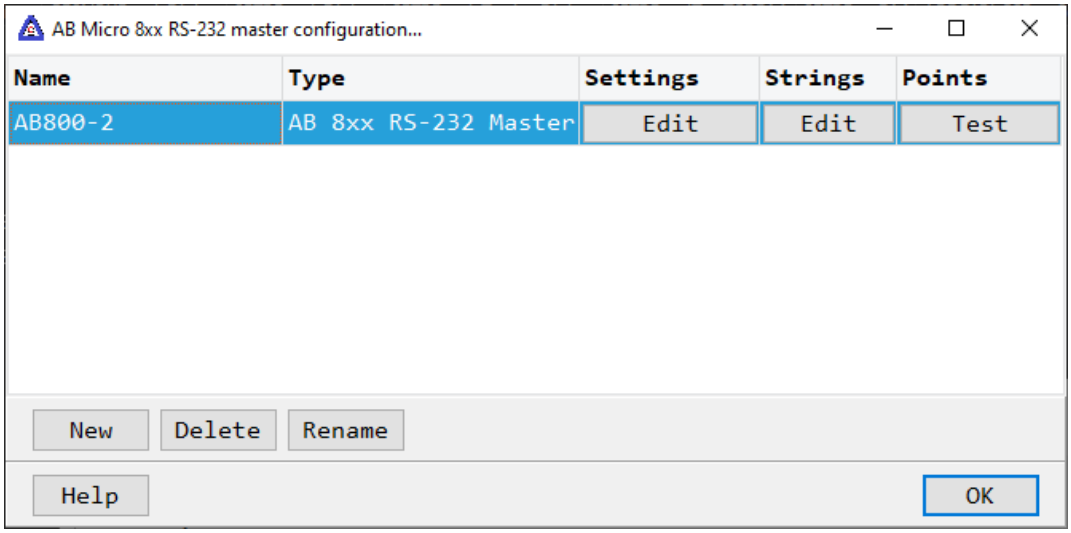
The L5K collection and point create is the same as the AB Logix (Original buffer).

Points

The points are the same as the AB Logix (Original buffer).

AB Micro 8xx RS-232

Each AB Micro 8xx RS-232 master object is listed in the window.

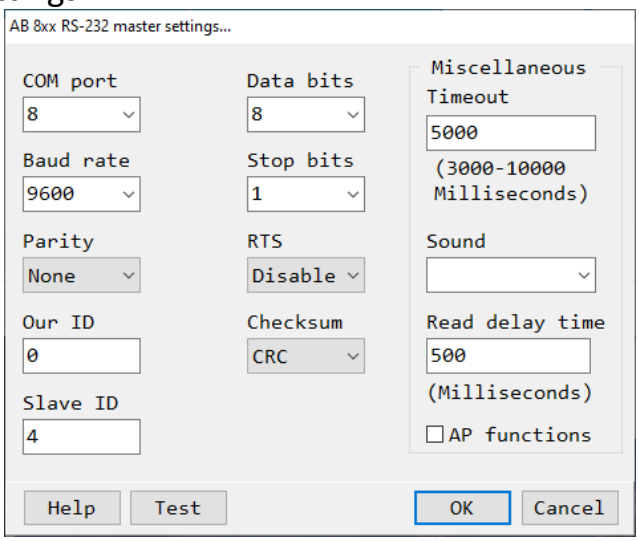


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an object select the "Delete" button.

Settings



Miscellaneous

RTS (Request to send)

Disabled: Disables the RTS line when the device is opened and leaves it disabled.

Enabled: Enables the RTS line when the device is opened and leaves it active.

Handshake: Enables RTS handshaking. The driver raises the RTS line when the "type-ahead" (input) buffer is less than one-half full and lowers the RTS line when the buffer is more than three-quarters full.

Toggle: Specifies that the RTS line will be high if bytes are available for transmission. After all buffered bytes have been sent, the RTS line will be low.

Timeout

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

The Micro 8xx supports a limited number of CIP services. To provide throttling of data request, all the points are “polled” and then the “Read delay timer” begins timing.

When the timer completes the process is repeated. Allowable values: 0 = no time delay, 100 – 2,147,483,647 milliseconds If a write command is issued while the timer is active, the timer is cancel and the read/write logic is executed

To prevent flooding due to write request the next read request is always issued after a write request.

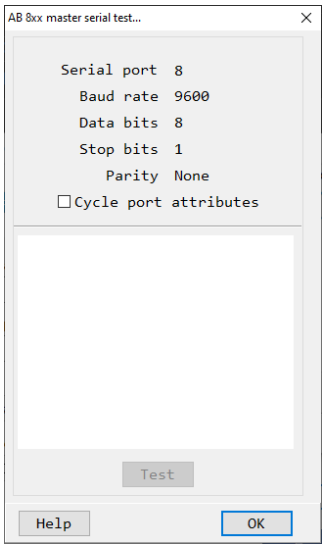
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

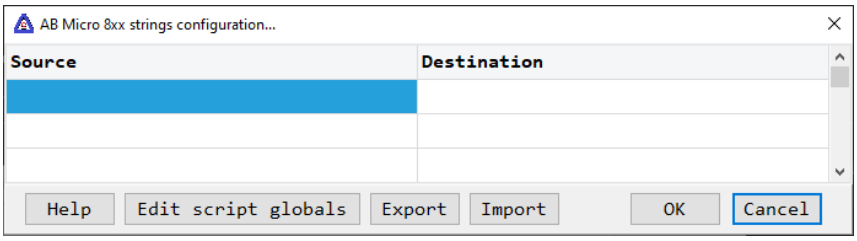
See analog functions.

Test button



When the test button is selected the program will attempt to connect to the device and read the identity of the device. The above is an example, other results will be different.

Strings



Strings are not regular points. They do not have alarms, printing, etc. Strings may be displayed in any window via the "Script Global" animation. Any other actions, parsing, comparing, etc. must be done in scripts.

Source

This is the tagname in the PLC. It must be a valid tagname with a datatype "string".

Destination (optional)

If desired, select a script global location and the string will be copied to the location when the string value is returned from the external device. If this option is used or not the string may still be accessed via the "StringGet" and "StringSet" script commands. If this is configured and access to the string, in a script, is desired, use the "GlobalGet" or "StringGet" to copy the string. The format of the destination is: <section>.<name>.
Note: Writing to the script global does not write the value to the PLC. "StringSet" must be used to write a string to the PLC.

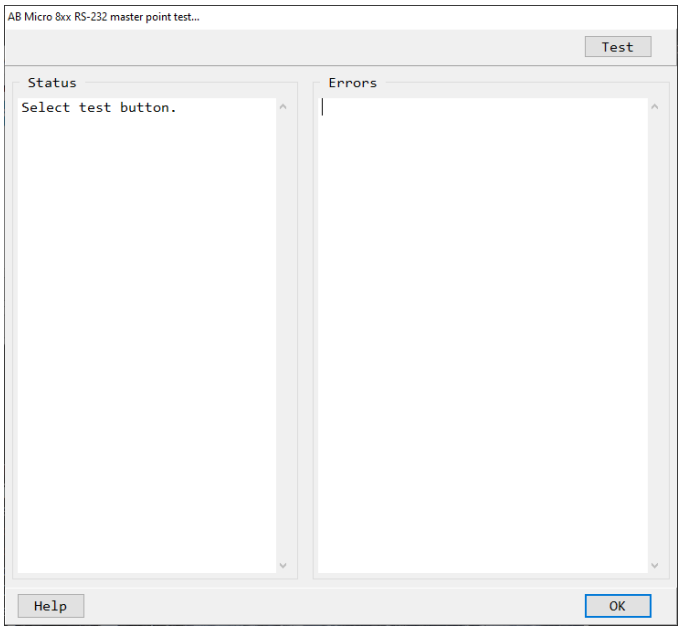
Export

This is used to export the configuration to an Excel worksheet.

Import

This is used to import string configurations from an Excel worksheet. The contents of the grid are completely replaced by the contents of the file.

Points



This feature is used to test if a point or string, with the source address entered, exist in the external device.

Test

Selecting this button will execute the test. All the configured points and strings will be tested. The test is very fast.

Note: Right click and copy the contents of either text area.

Status

This area is to display general information regarding the test and testing progress as well as displaying each point/string success. Each point/string tagname will be displayed with a 'success' suffix if the source address is in the external device.

Example:
Connecting...Connected
Session registered
CPU test successful
Point testing begins...
Alarm_Active...
Close_Discharge_Valve_CRT... success
CRT_Acknowledge... success

The first four lines are general status updates. The first point 'Alarm_Active', failed and the reason will be displayed in the 'Error' area. (See below) The remaining lines each indicate success.

Error

This area is to display the error that was generated for a source address. Several types of error are possible. Each point error has three lines. Examples:

Tagname: Alarm_Active
Source address: Alarm_Active1
Error: General Status Error:05

Tagname: Discharge_Value_Close_Timer.ACC
Source address: 1Discharge_Value_Close_Timer.ACC
Error: General Status Error:05

Tagname: N7[0]
Source address: N7[0]
Error: Point is digital, returned data type is not digital

The 'General Status Error:05' is the most common when a source address does not exist in the external device.

The other error in the example is the point type does not match the data type of the source address. In the above example the point type is digital and the data for the source address is not digital. In this case it was a two byte integer.
Each string type has two lines. Example:

Source address: string1
Error: General Status Error:05

The structure data type covers many different collections of data. If the data type returned is one of the atomic data types, i.e. Boolean, real, etc. an error will be flagged.

Other possible error codes:

Error code(hex)	Description
04	A syntax error was detected decoding the Request Path.
05	Request Path destination unknown: Probably instance number is not present.
06	Insufficient Packet Space: Not enough room in the response buffer for all the data.
13	Insufficient Request Data: Data too short for expected parameters.
26	The Request Path Size received was shorter or longer than expected.

Source address

The HMI uses controller tagnames to access controller data. The HMI supports: boolean, small integer (byte), integer, double integer, floats and strings. Arrays and structures (timers, control, etc.) are also supported at the structure field level.

There are two special characters. The colon (:) and the period (.). The colon is used to indicate a program name and the period is used to indicate a field name.

Format

The source address must match the controller tagname. If the data type is a simple type then the source address and controller tagname must match exactly.
Examples of possible source address.

Tagname	Source address	Data type
Door_Open	Door_Open	Boolean
Door_Closed	Door_Closed	Boolean
Water_Press	Water_Press	Real
Water_Temp	Water_Temp	Integer
Tank_Level	Tank_Level	String
Open_Time	Open_Time.ACC	Timer (accumulator)
Open_Done	Open_Done.DN	Timer (done)

For access to array elements add the array bounds.

Door_Open	Door_Open[0]	Array[8] of Boolean
Door_Closed	Door_Closed[2]	Array[8] of Boolean
Water_Press	Water_Press[3]	Array[8] of Real
Water_Temp	Water_Temp[7]	Array[8] of Integer
Open_Time	Open_Time[0].ACC	Array[8] Timer (accumulator)
Open_Done	Open_Done[0].DN	Array[8] Timer (done)

For multi-dimensional arrays.

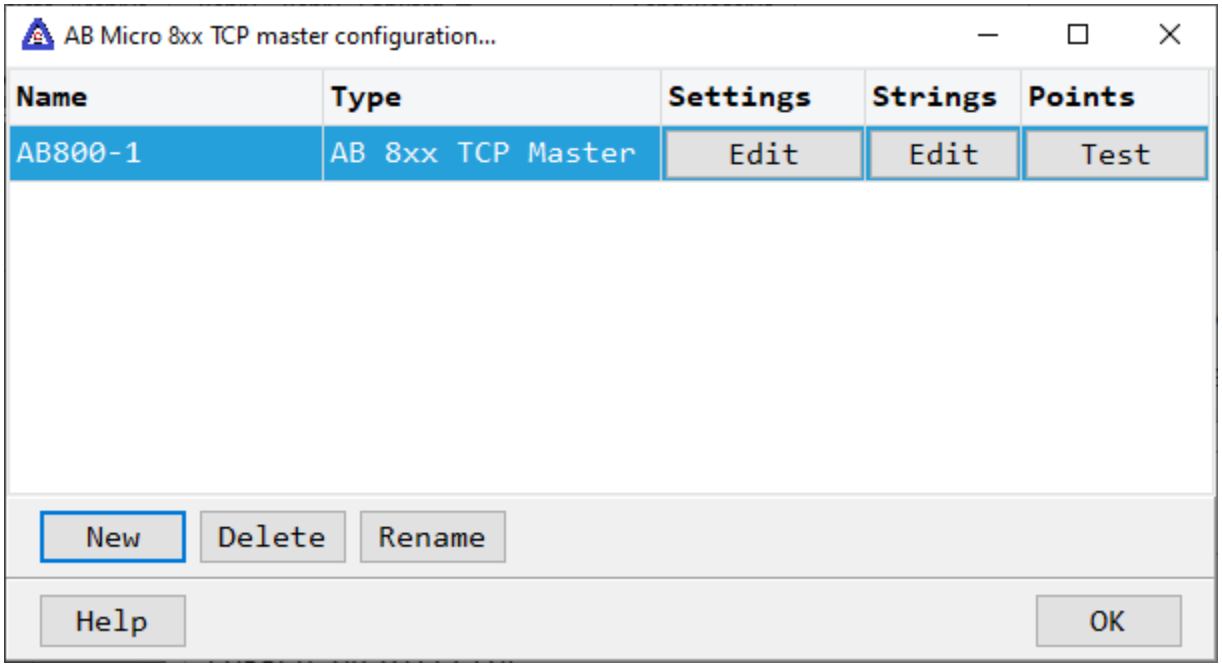
Door_Open	Door_Open[0,1,2]	Array[8,8,8] of Boolean
Water_Press	Water_Press[3,2]	Array[8,4] of Real

If the tag is in a program, prefix the program name with a colon <programName:>tagname.

First:Door_Open	Door_Open[0]	Array[8] of Boolean
First:Door_Closed	Door_Closed[2]	Array[8] of Boolean
First:Water_Press	Water_Press[3]	Array[8] of Real
First:Water_Temp	Water_Temp[7]	Array[8] of Integer

AB Micro 8xx TCP

Each AB Micro 8xxTCP master object is listed in the window.

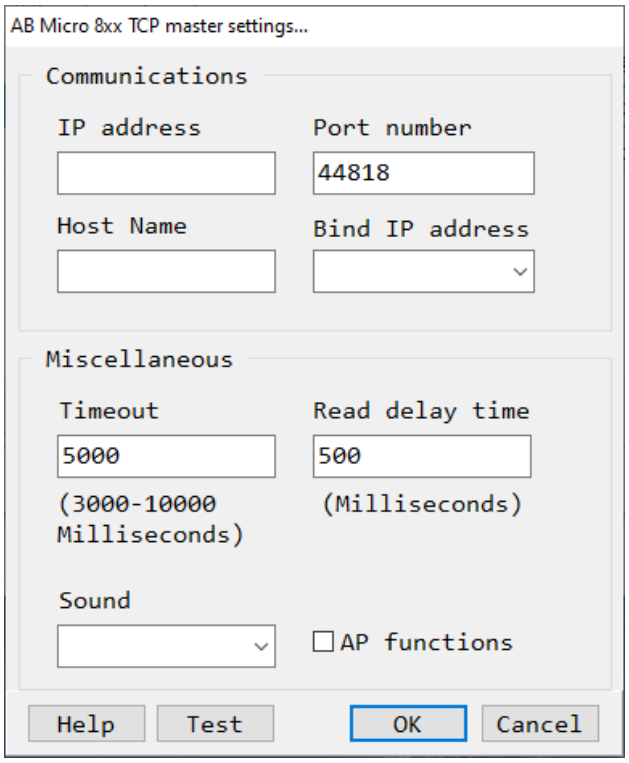


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an AB Micro 8xx TCP master object select the "Delete" button.

Settings



If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Miscellaneous

Timeout

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

The Micro 8xx supports a limited number of CIP services. To provide throttling of data request, all the points are “polled” and then the “Read delay timer” begins timing. When the timer completes the process is repeated. Allowable values: 0 = no time delay, 100 – 2,147,483,647 milliseconds If a write command is issued while the timer is active, the timer is cancel and the read/write logic is executed

To prevent flooding due to write request the next read request is always issued after a write request.

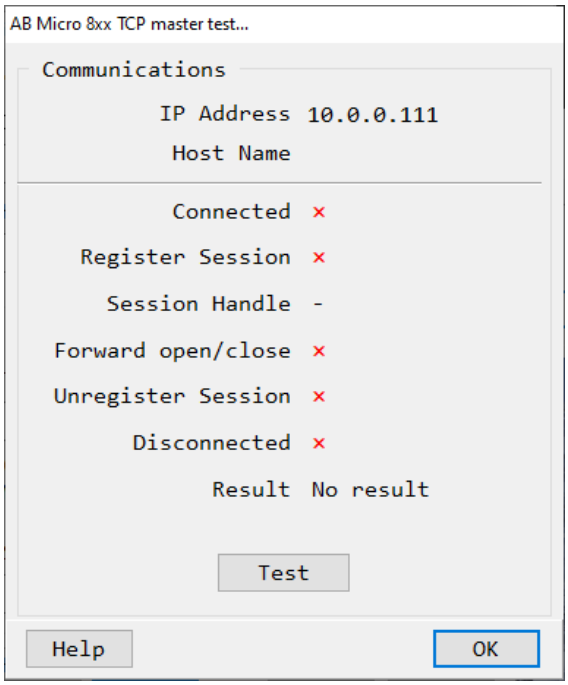
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button



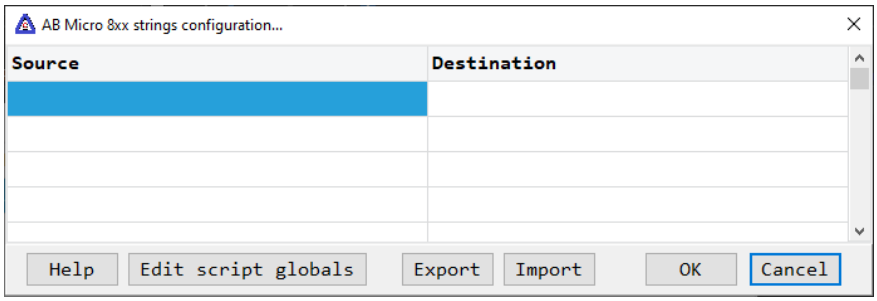
When the test button is selected the program will attempt to connect to the device, register a session perform a “forward open/close” and the, unregister the session and close the connection.

If a "host name" is entered, the IP address is ignored.

Connected:	Connection to device
Register Session:	Request a session handle
Session Handle:	Session handle value
Forward open/close:	Command to the controller
Unregister Session:	Release the session handle
Disconnected:	Disconnect from the ENET card
Result:	Test result
Failed:	Could not connect to the device card

Partial Success: Connected to the device, failed open/close response
Success: Connected to device, successful device response
A partial success result is normally due to, too many active connections.

Strings



Strings are not regular points. They do not have alarms, printing, etc. Strings may be displayed in any window via the "Script Global" animation. Any other actions, parsing, comparing, etc. must be done in scripts.

Source

This is the tagname in the PLC. It must be a valid tagname with a datatype "string".

Destination (optional)

If desired, select a script global location and the string will be copied to the location when the string value is returned from the external device. If this option is used or not the string may still be accessed via the "StringGet" and "StringSet" script commands. If this is configured and access to the string, in a script, is desired, use the "GlobalGet" or "StringGet" to copy the string. The format of the destination is: <section>.<name>.
Note: Writing to the script global does not write the value to the PLC. "StringSet" must be used to write a string to the PLC.

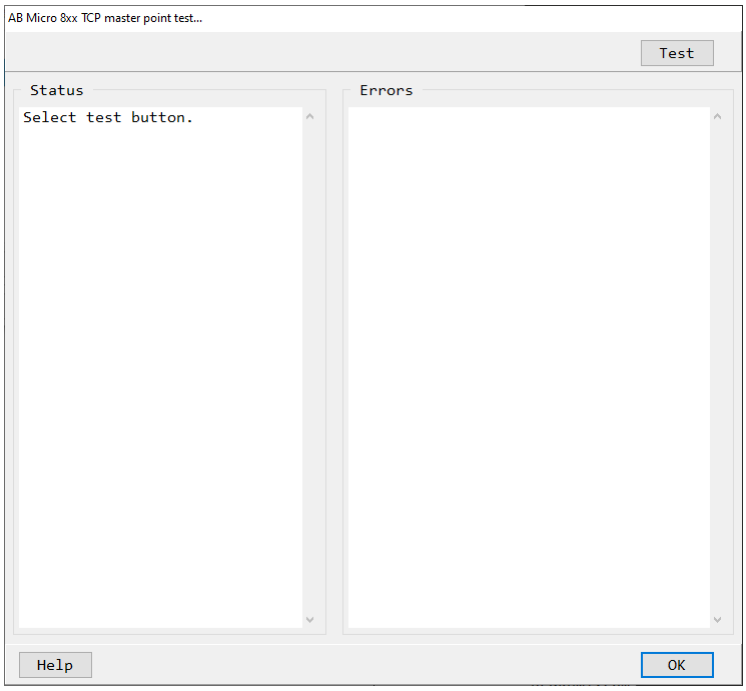
Export

This is used to export the configuration to an Excel worksheet.

Import

This is used to import string configurations from an Excel worksheet. The contents of the grid are completely replaced by the contents of the file.

Points



This feature is used to test if a point or string, with the source address entered, exist in the external device.

Test

Selecting this button will execute the test. All the configured points and strings will be tested. The test is very fast.

Note: Right click and copy the contents of either text area.

Status

This area is to display general information regarding the test and testing progress as well as displaying each point/string success. Each point/string tagname will be displayed with a 'success' suffix if the source address is in the external device.

Example:
Connecting...Connected
Session registered
CPU test successful
Point testing begins...
Alarm_Active...
Close_Discharge_Valve_CRT... success
CRT_Acknowledge... success

The first four lines are general status updates. The first point 'Alarm_Active', failed and the reason will be displayed in the 'Error' area. (See below) The remaining lines each indicate success.

Error

This area is to display the error that was generated for a source address. Several types of error are possible. Each point error has three lines. Examples:

Tagname: Alarm_Active
Source address: Alarm_Active1
Error: General Status Error:05

Tagname: Discharge_Value_Close_Timer.ACC
Source address: 1Discharge_Value_Close_Timer.ACC
Error: General Status Error:05

Tagname: N7[0]
Source address: N7[0]
Error: Point is digital, returned data type is not digital

The 'General Status Error:05' is the most common when a source address does not exist in the external device.

The other error in the example is the point type does not match the data type of the source address. In the above example the point type is digital and the data for the source address is not digital. In this case it was a two byte integer.
Each string type has two lines. Example:

Source address: string1
Error: General Status Error:05

The structure data type covers many different collections of data. If the data type returned is one of the atomic data types, i.e. Boolean, real, etc. an error will be flagged.

Other possible error codes:

Error code(hex)	Description
04	A syntax error was detected decoding the Request Path.
05	Request Path destination unknown: Probably instance number is not present.
06	Insufficient Packet Space: Not enough room in the response buffer for all the data.
13	Insufficient Request Data: Data too short for expected parameters.
26	The Request Path Size received was shorter or longer than expected.

Source address

The HMI uses controller tagnames to access controller data. The HMI supports: boolean, small integer (byte), integer, double integer, floats and strings. Arrays and structures (timers, control, etc.) are also supported at the structure field level.

There are two special characters. The colon (:) and the period (.). The colon is used to indicate a program name and the period is used to indicate a field name.

Format

The source address must match the controller tagname. If the data type is a simple type then the source address and controller tagname must match exactly.
Examples of possible source address.

Tagname	Source address	Data type
Door_Open	Door_Open	Boolean
Door_Closed	Door_Closed	Boolean
Water_Press	Water_Press	Real
Water_Temp	Water_Temp	Integer
Tank_Level	Tank_Level	String
Open_Time	Open_Time.ACC	Timer (accumulator)
Open_Done	Open_Done.DN	Timer (done)

For access to array elements add the array bounds.

Door_Open	Door_Open[0]	Array[8] of Boolean
Door_Closed	Door_Closed[2]	Array[8] of Boolean
Water_Press	Water_Press[3]	Array[8] of Real
Water_Temp	Water_Temp[7]	Array[8] of Integer
Open_Time	Open_Time[0].ACC	Array[8] Timer (accumulator)
Open_Done	Open_Done[0].DN	Array[8] Timer (done)

For multi-dimensional arrays.

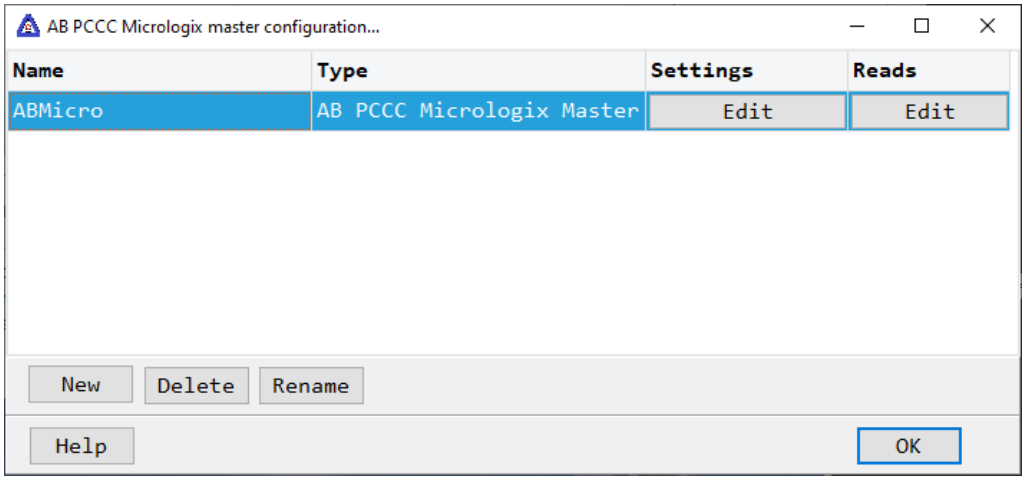
Door_Open	Door_Open[0,1,2]	Array[8,8,8] of Boolean
Water_Press	Water_Press[3,2]	Array[8,4] of Real

If the tag is in a program, prefix the program name with a colon
<programName:>tagname.

First:Door_Open	Door_Open[0]	Array[8] of Boolean
First:Door_Closed	Door_Closed[2]	Array[8] of Boolean
First:Water_Press	Water_Press[3]	Array[8] of Real
First:Water_Temp	Water_Temp[7]	Array[8] of Integer

AB PCCC Micrologix

Each AB PCCC Micrologix master object is listed in the window.

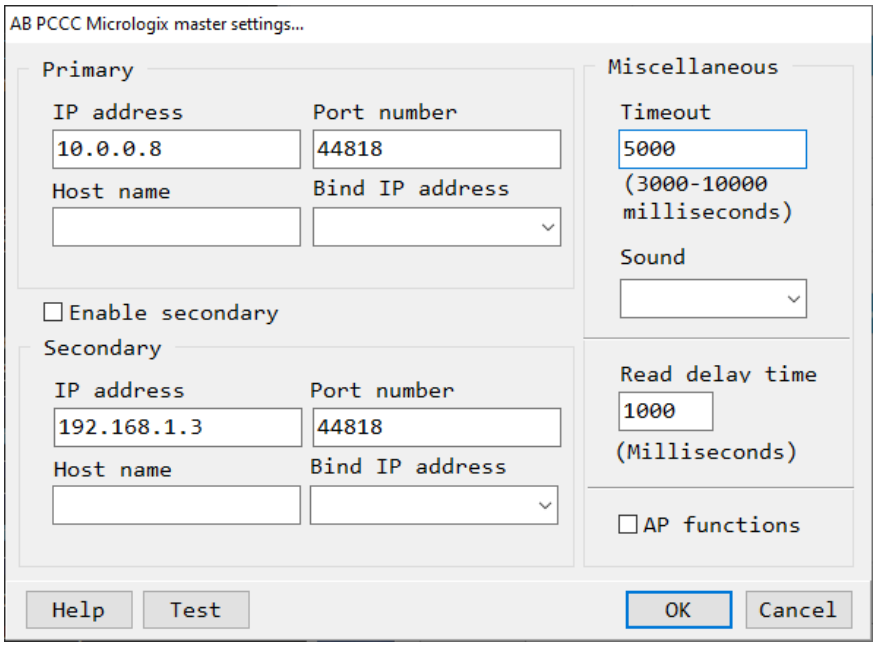


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an AB PCCC Micrologix master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

The TCP port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

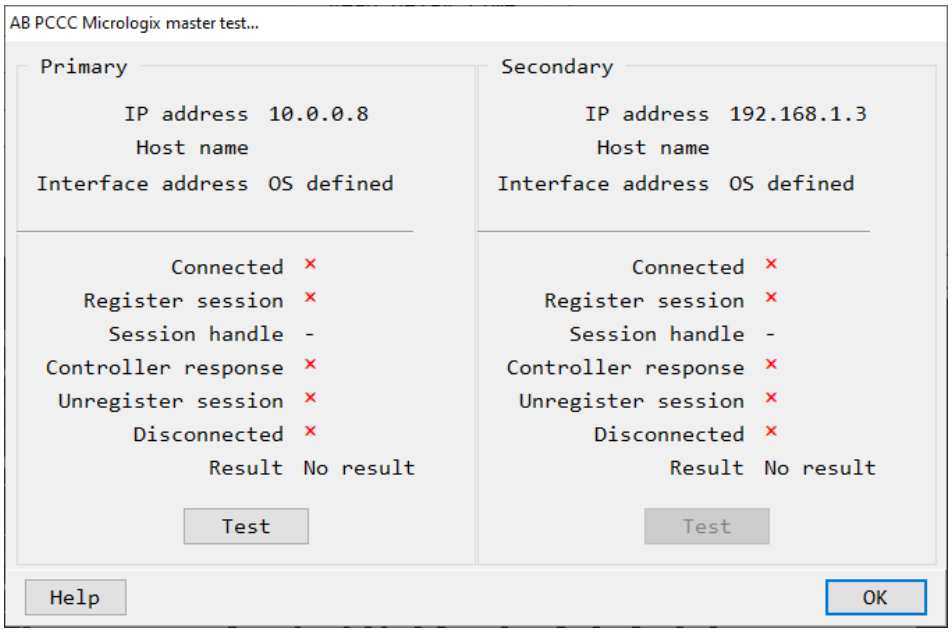
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button



The program will attempt to connect to the controller, register the session and perform a "Forward Open." Use the reads testing for more verification.

Reads

AB PCCC Micrologix reads...

#	File type	File/Slot #	Start element	Count	Enabled	Testing
1	Input	1	0	4	☐	Test
2	None				☐	Test
3	None				☐	Test
4	None				☐	Test
5	None				☐	Test
6	None				☐	Test
7	None				☐	Test
8	None				☐	Test
9	None				☐	Test
10	None				☐	Test
11	None				☐	Test
12	None				☐	Test

Help OK Cancel

The address ranges shown may or may not be present in the slave device.

The HMI uses the AB defined addressing for access to the data files in the PLC. Please refer to the AB documentation.

The DF1 “Duplicate Message Detection” logic may be enabled in the communication device. If more than one read command is enabled the detection logic will not cause a problem. If only one read command is enabled the “Duplicate Message Detection” logic must be disabled. (Or create another read command.)

Example address:

Common

N7:10, N7:451/0, N7:33/15
T4:152/TT,T4:152/EN,T4:152.ACC
B3:45/5
B3/567

I:0.0/00, I:1.1/12, I:4.4/15
O:0.0/01, O:1.0/16, O:8.1/31
The format of the address must be used.
I:X.X/XX
O:X.X/XX
File Type : Slot . Word / Bit

File type

Example: Integer (N), Input (I), Binary (B)

File/Slot

Legal file numbers are 3 to 255. For inputs and outputs the file number is the slot number.

Start element

This is the starting element for the file type to read. The different file types have different element counts. Example: Integer (N) has a one word element; Timer (T) has a three word element.

Count

This is the number of elements to read. The file type determines the maximum number of elements that can be read in a single command.

The maximum is 122 words (244 bytes) of data.

Enabled

The read requests are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test

AB PCCC Micrologix reads testing...

Address	17 - 0	Value : Int
I:1.0	0000 0000 0000 0000	0
I:1.1	0000 0000 0000 0000	0
I:1.2	0000 0000 0000 0000	0
I:1.3	0000 0000 0000 0000	0

Exit

When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

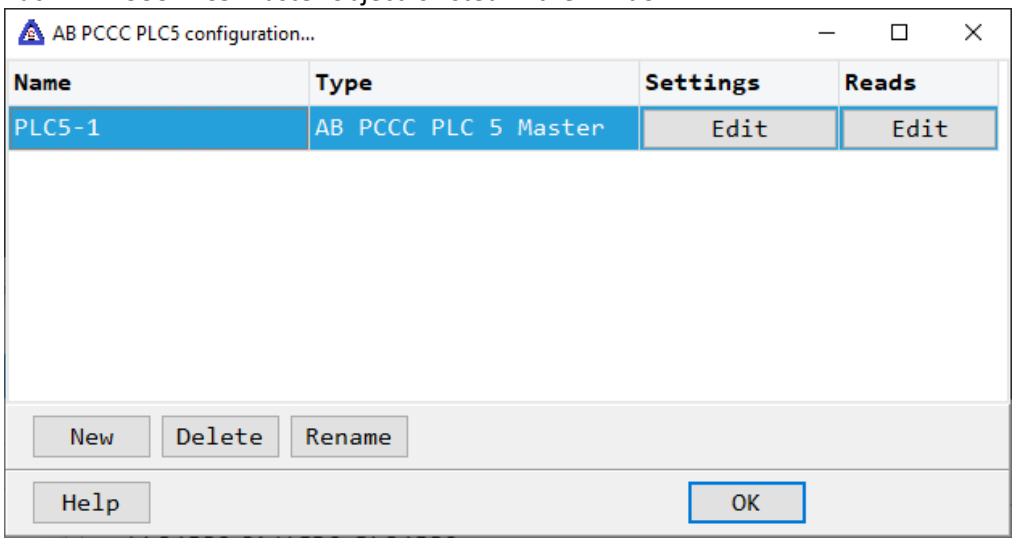
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. If the error string contains 'STS' or 'EXT STS' the text that follows is the error code ,in hexadecimal, returned from the external device. The HMI uses a function code $0F. PCCC, assuming the port test succeeded, if "Forward open success..." is displayed and does not change, close the testing window and reattempt the test.

Error messages

No file type selected	A file type must be selected.
Element out of range	Start element must be: 0-255
Count exceeds limit	The number of elements to read is too high for the file type.
Element start + count exceeds limit	Element start + count greater than file end.
Count < 1	Must read at least one element
File number out of range	The file number must be: 0-255
Invalid file type	The file number does not match the assigned file type.

AB PCCC PLC5

Each AB PCCC PLC5 master object is listed in the window.

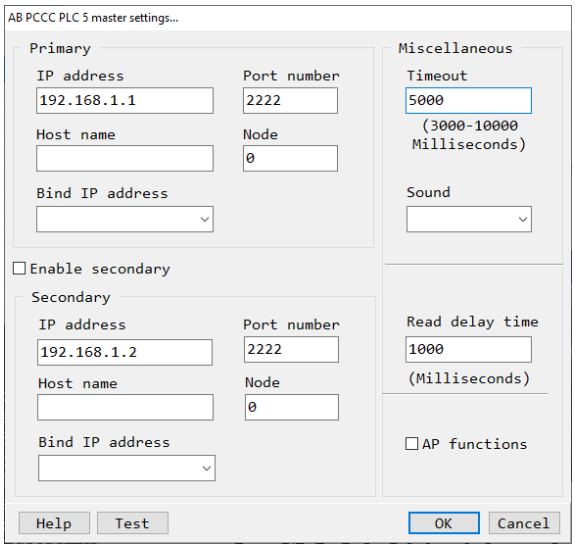


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an AB PCCC PLC5 master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

The TCP port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

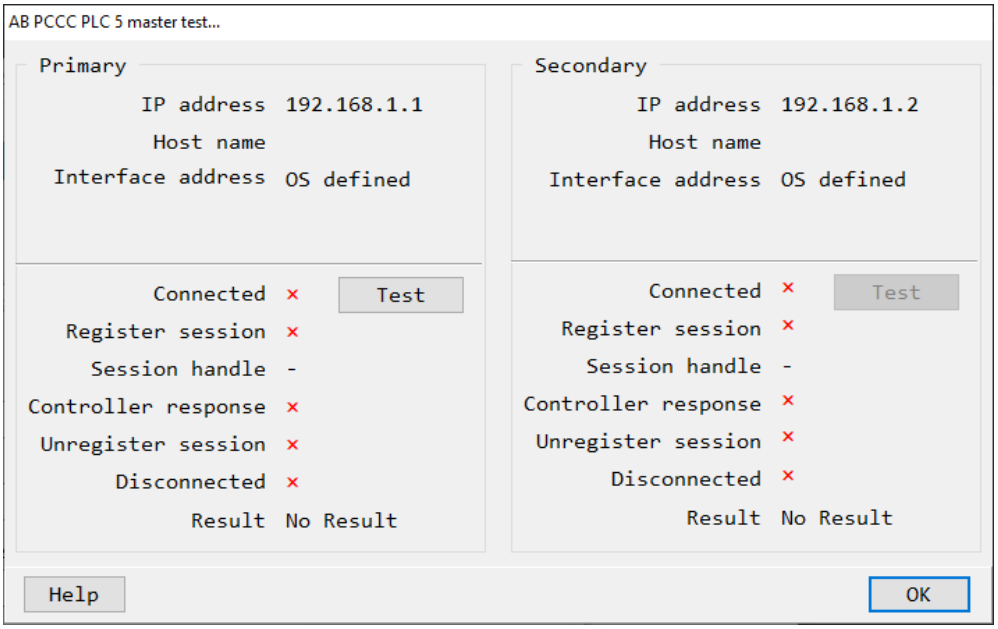
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

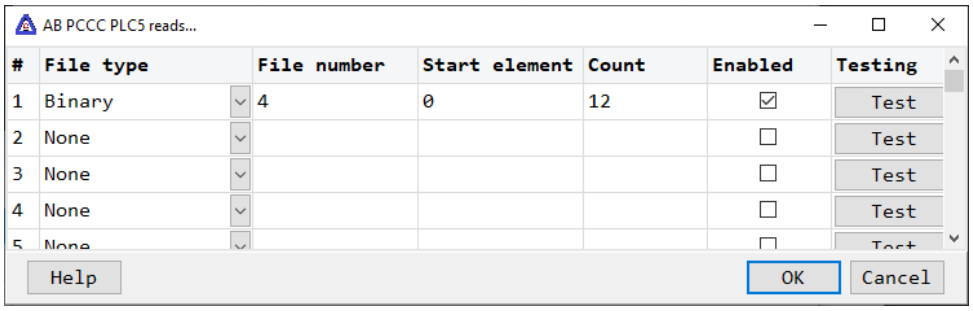
See analog functions.

Test button



The program will attempt to connect to the controller, register the session and perform a "Forward Open". Use the reads testing for additional verification.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the AB defined addressing for access to the data files in the PLC. Please refer to the AB documentation.

The DF1 Duplicate Message Detection' logic may be enabled in the communication device. If more than one read command is enabled the detection logic will not cause a problem. If only one read command is enabled the Duplicate Message Detection' logic must be disabled. (Or create another read command)

Example address:

Common

N7:10, N7:451/0, N7:33/15
T4:152/TT,T4:152/EN,T4:152.ACC
B3:45/5
B3/567

I:001/00, I:010/15
O:001/01, O:012/17

Note: Input and output references are in octal for point addressing. The input and output reference must contain the complete address. The value after the colon must be three digits and any bit references must be two digits. Example I:XXX or O:XXX I:001, O:005, I:001/01, O:005/17, I:022/01, O:019/17, I:167/00, O:177/17

File type

Example: Integer (N), Input (I), Binary (B). See notes.

File

Legal file numbers are 3 to 999. File number 0 (zero) is for outputs, file number 1 (one) is for inputs and file number 2 is for status. If the file type is input, output, or status the file number is set to the correct value when the 'OK' button is selected.

Start element

This is the starting element for the file type to read. The different file types have different element counts. Example: Integer (N) has a one word element; Timer (T) has a three word element.

Count

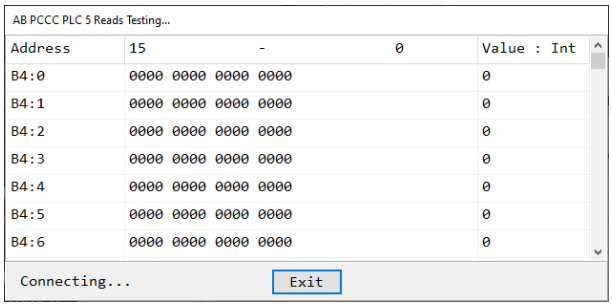
The maximum is 122 words (244 bytes) of data.

Enabled

The read requests are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. If the error string contains 'STS' or 'EXT STS' the text that follows is the error code ,in hexadecimal, returned from the external device. The HMI uses a function code $0F. PCCC, assuming the port test succeeded, if "Forward open success..." is displayed and does not change, close the testing window and reattempt the test.

Error messages

No file type selected	A file type must be selected.
Element out of range	Start element must be: 0-255
Count exceeds limit	The number of elements to read is too high for the file type.
Element start + count exceeds limit	Element start + count greater than file end.
Count < 1	Must read at least one element
File number out of range	The file number must be: 0-255
Invalid file type	The file number does not match the assigned file type.

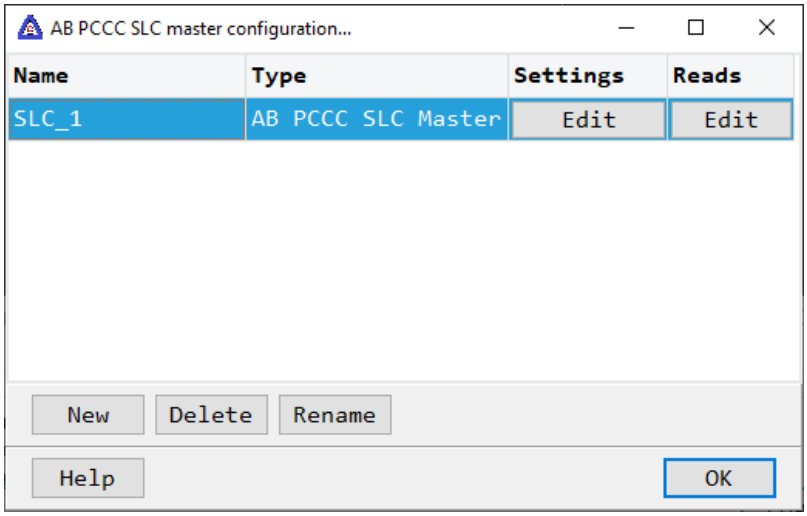
Notes:

1) For PID (PD) file type, only “float” is supported.

Digital				Analog (2 words each)		
Word	**Bit**	**Ext.**	**Description**	**Word**	**Ext.**	**Description**
0	15	/EN	Enabled	2	.SP	Setpoint
0	11	/NOBC	No back calculation	4	.KP	Proportional gain
0	9	/CT	Cascade type	6	.KI	Integral gain
0	8	/CL	Cascade loop	8	.KD	Derivative gain
0	7	/PVT	PV tracking	10	.BIAS	Output bias %,
0	6	/DO	Derivative	12	.MAXS	Engineering unit max.
0	4	/SWM	Set output	14	.MINS	Engineering unit min.
0	2	/CA	Control action	16	.DB	Deadband,
0	1	/MO	Station mode	18	.SO	Set output %,
0	0	/PE	PID equation	20	.MAXO	Output limit high %
1	12	/INI	PID initialized	22	.MINO	Output limit low %
1	11	/SPOR	SP Out of range	24	.UPD	Update time
1	10	/OLL	Output limit low	26	.PV	Process variable
1	9	/OLN	Output limit high	28	.ERR	Error
1	8	/EWD	Error within deadband	30	.OUT	Output %
1	3	/DVNA	Deviation high alarm	32	.PVH	PV alarm high
1	2	/DVPA	Deviation low alarm	34	.PVL	PV alarm low
1	1	/PVLA	PV Low Alarm	36	.DVP	Deviation alarm (+)
1	0	/PVHA	PV High Alarm	38	.DVN	Deviation alarm (-)
				40	.PVDB	PV alarm deadband
				42	.DVDB	Deviation alarm DB
				44	.MAXI	Input range maximum
				46	.MINI	Input range minimum
				48	.TIE	Tieback %

AB PCCC SLC

Each AB PCCC SLC master object is listed in the window.

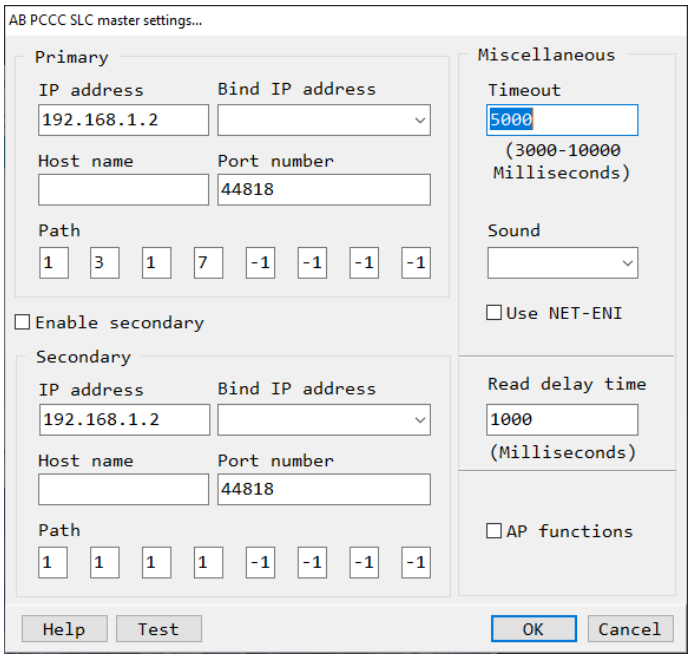


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an AB PCCC SLC master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

The TCP port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Path

Example 1

This is the path to the endpoint (controller). A "-1" value in the path is ignored. The last address in the path is the DH+ node address of the controller. The node address is in octal. Convert the address from octal to decimal and enter that value as the last digit. The testing we did was with a path of four values. We could not discovery a path shorter than four. For a path shorter than 4, please contact technical support.

Rack 1

Slot 0 1756-ENET
Slot 1 CPU
Slot 2 16 Point Digital in
Slot 3 1756-DHRIO

Rack 2

Slot 0 SLC 5/04 (DH+ node address octal 37)
Slot 1-n various cards

To access the SLC in rack 2 the path would be: 1 - 3 - 1 - 31 (with -1 in the remaining fields)

1 = Backplane port of 1756-ENET
3 = Slot 3, 1756-DHRIO
1 = Port A of the 1756-DHRIO, use a 2 for port B
31 = SLC 5/04 (31 is the decimal value for DH+ octal address 37)

Note: On the test setup we used the DH+ port on the 1756-DHRIO has pin 1 at the bottom of the connector and on the SLC 5/04 DH+ port has pin 1 at the top of the connector. When port testing a result of "GSE:1" is "Connection failure: A connection related service failed along the connection path."

Example 2

When using a 1761-NET-ENI set all the path values to -1 and enable the "Use NET-ENI" checkbox. The path is not used when using a 1761-NET-ENI.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

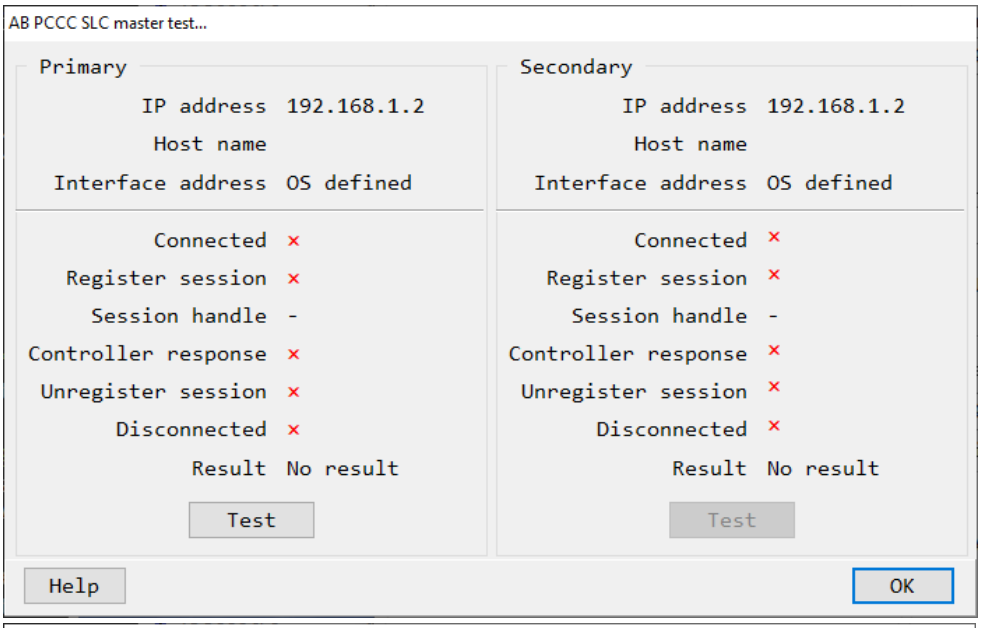
If the watchdog timer times out it fires the read/write processing logic.

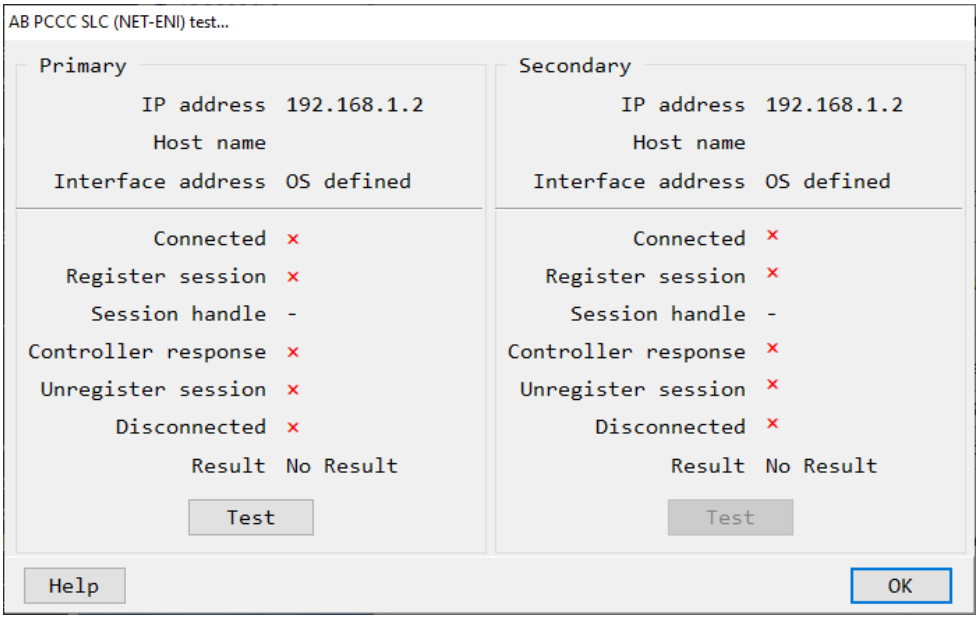
If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button

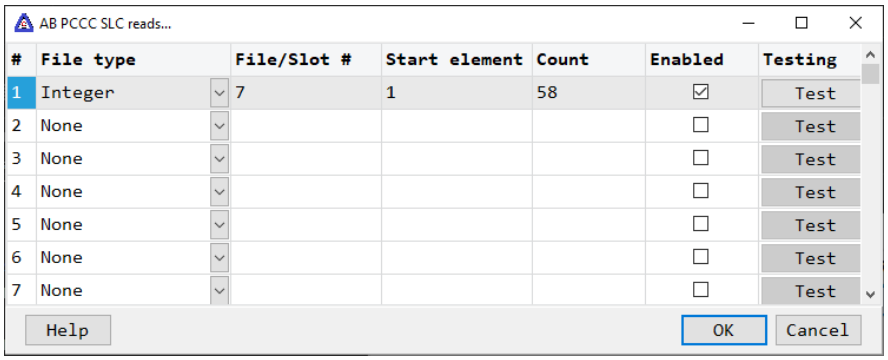




The program will attempt to connect to the controller, register the session and perform a "Forward Open." and after a successful "Forward Open" will attempt to read one word from "N7:0." Verify that this address does in exist in the PLC.

Use the “Reads” testing for more verification. Remember, the last value in the path is the decimal node number. DH+ node numbers are in octal. Convert the octal node number to decimal and enter the value as the last value in the path.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the AB defined addressing for access to the data files in the PLC. Please refer to the AB documentation.

The DF1 “Duplicate Message Detection” logic may be enabled in the communication device. If more than one read command is enabled the detection logic will not cause a problem. If only one read command is enabled the “Duplicate Message Detection” logic must be disabled. (Or create another read command.)

Common

Example address:

N7:10, N7:451/0, N7:33/15
T4:152/TT,T4:152/EN,T4:152.ACC
B3:45/5
B3/567

I:1.0/00, I:1.1/12, I:4.4/15
O:2.0/01, O:7.0/16, O:8.1/31
The format of the address must be used.

I:X.X/XX
O:X.X/XX

File Type : Slot . Word / Bit

Note: When configuring reads if the file type is input or output be aware that setting the count greater than the number of words the card supports, may or may not return data (a successful read). In some cases the PLC will return the data for the selected card (slot) and additional cards (slots) up to the count value.

Example:
Slot 1 1746-IA16 1 word

Slot 4 1747-KE 45 words
Slot 5 1746-NI8 8 words
Slot 6 1746-IA16 1 word

A read
Slot = 1
Start element = 0
Count = 45

For this read the PLC will return 45 words of data. The data past the first word I:1.0 is for other cards.
For the HMI point source address and PLC addressing to match, configure one read for each populated slot (if data is needed) and set the count to the number of words the card utilizes.

File type

Example: Integer (N), Input (I), Binary (B)

File/Slot

Legal file numbers are 3 to 255. For inputs and outputs the file number is the slot number.

Some file numbers are assigned.

#	Type
2	Status
3	Binary
4	Timer
5	Counter
6	Control
7	Integer
8	Floating Point
9-255	User Assigned

If the file type is status the file number is set to the correct value when the 'OK' button is selected.

Start element

This is the starting element for the file type to read. The different file types have different element counts. Example: Integer (N) has a one word element; Timer (T) has a three word element.

Count

For SLC 5/01 or 5/02 = 82 bytes (41 words).

For SLC 5/03 or 5/04 = 236 bytes (118 words).

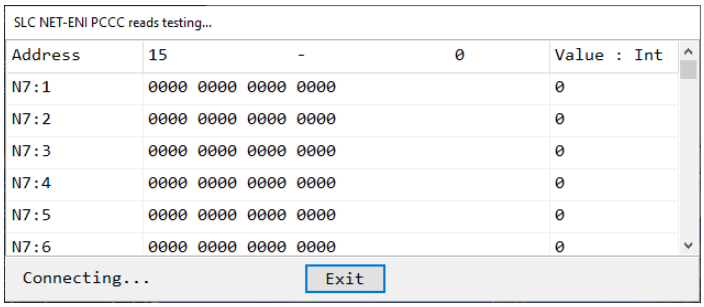
Example: Timer (T) 3 words per element. Maximum count is 40.
Float (F) 2 words per element. Maximum count is 61.

Enabled

The read requests are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. If the error string contains 'STS' or 'EXT STS' the text that follows is the error code ,in hexadecimal, returned from the external device. The HMI uses a function code $0F. PCCC, assuming the port test succeeded, if "Forward open success..." is displayed and does not change, close the testing window and reattempt the test.

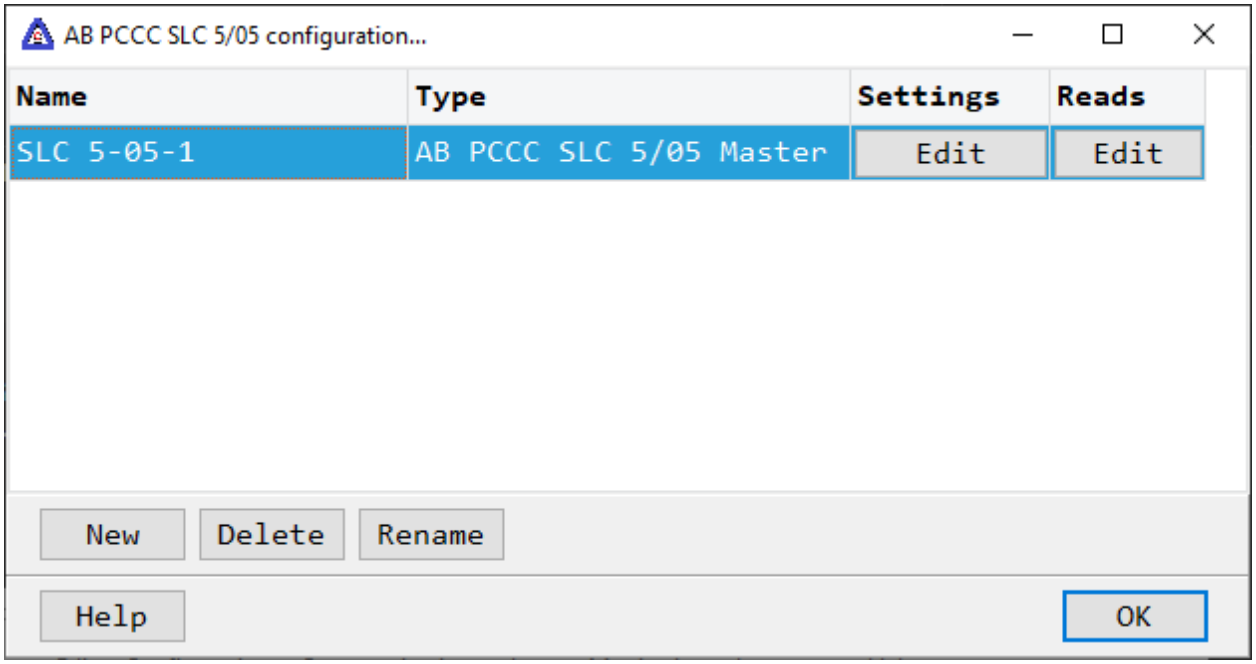
Error messages

No file type selected	A file type must be selected.
Element out of range	Start element must be: 0-255
Count exceeds limit	The number of elements to read is too high for the file type.
Element start + count exceeds limit	Element start + count greater than file end.
Count < 1	Must read at least one element

File number out of range	The file number must be: 0-255
Invalid file type	The file number does not match the assigned file type.

AB PCCC SLC 5/05

Each AB PCCC SLC 5/05 master object is listed in the window.

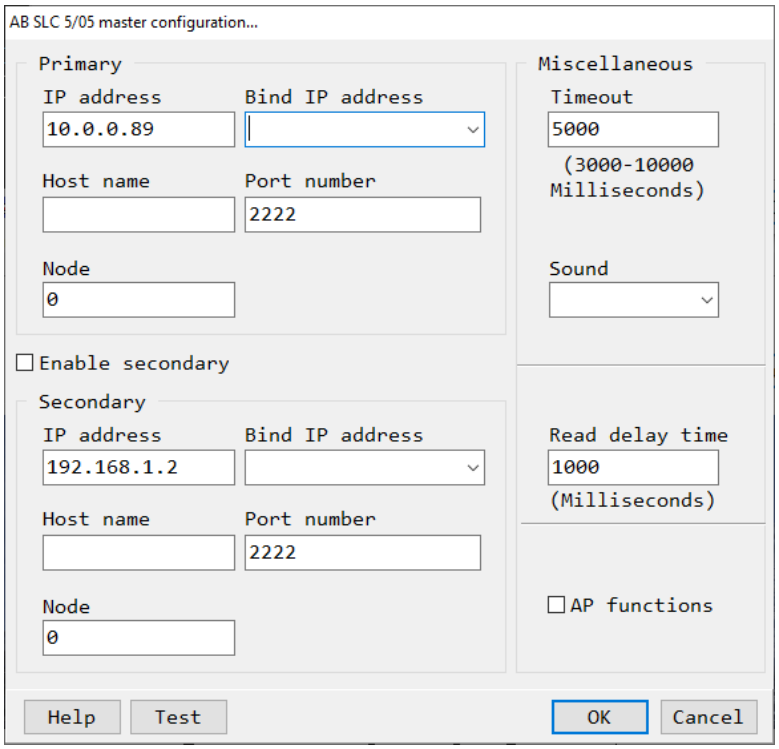


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an AB PCCC SLC 5/05 master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

The TCP port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

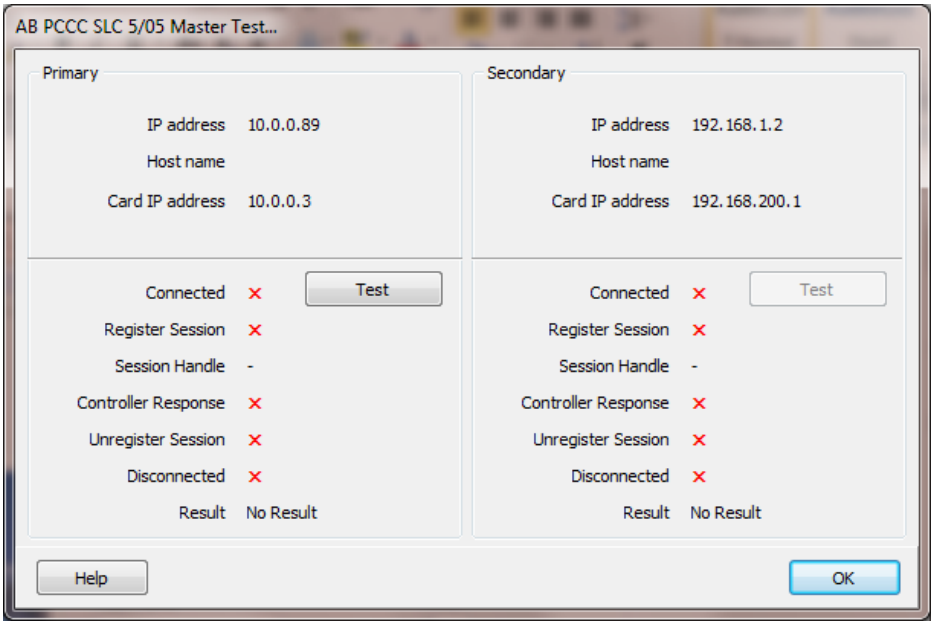
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

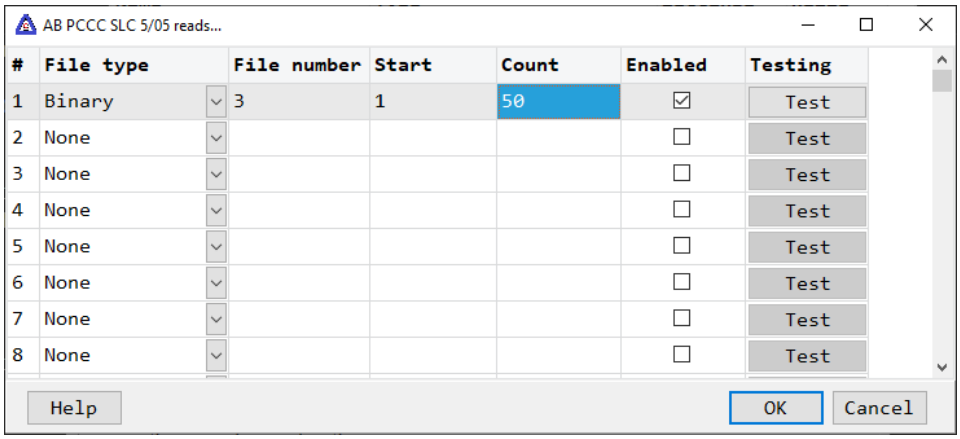
Test button



The program will attempt to connect to the controller, register the session and will attempt to read one word from "N7:0." Verify that this address does in exist in the PLC.

Use the reads testing for more verification.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the AB defined addressing for access to the data files in the PLC. Please refer to the AB documentation.

The DF1 “Duplicate Message Detection” logic may be enabled in the communication device. If more than one read command is enabled the detection logic will not cause a problem. If only one read command is enabled the “Duplicate Message Detection” logic must be disabled. (Or create another read command.)

Common

Example address:

N7:10, N7:451/0, N7:33/15
T4:152/TT,T4:152/EN,T4:152.ACC
B3:45/5
B3/567

I:1.0/00, I:1.1/12, I:4.4/15
O:2.0/01, O:7.0/16, O:8.1/31
The format of the address must be used.

I:X.X/XX
O:X.X/XX

File Type : Slot . Word / Bit

Note: When configuring reads if the file type is input or output be aware that setting the count greater than the number of words the card supports, may or may not return data (a successful read). In some cases the PLC will return the data for the selected card (slot) and additional cards (slots) up to the count value.

Example:
Slot 1 1746-IA16 1 word
Slot 4 1747-KE 45 words
Slot 5 1746-NI8 8 words
Slot 6 1746-IA16 1 word

A read
Slot = 1
Start element = 0
Count = 45

For this read the PLC will return 45 words of data. The data past the first word I:1.0 is for other cards.
For the HMI point source address and PLC addressing to match, configure one read for each populated slot (if data is needed) and set the count to the number of words the card utilizes.

File type

Example: Integer (N), Input (I), Binary (B)

File number

Legal file numbers are 3 to 255. For inputs and outputs the file number is the slot number.

Some file numbers are assigned.

#	Type
2	Status
3	Binary
4	Timer
5	Counter
6	Control
7	Integer
8	Floating Point
9-255	User Assigned

If the file type is status the file number is set to the correct value when the 'OK' button is selected.

Start element

The starting element to read. The different file types have different element counts. Example. Integer (N) has a one word element, Timer (T) has a three word element. This is the starting element for the file type.

Count

For SLC 5/01 or 5/02 = 82 bytes (41 words).
For SLC 5/03 or 5/04 = 236 bytes (118 words).

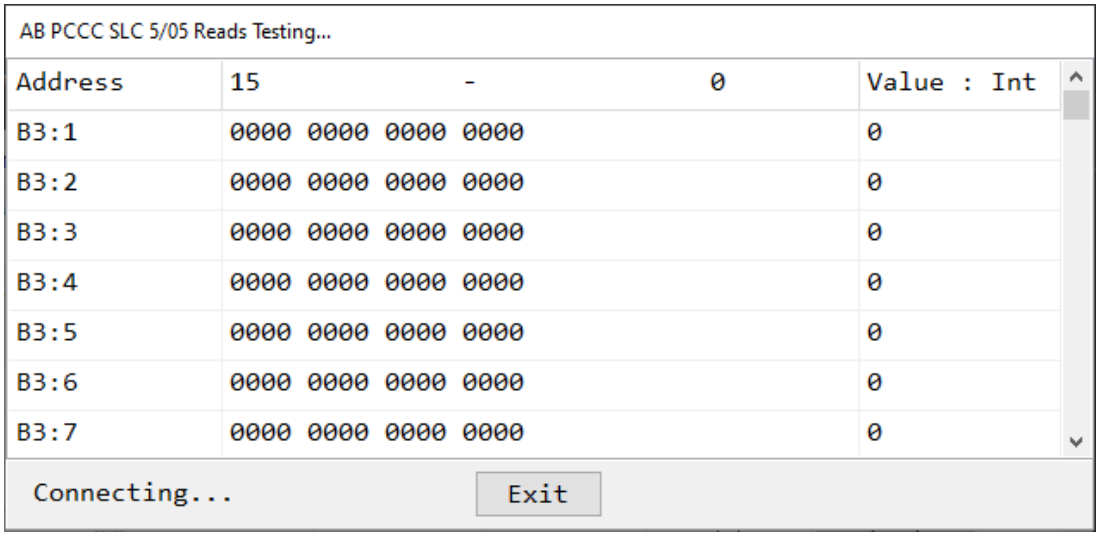
Example: Timer (T) 3 words per element. Maximum count is 40.
Float (F) 2 words per element. Maximum count is 61.

Enabled

The read requests are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

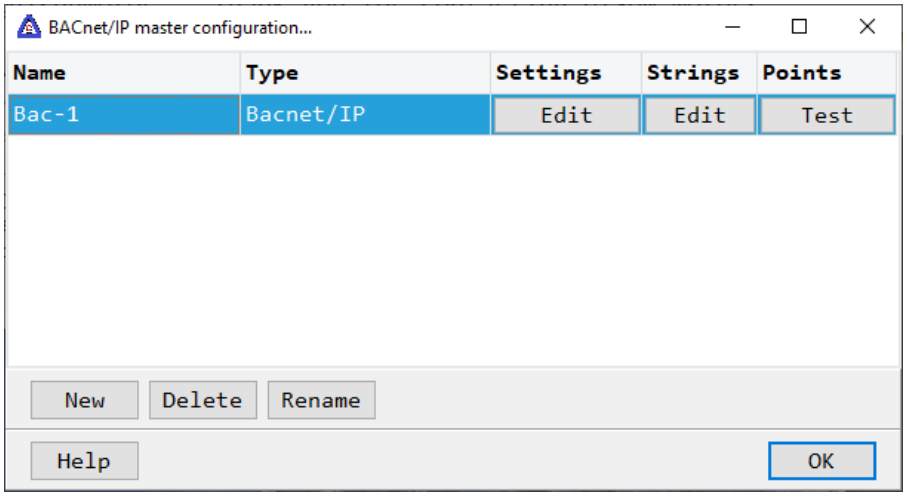
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. If the error string contains 'STS' or 'EXT STS' the text that follows is the error code ,in hexadecimal, returned from the external device. The HMI uses a function code $0F. PCCC, assuming the port test succeeded, if "Forward open success..." is displayed and does not change, close the testing window and reattempt the test.

Error messages

No file type selected	A file type must be selected.
Element out of range	Start element must be: 0-255
Count exceeds limit	The number of elements to read is too high for the file type.
Element start + count exceeds limit	Element start + count greater than file end.
Count < 1	Must read at least one element
File number out of range	The file number must be: 0-255
Invalid file type	The file number does not match the assigned file type.

BACNET/IP

Each Bacnet/IP master object is listed in the window.

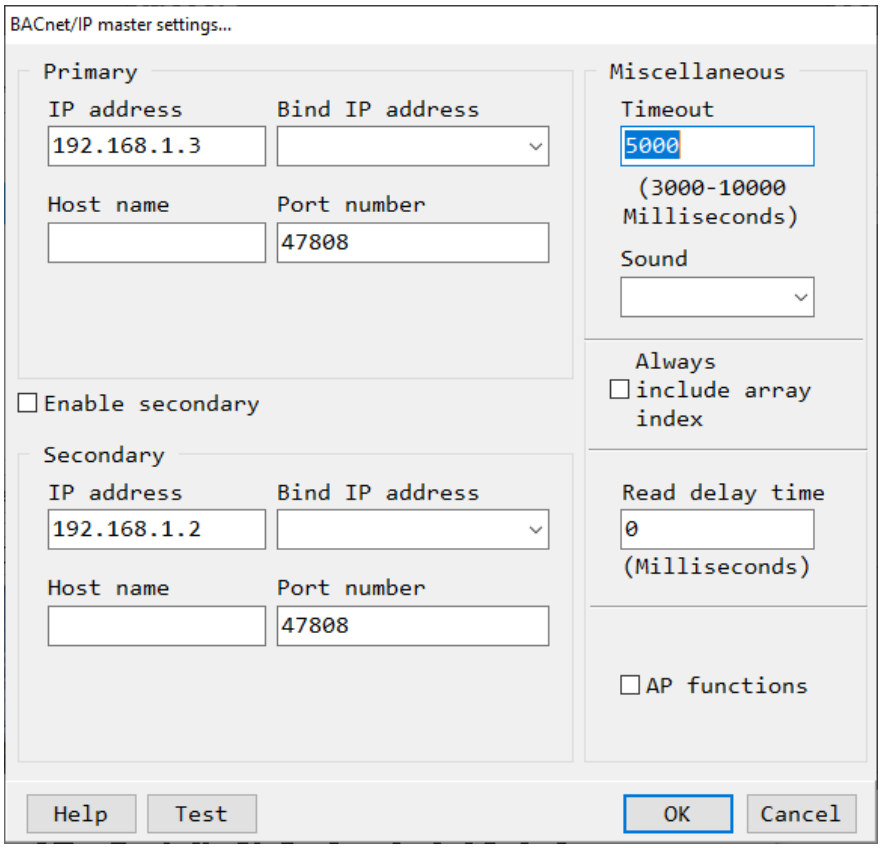


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Bacnet/IP master object select the "Delete" button.

Settings



The port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second path to be used as a hot backup. When in run mode the primary configuration is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary configuration. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary configuration, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary

configuration an entry will be made in the event log and operations will return to the primary port.

If the computer has two network interface cards (NIC) the program will use the first one detected for the primary and the second one detected for the secondary. If only one NIC is detected the program will use it for the primary and the secondary.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Always include array index

Some devices want an array index as part of the packet when it is not needed. When enabled the array index will always be part of the message transmitted (-1 default). If it is not enabled the array index will only be transmitted when it is greater than zero (0). This property does not apply to the BACnetNakedWrite script function.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

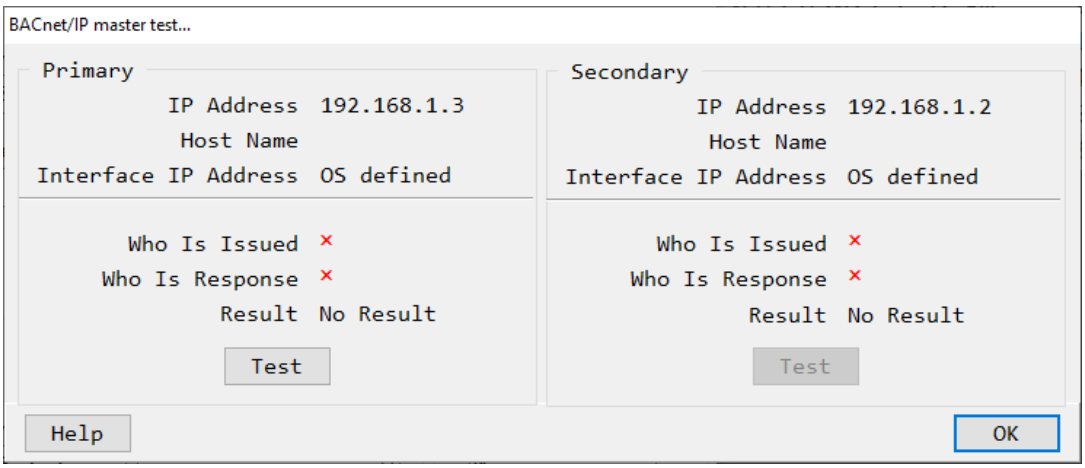
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

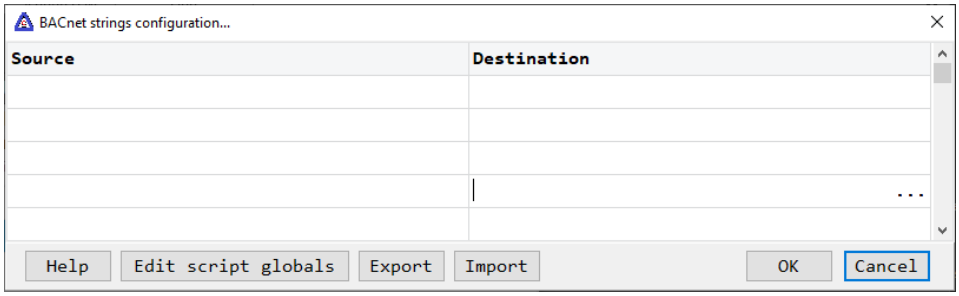
See analog functions.

Test button



When the test button is selected the program will issue a "Who-Is" command to the configured host and listen for the "I-Am" response from the client. Most clients broadcast the response so the program "listens" for a broadcast message.

Strings



Strings are not regular points. They do not have alarms, printing, etc. Strings may be displayed in any window via the "Script Global" animation. Any other actions, parsing, comparing, etc. must be done in scripts.

Source

This is the source address in the device. It must be a valid device object with a datatype "string".

Destination (optional)

If desired, select a script global location and the string will be copied to the location when the string value is returned from the external device. If this option is used or not the string may still be accessed via the "StringGet" and "StringSet" script commands. If this is configured and access to the string, in a script, is desired, use the "GlobalGet" or "StringGet" to copy the string. The format of the destination is: <section>.<name>.
Note: Writing to the script global does not write the value to the device. "StringSet" must be used to write a string to the device.

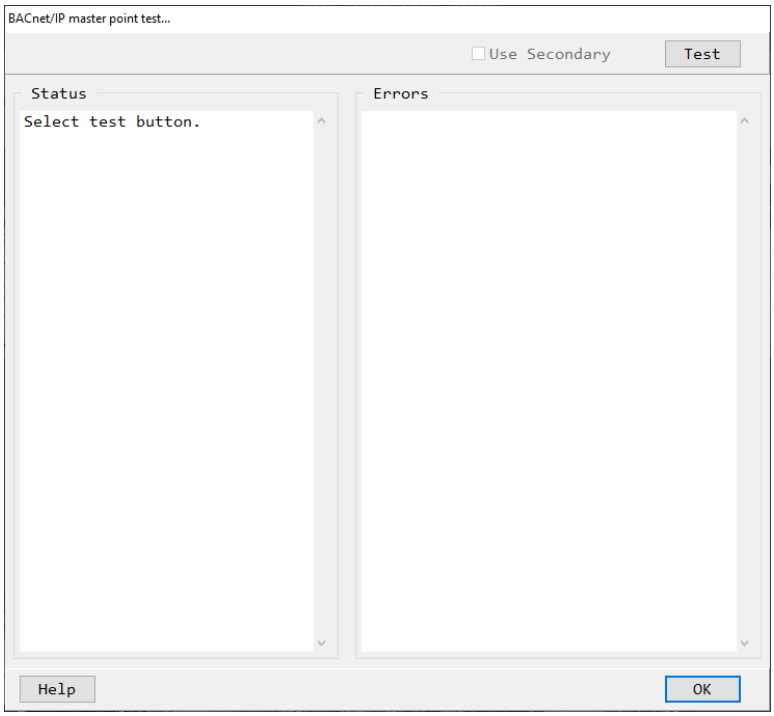
Export

This is used to export the configuration to an Excel worksheet.

Import

This is used to import string configurations from an Excel worksheet. The contents of the grid are completely replaced by the contents of the file.

Points



This feature is used to test if a point or string, with the source address entered, exist in the external device.

Use Secondary

If enabled the secondary port settings will be used.

Test

Selecting this button will execute the test. All the configured points and strings will be tested. The test is very fast.

Note: Right click and copy the contents of either text area.

Status

This area is to display general information regarding the test and testing progress as well as displaying each point/string success. Each point/string tagname will be displayed with a 'success' suffix if the source address is in the external device.

Example:
Connecting...Connected
Point testing begins...
Close_Discharge_Valve_CRT... success

CRT_Acknowledge... success

The first three lines are general status updates.

Error

This area is to display the error that was generated for a source address. Several types of error are possible. Each point error has three lines. Examples:

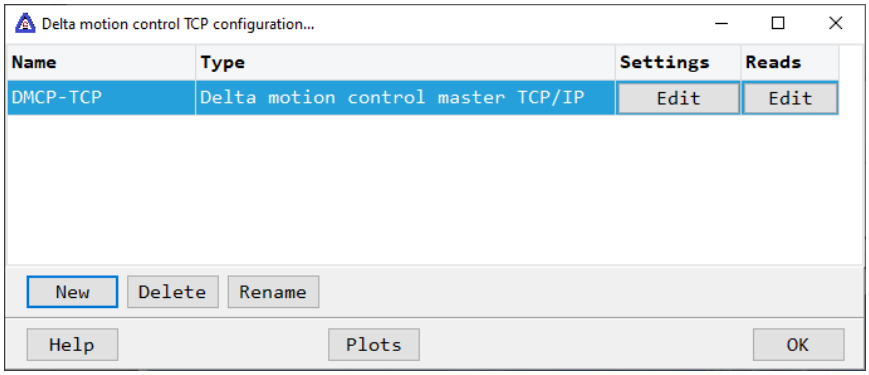
Tagname: Alarm_Active
Source address: Analog Output.33.Present Value
Error: ADPU Error (5)

The ' ADPU Error (5)' is the most common when a source address does not exist in the external device.

Source address: string1
Error: ADPU Error (5)

Delta Motion Control TCP

Each DMCP TCP master object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

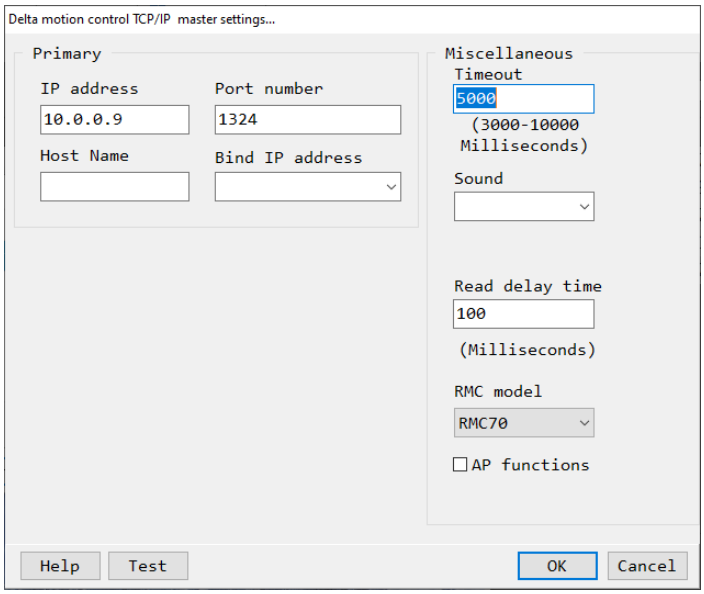
To delete a DMCP TCP master object select the "Delete" button.

Plots

Plots collected from a controller are displayed using a theme. Select this button to manage plot themes.

Plot settings are covered after the DMCP UDP port section.

Settings



If a "host name" is entered, the IP address is ignored.

Port number

This is the port number used for TCP/UDP communication. The default port number is 1324.

Bind IP address See here.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

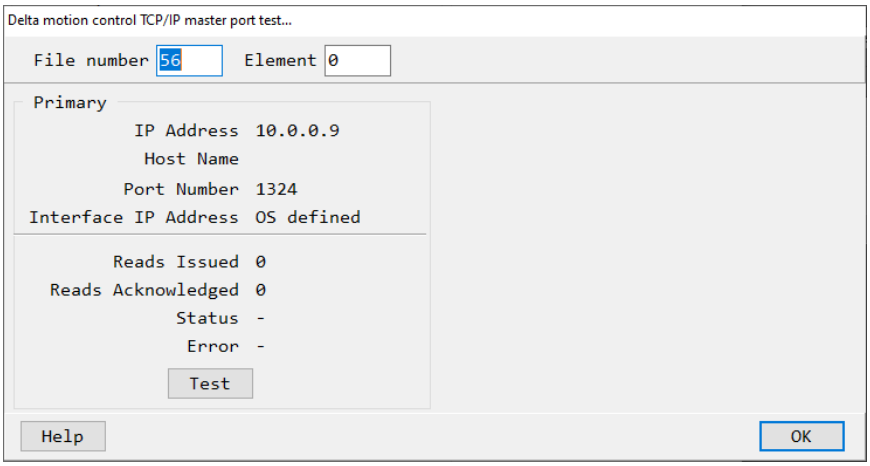
AP functions

See analog functions.

RMC model

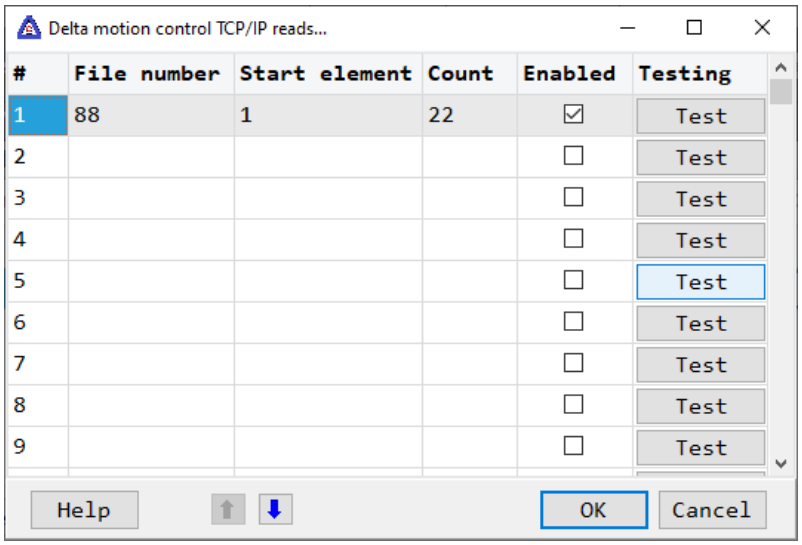
Select the correct Delta Motion Controller model number.

Test button



When the test button is selected the program will attempt to read one point of data from the device at the address and element entered.

Reads



The address ranges shown may or may not be present in the device.

Registers

<u>Address Examples</u>
7.4, 5.90
58.8.6 file 58 element 8 bit 6

Note:
1) DMCP does not provide a function for single bit writes to a register. A bit set write will set the desired bit and clear all other bits in the register. A bit reset will clear the register.
2) If the HMI will be used to collect plots from the controller <u>do not</u> configure any read to access the "Dynamic Plot Upload" files/registers. The HMI uses "Method 2: Read a Captured Plot - Advanced" to collect the plot from the controller. Any read configured to access these file/registers will interfere with the handshaking required to collect plots from the controller.

File number

This is the file number in the device.

Start element

This is the start element value for the read. Each line is one read from the HMI to the device requesting data from the device.

Count

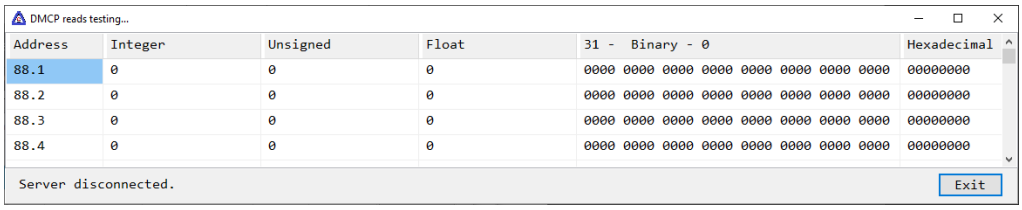
This is the number of 32 bit registers to return. For TCP the limit is 1024, for UDP the limit is 256.

Enabled

The read requests are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



Address	Integer	Unsigned	Float	31 - Binary - 0	Hexadecimal
88.1	0	0	0	0000 0000 0000 0000 0000 0000 0000 0000	00000000
88.2	0	0	0	0000 0000 0000 0000 0000 0000 0000 0000	00000000
88.3	0	0	0	0000 0000 0000 0000 0000 0000 0000 0000	00000000
88.4	0	0	0	0000 0000 0000 0000 0000 0000 0000 0000	00000000

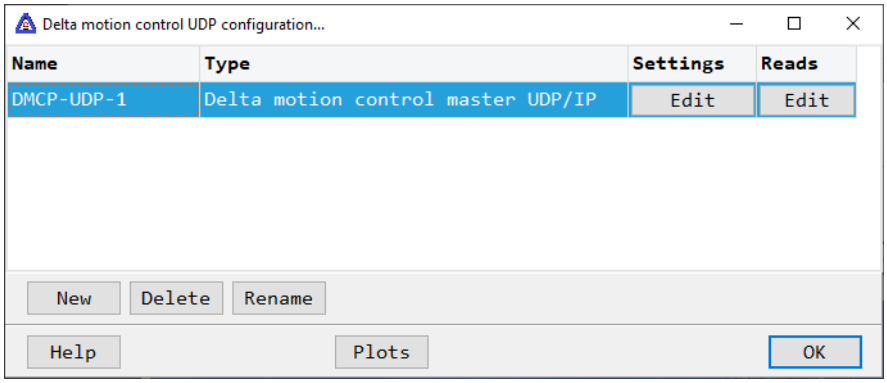
When the button is selected the program will attempt to access the data in the device using the communication parameters configured.

Error messages

Start out of range	The value is out of range for the type.
Count exceeds limit	The count must be 1024 or less for TCP and 256 or less for UDP.
Start + count exceeds register limit	The register plus the count exceeds the maximum address range.
Count < 1	The line must read at least one register.

Delta Motion Control UDP

Each DMCP UDP master object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

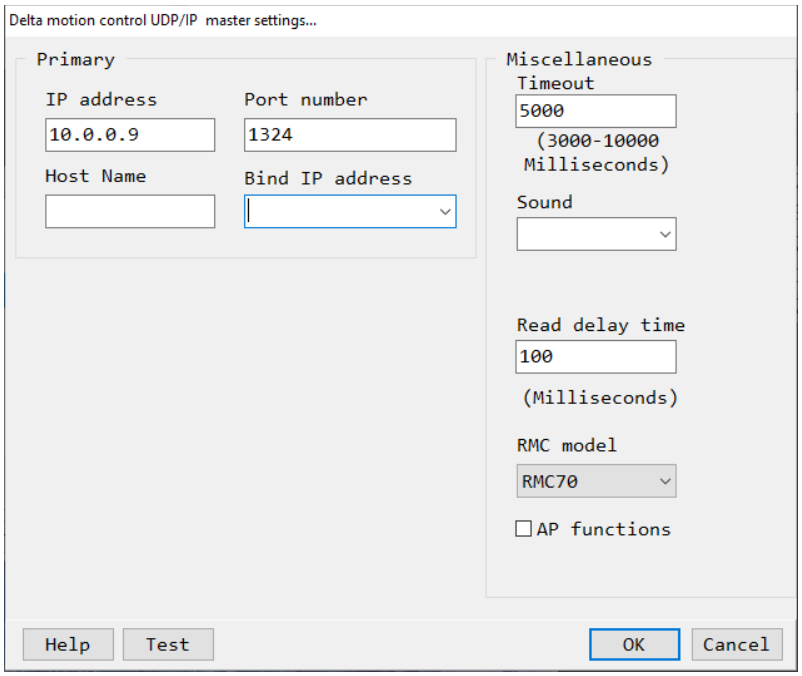
To delete a DMCP UDP master object select the "Delete" button.

Plots

Plots collected from a controller are displayed using a theme. Select this button to manage plot themes.

Plot settings are covered at the end of this section.

Settings



If a "host name" is entered, the IP address is ignored.

Port number

This is the port number used for TCP/UDP communication. The default port number is 1324.

Bind IP address See here.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

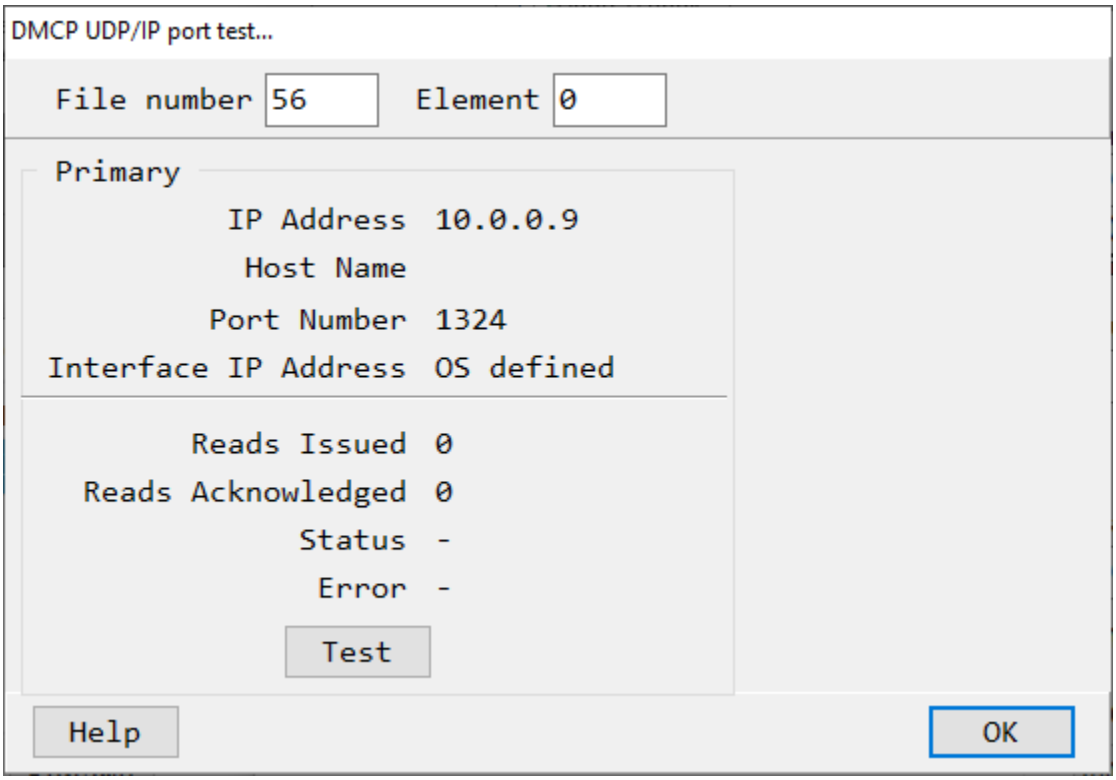
AP functions

See analog functions.

RMC model

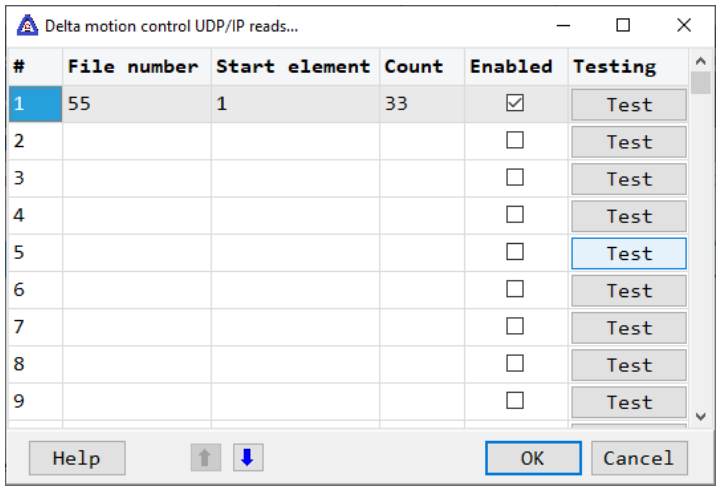
Select the correct Delta Motion Controller model number.

Test button



When the test button is selected the program will attempt to read one point of data from the device at the address and element entered.

Reads



The address ranges shown may or may not be present in the device.

Registers

<u>Address Examples</u>
7.4, 5.90
58.8.6 file 58 element 8 bit 6

Notes:
1) DMCP does not provide a function for single bit writes to a register. A bit set write will set the desired bit and clear all other bits in the register. A bit reset will clear the register.
2) If the HMI will be used to collect plots from the controller <u>do not</u> configure any read to access the "Dynamic Plot Upload" files/registers. The HMI uses "Method 2: Read a Captured Plot - Advanced" to collect the plot from the controller. Any read configured to access these file/registers will interfere with the handshaking required to collect plots from the controller.

File number

This is the file number in the device.

Start element

This is the start element value for the read. Each line is one read from the HMI to the device requesting data from the device.

Count

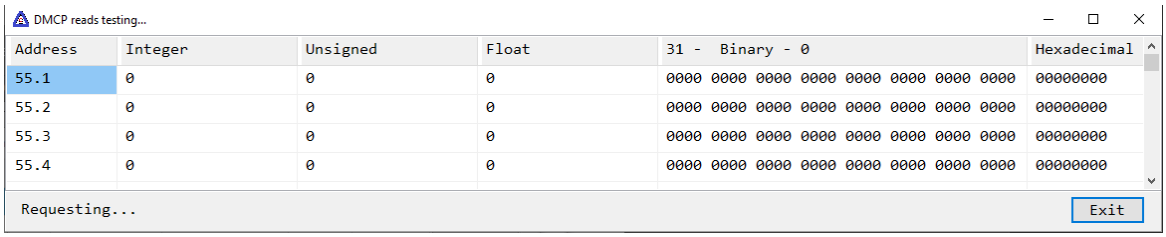
This is the number of 32 bit registers to return. For TCP the limit is 1024, for UDP the limit is 256.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test

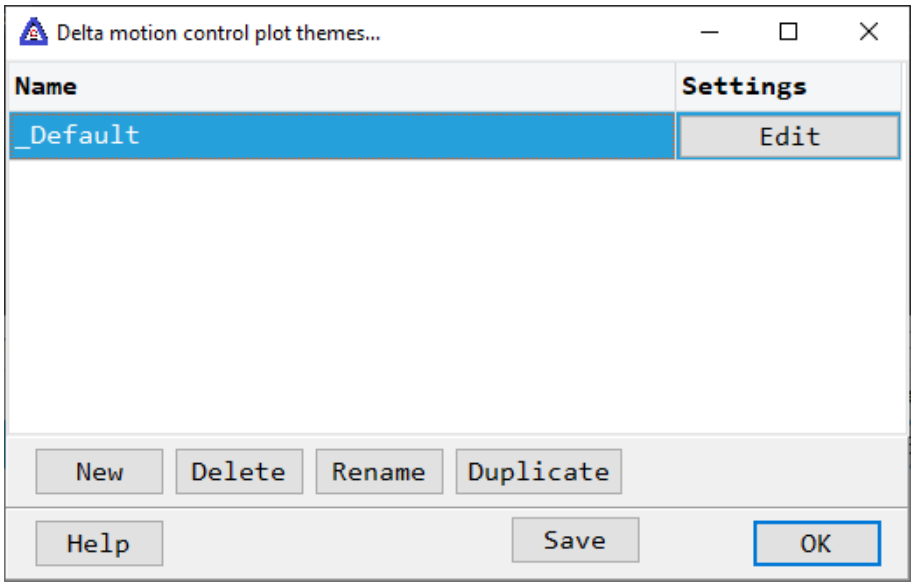


When the button is selected the program will attempt to access the data in the device using the communication parameters configured.

Error messages

Start out of range	The value is out of range for the type.
Count exceeds limit	The count must be 1024 or less for TCP and 256 or less for UDP.
Start + count exceeds register limit	The register plus the count exceeds the maximum address range.
Count < 1	The line must read at least one register.

DMCP Plots



Each DMCP Plot theme defines how a plot will appear.

To create a new theme select the "New" button and enter a name. Each name must be unique.

To rename a theme select the "Rename" button and supply a new name.

To delete a theme select the "Delete" button.

Notes:

A theme named "_Default" is always present and cannot be renamed or deleted.

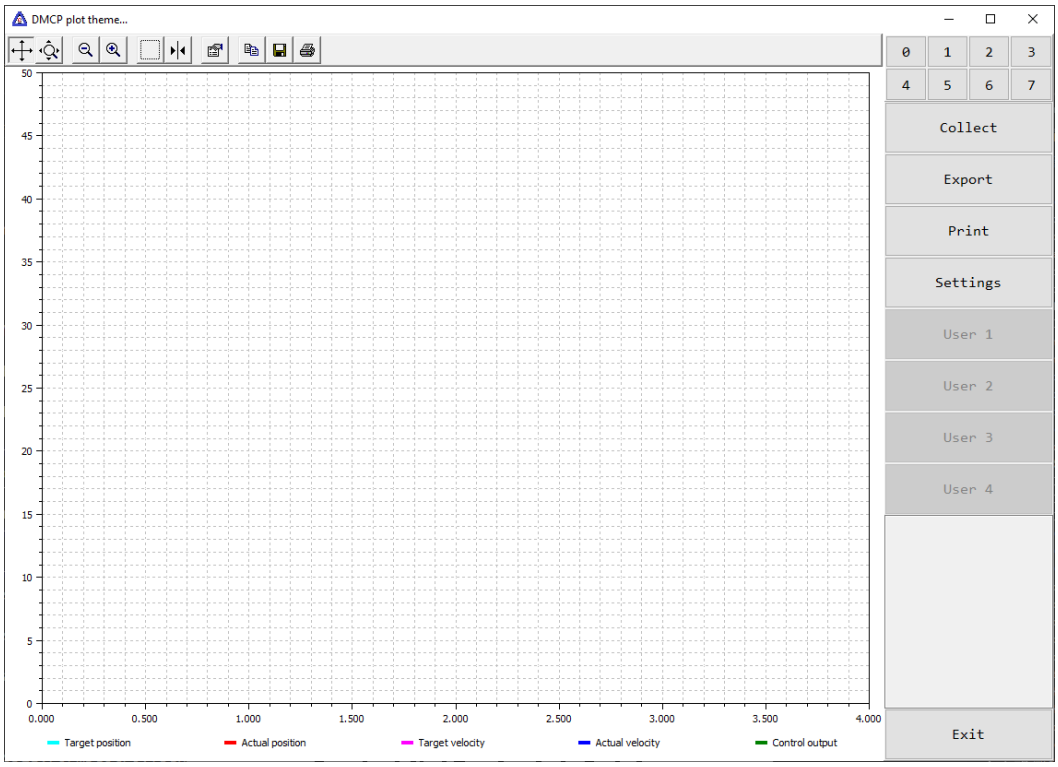
The theme can be changed.

To revert to the original settings, delete the "_Default" theme and it will be recreated with the default settings.

Select the "Edit" button to edit the theme.

At runtime the area between the "Print" button and the "Exit" button is a status area. It will contain information about the plot or other data about the plot/controller.

Settings



Collect

This will collect the plot from the controller.

Export

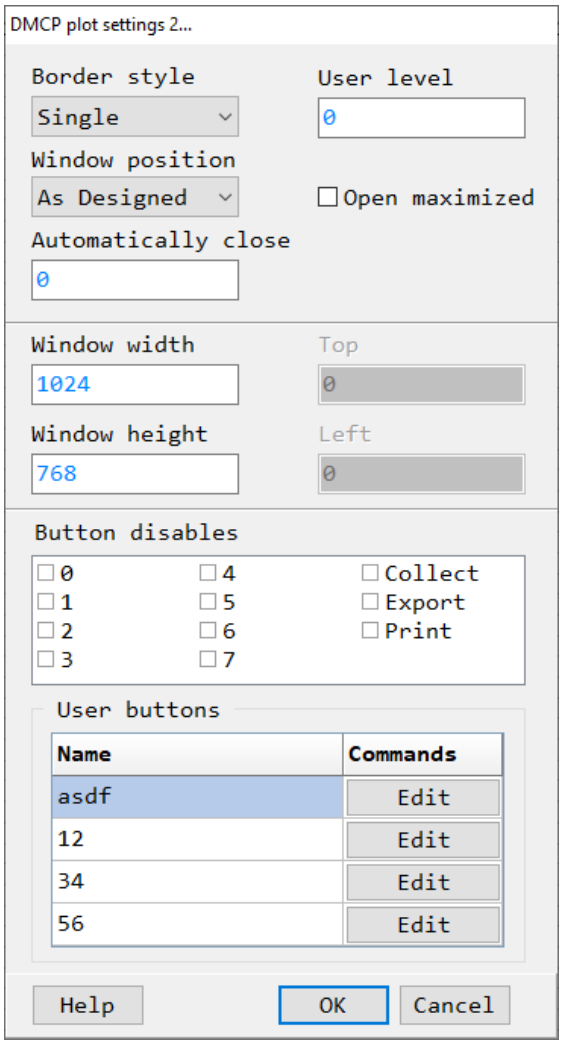
This will display a dialog to save the plot to the selected file. If a file with the same name exists, it will be deleted.

Print

This will print the screen to the selected printer. To print only the trend, use the print button on the trend.

Settings

This button is not present at runtime. It is present in configuration to allow for additional theme configuration settings.



Border style

Windows provides for several window border styles. None, single, sizable and dialog.

Window position

Select the position to open the window. 'As designed' is the same as 'Top Center'.

Open maximized

If enabled, when the window opens, it will maximize to fill the screen.

Window width/height

This size of the window, including any border and title bar.

User Level

At runtime the logged on user must have at least the level entered to view the plot with the theme.

Automatically close

The number of seconds the window will be open. If the value is 0 the window will not automatically close.

Button disables

Each button can be disabled. The plot buttons 0,1..7 will be visible but disabled.
The remaining buttons will not be visible if disabled.

Note: If all the plot buttons are disabled, the only plot to be displayed is the plot number in the script command.

If the "Collect" button is disabled and the "Collect on open" variable is false, the plot shown will only be a plot that was previously collected.

Disabling the "Print" button does not disable the "Print" button on the trend. The "Print" button on the trend is disabled via the "Proprieties" of the trend.

User buttons are disabled/not visible when the button name is blank.

User buttons

Each button can be configured with mouse commands.

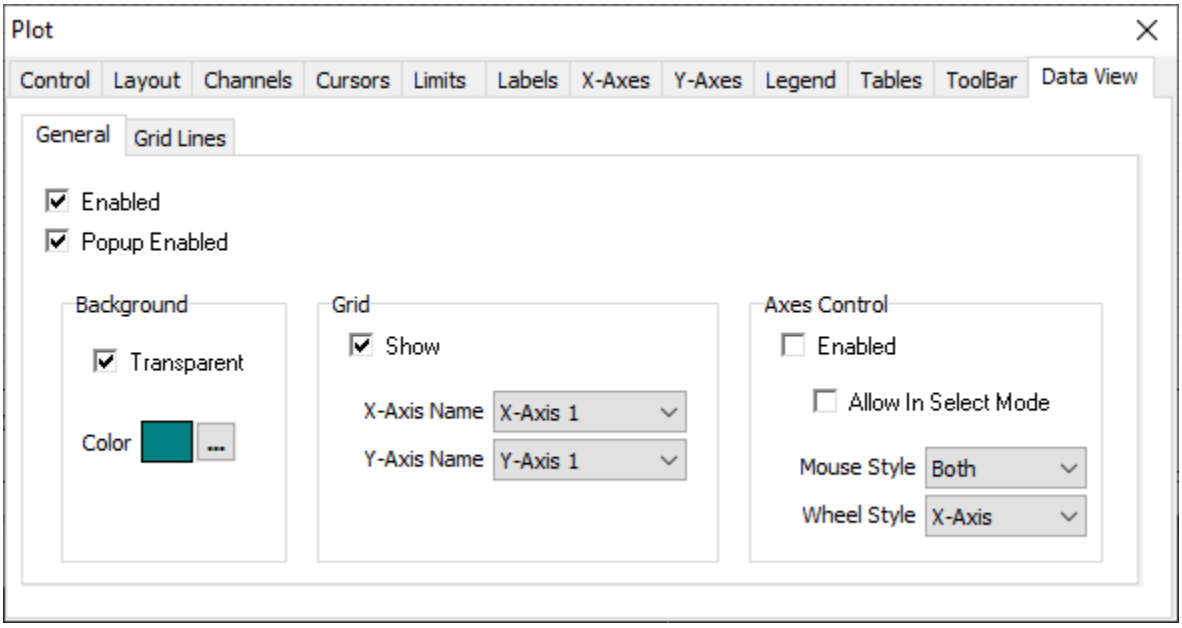
User 1/2/3/4

These buttons may or may not be present based on the settings configuration below.
The buttons are configured in the 'Settings 2' dialog.

Exit

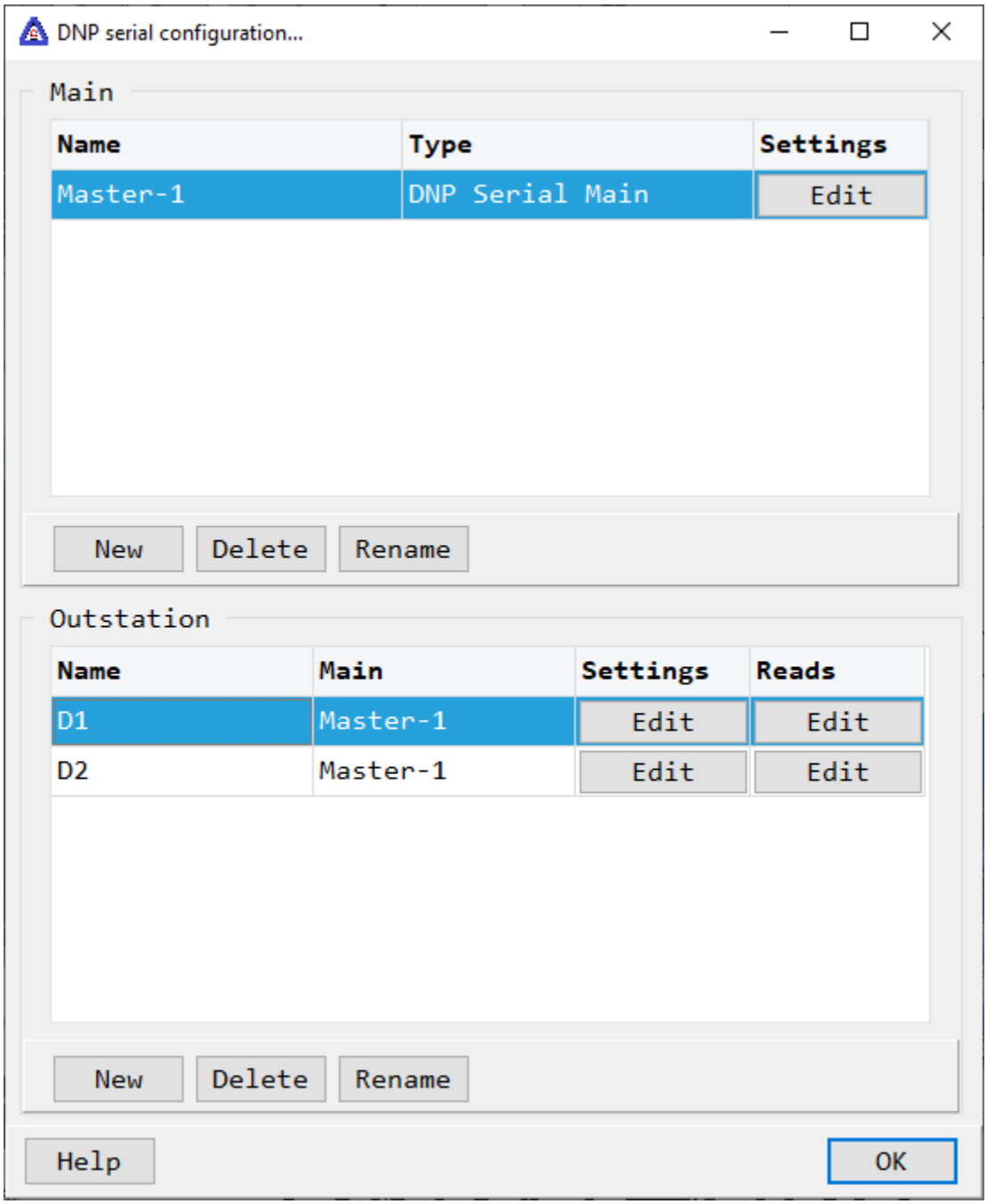
Closes the plot window.

Right click on the trend and the trend settings dialog appears.



DNP3 Master Serial

Each DNP3 master (Main) serial object controls one serial port and can have one or many outstation objects. The outstations are configured to address one device.



Note: If using the port as RS-485, it is sensitive to improper wiring and/or terminations. Unpowered units can cause data echoes and other reliability issues. Please follow all RS-485 wiring guidelines.

DNP serial main

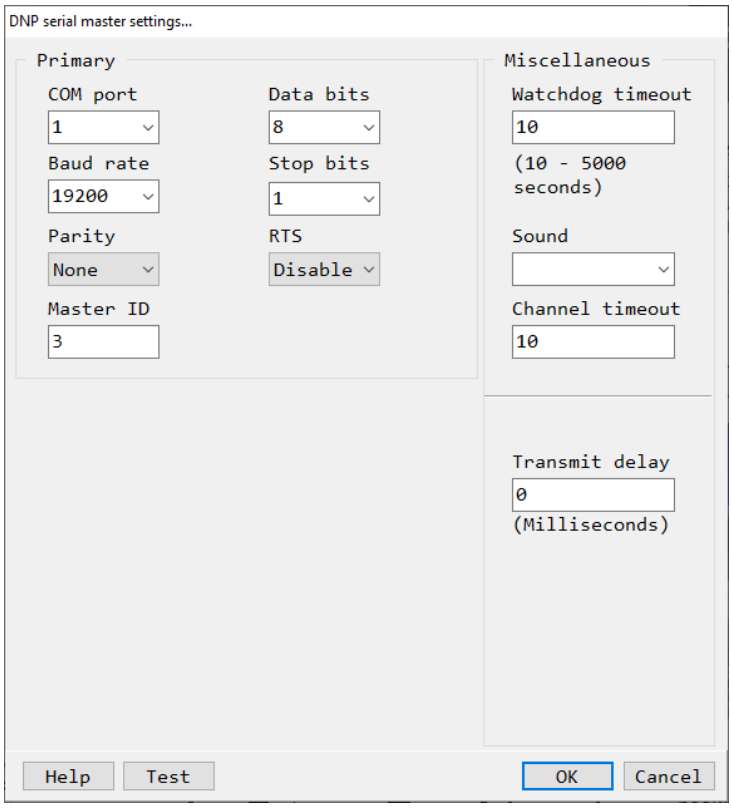
Each DNP serial master (main) object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a DNP serial master object select the "Delete" button.

Settings



Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

RTS

See here.

Master ID

This is the DNP address of the master.

Miscellaneous

Watchdog timeout (seconds)

This is the amount of time a received fragment must be completed. The timer starts when the first byte is received. If the timer expires, any received segments are discarded and all receiving buffers are cleared.

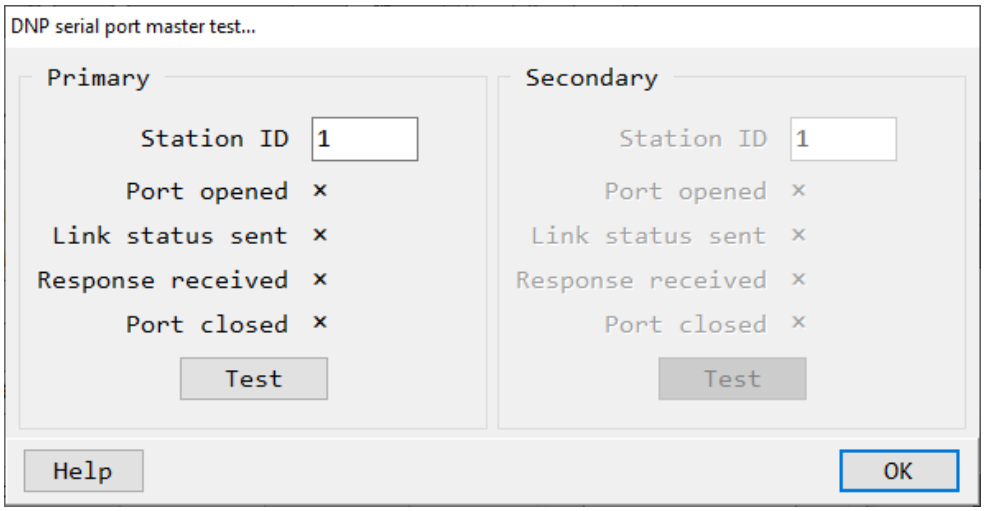
Channel timeout (seconds)

This is the amount of time in which a message must be completed. The timer starts when the message is transmitted. If the timer expires the message request is discarded.

Transmit delay (milliseconds)

Some RS-485 devices need time to switch from transmit mode to receive mode. If this value is > 0 the next data to be transmitted will be delayed by the number of milliseconds. If this communications port is RS-232 then the delay should not be needed.

Test button



When the test button is selected the program will issue a 'REQUEST_LINK_STATUS', function code 9, to the outstation ID entered in the 'Station ID' edit field. The fields below the edit field will display the result of the command.

DNP serial outstation

Each DNP serial outstation object is listed in the window.

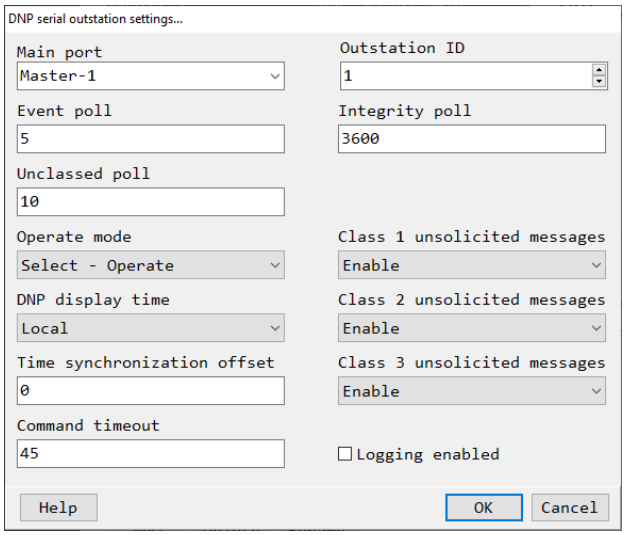
To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a DNP serial outstation object select the "Delete" button.

Each DNP serial outstation slave object listed has buttons.

Settings



Main port

This is the Master (Main) DNP serial port the outstation is linked to. A port must be linked to a master for runtime operations.

Outstation ID

This is the DNP address of the outstation.

Event poll

This is the frequency, in seconds, the outstation will be polled for event data. Enter a value of zero to disable event polling. The outstation must be configured to report events as one of the class types.

Integrity poll

This is the frequency, in seconds, the outstation will be requested to send all data. Enter a value of zero to disable integrity polling. NOTE: This does not include any data the outstation has not configured to be reported.

Unclassed poll

This is the frequency, in seconds, the unclassed reads will be executed. Enter a value of zero to disable unclassed reads. If one or more reads have not been configured this timer is disabled. NOTE: All the enabled reads will be executed starting with the first one in the list and ending with the last one in the list. The timer does not restart until all the enabled reads have completed or timed out.

Operate mode

This selects the mode used to modify writable I/O objects. (10 and 40)

Class 1, 2, 3 unsolicited messages

Enable - Unsolicited messages are enabled for the class.
Disable - Unsolicited messages are disabled for the class.
No change - The outstation selects if unsolicited messages are enabled or disabled for the class.

DNP display time

This determines how time values are handled.

If UTC is selected:
Time values from the outstations are <u>not</u> converted to local time.
Time values for internal alarms are converted from local time to UTC.

If local time is selected:
Time values from the outstations are converted to local time.
Time values for internal alarms are <u>not</u> converted from local time to UTC.

Time synchronization offset

When the time is sent to an outstation this is the offset, in milliseconds that is sent to the outstation. This provides for communication lag.

Command timeout

This is the amount of time to allow a command to complete. This value should take into account the number of outstations configured for a master. If this timer completes the

masters connection is reset to 'master start' mode. This timer should be greater than the master channel timeout to allow other outstation message processing.

Logging enabled

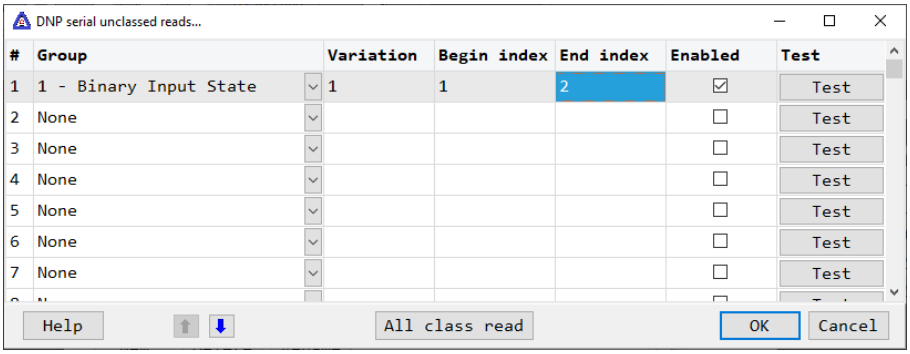
When this feature is enabled and a DNP event object is received it will be logged to a file. Each outstation has a separate log file. See Log File Settings

Only events that are configured as points will be logged.

All time values are in UTC. This is the value reported from the outstation without conversion to human readable time or local time.

Each line has five columns and each column is separated with a 'TAB' character.
Line format:<UTC time> TAB <group number . index number> TAB <point tagname> TAB <value> TAB <point description> CRLF(carriage return + line feed)

Reads



NOTE: Normal operation of DNP is report-by-exception. Some devices may contain data that is not included in report-by-exception, event polling or integrity polling. This is not normally the case and special care must be taken to properly configure any reads to prevent a failure of DNP communications.

The groups and ranges shown may or may not be present in the outstation device.

Group

The DNP object group.
1 - Binary Input State
3 - Double Bit Input State
10 - Binary Output State

20 - Counter Value
21 - Frozen Counter Value
30 - Analog Input Value
40 - Analog Output Value
50 - Time and Date
110 - Octet String Value

Variation

Variation 0 is the default variation.

Begin/End index

The beginning and ending index number.
NOTE: Use caution. This can disrupt all DNP communications.

Enabled

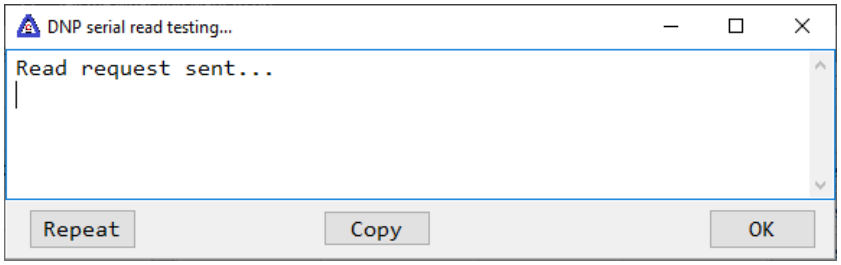
The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test

When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

All class read



When this button is selected a Class 0 (Group 60, variation 1) read will be issued to the outstation. The outstation will return the static data of all its points that have been assigned to one of the four classes (static class 0, events classes 1, 2, or 3). Any points

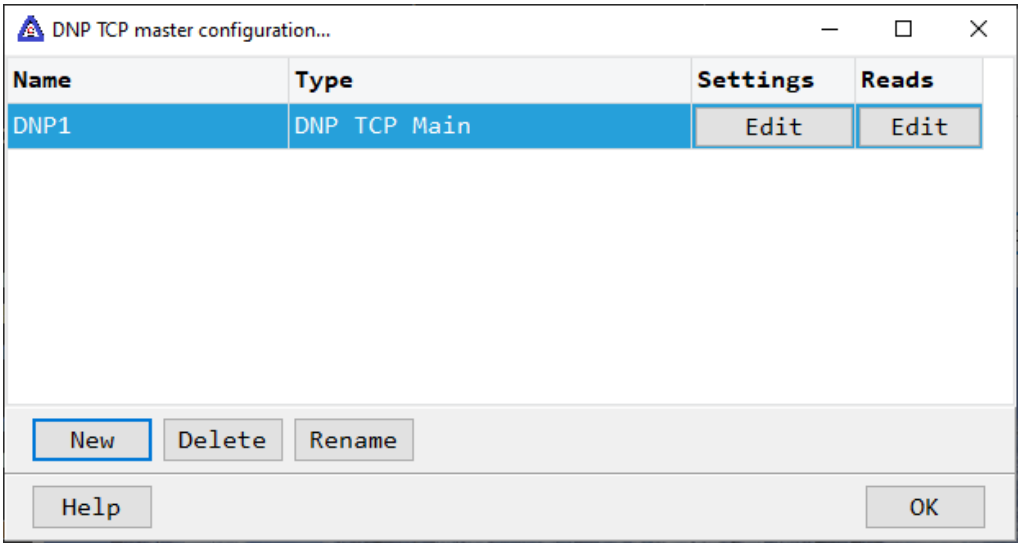
that are not assigned to one of the four classes will be excluded from the response, with the result that the response to the Class 0 request may contain a subset of all the outstation's static data. The outstation will include some or all of the following objects in its response: groups 1, 3, 10, 20, 21, 30, 31, 40, 87, 101, 102, and 110. Please see the DNP specification for more information.

Error messages

Invalid begin index	The begin index must be 0 - 65535.
Invalid end index	The end index must be 0 - 65535 and greater than or equal to the 'begin index'.
Variation not allowed	Not all objects support all variations.

DNP3 Master TCP

Each DNP3 Master TCP object is listed in the window.

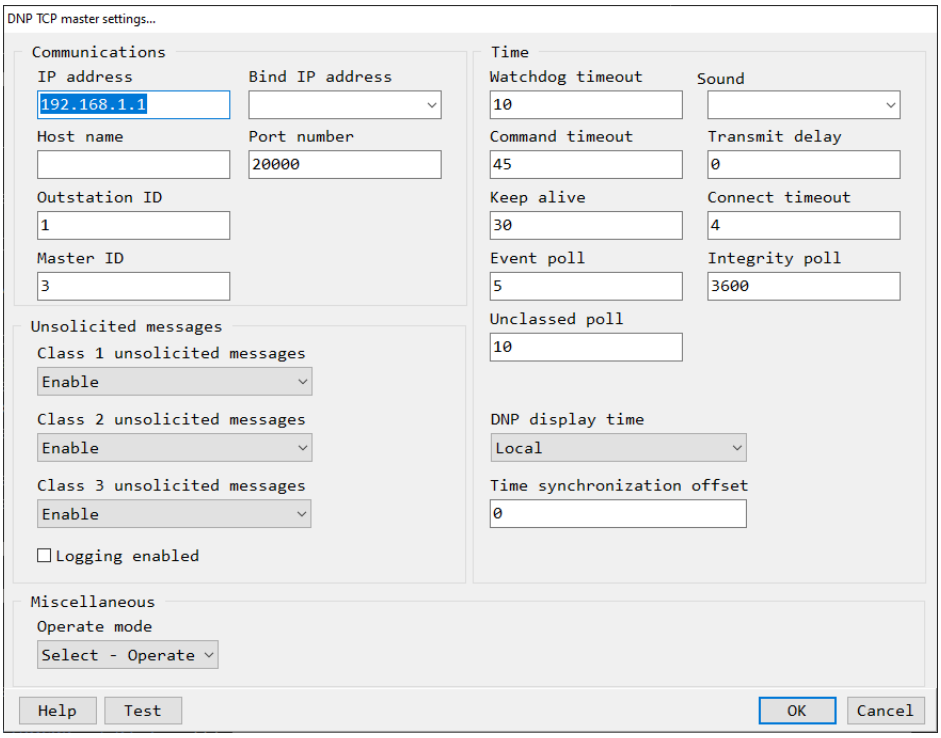


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a DNP TCP master object select the "Delete" button.

Settings



Each master configuration communicates with one outstation.

Communications

IP address

This is the address of the outstation.

Bind IP address See here.

Host name

This is the host name of the outstation. If the host name is blank the IP address is used.

Port number

This is the port number to use for communications with the outstation for TCP connections initiated by the master.

Master ID

This is the DNP address of the master.

Outstation ID

This is the DNP address of the outstation.

Time

Watchdog timeout (seconds)

This is the amount of time a received fragment must be completed. The timer starts when the first byte is received. If the timer expires, any received segments are discarded and all receiving buffers are cleared.

Command timeout (seconds)

This is the amount of time to allow a command to complete. If this timer completes the last message is discarded and if another is pending it is processed.

Transmit delay (milliseconds)

If this value is > 0 the next data to be transmitted will be delayed by the number of milliseconds.

Keep alive (seconds)

If this timer completes a 'request link status' message is transmitted to the outstation. This timer is restarted each time data is received from the outstation. If for example the event timer is set to 5 seconds and this timer is set to 10 seconds and the outstation connection is valid this timer will never fire. If for example, the event timer is set to 10 minutes this timer can be used to verify the link is still active. Enter a value of zero to disable this timer.

Connect timeout (seconds)

This is the amount time the outstation must respond to a connection request before the attempt is aborted and restarted.

Event poll

This is the frequency, in seconds, the outstation will be polled for event data. Enter a value of zero to disable event polling. The outstation must be configured to report events as one of the class types.

Integrity poll

This is the frequency, in seconds, the outstation will be requested to send all data. Enter a value of zero to disable integrity polling. NOTE: This does not include any data the outstation has not configured to be reported.

Unclassed poll

This is the frequency, in seconds, the unclassed reads will be executed. Enter a value of zero to disable unclassed reads. If one or more reads have not been configured this timer is disabled. NOTE: All the enabled reads will be executed starting with the first one in the list and ending with the last one in the list. The timer does not restart until all the enabled reads have completed or timed out.

DNP display time

This determines how time values are handled.

If UTC is selected:
Time values from the outstations are not converted to local time.
Time values for internal alarms are converted from local time to UTC.

If local time is selected:
Time values from the outstations are converted to local time.
Time values for internal alarms are not converted from local time to UTC.

Time synchronization offset

When the time is sent to an outstation this the offset, in milliseconds, that is sent to the outstation. This provides for communication lag.

Unsolicited messages

Class 1, 2, 3 unsolicited messages

Enable - Unsolicited messages are enabled for the class.
Disable - Unsolicited messages are disabled for the class.
No change - The outstation selects if unsolicited messages are enabled or disabled for the class.

Logging enabled

When this feature is enabled and a DNP event object is received it will be logged to a file. Each outstation has a separate log file. See Log File Settings

Only events that are configured as points will be logged.

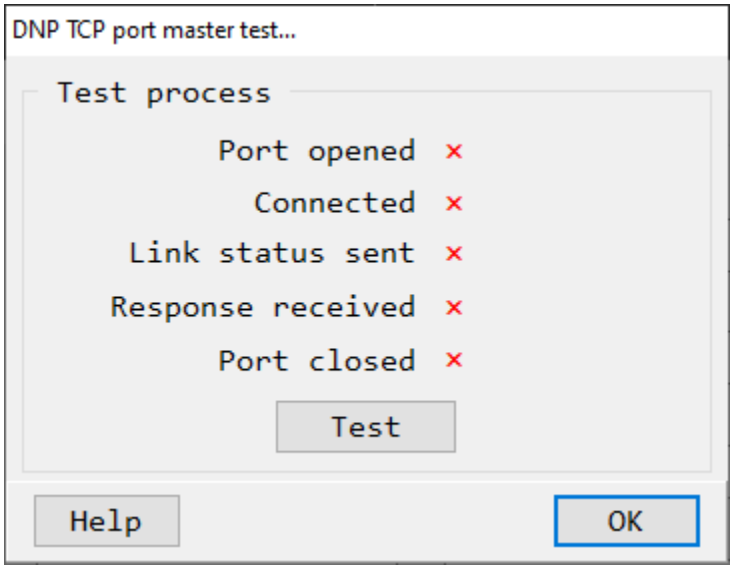
All time values are in UTC. This is the value reported from the outstation without conversion to human readable time or local time.
Each line has five columns and each column is separated with a 'TAB' character.
Line format:<UTC time> TAB <group number . index number> TAB <point tagname> TAB <value> TAB <point description> CRLF(carriage return + line feed)

Miscellaneous

Operate mode

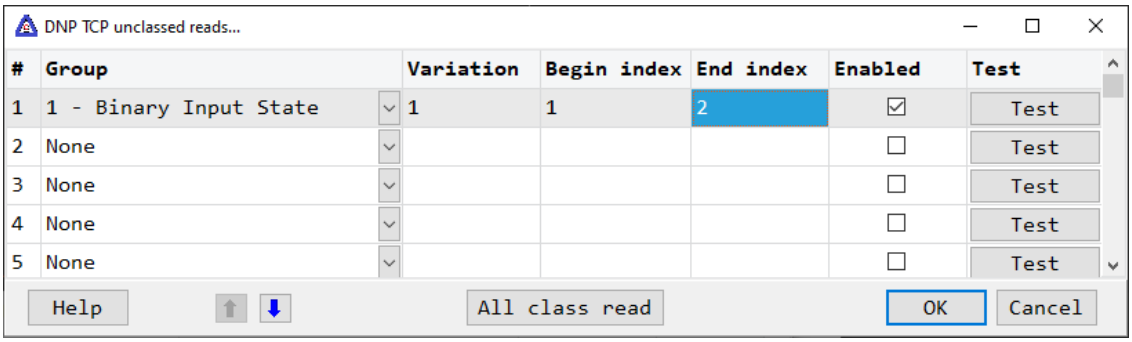
This selects the mode used to modify writable I/O objects. (10 and 40)

Test button



When the test button is selected the program will issue a 'REQUEST_LINK_STATUS', function code 9, to the outstation ID.

Reads



NOTE: Normal operation of DNP is report-by-exception. Some devices may contain data that is not included in report-by-exception, event polling or integrity polling. This is not normally the case and special care must be taken to properly configure any reads to prevent a failure of DNP communications.

The groups and ranges shown may or may not be present in the outstation device.

Group

The DNP object group.
1 - Binary Input State
3 - Double Bit Input State
10 - Binary Output State
20 - Counter Value
21 - Frozen Counter Value
30 - Analog Input Value
40 - Analog Output Value
50 - Time and Date
110 - Octet String Value

Variation

Variation 0 is the default variation.

Begin/End index

The beginning and ending index number. NOTE: Use caution. This can disrupt all DNP communications.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test

When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

All class read

When this button is selected a Class 0 (Group 60, variation 1) read will be issued to the outstation. The outstation will return the static data of all its points that have been assigned to one of the four classes (static class 0, events classes 1, 2, or 3). Any points that are not assigned to one of the four classes will be excluded from the response, with the result that the response to the Class 0 request may contain a subset of all the outstation's static data. The outstation will include some or all of the following objects in its response: groups 1, 3, 10, 20, 21, 30, 31, 40, 87, 101, 102, and 110. Please see the DNP specification for more information.

Error messages

Invalid begin index	The begin index must be 0 - 65535.
Invalid end index	The end index must be 0 - 65535 and greater than or equal to the 'begin index'.
Variation not allowed	Not all objects support all variations.

DNP Objects

The HMI supports the listed objects:

Groups	Point type	Default Variation
1 - Binary Input State	Digital	1

3 - Double Bit Input State	Analog	1
10 - Binary Output State	Digital	1
20 - Counter Value	Analog	5
21 - Frozen Counter Value	Analog	9
30 - Analog Input Value	Analog	6
31 - Frozen Analog Input	Analog	8
34 - Analog Input Deadband	Analog	3
40 - Analog Output Value	Analog	4
50 - Time and Date	Analog	*1, 2 1
110 - Octet String Value	N/A	*2 N/A

All event objects associated with the above groups are also supported.
The point addressing format is **<group number> . <variation number> . <index number>**

Group Number

The group number is one of the above groups.

Variation Number

The outstation transmits the variation it is configured to transmit. The variation number is used to define the data type/size to be used for writes to the outstation. A variation of 0 (zero) is used for the default variation. Group 110 uses the variation as the length of the string. Allowed range is 1 - 255.

Index number

The allowed range is 0 - 65535. The outstation will define the actual permitted range.

Notes:
*1 This is the time in the outstation.
*2 The first release does not fully support this object type.

Point status defines for B0 - B15

Group 1, 10, 20, 30, 31 and 40

B0 Bit 0: ONLINE
B1 Bit 1: RESTART
B2 Bit 2: COMM_LOST
B3 Bit 3: REMOTE_FORCED
B4 Bit 4: LOCAL_FORCED

Group 1

B5 Bit 5: CHATTER_FILTER

Group 20

B5 Bit 5: ROLLOVER
B6 Bit 6: DISCONTINUITY

Group 30, 31 and 40

B5 Bit 5: OVER_RANGE
B6 Bit 6: REFERENCE_ERR

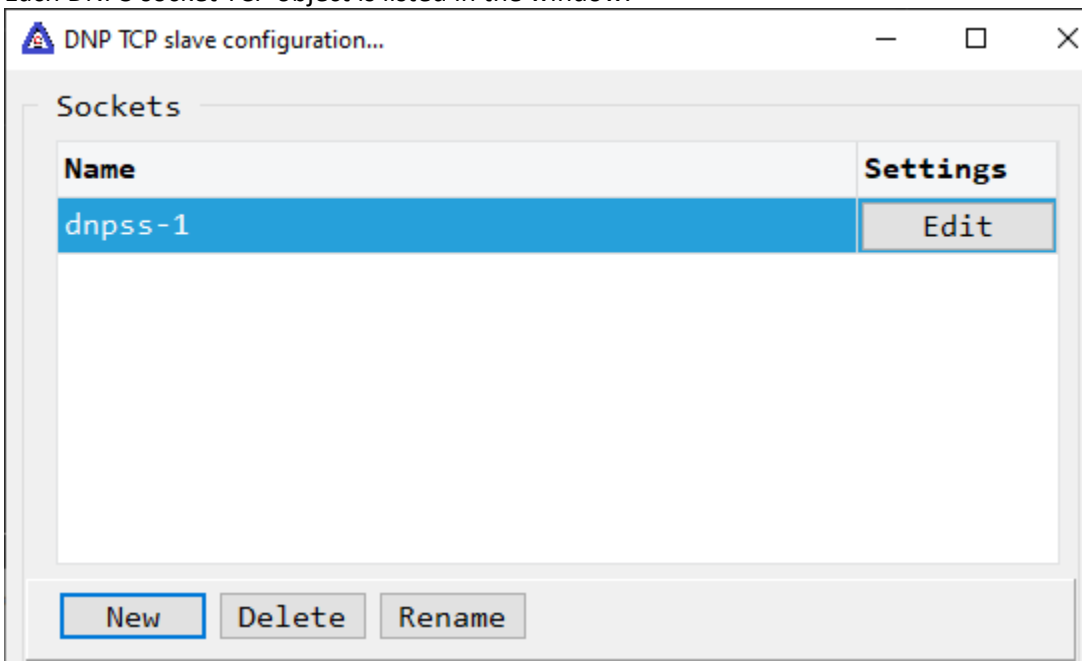
Point status defines for “Write status”

Group 12

Write status: The result of the write command. See the DNP3 specification for possible values.

DNP3 Outstation TCP

Each DNP3 socket TCP object is listed in the window.



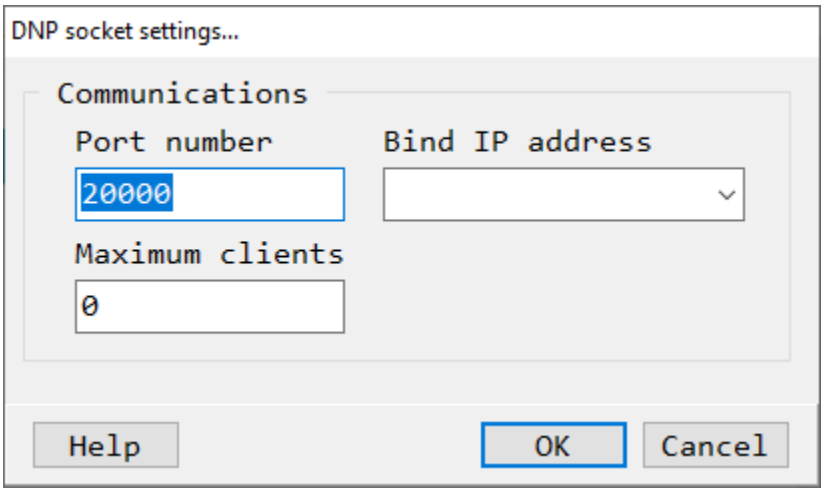
All sockets can access all outstations.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a DNP3 socket select the "Delete" button.

Settings



Port number

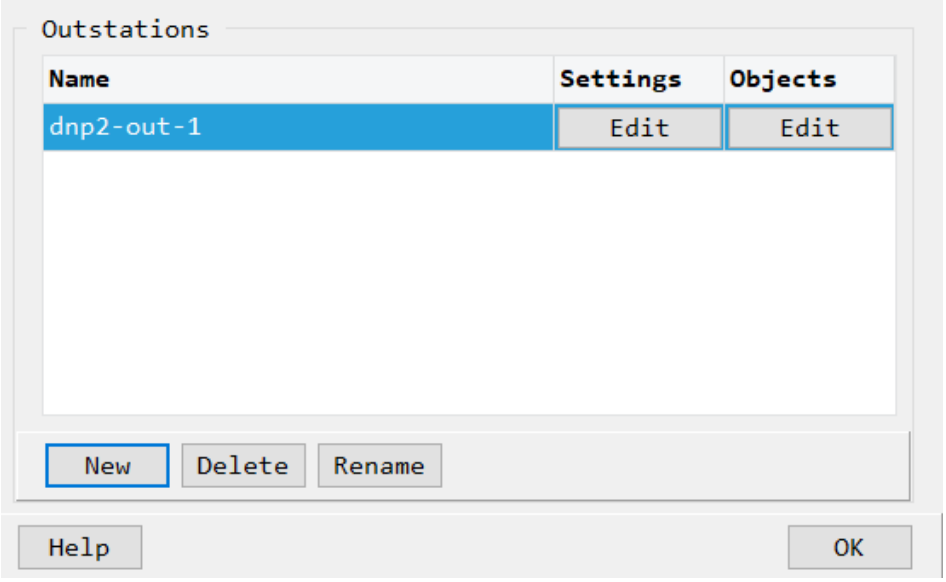
The TCP port number the master will connect to on the computer. 20000 is the default DNP3 port number.

Bind IP address See here.

Maximum clients

This is used to limit the number of masters that can connect using this socket. 0 = unlimited

Each DNP3 outstation object is listed in the window.

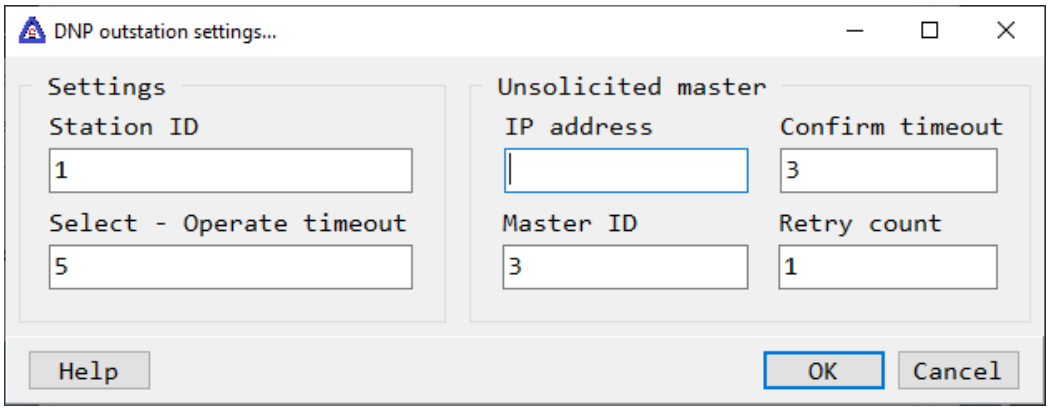


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a DNP3 outstation object select the "Delete" button.

Settings



Station ID

The outstation ID.

Select-Operate timeout (seconds)

When a "select" request is received, this timer begins timing. The "operate" request must be received before the timer completes or the "operate" request will fail.

Unsolicited master

This is the master station to send events. If event processing is not required, leave the IP address blank.
An outstation can send events to one master.
An outstation can respond to read and write request from any master.

IP address

The IP address of the master station configured to accept unsolicited events.

Master ID

The station ID of the master station configured to accept unsolicited events.

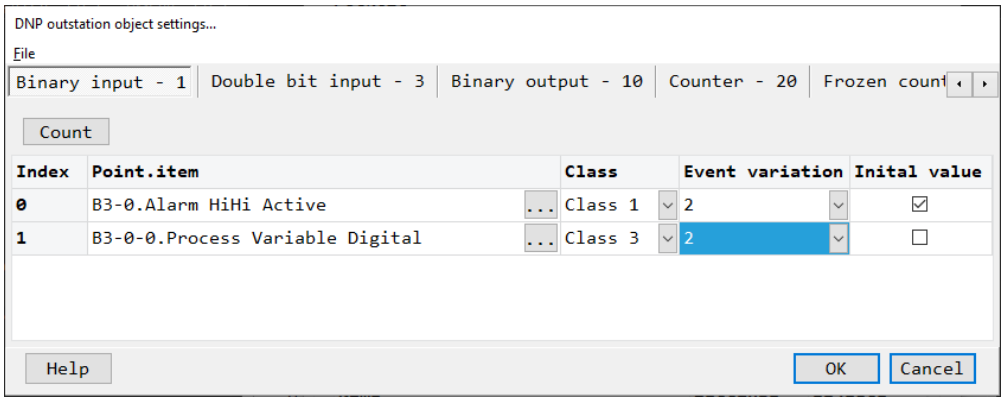
Confirm timeout (seconds)

The master must confirm receipt of each unsolicited message. If the master does not acknowledge the message it will be sent "Retry count" times.

Retry count

The number of times the outstation logic will resend unsolicited messages.

Objects



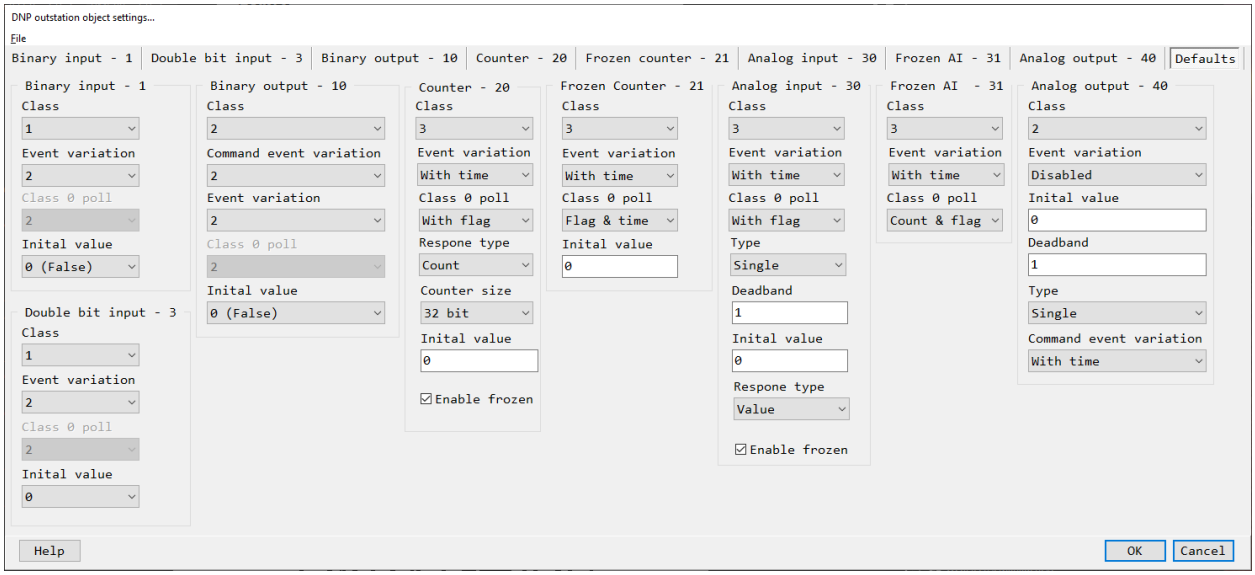
Response type (Counter - 20, Analog input - 30)

This applies to indexes that are configured as "Class 0" and returns the count or frozen count for a "Class 0" read.

Notes:

1) Binary output writes: Only "Latch On" and "Latch Off" are supported. "Zero count" functionality is supported.
2) Frozen counters do not require a point.item reference.
3) Frozen analog inputs do not require a point.item reference.
4) "All stations" addresses are not supported.

Defaults



See DNP3 specification Volume 2 Part 1, section 1.5.3 Classes, for class descriptions.
See DNP3 specification Volume 6 Part 2, for object types and data variation types.

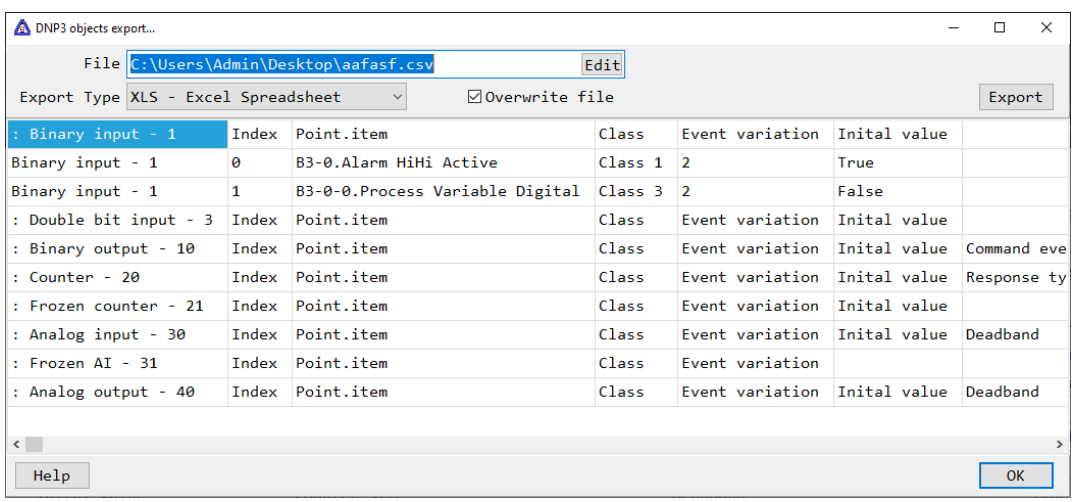
#	Object type	Class	Event variation	Class 0 poll variation	Initial value	Note
1	Binary input (BI)	1		2	0 (false)	
2	BI event		2			Any value change creates event
3	Double bit input	1		2	0	
4	Double bit input event		2			
10	Binary output (BO)	2		2	0 (false)	Note 10
11	BO event		2			Only flag changes generate event
13	BO command event		2			
20	Counter	3		1/2	0	Note 4
21	Frozen counter (FC)	3		5/6	0	Note 5
22	Counter event		5/6			Note 5
23	FC event		5/6			Note 5
30	Analog input (AI)	3		1/2/5/6	0	Note 6
31	Frozen AI	3		1/2/7/8		Note 7
32	AI event		3/4/7/8			Note 8
33	Frozen AI event		3/4/7/8			Note 8
40	Analog output status (AO)	2		1/2/3/4	0	Note 11,13
42	AO output event		3/4/7/8			Note 12
43	AO output command event		3/4/7/8			Note 12

Notes:

1) Variations with "relative time-of-occurrence" are not recommended. Use a variation with "time-of-occurrence".

2) Setting the "Class" to "No Class (4)" prevents event generation for the objects and static data read (Class 0 poll) will not return the static values.
3) Setting an event variation to "0" (zero) disables event generation for that object type.
4) If the counter is: 32 bit - variation 1, 16 bit - variation 2.
5) If the counter is: 32 bit - variation 5, 16 bit - variation 6.
6) If the input is: 32 bit - variation 1, 16 bit - variation 2, single precision - variation 5, double precision - variation 6.
7) If the input is: 32 bit - variation 1, 16 bit - variation 2, single precision - variation 7, double precision - variation 8.
8) If the input is: 32 bit - variation 3, 16 bit - variation 4, single precision - variation 7, double precision - variation 8.
9) Group 34 (Analog input deadband) is supported. This object type is not returned for a Class 0 static data poll, the class is 4.
10) Group 12, variation 1 is supported for Binary output commands.
11) Group 41, variations 1-4 are supported for Analog output commands.
12) If the output is: 32 bit - variation 3, 16 bit - variation 4, single precision - variation 7, double precision - variation 8.
13) If the output is: 32 bit - variation 1, 16 bit - variation 2, single precision - variation 3, double precision - variation 4.
14) The initial value does not issue a write to the connected point. It only sets the internal DNP3 storage location to the initial value. This value is used for change (comparison) event generation.
15) Group 40 deadband is only used for generation of group 42 events.

Export



: Binary input - 1	Index	Point.item	Class	Event variation	Inital value	
Binary input - 1	0	B3-0.Alarm HiHi Active	Class 1	2	True	
Binary input - 1	1	B3-0-0.Process Variable Digital	Class 3	2	False	
: Double bit input - 3	Index	Point.item	Class	Event variation	Inital value	
: Binary output - 10	Index	Point.item	Class	Event variation	Inital value	Command eve
: Counter - 20	Index	Point.item	Class	Event variation	Inital value	Response ty
: Frozen counter - 21	Index	Point.item	Class	Event variation	Inital value	
: Analog input - 30	Index	Point.item	Class	Event variation	Inital value	Deadband
: Frozen AI - 31	Index	Point.item	Class	Event variation		
: Analog output - 40	Index	Point.item	Class	Event variation	Inital value	Deadband

The export command exports all group/items to the filename and type selected.

File

Enter the destination file name. Use the edit button as needed to select a file and path. The file extension will automatically be added when the export button is selected.

Comma Separated Values
Excel Spreadsheet
Tab Separated Values

Overwrite file

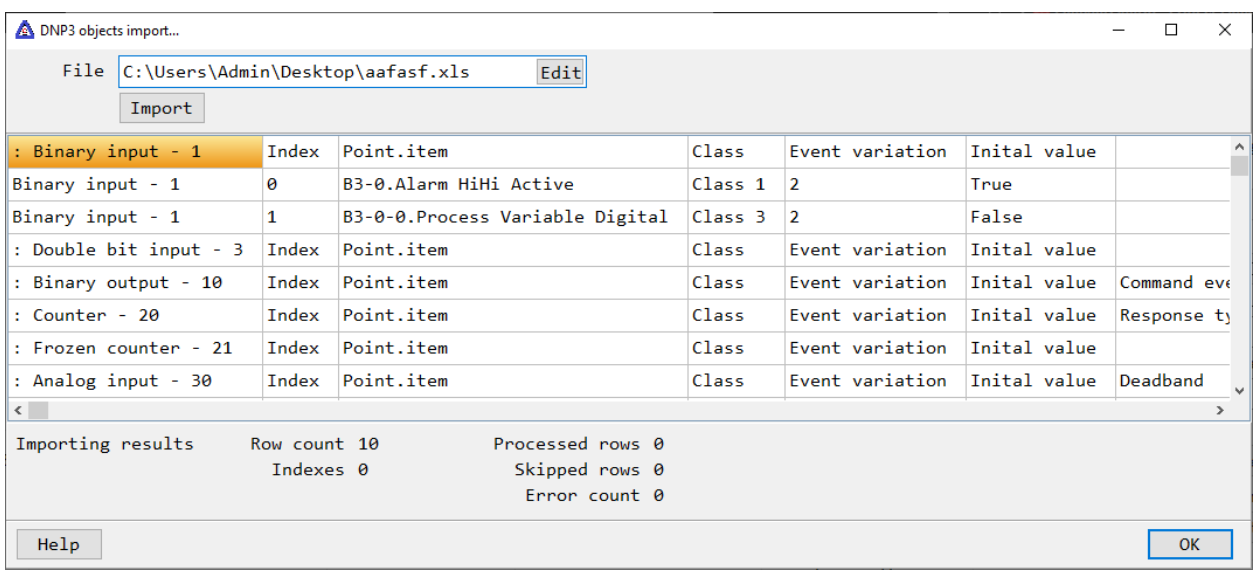
If enabled, and a file with the name supplied is detected the file will be deleted automatically. If disabled, and a file with the same name exists, a prompt will appear.

Export button

Select the export button to export the data in the grid.

Notes:
Each group type has a header row that defines the columns for the group type.
The indexes are grouped after the header row for that group type.
The header rows are always exported.
Excel does not need to be installed to export to XLS.
The first column is the type. The text matches the caption of the tab for the group.
If the first column begins with a ":" (colon character) the row/line will be not be processed.

Import



The import command imports all group/items from the filename and type selected.

File

Enter the input file name. Use the edit button as needed to select the file and path. The file extension will automatically determine the format of the file. Once the file is selected an attempt to import the contents to the grid will be performed. If the gird has data select the import button to attempt to import the data to the project database.

CSV	Comma Separated Values
XLS	Excel Spreadsheet
TXT	Tab Separated Values

Import button

When the grid has been filled, select the import button and an attempt to import the data will be performed. If errors are detected a window will be displayed indicating the row number and error message.

Importing results

Row count

The number of rows imported to the grid.

Indexes

The number of indexes successfully imported.

Processed rows

The number of grid rows processed.

Skipped rows

The number of gird rows skipped because the first character was a colon ':' or the first column of the row as blank.

Notes

1) The first column is the type. The text matches the caption of the tab for the group.
2) Text comparisons must match exactly.
3) If the first column begins with a ":" (colon character) the row/line will be not be processed.
4) All imported group/indexes overwrite existing indexes. If an index does not exist in the import file the index is not modified.
5) Excel does not need to be installed to import from an XLS file. Excel 2007 file format is not supported at present.
6) The importing of indexes does not perform extensive error checking. It is much like the editor. For example in the editor a "point.item" can be entered that does not exist. This is

permitted because the point.item may be created at a later time and the runtime engine would disregard any point.item that are not valid.

Enron RTU over TCP/IP

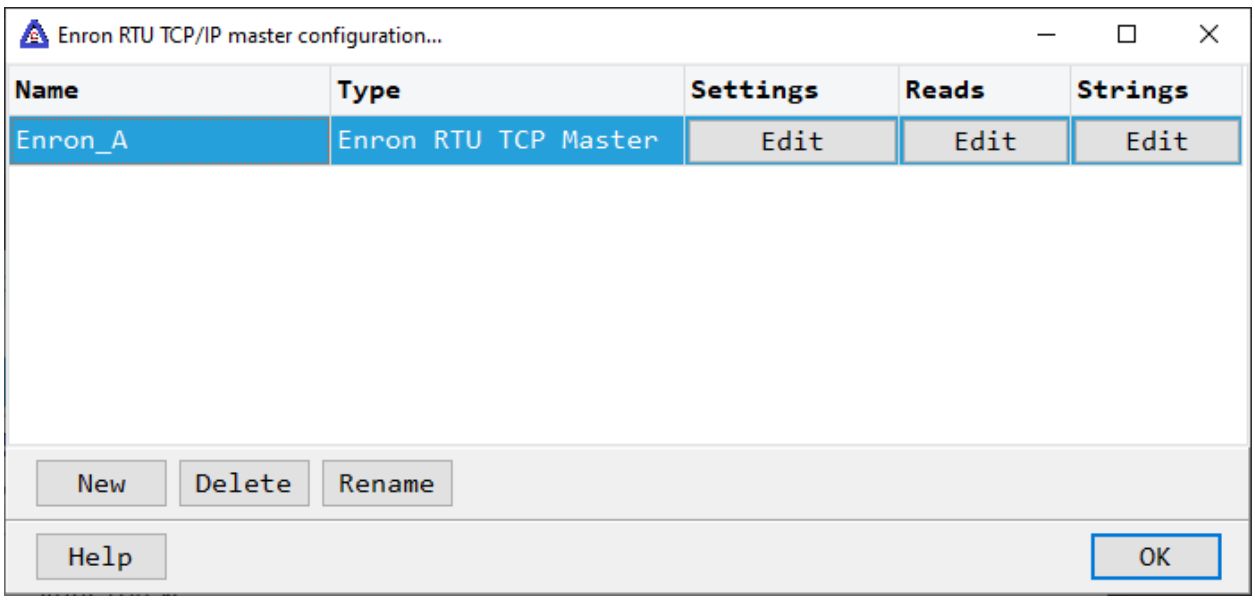
Each Enron RTU over TCP/IP master object is listed in the window.

There are two protocols for Enron; Enron RTU serial and Enron TCP/IP.
The Enron RTU over TCP/IP port is the Enron RTU serial protocol carried by the TCP/IP protocol.

Enron RTU serial

Enron TCP/IP

For example: a device may only support the Enron RTU serial protocol. A need to access the device via TCP/IP exists. This allows for a serial to TCP/IP converter to be placed at the device and accessed using the Enron over TCP/IP port type.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

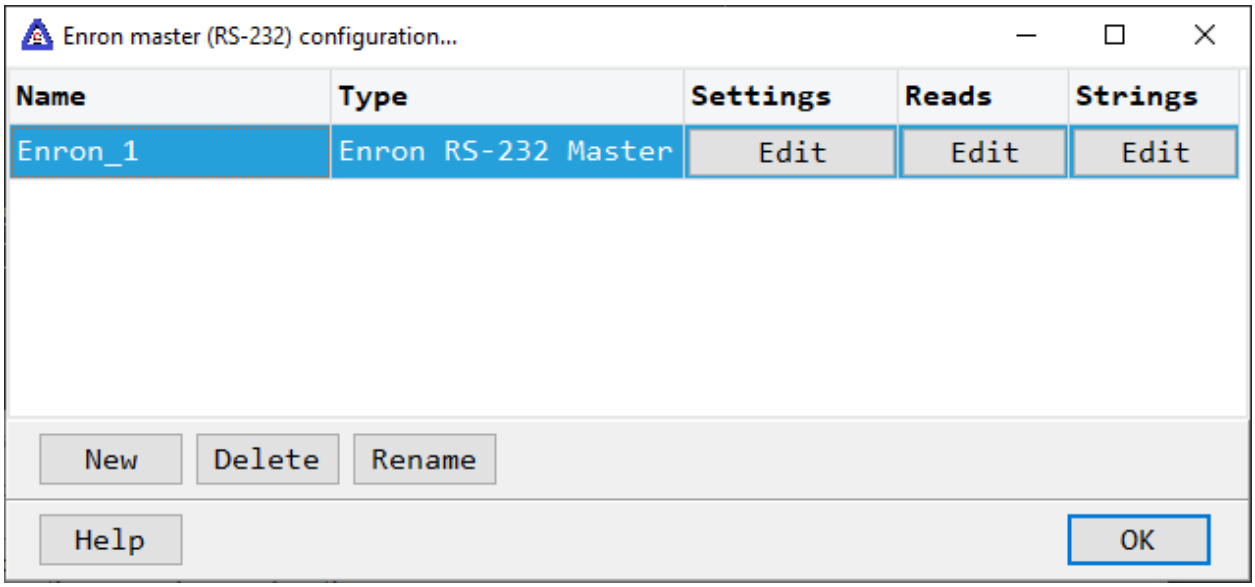
To delete an Enron RTU over TCP/IP master object select the "Delete" button.

The settings are the same as Enron TCP/IP.

The reads and strings are the same as Enron Serial.

ENRON SERIAL

Each Enron serial master object is listed in the window.

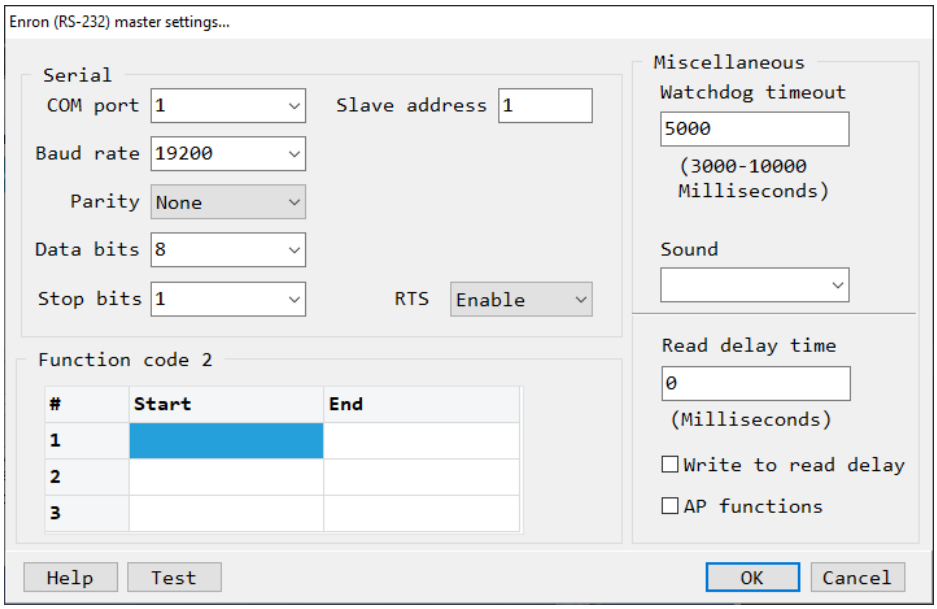


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an Enron serial master object select the "Delete" button.

Settings



Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Function code 2

Enron 4/5 digit addressing uses each “thousands” as a section with 999 elements. (e.g. 00xxx, 13xxx, 07xxx) **Note:** A zero address does not exist for any section. e.g. 1000, 17000, 32000 are all illegal addresses.

Function code 1 is used to read Boolean data types and function code 3 is used to read all other data types.

If required, use these fields to specify 1 to 3 areas of Boolean data. If a Boolean read is configured, with a starting register within the bounds of the fields, the HMI will use function code 2 to read the data.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

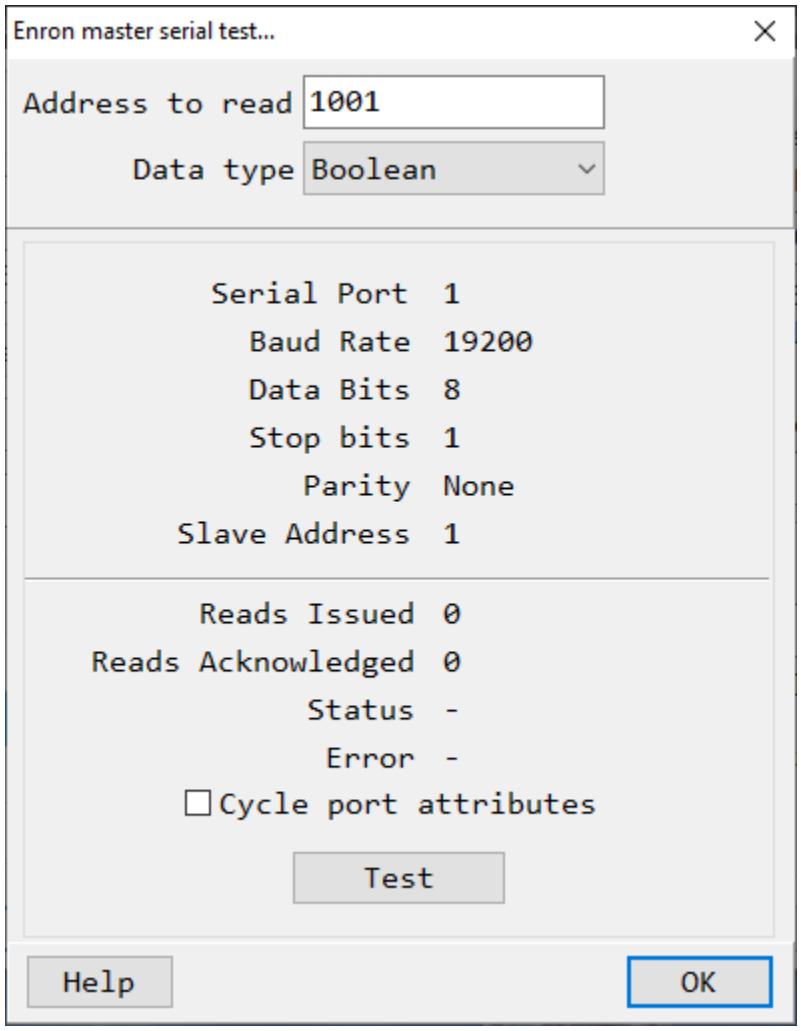
Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

AP functions

See analog functions.

Test button

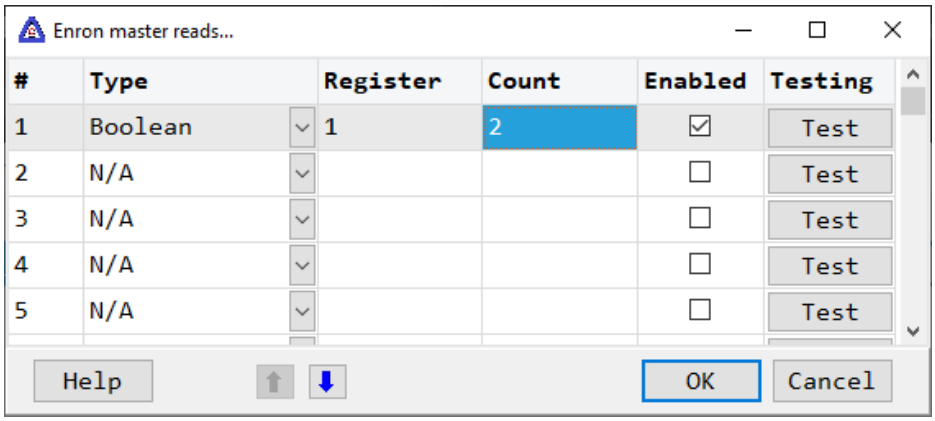


When the test button is selected the program will attempt to read one element of data from the device at the address entered.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads



The address ranges shown may or may not be present in the slave device.

Type

Any of the following register types are valid:

Boolean, 16 bit short integers, 32 bit long integers, 32 bit floats
8 character strings, 16 character strings

Note: Do not configure reads that overlap. Two reads that contain the same address will cause the second address to be ignored.

Register range

The HMI uses the Enron register address numbering system.

```
0001 – 0999
...
99001 – 99999
```

Notes:

1) A zero address does not exist. e.g. 1000, 17000, 32000 are all illegal addresses.
2) The address must contain at least 4 digits and less than 6. i.e. 0001, 99999

Count

The count is the number of a 'type' to read. Each read can request up to 125 words of data. Each data type has a limit.

Data type	**Max count per request**
Booleans	2000

16 bit short integers	125
8 character strings	31 (Each element contains 2 characters)
32 bit long integers	62
32 bit floats	62
16 character strings	15 (Each element contains 2 characters)

Enabled

The read requests are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

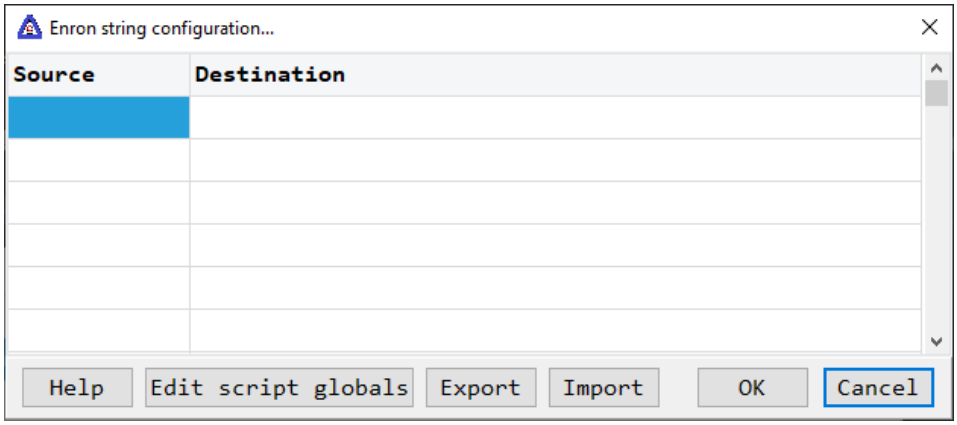
Test

When the button is selected the program will attempt to access the data in the device using the communication parameters configured.

Error messages

No register address	The value in the field is not a Enron address.
Invalid count	The count must be between 1 and 2000.
Exceeds 999	The register plus the count exceeds the maximum address range.
Invalid Enron address	The value in the field is an invalid Enron address.

Strings



Strings are not regular points. They do not have alarms, printing, etc. Strings may be displayed in any window via the "Script Global" animation. Any other actions, parsing, comparing, etc. must be done in scripts.

Source

This is the register address in the PLC.

Destination (optional)

If desired, select a script global location and the string will be copied to the location when the string value is returned by the external device. If this option is used or not the string may still be accessed via the "StringGet" and "StringSet" script commands. If this is configured and access to the string, in a script, is desired use the "GlobalGet" or "StringGet" to copy the string. The format of the destination is: <section>.<name>.
Note: Writing to the script global does not write the value to the PLC. "StringSet" must be used to write a string to the PLC.

Export

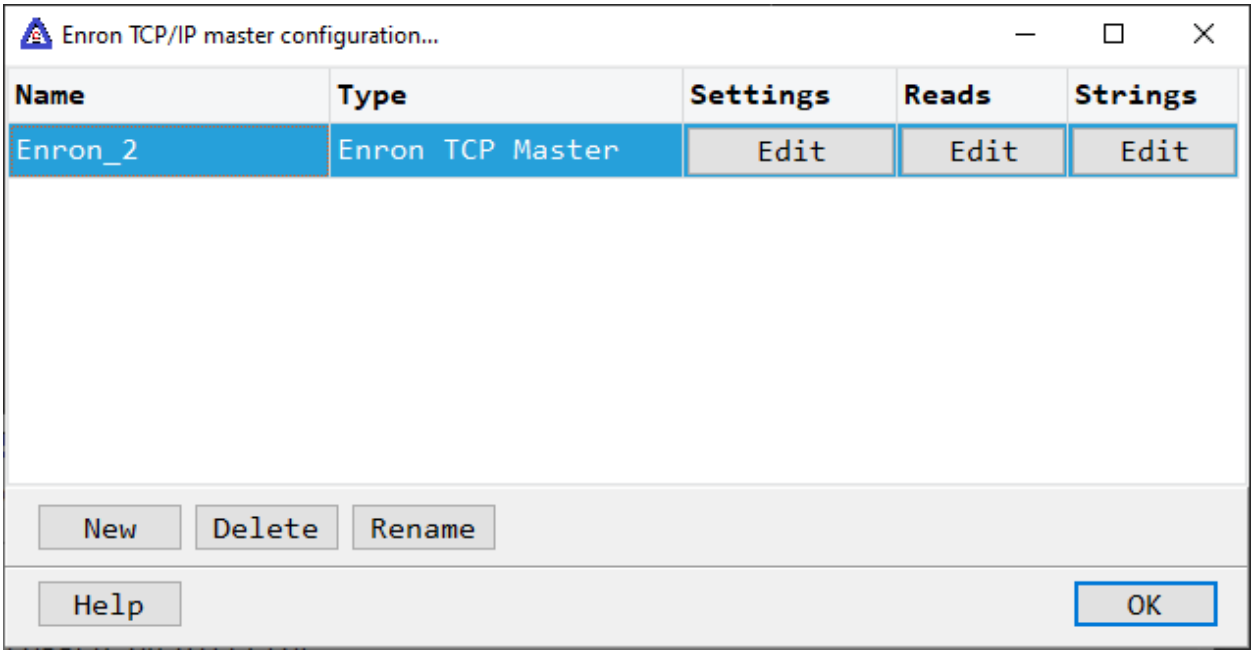
This is used to export the configuration to an Excel worksheet.

Import

This is used to import string configurations from an Excel worksheet. The contents of the grid are completely replaced by the contents of the file.

Enron TCP/IP

Each Enron TCP/IP master object is listed in the window.

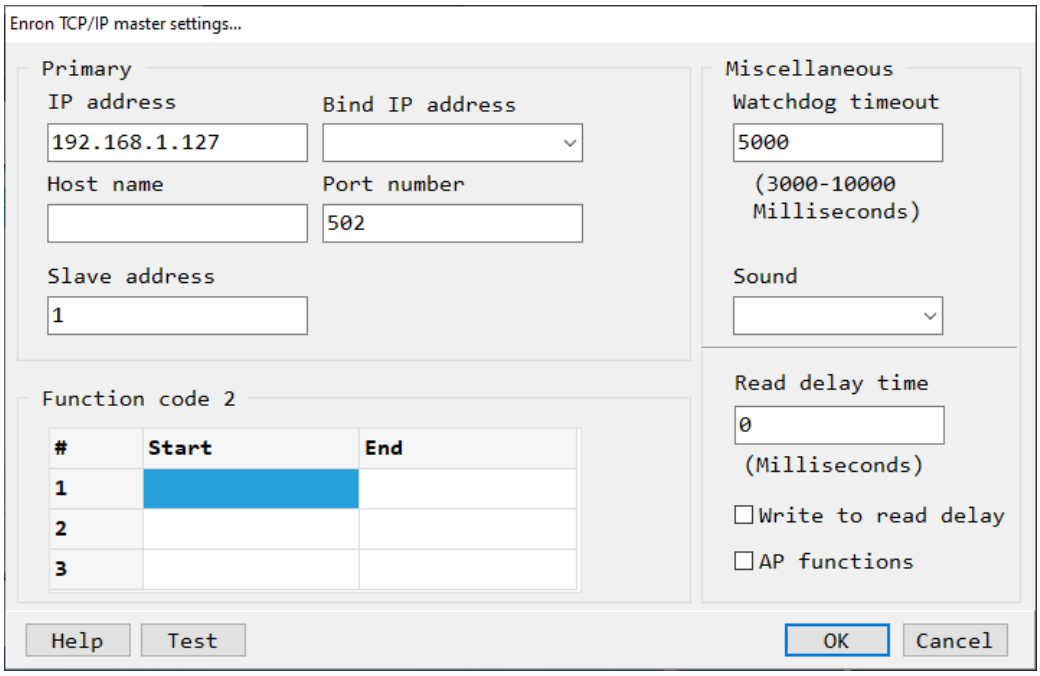


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an Enron TCP/IP master object select the "Delete" button.

Settings



Select the port attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Function code 2

Enron 4/5 digit addressing uses each “thousands” as a section with 999 elements. (e.g. 00xxx, 13xxx, 07xxx) **Note:** A zero address does not exist for any section. e.g. 1000, 17000, 32000 are all illegal addresses.

Function code 1 is used to read Boolean data types and function code 3 is used to read all other data types.

If required, use these fields to specify 1 to 3 areas of Boolean data. If a Boolean read is configured, with a starting register within the bounds of the fields, the HMI will use function code 2 to read the data.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

AP functions

See analog functions.

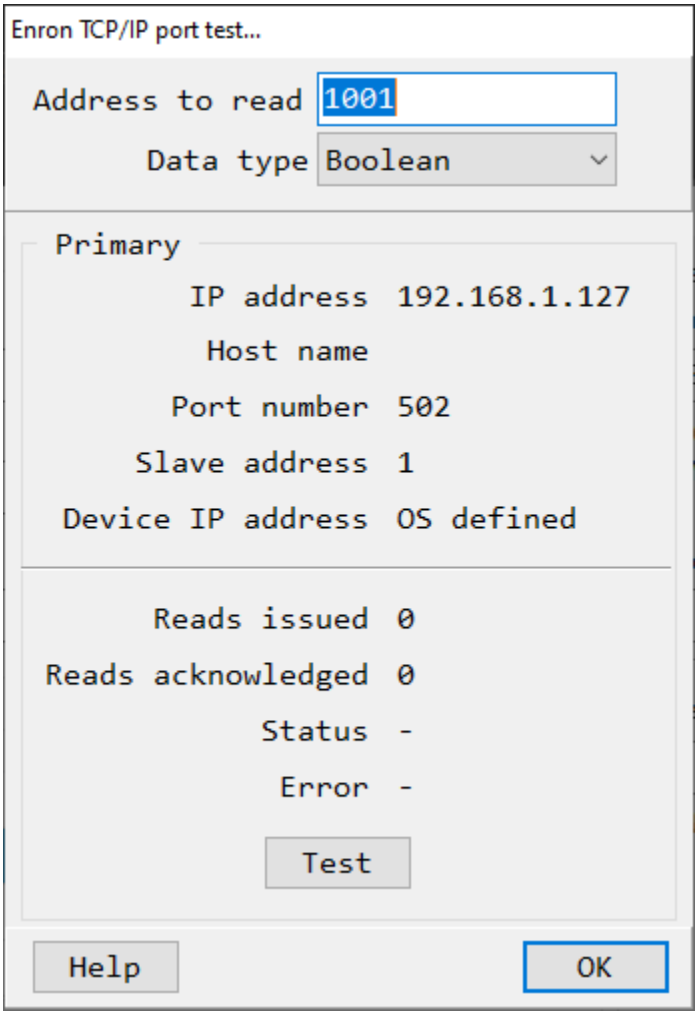
Reads

The reads configuration is the same as Enron serial.

Strings

The strings configuration is the same as Enron serial.

Test button

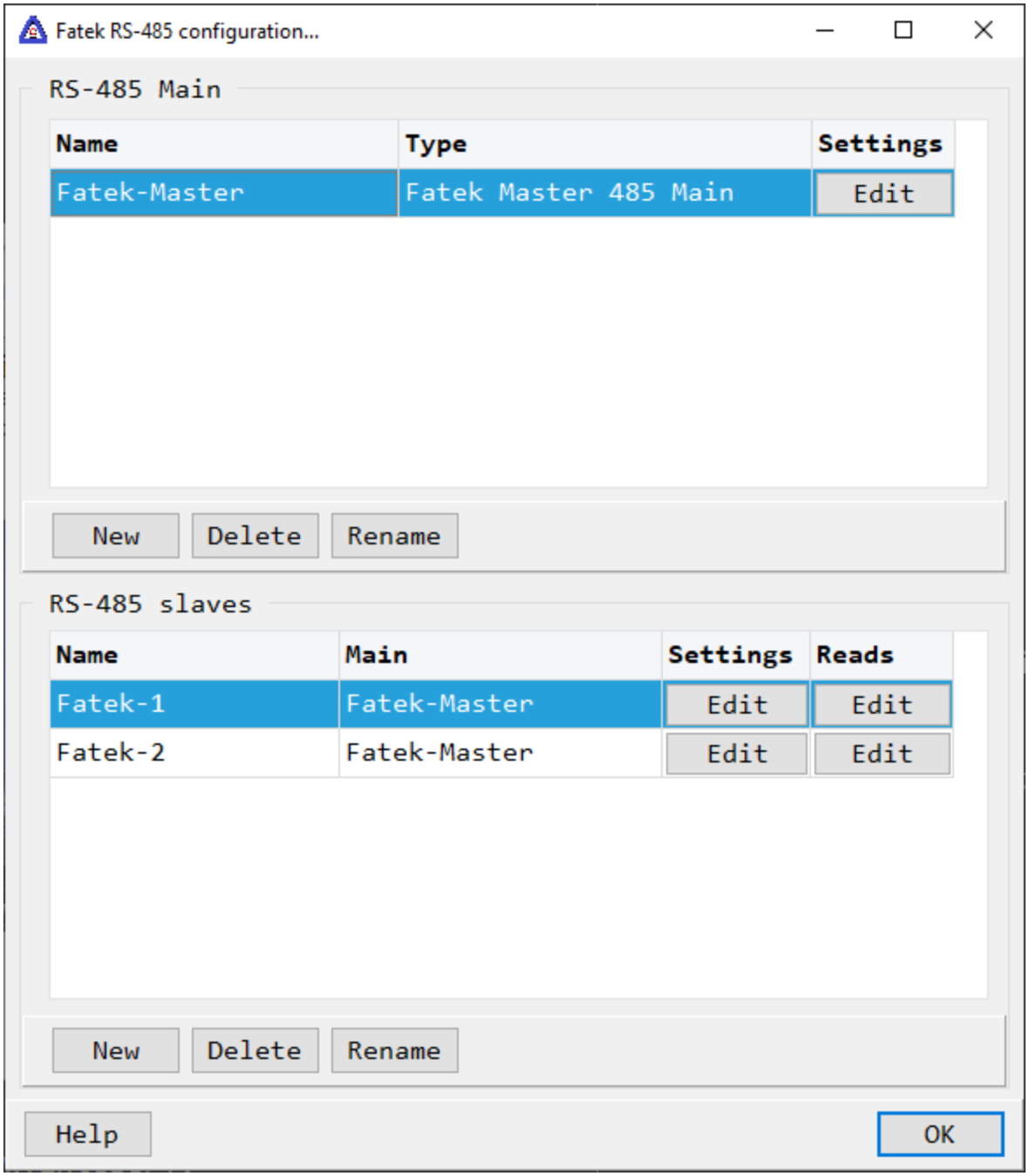


When the test button is selected the program will attempt to read one element of data from the device at the address entered.

The program will attempt to use the communication parameters configured.

FATEK SERIAL

Each Fatek RS-485 master object controls one serial port and can have one or many slave objects. The slaves are configured to address one device.



Note: RS-485 is sensitive to improper wiring and/or terminations. Unpowered units can cause data echoes and other reliability issues. Please follow all RS-485 wiring guidelines.

Fatek RS-485 serial main

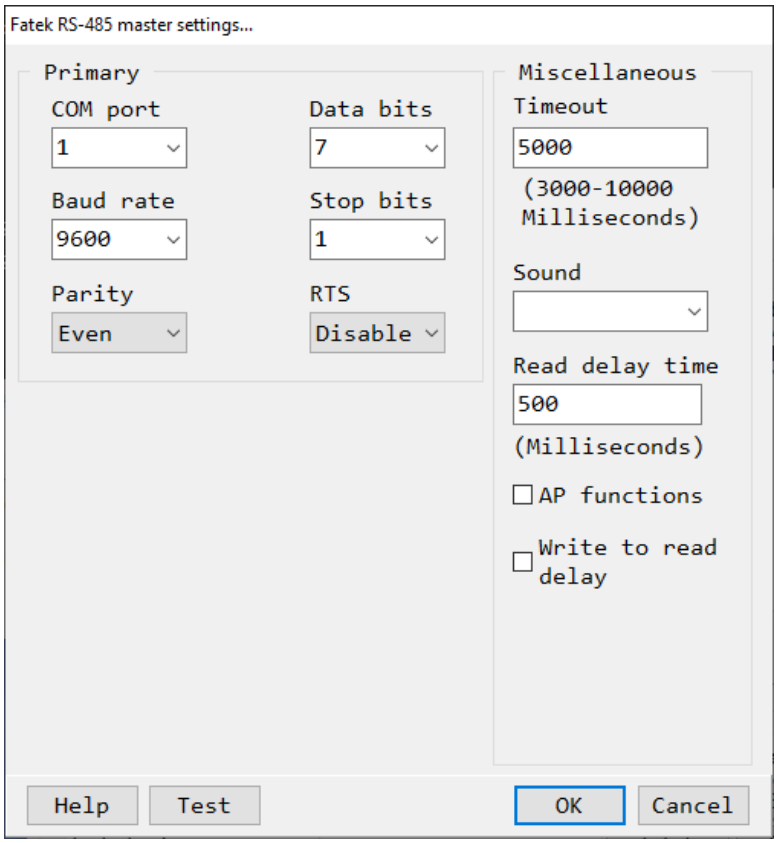
Each Fatek RS-485 master object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Fatek master object select the "Delete" button.

Settings



Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

RTS

See here.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

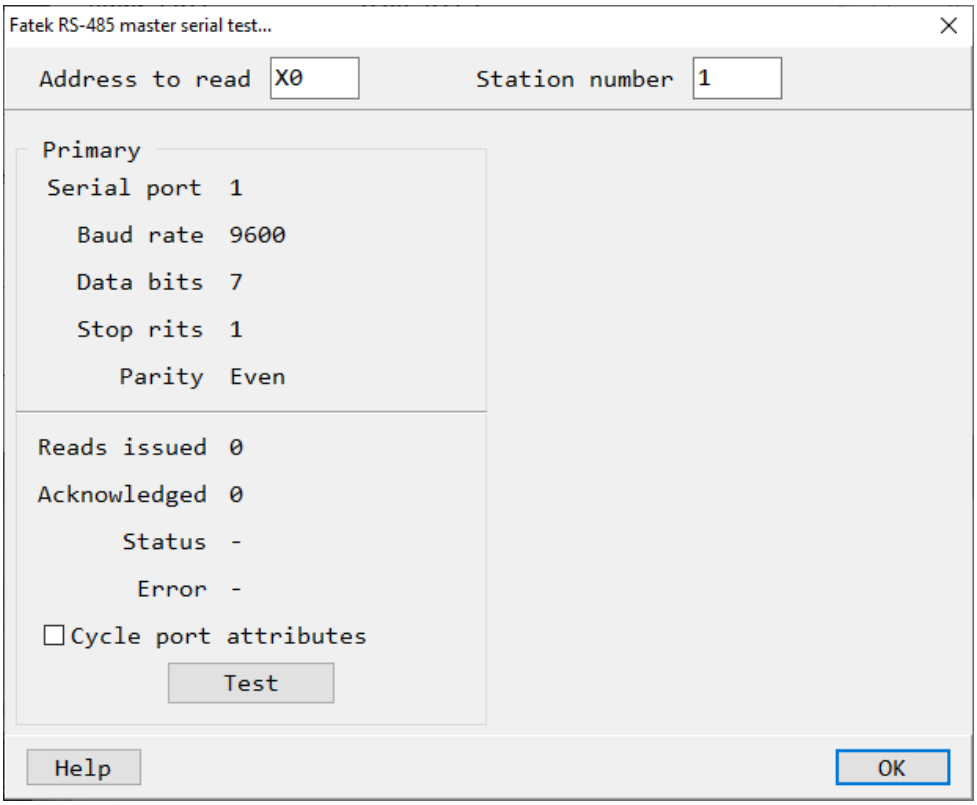
Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

AP functions

See analog functions.

Test button



When the test button is selected the program will attempt to read one point of data from the device at the address entered.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Fatek RS-485 slaves

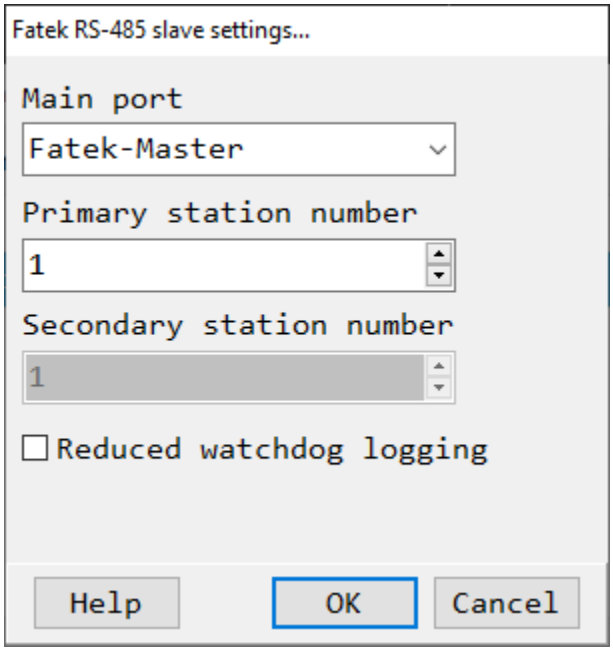
Each Fatek RS-485 slave object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Fatek slave object select the "Delete" button.

Settings



Main port

This is the master RS-485 port the slave is linked to. A port must be linked to a master for runtime operations.

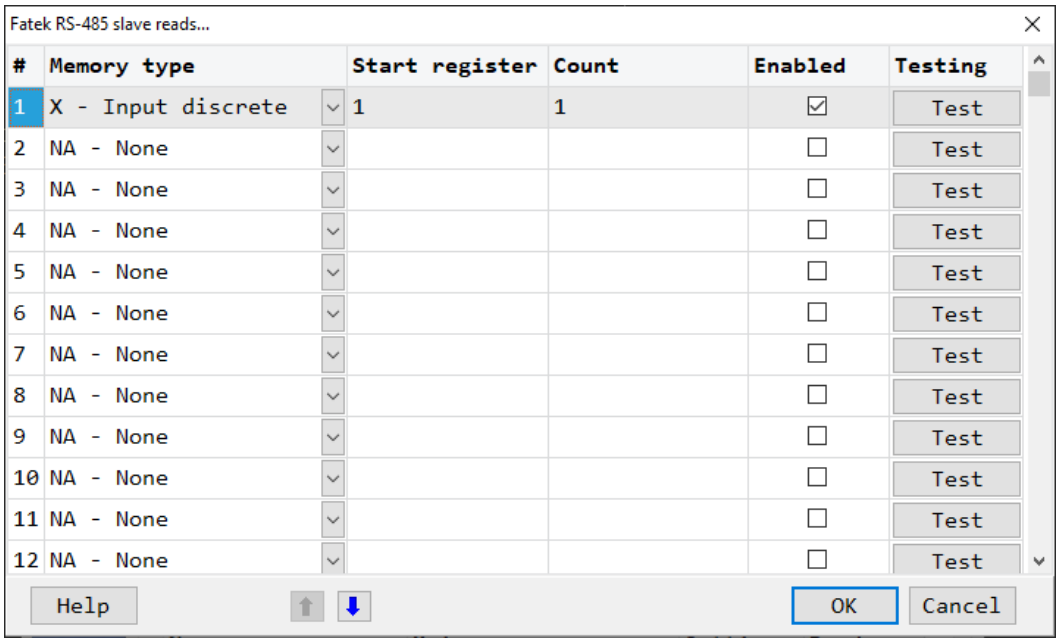
Primary station number
Secondary station number (Future)

This is the station number of the slave device for the primary and secondary. (1 - 255)

Reduced watchdog logging

When enabled and a watchdog timeout occurs, a single entry will be placed in the event log and the sound will be queued (if configured). When the slave device responds, after a watchdog timeout, a single entry will be placed in the event log. The watchdog counter will continue to count each time the watchdog timer expires.

Reads



The address ranges shown may or may not be present in the slave device.

Registers

Address Examples
X0, X28
Y0, Y4
TMR0, CTR105

Memory type

Register Type	Prefix	Data Type
Input discrete	X	Boolean
Output relay	Y	Boolean
Internal relay	M	Boolean
Step relay	S	Boolean
Timer discrete	T	Boolean
Counter discrete	C	Boolean
Timer register	TMR	Word

Counter register	CTR	Word
Register	R	Word
Data Register	D	Word

Notes:
1) The analog point 'source type' will determine if 1 or 2 words are used.
2) In WinProLadder, using a FBs-10MA PLC, the status window, R uses 1 word and DR uses 2 words, both using the same memory area.
Caution: The addressing has overlap.
R0 is word 0.
R1 is word 1.
DR0 is word 0 and 1.
DR1 is word 1 and 2.

Start register

This is the start register value for the read. Each line is one read from the HMI to the PLC requesting data from the PLC.

Count

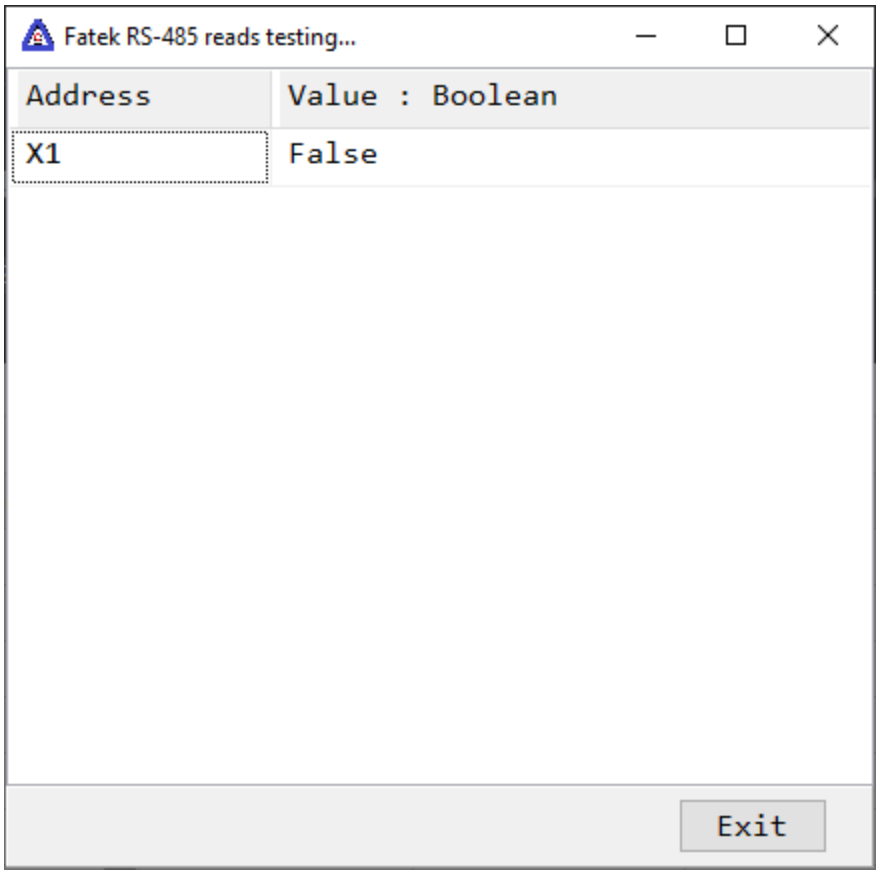
This is the number of points to return. For Boolean types the per read point limit is 256. For Word types the per read point limit is 64.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



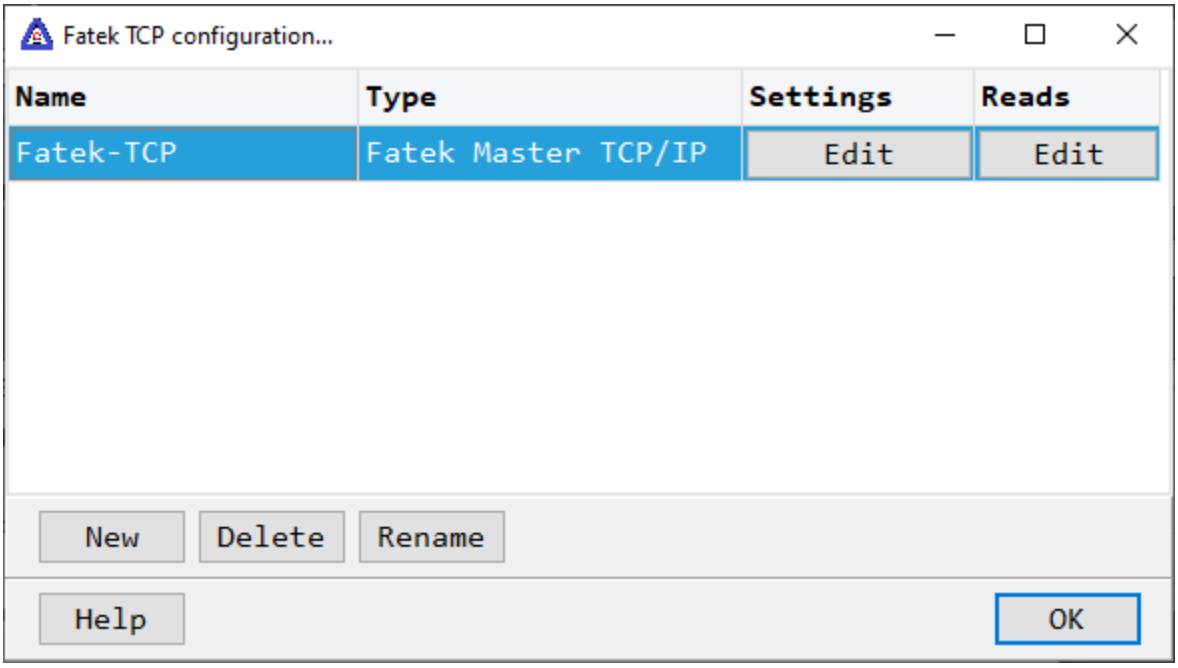
When the button is selected the program will attempt to access the data in the device using the communication parameters configured.

Error messages

No memory type selected	The memory type has not been selected.
Start out of range	The value is out of range for the type.
Count exceeds limit	The count must be 64 or less for words types; 256 or less for Boolean types.
Start + count exceeds register limit	The register plus the count exceeds the maximum address range.
Count < 1	The line must read at least on register.

FATEK TCP

Each Fatek TCP/IP object is listed in the window.

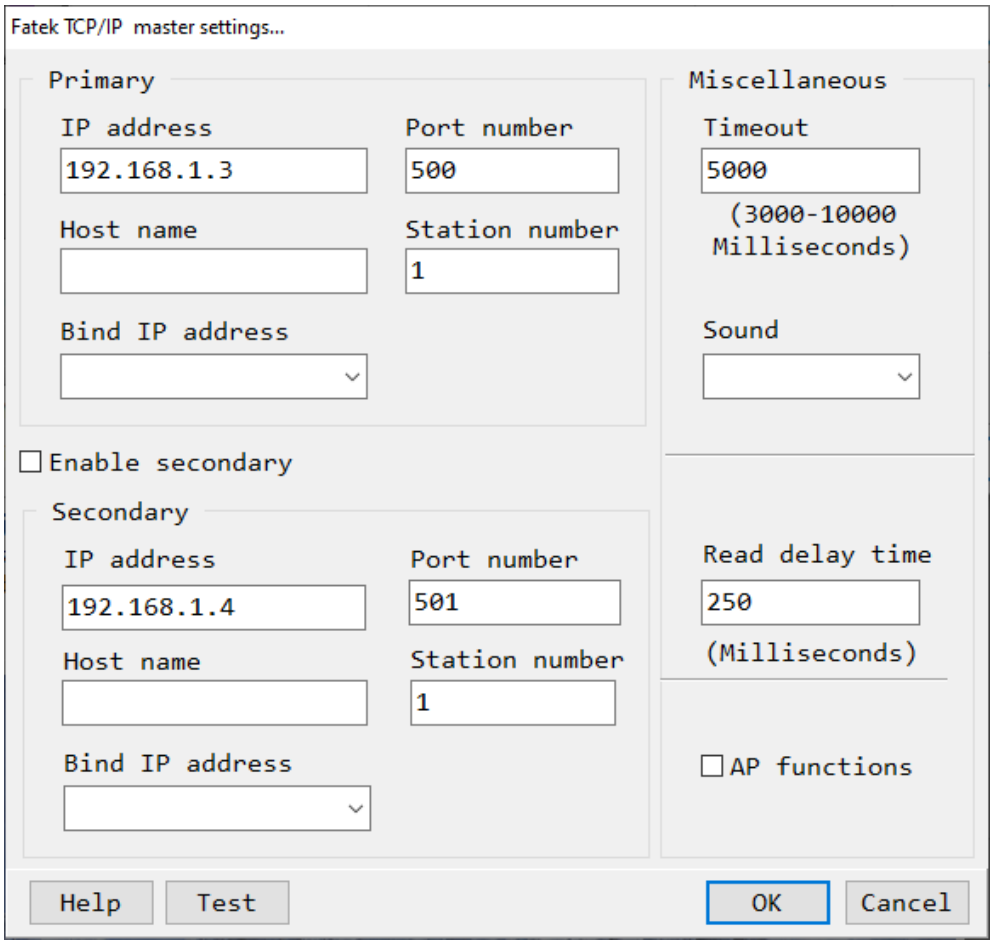


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Fatek object select the "Delete" button.

Settings



The port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Port number

This is the port number used for TCP communication. The default port number is 500.

Station number

This is the station number of the slave device. (1 - 255)

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second path to be used as a hot backup. When in run mode the primary configuration is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary configuration. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary configuration, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary configuration an entry will be made in the event log and operations will return to the primary port.

If the computer has two network interface cards (NIC) the program will use the first one detected for the primary and the second one detected for the secondary. If only one NIC is detected the program will use it for the primary and the secondary.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sounds delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

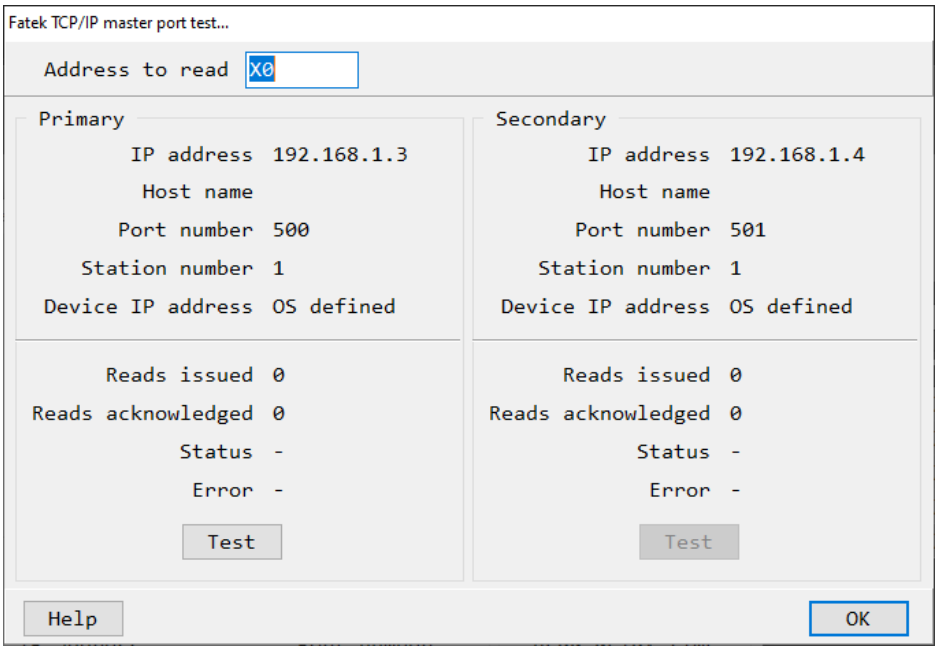
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button



When the test button is selected the program will attempt to read one point of data from the device at the address entered.

If a "host name" is entered, the IP address is ignored.

Reads

Fatek TCP/IP reads...

#	Memory type	Start register	Count	Enabled	Testing
1	X - Input discrete	1	1	☑	Test
2	NA - None			☐	Test
3	NA - None			☐	Test
4	NA - None			☐	Test
5	NA - None			☐	Test
6	NA - None			☐	Test
7	NA - None			☐	Test
8	NA - None			☐	Test
9	NA - None			☐	Test
10	NA - None			☐	Test
11	NA - None			☐	Test
12	NA - None			☐	Test

Help OK Cancel

Registers

Address Examples
X0, X28
Y0, Y4
TMR0, CTR105

Memory type

Register Type	Prefix	Data Type
Input discrete	X	Boolean
Output relay	Y	Boolean
Internal relay	M	Boolean
Step relay	S	Boolean
Timer discrete	T	Boolean
Counter discrete	C	Boolean
Timer register	TMR	Word
Counter register	CTR	Word
Register	R	Word
Data Register	D	Word

Notes:
1) The analog point 'source type' will determine if 1 or 2 words are used.
2) In WinProLadder, using a FBs-10MA PLC, the status window, R uses 1 word and DR uses 2 words, both using the same memory area.
Caution: The addressing has overlap.

R0 is word 0.
R1 is word 1.
DR0 is word 0 and 1.
DR1 is word 1 and 2.

Start register

This is the start register value for the read. Each line is one read from the HMI to the PLC requesting data from the PLC.

Count

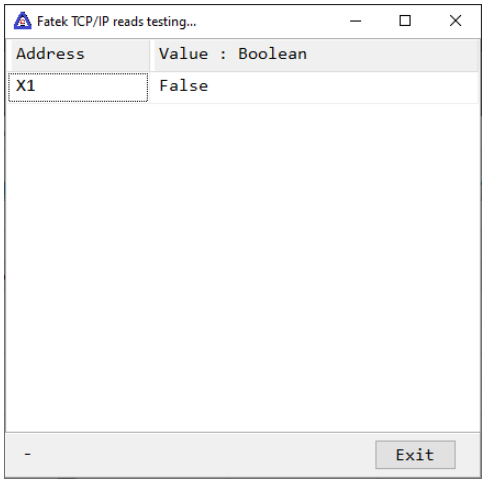
This is the number of points to return. For Boolean types the per read point limit is 256. For Word types the per read point limit is 64.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that show some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



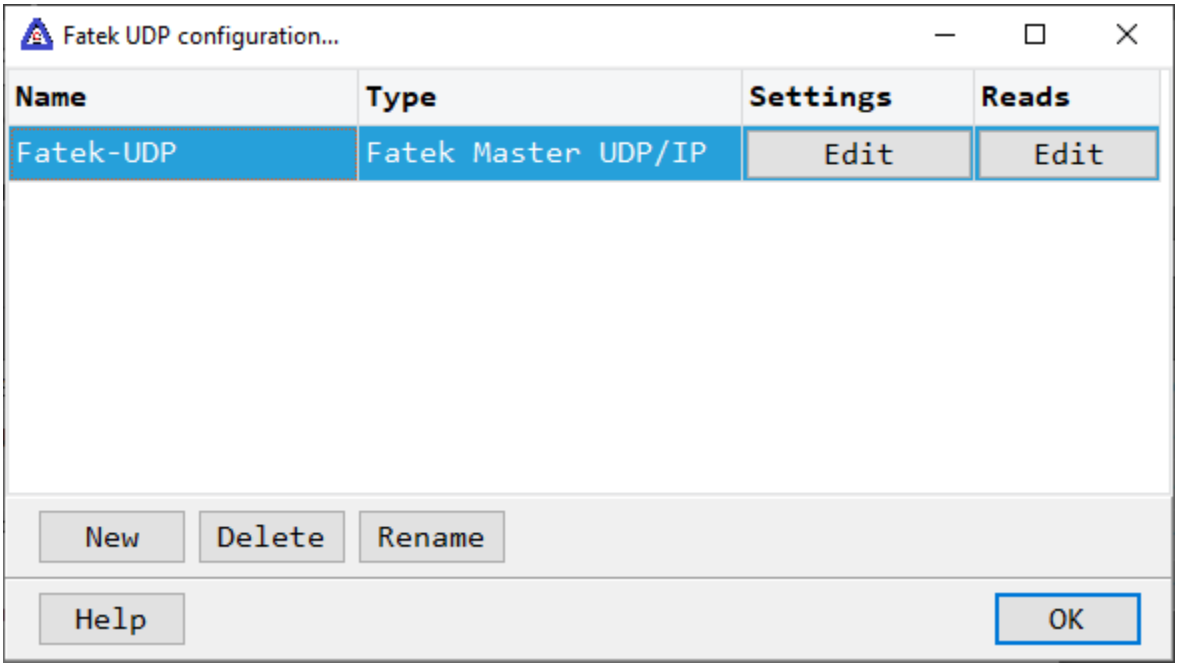
When the button is selected the program will attempt to access the data in the device using the communication parameters configured.

Error messages

No memory type selected	The memory type has not been selected.
Start out of range	The value is out of range for the type.
Count exceeds limit	The count must be 64 or less for words types; 256 or less for Boolean types.
Start + count exceeds register limit	The register plus the count exceeds the maximum address range.
Count < 1	The line must read at least on register.

FATEK UDP

Each Fatek UDP/IP object is listed in the window.

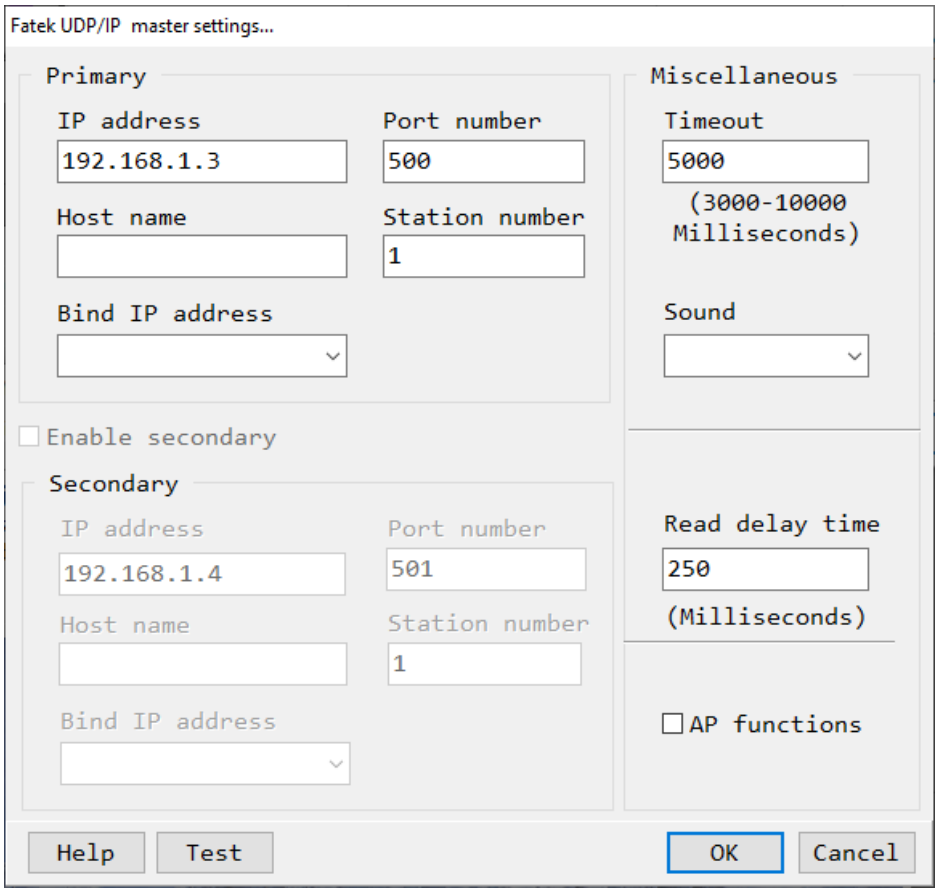


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Fatek object select the "Delete" button.

Settings



The port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Port number

This is the port number used for UDP communication. The default port number is 500.

Note: For UDP, port 500 might be in use by the operating system. If using the 'Test' button for checking port settings and the status field is displaying 'Port opening...' the port number selected might be in use.

Station number

This is the station number of the slave device. (1 - 255)

Bind IP address See here.

Enable secondary (disabled)

The "Enable secondary" checkbox provides for a second path to be used as a hot backup. When in run mode the primary configuration is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary configuration. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary configuration, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary configuration an entry will be made in the event log and operations will return to the primary port.

If the computer has two network interface cards (NIC) the program will use the first one detected for the primary and the second one detected for the secondary. If only one NIC is detected the program will use it for the primary and the secondary.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sounds delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

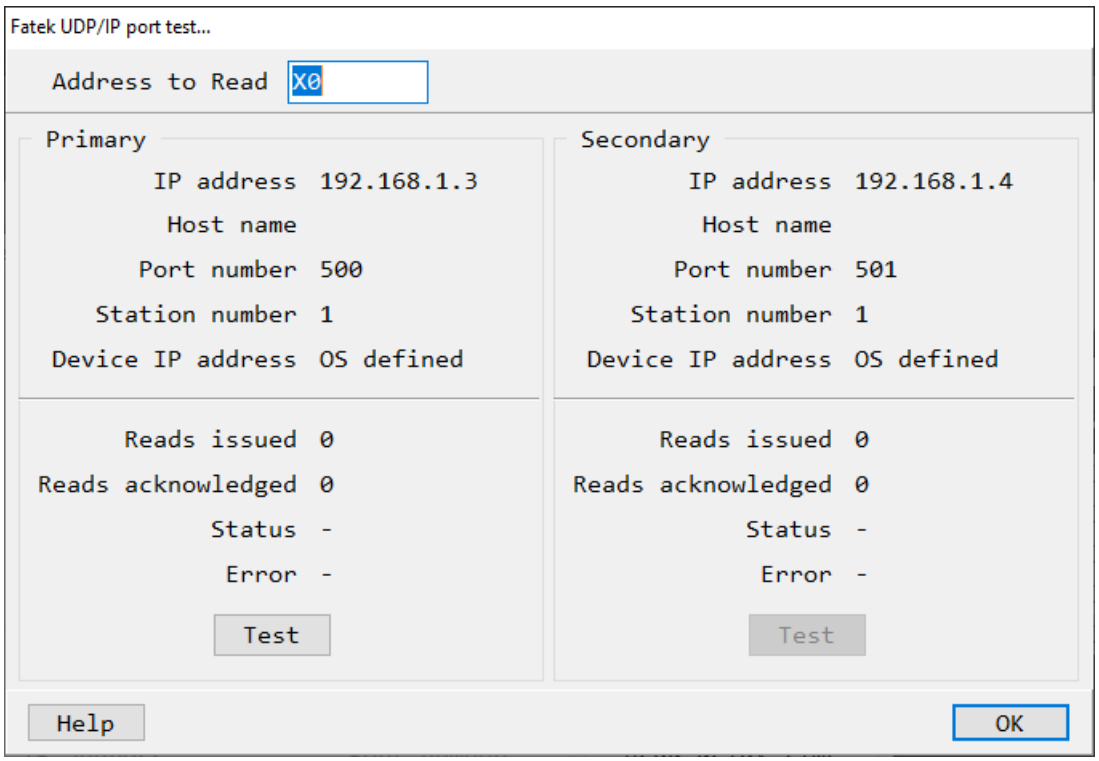
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

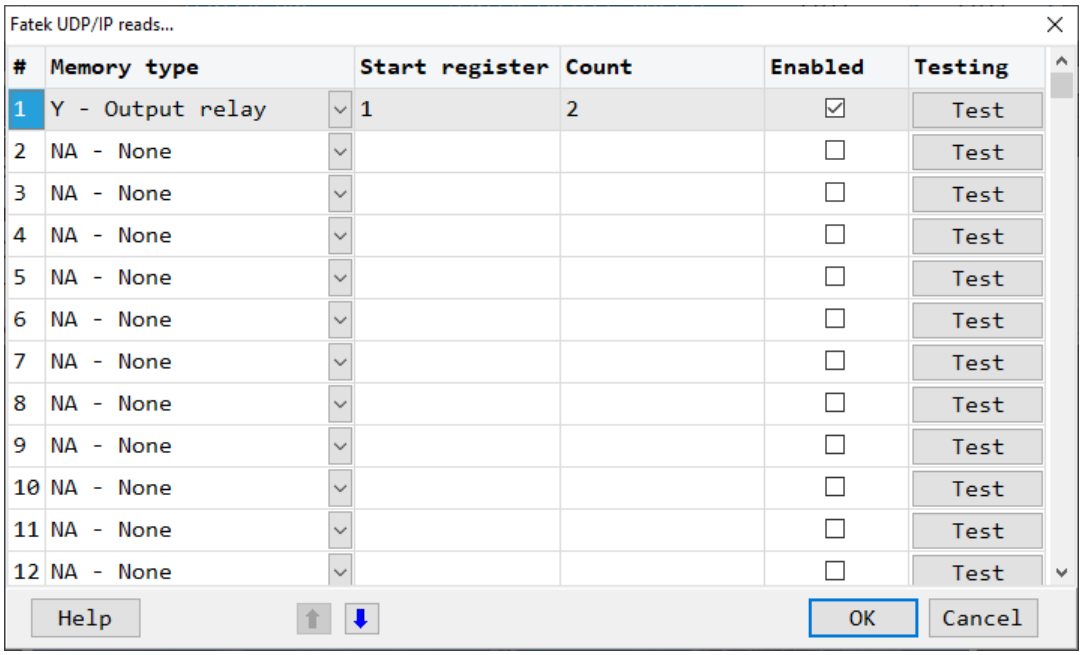
See analog functions.

Test button



When the test button is selected the program will attempt to read one point of data from the device at the address entered.

Reads



Registers

Address Examples
X0, X28
Y0, Y4
TMR0, CTR105

Memory type

Register Type	Prefix	Data Type
Input discrete	X	Boolean
Output relay	Y	Boolean
Internal relay	M	Boolean
Step relay	S	Boolean
Timer discrete	T	Boolean
Counter discrete	C	Boolean
Timer register	TMR	Word
Counter register	CTR	Word
Register	R	Word
Data Register	D	Word

Notes:
1) The analog point 'source type' will determine if 1 or 2 words are used.

2) In WinProLadder, using a FBs-10MA PLC, the status window, R uses 1 word and DR uses 2 words, both using the same memory area.
Caution: The addressing has overlap.
R0 is word 0.
R1 is word 1.
DR0 is word 0 and 1.
DR1 is word 1 and 2.

Start register

This is the start register value for the read. Each line is one read from the HMI to the PLC requesting data from the PLC.

Count

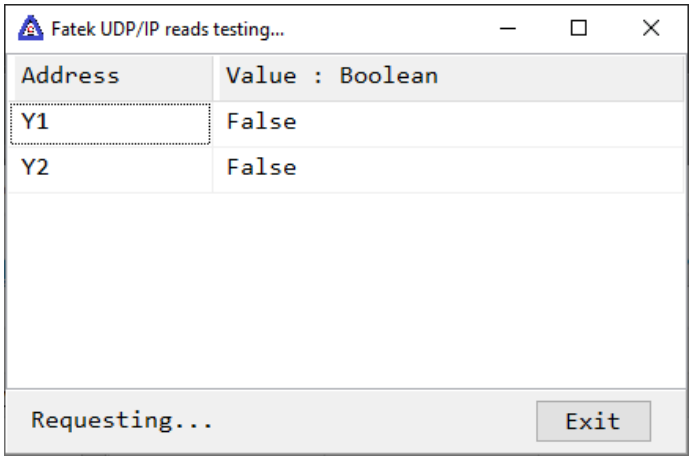
This is the number of points to return. For Boolean types the per read point limit is 256. For Word types the per read point limit is 64.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



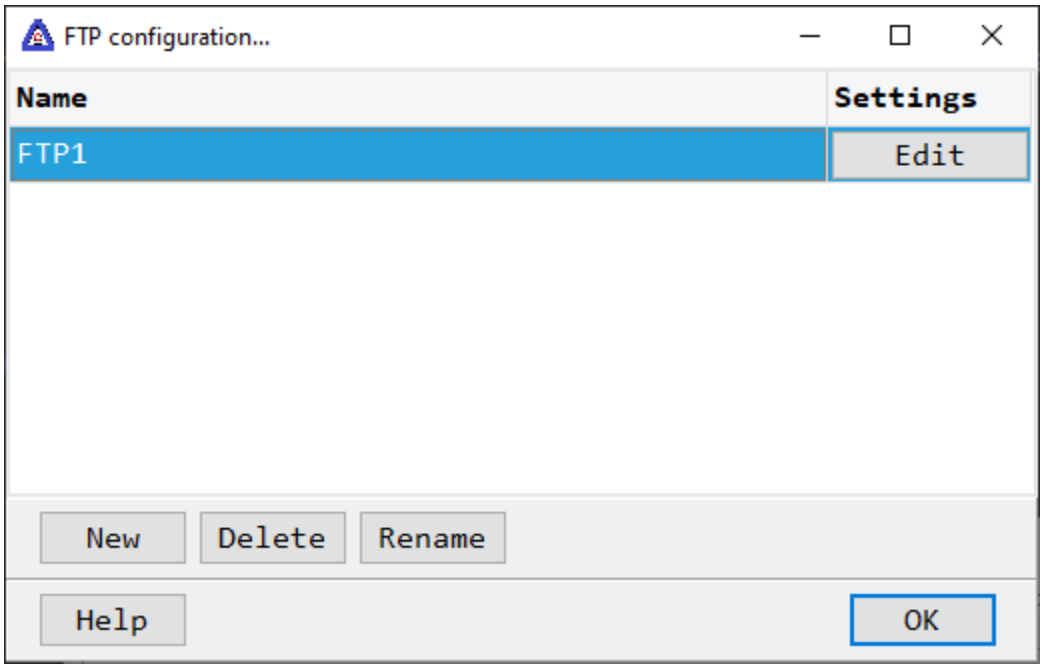
When the button is selected the program will attempt to access the data in the device using the communication parameters configured.

Error messages

No memory type selected	The memory type has not been selected.
Start out of range	The value is out of range for the type.
Count exceeds limit	The count must be 64 or less for words types; 256 or less for Boolean types.
Start + count exceeds register limit	The register plus the count exceeds the maximum address range.
Count < 1	The line must read at least on register.

FTP Clients

Each FTP client master object is listed in the window.



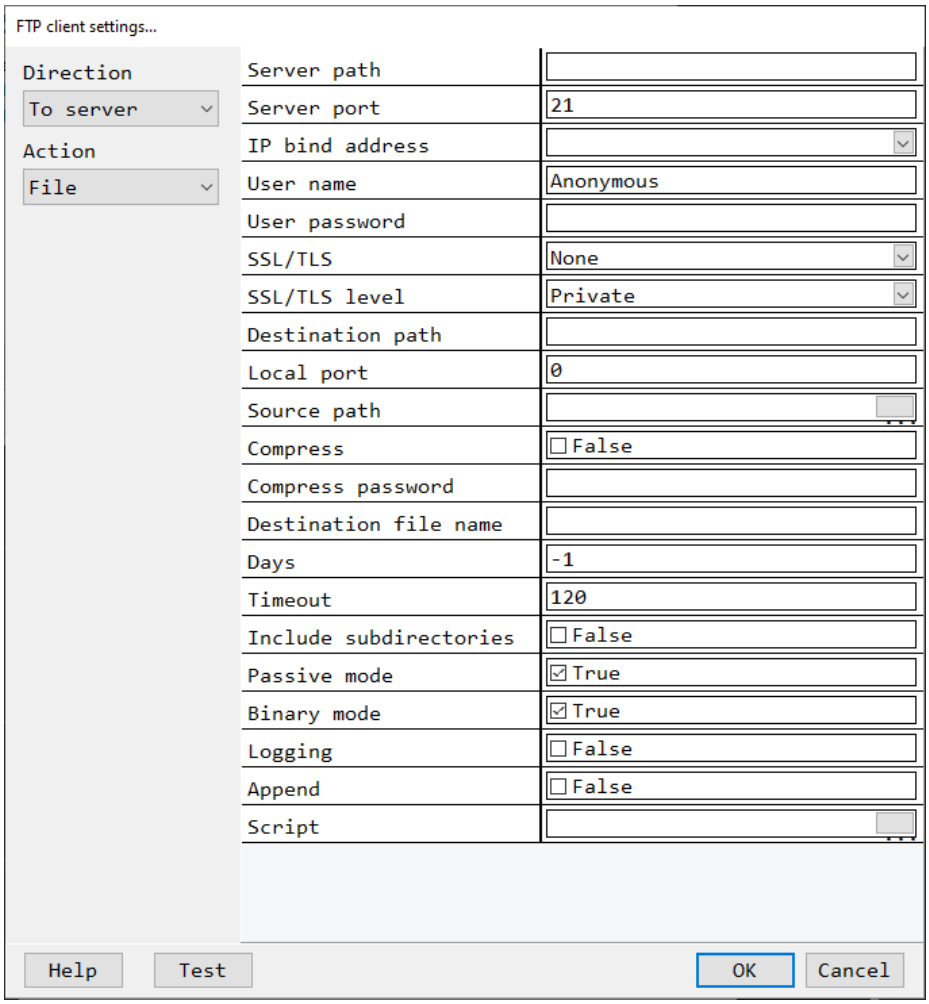
Each file transfer protocol (FTP) object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an FTP object select the "Delete" button.

Settings



The FTP object has many properties. The properties can be modified at runtime via scripting, mouse commands or a scheduled task.

Note: If the destination file exists it will be overwritten, unless the FTP host or PC OS prevents it or "Append" is enabled.

Direction

Select if the file(s) are to be transferred from the PC to a server or from a server to the PC.

Action

Some actions provide for easier use of the FTP option.
File = Copy a file or files to or from the FTP server.

The following actions are only valid when the direction is "To server".

Alarm log Copy an alarm log to the FTP server.
Event log Copy an event log to the FTP server.

Server path

This is the path to the FTP server. Examples: ftp.somesite.com, anothersite.com.

Server port

The server port number. The default FTP port number is 21.

Bind IP address See here.

User name

The name used to logon to the FTP server.

User password

The password used with the "user name" to logon to the FTP server. The password may not be used.

SSL/TLS (secure socket layer/transport layer security)

The FTP connection can be configured to use a more secure (encrypted) protocol than plain FTP or plain FTP can be used.

None (Plain FTP)
Implicit
Explicit (SSL)
Explicit (TLS)

SSL/TLS level

If encryption is used the encryption level can be “Clear” or “Private”.

Clear The control connection is encrypted and the file transfer is not encrypted. The file itself could be encrypted.

Private The control connection and the file transfer are encrypted.

Destination path

To server: This is the path on the FTP server to place the file.
From server: This is the path on the PC to place the file or files.

If the path does not exist the transfer will fail on the FTP server or the PC.

Local port

The PC port number to use for the operation. A "0" commands the program to request any free port available from the PC OS and is the recommended value.

Passive mode: If a value other than "0" is used the program will attempt to use that port number. The port number may be in use and the operation will fail.

Active mode: If a value other than "0" is used the program will attempt to use that port number and the subsequent port number. The port numbers may be in use and the operation will fail.

If two or more FTP objects using the same port number, other than "0", the objects cannot be active concurrently.

Source path

To server: This is the path on the PC of the file or files to transfer. A single path is used and wild card characters are allowed.

Use wild card characters with care.
Examples:

C:\testfile.txt	A single file.
C:*.txt	All files in the "C" path that use "txt" as a file extension.
C:\MyFiles\	All files in the "C:\MyFiles\" path.
C:*.*	This example would copy the complete "C" drive to the FTP server.

Note: If wild card characters or a directory path are used, "compress" (below) must be enabled.

From server: This is the path on the FTP server of the file to transfer from the server to the PC. Only a single file can be selected.

When the action is one of the "log" actions the "Source path" attribute is not used. The source file will be calculated in the FTP object based on the action and the "Days" attribute (below).

Compress

This is only used if the direction is "To server". The source is "Zip" compressed and transferred to the server.
Note: The 'zip' extension is not added to the destination file name.

Compress password

If the attribute to compress the file is enabled and a password is provided the file is compressed using the password.

Destination file name

To server: The file name on the server. This is the name the file will be saved as on the server.

From server: This file name on the PC. This is the name the file will be saved as on the PC.

Compress password

If the attribute to compress the file is enabled and a password is provided the file is compressed using the password.

Days

If the action is "File" this attribute is not applicable. For the "log" actions this attribute commands the FTP object to determine the log date/file name based on the "days" value. The value is the number of previous days. A value of "0" is the current day. A value of "-1" is the day before the current day. A value of "-2" is two days before the current day.

Example: A script is executed every night at 1:30 AM to transfer the previous days alarm log to the FTP server, set this value to -1.

Timeout

This is the number of seconds to allow the operation to complete. The underlying socket might timeout before the value is reached. The value must be 15 - 86400 seconds.

Include subdirectories

If a wild card character is used in the source path or the source path is to a directory, this attribute commands the operation to include ALL subdirectories in the operation. Use wild card characters with care. This attribute will include ALL files and ALL directories in the specified path. It will seek down to the lowest level of the directory tree.

Passive mode

The FTP server may support passive or active mode. Passive is the default mode. Non-passive mode will not normally work through a router without configuration in the router.

Binary mode

The FTP server may support ASCII or binary mode. Binary is the default mode.

Logging

This attribute is normally not used. If this attribute is enabled the FTP operation will log data to a file in the "Log file settings". The log is written to disk after the transfer operation is finished.

Append

This attribute is used to append the file to an existing file at the destination location. This attribute is only applicable for transmitting a file.

Script (optional)

If specified, the script to execute when the FTP session has ended and the controlling thread is released.

State

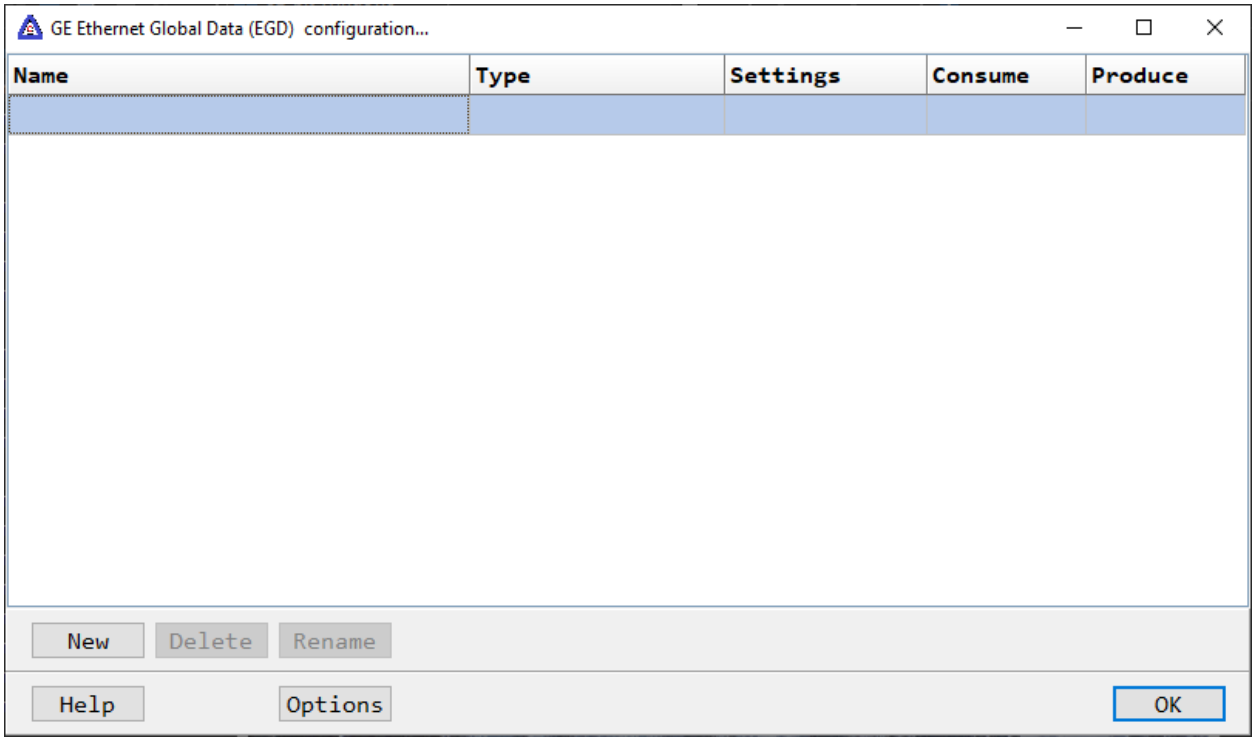
State is a special setting. It is only used at runtime to start or cancel a transfer. "True" starts the transfer and "False" cancels the transfer. It can also be used to get the current state of the port, active or inactive.

Test button

When the test button is selected the configuration settings will be applied and an attempt to transfer the files will occur.

GE EGD (Ethernet Global Data)

Each EGD object is listed in the window.



To create a new object, select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an EGD object select the "Delete" button.

Note:
The EGD protocol uses a produce/consume communication protocol. Data is transmitted in blocks of bytes. The contents of a block are configured at the producer end and sent to consumers, via UDP, at a frequency set in the producer.

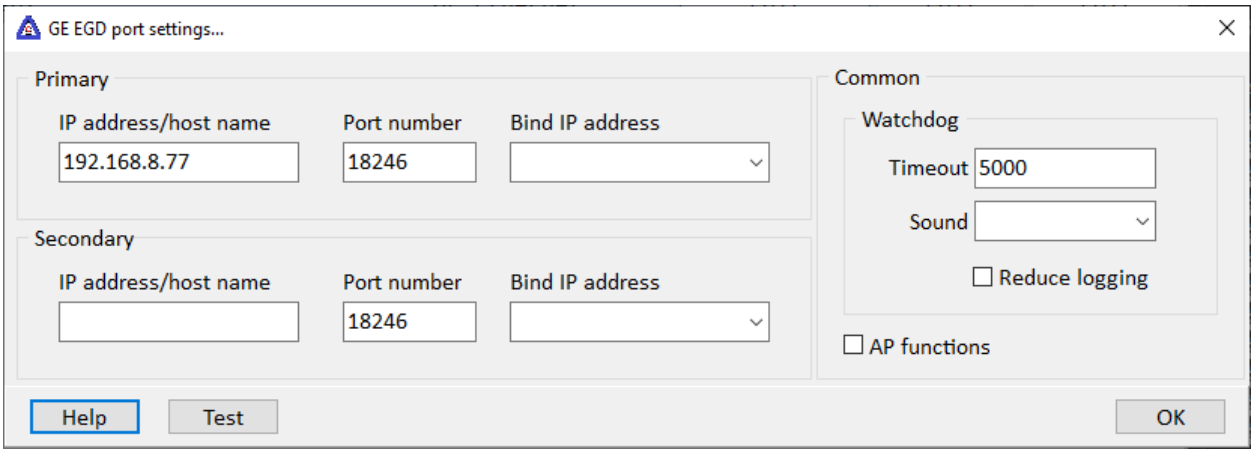
The HMI consumes messages and parses the data into “points” and/or “script globals.” See below for more information.

The HMI produces messages from a configured list of sources. See below for more information.

Options

The options button sets the colors for the “source” column in the “Produce settings” configuration window.

Settings



The UDP port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

Secondary

The secondary port provides for a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the secondary is properly configured, the HMI will open the port and listen for messages.

If the primary port watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. When communication is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Bind IP address

This field specifics the IP address of an interface device on the computer, used to “bind” the communications path. Leave the field blank to allow the OS to select the device. **Note:** For EGD this needs consideration. The EGD protocol includes the IP address in the header of all messages. If the sender IP address does not match the IP address in the header the message is ignored. If the “Bind IP address” is not specified the OS will select the device. The address might not be correct. Configuring the “Bind IP address” field might prevent issues.

Miscellaneous

Watchdog timeout

The timer begins timing when the port opens the communication socket. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

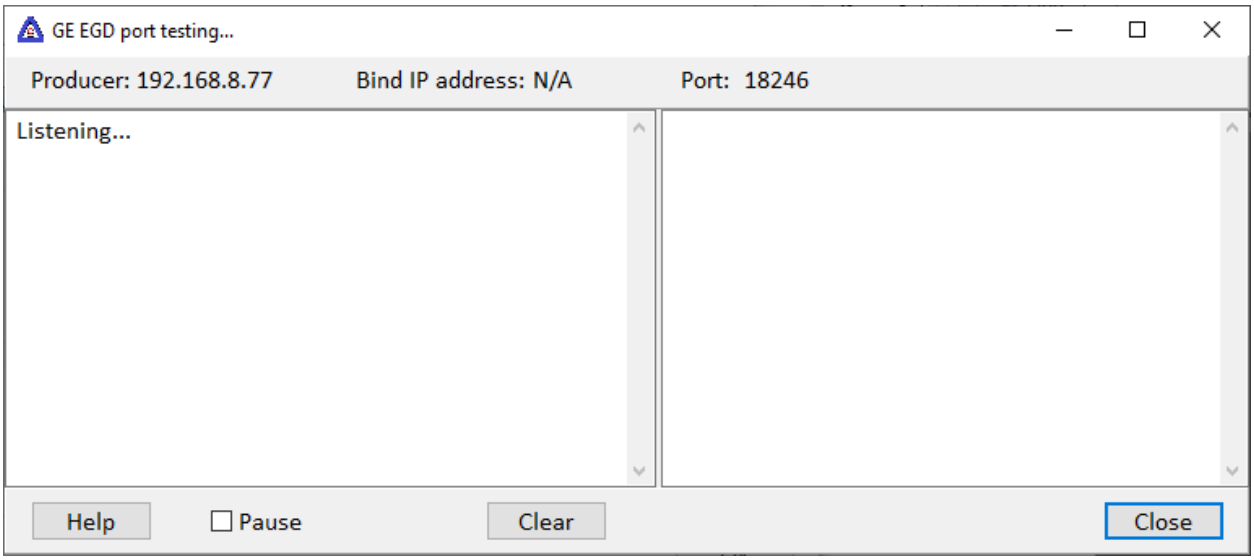
Reduced watchdog logging

When enabled and a watchdog timeout occurs, a single entry will be placed in the event log and the sound will be queued (if configured). When the external device responds, after a watchdog timeout, a single entry will be placed in the event log. The watchdog counter will continue to count each time the watchdog timer expires.

AP functions

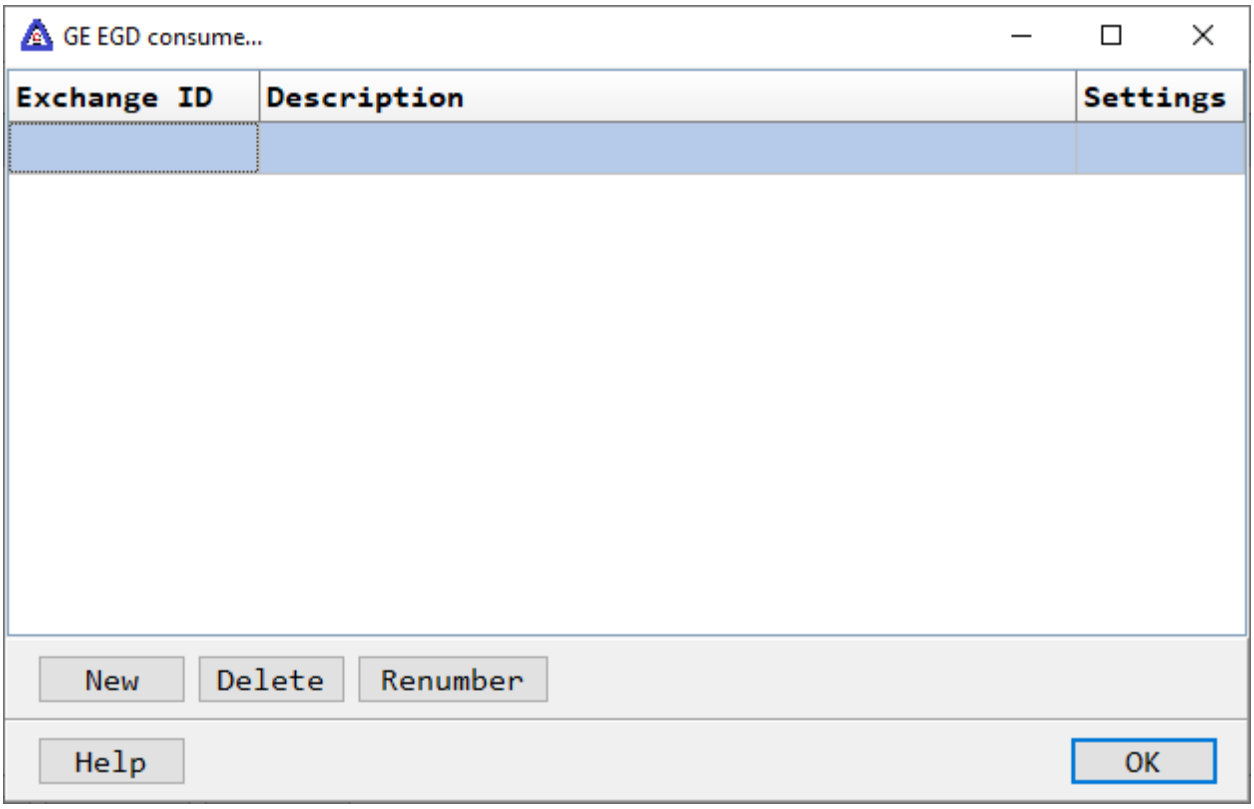
See analog functions.

Test button



When the test button is selected the program will attempt to open a UDP port and listen for messages. If a message is received, the header is parsed and displayed in the right panel.

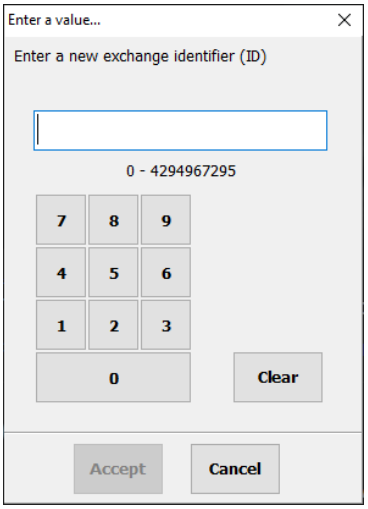
Consume



The external device will “produce” a message and the HMI can “consume” the message.

Exchange ID

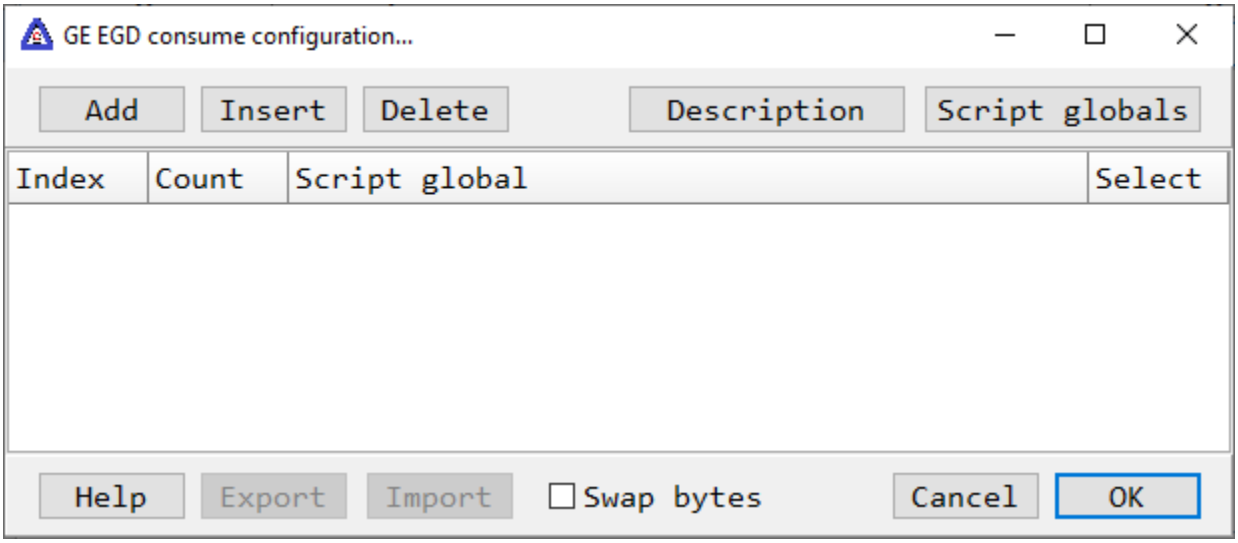
The “producer” assigns the exchange ID. Select the “New” button and enter the exchange ID to create a new “consume” message configuration. Use the “Renumber” button to change the ID of an existing consume message configuration.



Description (optional)

This is a text field to be used as needed.

Settings



Note: If the consume message does not contain any strings, strings are stored in script globals, no configuration data is required.

Add Select a script global to receive the string. After the script global is selected a prompt for the byte count will be displayed. If later, a count needs to be adjusted, double click on the "count" cell for the script global row.

Insert Insert a script global in the list

Delete Remove a script global from the list.

Select Can be used to select another script global for the current item in the list.

The "Script globals" button provides access to editing the script global configuration.

The index is automatically calculated from list item "counts".

Swap bytes String data arrives as a "word" (2 bytes) field. The byte order might need adjusting. For example, AB might arrive ordered as AB or BA. Use this property to "swap" (reverse) the byte order of the word.

Points

Points have a “Source” field and a “Source” type field. (Points configuration)

Source (analog)

This field, for analog points, is specified as <exchange ID>:<index>.
Example 123:7
The index is the byte offset from the start of the message data section. 0 (zero) is the first index. The highest index, for a 16-bit type, is 1398.

The analog “types” supported: * default

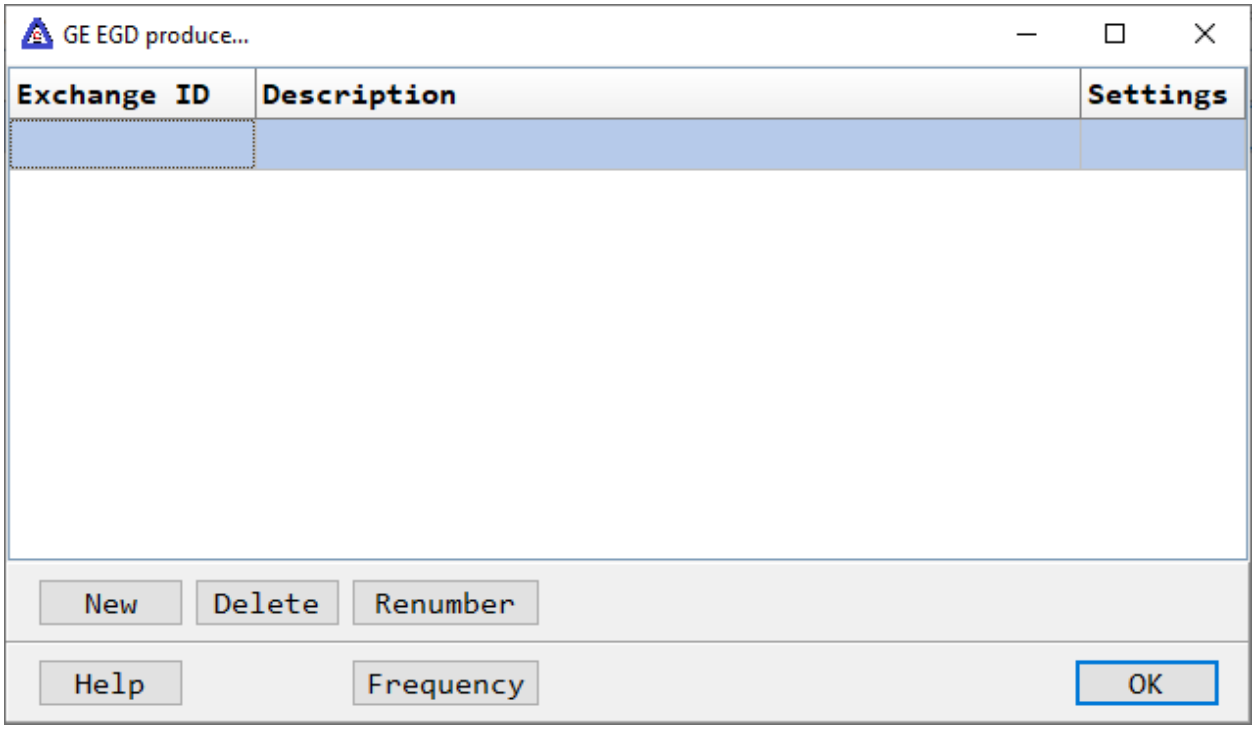
Type	Description	Range
Integer*	32-bit	-2,147,483,648 .. 2,147,483,647
Float	32-bit	IEEE-754 standard format
Small integer	16-bit	-32,768 .. 32,767
Unsigned integer	32-bit	0 .. 4,294,967,295

Source (Digital)

This field, for digital points, is specified as <exchange ID>:<index>.<bit number>
Bits are numbered 0-7.

Example: 123:7.3

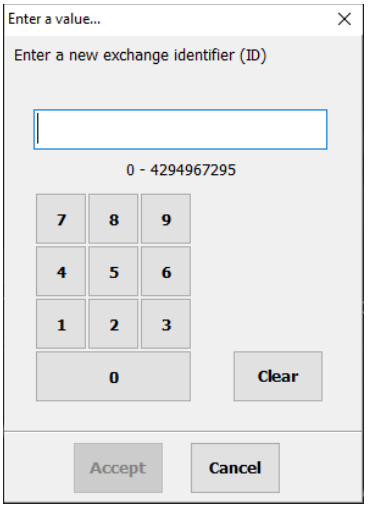
Produce



The external device will “consume” a message “produced” by the HMI.

Exchange ID

The “producer” (HMI) assigns the exchange ID. Select the “New” button and enter the exchange ID to create a new “produce” message configuration. Use the “Renumber” button to change the ID of an existing “produce” message configuration. **Note:** The exchange ID must be unique to produced messages for this connection in the HMI. A consume ID (created by the external device) and a produce ID (created by the HMI) can be the same value.



Frequency

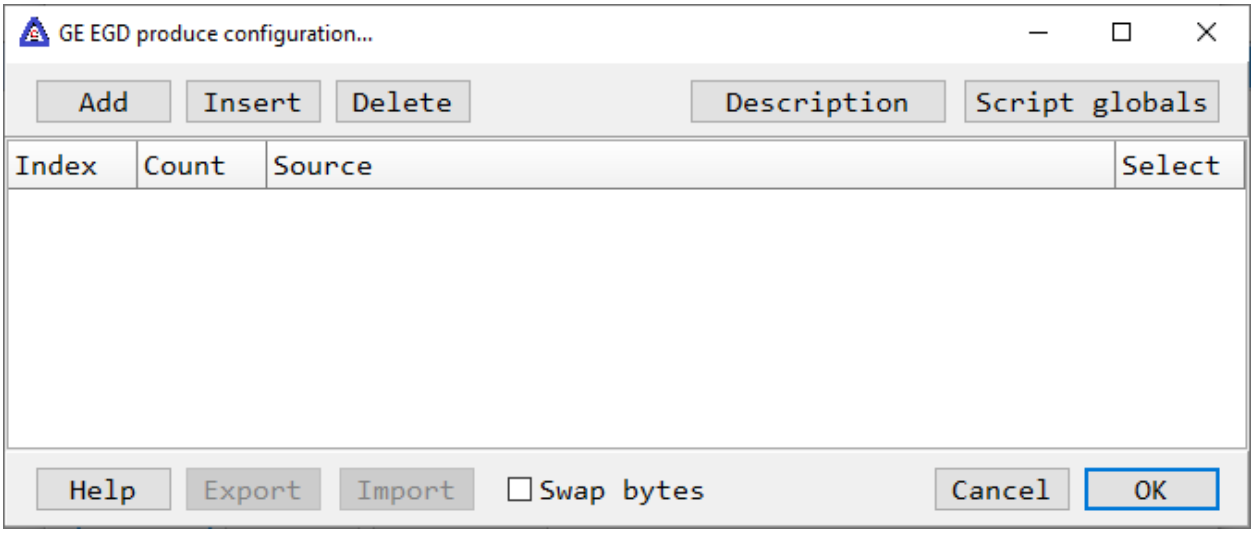
This value, in milliseconds, is how frequently to send the list of produced exchanges. The complete list is sent, in order and after the last exchange is sent the timer is restarted.

Note: If the exchanges are all commands, meaning data that only needs to be sent when a command (e.g., stop pump, set a set point, etc.) is issued, the frequency can be set high (lowering network load) and the "SetPortReadEnable" can be called to send the exchange.

Example:
Set the point(s) or script global(s) to the required value(s).
Call SetPortReadEnable(<port name>, <exchange ID>, true);

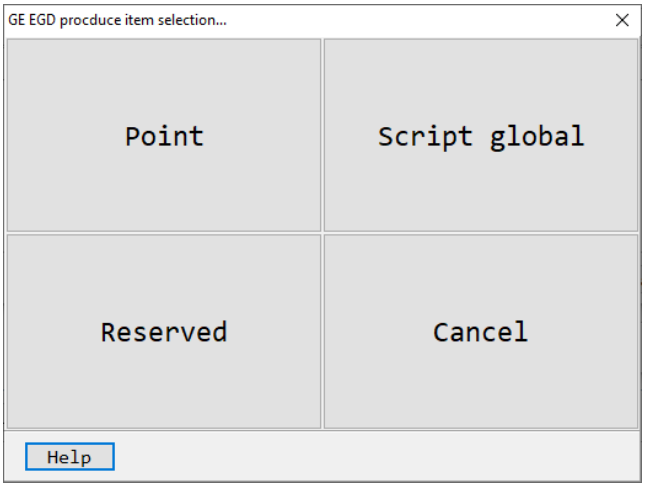
The exchange ID specified will immediately be transmitted outside the normal frequency/timer logic.

Settings



Note: The index and count fields are automatically calculated from the point item (defines size), script global count and reserved counts.

Add Select the add button to add a script global or point to the list or reserve space in the message data block.



Point Select the button and the point selection dialog will appear. Select a point and item.

Script global Select the button and the script global selection dialog will appear. Select a script global and then a count.

Reserved Select the number of bytes to reserve

Insert Insert a new item in the list
Delete Remove an item from the list.

The “Script globals” button provides access to editing the script global configuration.

Swap bytes String data is transmitted as a “word” (2 bytes) field. The byte order might need adjusting. For example, AB might need to be ordered as AB or BA. Use this property to “swap” (reverse) the byte order of the word.

General purpose - Serial

Each general purpose serial object controls one serial port and is listed in the window. This port type can use a self-contained script to control the serial port operation or an external script. Each port can have its own script engine and executes a single script

The design of this port is “event driven” or “polled”.

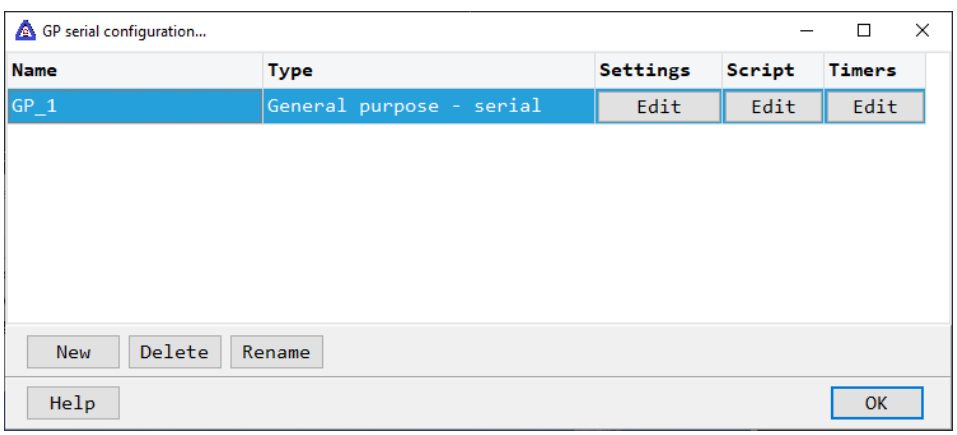
Event driven

There is no main loop and the procedures in the script are executed based on an “event” occurring. Not all event types are required and may not be required for successful control of the serial port. The events/procedures are listed below. Any other required procedures can be created and used in the script. **Note:** The amount of time each procedure uses, to execute its code, should be kept to a minimum. A time consuming procedure can disrupt serial port control and possibly disrupt overall HMI operations. The script is executed by the port script engine.

Polled

The serial port is controlled via scripting external to the port using the “GPPCommand” script command.

Note: Do not mix “event driven” and “polled” methods. “Polled” might be better to develop and prove the required code and switch to “event” when all the functions are proofed.

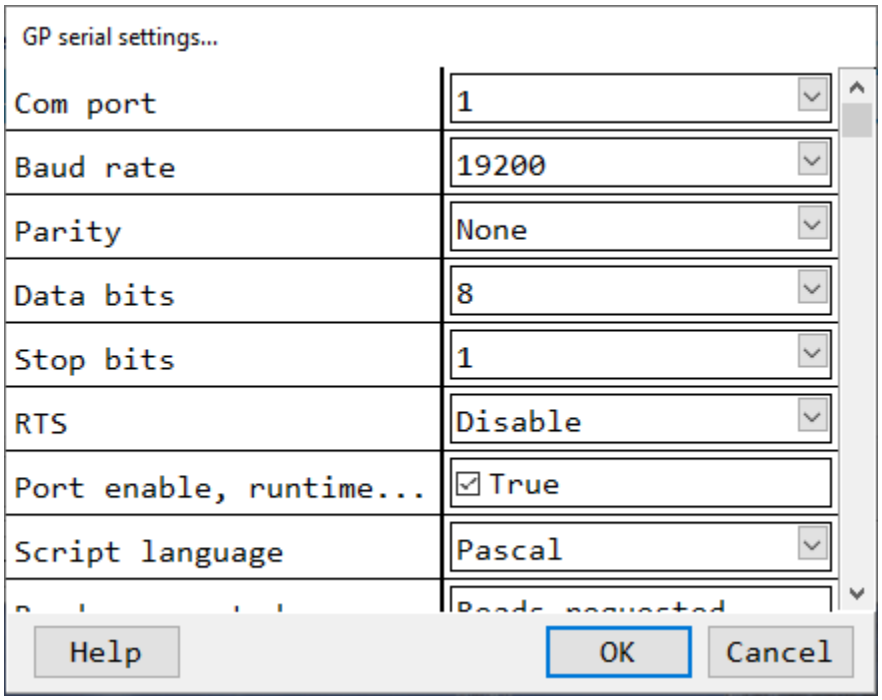


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a general purpose serial port object select the "Delete" button.

Settings



Select the serial port attributes as required. The serial port settings are applied when runtime monitoring starts.

RTS

See here.

Port enable, runtime start

When this attribute is true and runtime monitoring starts, the serial port will be opened using the port settings configured.
Note: This attribute is applied after the "Initialize" event (if present) is called and returns.

Reads requested, Reads completed, Writes requested, Writes completed, Watchdog timeouts

Each port has a diagnostic window. These integer values are provided to display to the user, some data. The name of each integer value can be changed to define what the value represents. The values are under the control of a script. If the ClearPortCounters is issued for the port, the values will be reset to zero (0).

Watchdog timed out

Each port has a diagnostic window. This boolean value is provided to display to the user, some data. The name of the boolean value can be changed to define what the value

represents. The value is under the control of a script. If the ClearPortCounters is issued for the port, the value will be reset to zero (false).

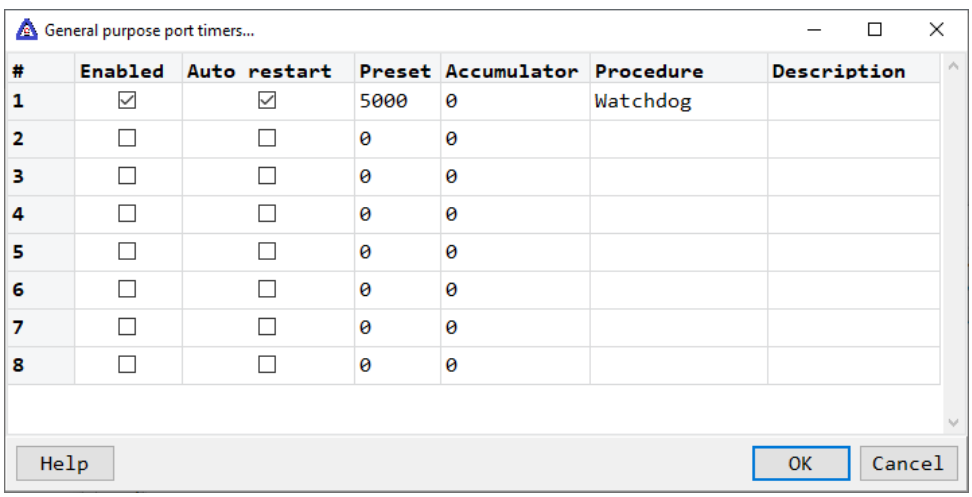
Error

Each port has a diagnostic window. This string value is provided to display to the user, some data. The name of the string value can be changed to define what the value represents. The value is under the control of a script. If the ClearPortCounters is issued for the port, the value will be reset to empty (no value).

Diagnostics enabled

If this attribute is enabled the HMI will record the incoming and outgoing serial data and the data can be viewed at runtime.
Note: If this data is not required, disable this attribute. If this attribute is enabled and not required, it is a waste of resources, processing time, etc.

Timers



#	Enabled	Auto restart	Preset	Accumulator	Procedure	Description
1	☑	☑	5000	0	Watchdog	
2	☐	☐	0	0		
3	☐	☐	0	0		
4	☐	☐	0	0		
5	☐	☐	0	0		
6	☐	☐	0	0		
7	☐	☐	0	0		
8	☐	☐	0	0		

These timers can be used to execute a script procedure after a set time has elapsed or the timer fields can be monitored in the port script and some action taken based on the timer state.

The time base is milliseconds. The timers are updated every 100 milliseconds. Use increments of 100 milliseconds. **Note:** When a procedure is executed from a timer completion, the other timers are still timing but, execution of a configured procedure is paused until the first procedure is completed. Depending on the timer order, the timer might not time out until the next timer update. Using one timer and a logic state machine can produce the best results.

The preset and accumulator limit:
2,147,483,647 milliseconds = 2,147,483 seconds = 35,791 minutes = 596 hours = 24 days = 0.068 years

Timer Fields

Name	Type	Description
TT	Boolean	Timer enabled and not done. (Timer timing)
DN	Boolean	Timer enabled and accumulator >= preset. (Timer done)
EN	Boolean	Timer enabled
ACC	Integer	Accumulator
PRE	Integer	Preset
ARS	Boolean	Timer Auto Restart

Auto restart

When enabled, when the timer is timed out (ACC is greater than/equal to the PRE) the timer begins timing again. The done field is never set true.

Procedure

A procedure to execute when the timer times out. (optional)

Description

A user field to store information about the timer. (optional)

Timer operation

When the enabled field (EN) becomes true:

1. The timer begins timing
2. The timer timing (TT) field is set true.
3. The accumulator (ACC) begins incrementing.

When the ACC is greater than/equal to the preset (PRE):

1. The DN field is set true.
2. The TT is set false.
3. The ACC stop incrementing.

When the enabled field (EN) becomes false:

1. The ACC is set to zero.
2. The DN is set false.
2. The TT is set false.

General purpose scripting

Scripting for port control is much like general scripting in the HMI. See the paragraph in the port section for more information. Serial or TCP Client

The main difference is the port script is designed to be self-contained and not call or access any other sections of the HMI. It is possible to access other sections of the HMI but, not recommend without taking extreme care. If access to other sections of the HMI is required, it is much better to set a flag and have an external script monitor the flag and execute the code in the external script.

Caution should be exercised when creating script code that is not specific to port operations. For example, executing a script function that displays a modal window will halt the script which halts port reading/writing.

General purpose – serial properties

All properties of the serial port are accessed via the "p." notation.

Active (read/write)

This property indicates the opened/closed state of the serial port and is used to open/close the serial port in a script.

```
if p.Active then
 begin
  //the port is open
end;
```

Examples:

```
p.Active:=true;      //open the port
p.Active:=false;     //close the port

procedure Initialize;
begin
 p.Active:=true;      //open the port
end;
```

Error (read/write)

This property is described above.

Examples:

```
p.Error:='wrong byte code';
```

Reads requested, Reads completed, Writes requested, Writes completed, Watchdog timeouts (read/write)

This property is described above.

Examples:

```
p.ReadRequest:=33;
p.ReadCompleted:=33;
p.WriteRequest:=1;
p.WritesCompleted:=1;
p.WatchDogTimeouts:=0;
```

Watchdog (read/write)

This property is described above.

Examples:

```
p.Watchdog:=false;

if p.Watchdog then
 ;
```

General purpose – serial events/callbacks/procedures

The events/callbacks/procedures listed may or may not be required.

Initialize

This procedure is called when runtime monitoring is started.
If "Port enable, runtime start" is enabled, this procedure is called before the attribute is applied (the port opened).

```
procedure Initialize;
```

LogString

This procedure is used to log a string value to the runtime diagnostics window.

```
p.LogString('Bad byte value');
```

OnOpen

This procedure is called when the serial port was successfully opened.

```
procedure OnOpen;
```

OnTxEmpty

This procedure is called when the last character in the output buffer was sent.

```
procedure OnTxEmpty;
```

OnRx80Full

This procedure is called when the serial port receive buffer is 80% full.

```
procedure OnRx80Full;
```

OnCts

This procedure is called when the CTS (clear-to-send) signal changed state.

```
procedure OnCts;
```

OnClose

This procedure is called when the serial port was successfully closed.

Note: This event is not called when runtime monitoring is stopped and the port is closed.

```
procedure OnClose;
```

OnRxChar

This procedure is called when the serial port receives a byte/character. The procedure can be called with more than one byte/character in the buffer.

```
procedure OnRxChar(buffer; count: integer);
```

"buffer" is a pointer to an array of integer. "count" is the number of bytes/characters in the buffer. The buffer index begins at 0 and ends at count – 1.
E.g. count = 4, buffer[0].. buffer[3] will contain the data bytes/characters.

Note: This event is required if receiving data from the serial port is needed and the callback method is used.

Send

"Send" is used to transmit an array of bytes out the port. "Value" is number of bytes transmitted. If the value is zero (0) an error has occurred.

```
value:=p.Send(bufferArray,8);     //send a buffer
```

TimerGet

TimerGet is the same function as TimerGet. The difference is the timer source.
Port timers are accessed with the “p.” notation. “Value” is true if no error was detected reading the timer.

```
value:=p.TimerGet(2,'TT');     //is the timer timing
```

Note: Verify the “p.” notation is used. If “p.” is not used a script global timer is accessed.

TimerSet

TimerSet is the same function as TimerSet. The difference is the timer source.
Port timers are accessed with the “p.” notation. “Value” is true if no error was detected writing the timer.

```
value:=p.TimerSet(2,'EN',True);     //enable the timer
```

Note: Verify the “p.” notation is used. If “p.” is not used a script global timer is accessed.

StringGet,StringSet

These procedure parameters are the same as the global script commands StringGet and StringSet. If a port script exists, these function calls are passed to the script as defined. If a port script does not exists StringGet returns an empty string and StringSet returns false.

General purpose – TCP Client

Each general purpose TCP Client object controls one TCP Client port and is listed in the window. This port type can use a self-contained script to control the port operation or an external script. Each port can have its own script engine and executes a single script

The design of this port is “event driven” or “polled”.

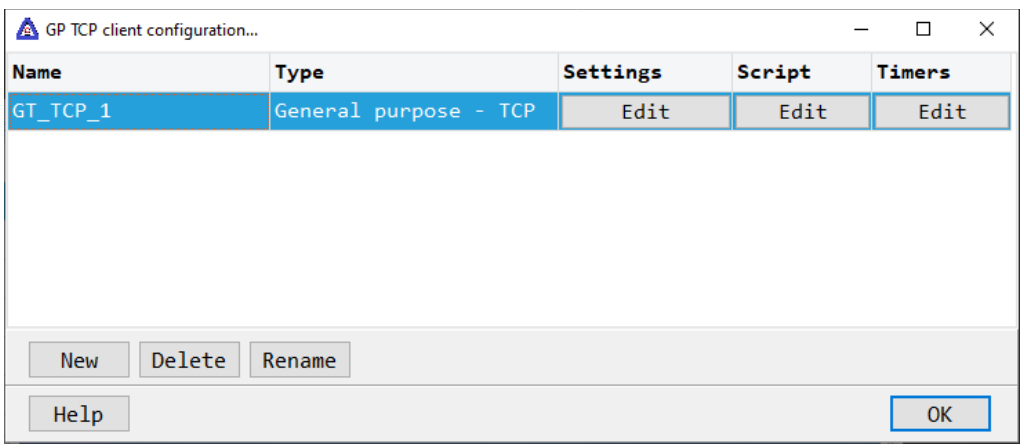
Event driven

There is no main loop and the procedures in the script are executed based on an “event” occurring. Not all event types are required and may not be required for successful control of the port. The events/procedures are listed below. Any other required procedures can be created and used in the script. **Note:** The amount of time each procedure uses, to execute its code, should be kept to a minimum. A time consuming procedure can disrupt port control and possibly disrupt overall HMI operations. The script is executed by the port script engine.

Polled

The port is controlled via scripting external to the port using the “GPPCommand” script command.

Note: Do not mix “event driven” and “polled” methods. Polled” might be better to develop and prove the required code and switch to “event” when all the functions are proofed.

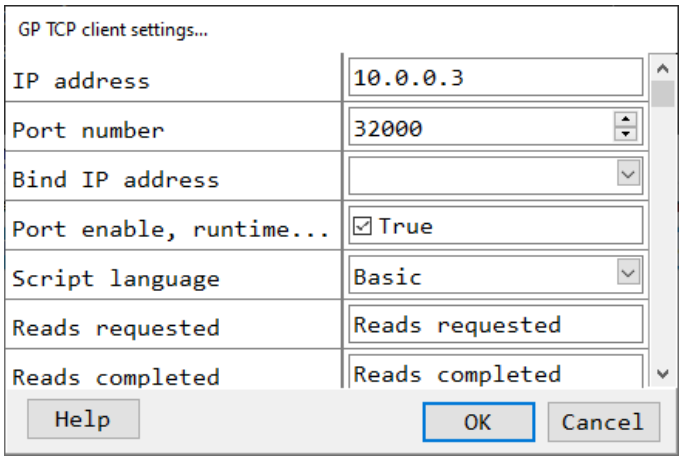


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a general purpose TCP Client port object select the "Delete" button.

Settings



The port settings are applied when runtime monitoring starts.

IP address

The IP address of the server.

Port number

The port number of the server.

Bind IP address See here.

Port enable, runtime start

When this attribute is true and runtime monitoring starts, a connection request will be issued.
Note: This attribute is applied after the "Initialize" event (if present) is called and returns.

Script language

Pascal or Basic script language.

Reads requested, Reads completed, Writes requested, Writes completed, Watchdog timeouts

Each port has a diagnostic window. These integer values are provided to display to the user, some data. The name of each integer value can be changed to define what the value represents. The values are under the control of a script. If the ClearPortCounters is issued for the port, the values will be reset to zero (0).

Watchdog timed out

Each port has a diagnostic window. This boolean value is provided to display to the user, some data. The name of the boolean value can be changed to define what the value represents. The value is under the control of a script. If the ClearPortCounters is issued for the port, the value will be reset to zero (false).

Error

Each port has a diagnostic window. This string value is provided to display to the user, some data. The name of the string value can be changed to define what the value represents. The value is under the control of a script. If the ClearPortCounters is issued for the port, the value will be reset to empty (no value).

Diagnostics enabled

If this attribute is enabled the HMI will record the incoming and outgoing data and the data can be viewed at runtime.
Note: If this data is not required, disable this attribute. If this attribute is enabled and not required, it is a waste of resources, processing time, etc.

Timers

See here.

General purpose scripting

Scripting for port control is much like general scripting in the HMI. See the paragraph in the port section for more information. TCP Client

Most of the events, names, functions, etc. are the same as the serial port and are covered here. Any differences are listed below.

OnOpen

This procedure is called when a TCP connection is established.

```
procedure OnOpen;
```

OnClose

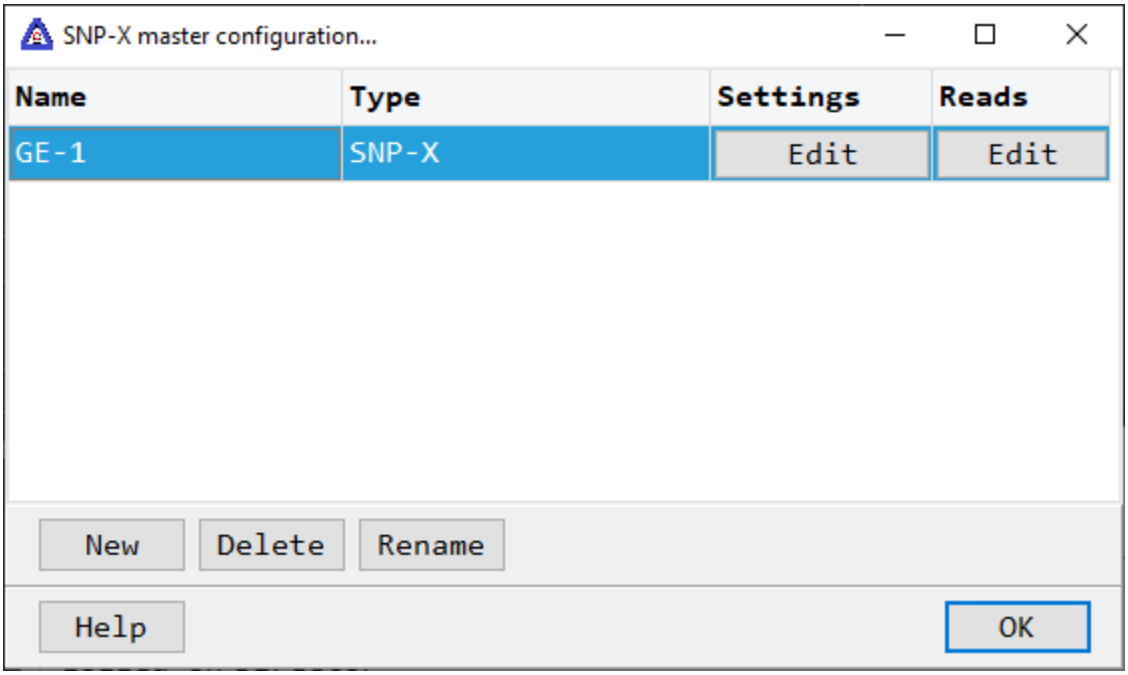
This procedure is called when the TCP connection state changes to “closed”.

Note: This event is not called when runtime monitoring is stopped and the port is closed.

```
procedure OnClose;
```

GE SNP-X

Each SNP-X master object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

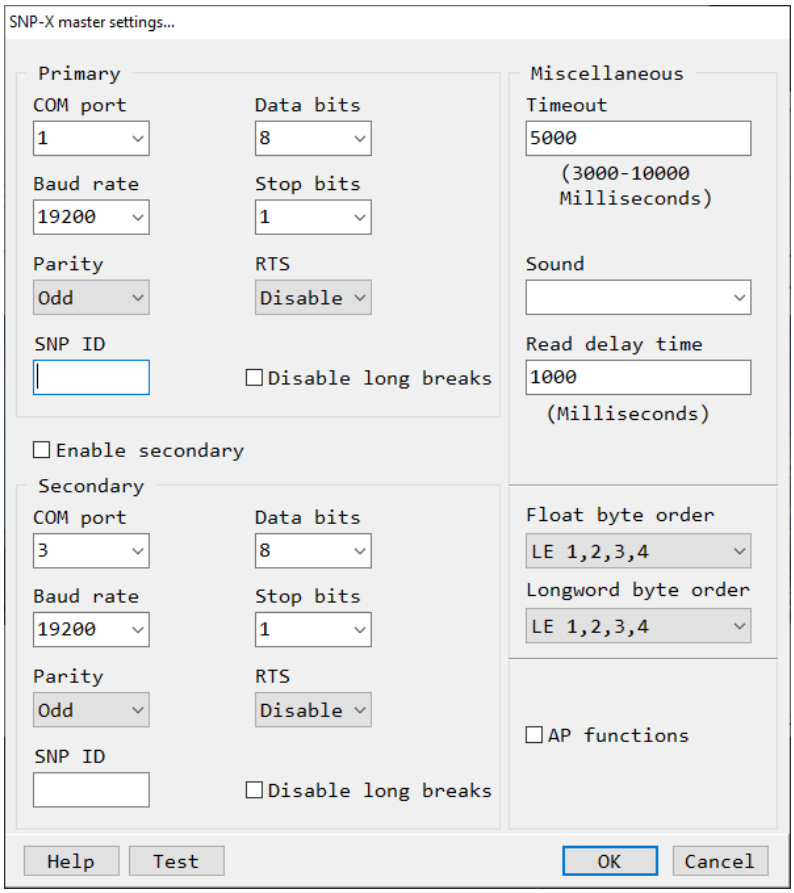
To delete a SNP-X master object select the "Delete" button.

SNP-X Commands

Name	**Command**	**Request Code**
X-Attach	88(58h)	00
X-Read	88(58h)	01
X-Write	88(58h)	02
X-Write	Buffer 84(54h)	N/A

If the amount of data to write is two bytes or less the X-Write command is used. Otherwise the X-Write and X-Write Buffer command is used.

Settings



The serial port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

RTS

See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.
If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if

configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

SNP ID

The maximum length is seven bytes. This may be limited by the slave device. The values of the 7 bytes are further restricted to the ASCII characters `0' through `9' inclusive, and `A' through `F' inclusive (must be capital letters). Null (blank SNP ID) can be used for point to point serial connections.

Disable long breaks

The protocol requires the serial transmit line to be put in a "space" state for 3 character times before the attach command is issued to the slave. Some devices do not require the "long break" to accept and respond to the attach command. (Under Windows the length of time for 3 characters, at the current configured baud rate, is approximate.)

Miscellaneous

Timeout

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:
The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Float byte order, longword byte order

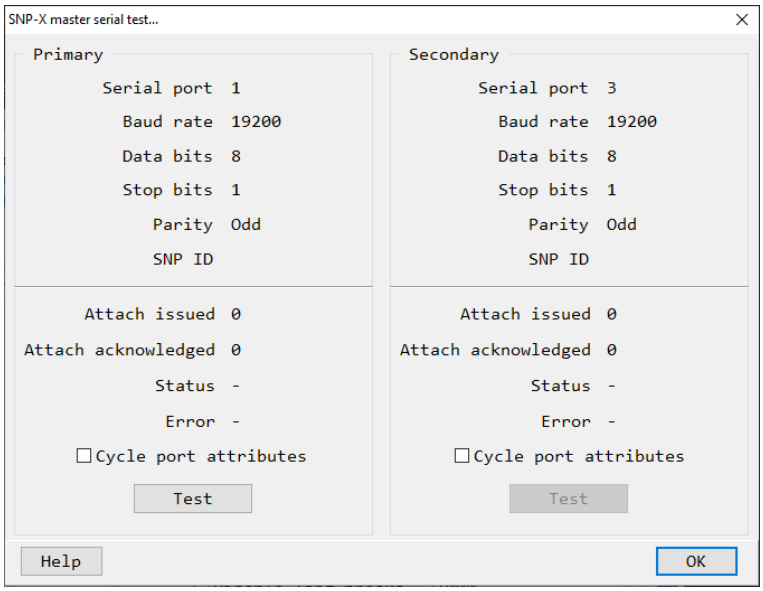
Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The register data received from the slave device is in LO-HI byte order. Regular integer data and floating data values can be in the same data read.

The 4 byte float/longword option is configured on a point level. The integer data is converted into a floating point value at the point level.

When using floating point values the byte order of the slave device might be different than the regular register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

Test button



When the test button is selected the program will attempt to "attach" to the slave device.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads

SNP-X reads...

#	Memory type	Start element	Count	Enabled	Testing
1	Registers	1	1	☑	Test
2	None			☐	Test
3	None			☐	Test
4	None			☐	Test
5	None			☐	Test
6	None			☐	Test
7	None			☐	Test
8	None			☐	Test
9	None			☐	Test
10	None			☐	Test
11	None			☐	Test
12	None			☐	Test

Help OK Cancel

The address ranges shown may or may not be present in the slave device.

The HMI uses the GE defined addressing for access to the data in the slave. Please refer to the GE documentation. Exception: The "%" is not required.

Example address:

I1, AQ6789, R45909
AI421/1, R876/16

Notes:
1. Do not use leading zeros in the address string.
2. When modifying a bit in a non-bit register, AI , AQ , R, all other bits in the register are cleared.

Memory Type

The supported GE defined memory areas.

Register Type	Prefix
Registers	R
Analog Inputs	AI
Analog Outputs	Q
Discrete Inputs	I
Discrete Outputs	Q
Discrete Temporaries	T
Discrete Internals	M
SA Discretes	SA
SB Discretes	SB
SC Discretes	SC
S Discretes	S
Genius Global Data	G

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values.

Count

The number of registers to read for the request. The memory type and slave device determine the maximum number of registers that can be read in a single request. The protocol response has a limit of 1000 bytes of data.

Registers, Analog Inputs and Analog Outputs have a maximum of 500 registers per request. (2 bytes per register)

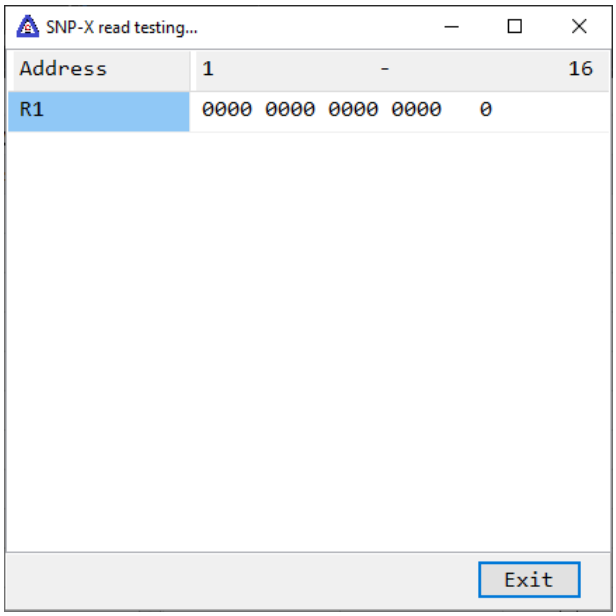
Discrete Inputs, Discrete Outputs, Discrete Temporaries, Discrete Internals, SA Discretes, SB Discretes, SC Discretes, S Discretes and Genius Global Data have a limit of 8000 registers per request. (1 bit per register)

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

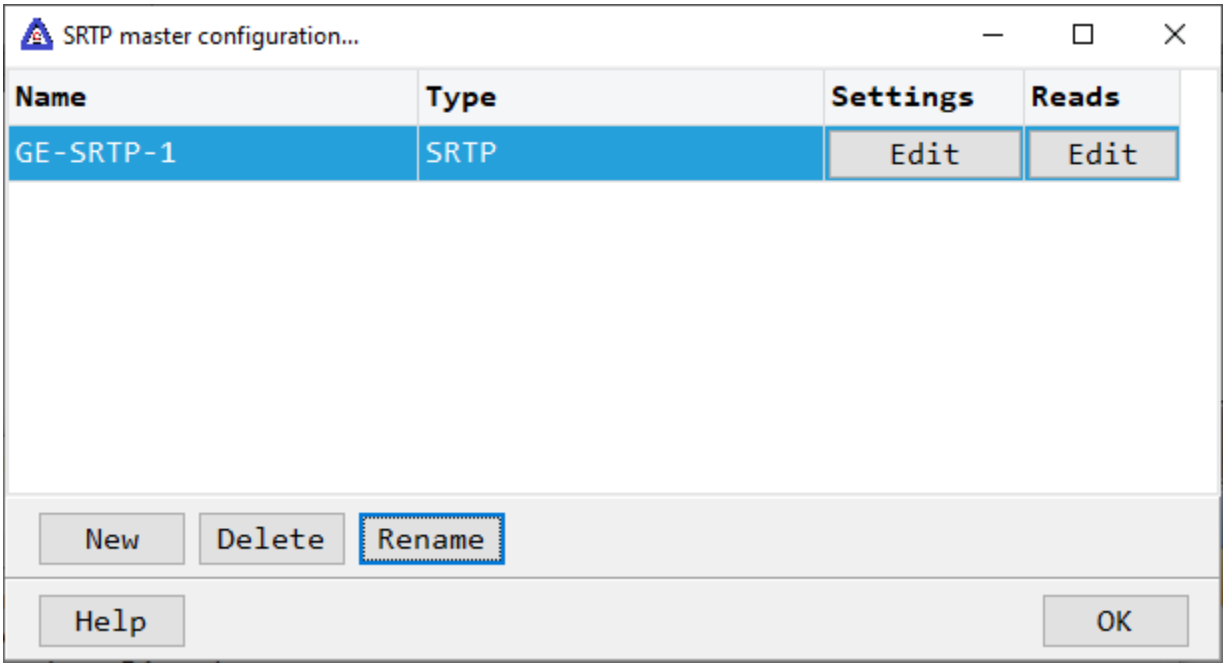
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. If the error string contains "Error : 0F:04" means "Invalid slave memory address or range in X-Request message". For other "codes" refer to the SNP-X specification. (GE document GFK-0582C)

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 1 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.
Bit data must be byte aligned	The start register for bit data must be on a byte boundary. (1, 9, 17, ...)

GE SRTP

Each SRTP object is listed in the window.

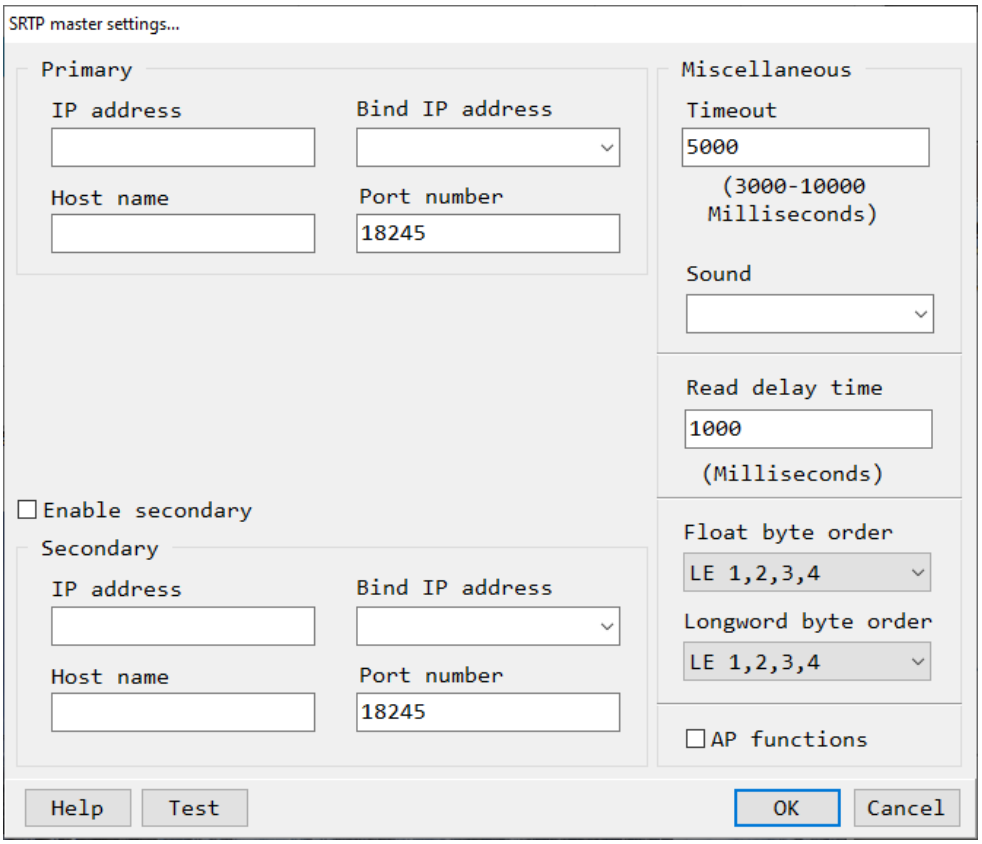


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a SRTP master object select the "Delete" button.

Settings



The TCP port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not

replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Miscellaneous

Timeout

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Float Byte Order, Longword Byte Order

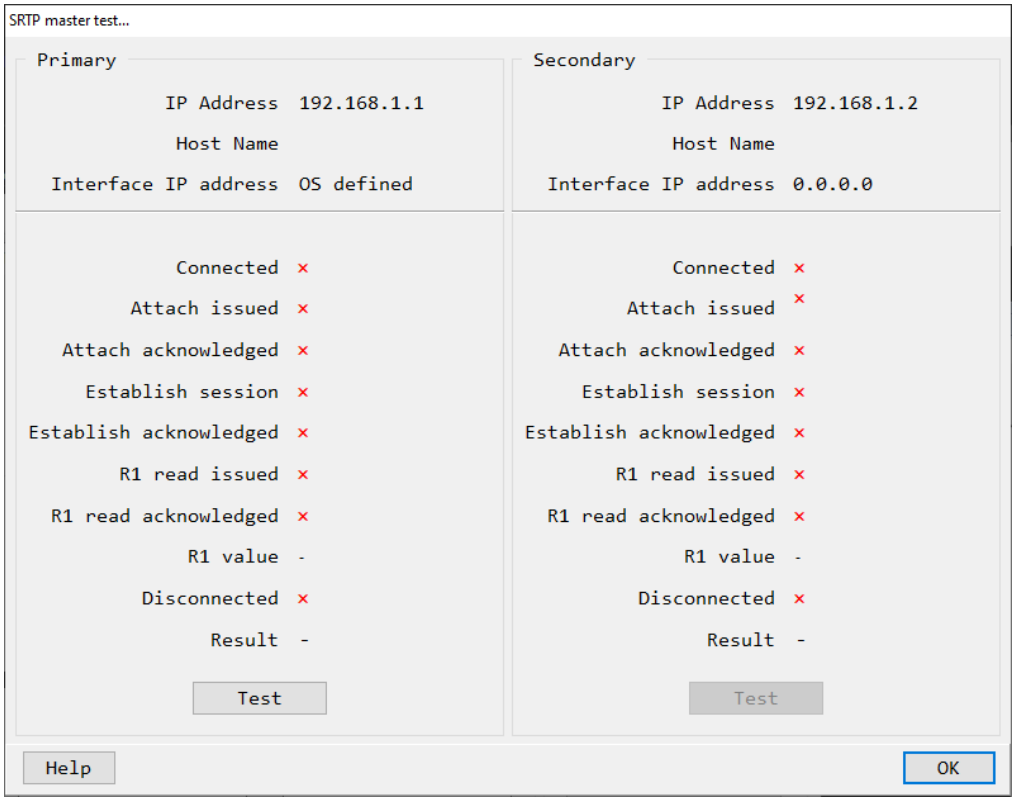
Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The register data received from the slave device is in LO-HI byte order. Regular integer data and floating data values can be in the same data read.

The 4 byte float/longword option is configured on a point level. The integer data is converted into a floating point value at the point level.

When using floating point values the byte order of the slave device might be different than the regular register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

Test button



When the test button is selected the program will attempt to "attach", establish a connection and read the contents of register 1.

The program will attempt to use the communication parameters configured.

Reads

SRTP reads...

#	Memory type	Start element	Count	Enabled	Testing
1	Discrete inputs	1	12	☑	Test
2	None			☐	Test
3	None			☐	Test
4	None			☐	Test
5	None			☐	Test
6	None			☐	Test
7	None			☐	Test
8	None			☐	Test
9	None			☐	Test
10	None			☐	Test
11	None			☐	Test
12	None			☐	Test

Help OK Cancel

The address ranges shown may or may not be present in the slave device.

The HMI uses the GE defined addressing for access to the data in the slave. Please refer to the GE documentation. Exception: The "%" is not required.

Example address:

I1, AQ6789, R45909
AI421/1, R876/16

Notes:
1. Do not use leading zeros in the address string.
2. When modifying a bit in a non-bit register, AI , AQ , R, all other bits in the register are cleared.

Memory Type

The supported GE defined memory areas.

Register Type	**Prefix**
Registers	R
Analog Inputs	AI
Analog Outputs	Q
Discrete Inputs	I
Discrete Outputs	Q
Discrete Temporaries	T

Discrete Internals	M
SA Discretes	SA
SB Discretes	SB
SC Discretes	SC
S Discretes	S
Genius Global Data	G

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values.

Count

The number of registers to read for the request. The memory type and slave device determine the maximum number of registers that can be read in a single request. The protocol response has a limit of 1000 bytes of data.

Registers, Analog Inputs and Analog Outputs have a maximum of 500 registers per request. (2 bytes per register)

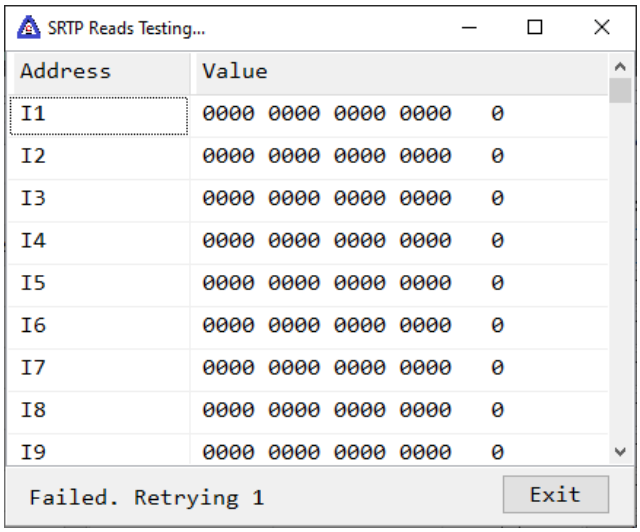
Discrete Inputs, Discrete Outputs, Discrete Temporaries, Discrete Internals, SA Discretes, SB Discretes, SC Discretes, S Discretes and Genius Global Data have a limit of 8000 registers per request. (1 bit per register)

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

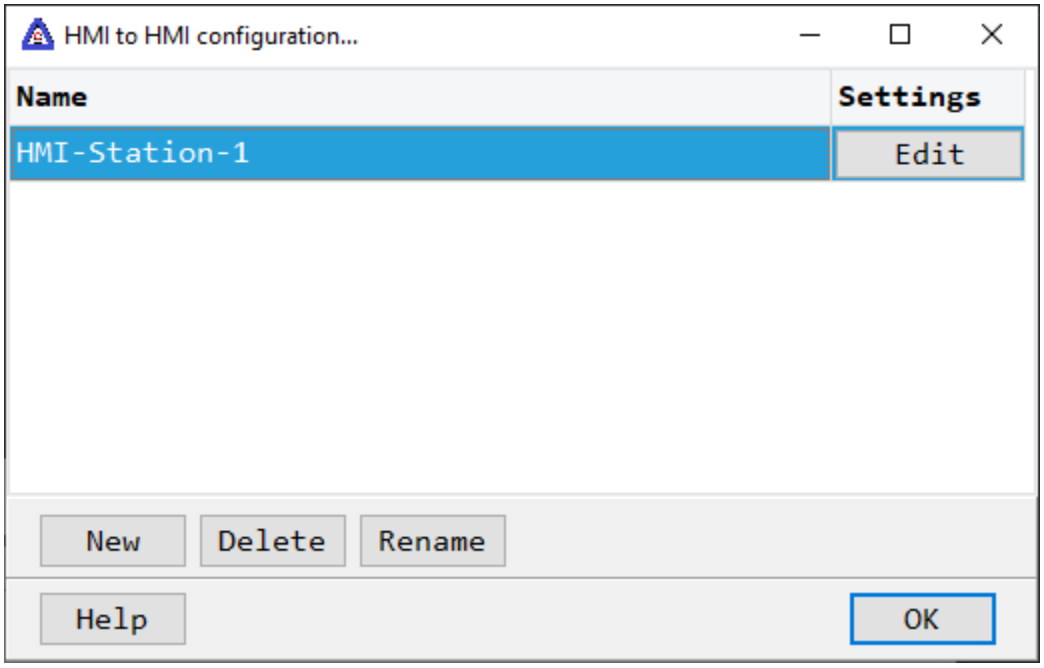
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. If the error string contains "Error : 0F:04" means "Invalid slave memory address or range in X-Request message". For other "codes" refer to the SNP-X specification. (GE document GFK-0582C)

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 1 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.
Bit data must be byte aligned	The start register for bit data must be on a byte boundary. (1, 9, 17, ...)

HMI <-> HMI

Each HMI <-> HMI master object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a HMI <-> HMI master object select the "Delete" button.

Question: I have an HMI project on a computer (A) and I want to use a second computer (B) to access the points on computer A.

Answer:

The fastest method is to clone the complete project from computer A, then modify the project as necessary for the B computer.

1) Duplicate the project directory. This new directory/project will be the project for B computer.
2) On computer A, enable the HMI <> HMI server. That is the only step for computer A.
3) On computer B, delete all communication ports. Computer B can use other communications ports but, for this example, computer B will only communicate with computer A.
4) Create an HMI <-> HMI port and give it a name. For this example we will call it "ClientOne". Enter the IP address of computer A in the settings.
5) Depending on how many points the project contains, these next steps can be done by hand or with Excel. On computer B,

A) Export the points to an Excel file.
B) Open the file.

C) Copy all the tagnames for the digital points (excluding digital host points).
D) Paste the tagnames in the source address for the digital points. Now the tagname and source address are the same. The point tagname changed as needed. The source address must be an exact match to the point tagname in computer A.
E) Repeat step D for the analog points (excluding analog host points), as required.
F) Change the port name to "ClientOne" for all the digital and analog points (excluding host points).
H) Save the file and exit Excel.
I) Import the file, overwriting the points.
J) Save the project.

That is all. Start runtime monitoring for computer A; repeat for computer B and the same data should be on both computers.

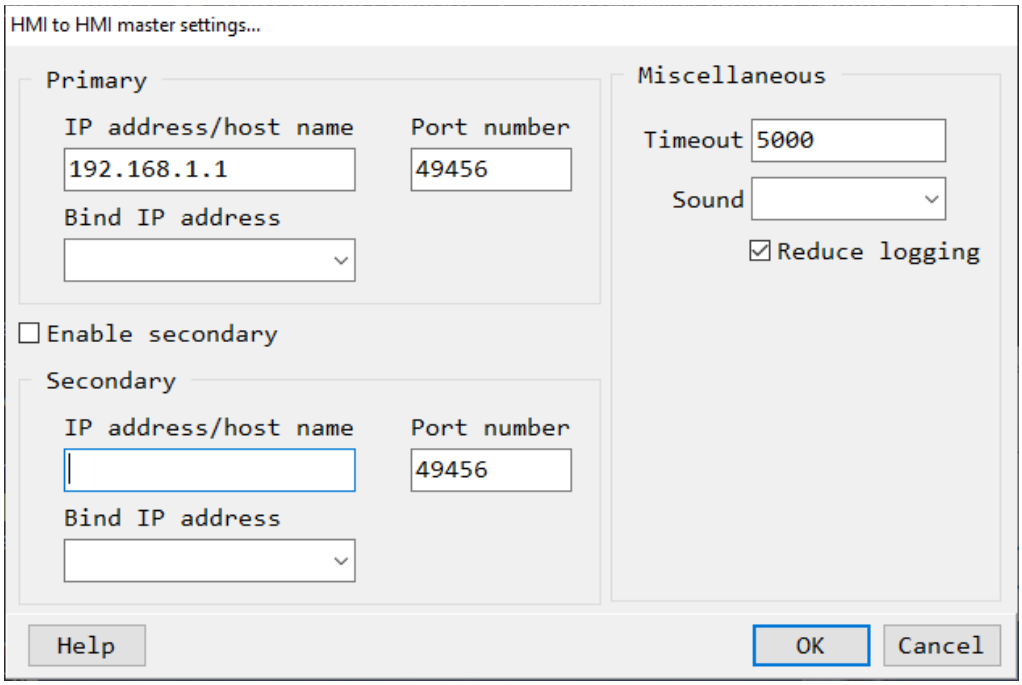
If the computers are not communicating, watchdog timeouts, etc. the most common causes:
1) The IP address in the slave port "ClientOne" for this example is not correct.
2) The port is blocked on A and/or B computer (firewall or virus protection) or blocked in a router. The port number to use is configured on the A computer and in "ClientOne" of computer B.
3) There are many other possible reasons for failure to communicate but, they are outside the scope of the HMI.

Addressing

The source address used for the point configuration, in the client, must be the exact point tagname used in the server. Do not include any attribute fields in the source string.

Based on the point type in the server and client the 'Process Variable Analog' or the 'Process Variable Digital' will be automatically selected.

Settings



The port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

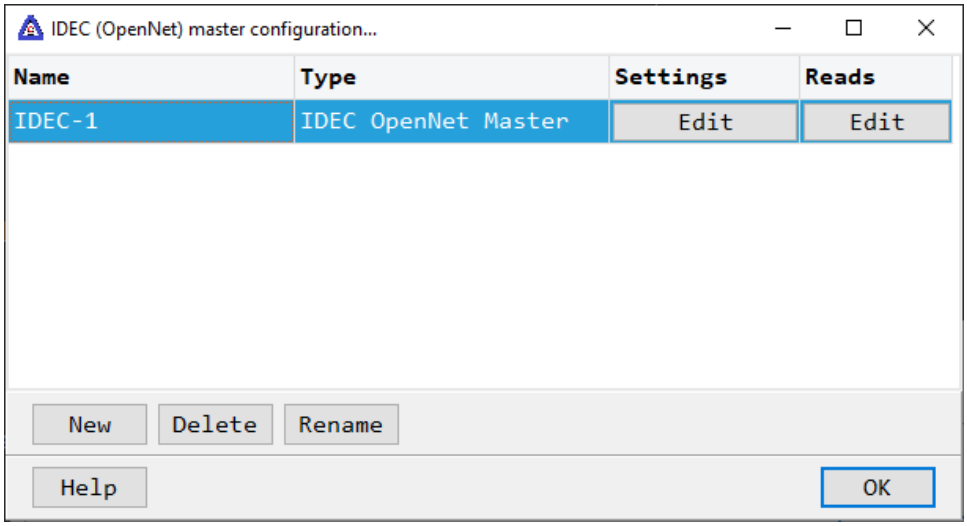
When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sounds delete the value in the sound name field.

Reduced logging

When enabled and a watchdog timeout occurs, a single entry will be placed in the event log and the sound will be queued (if configured). When the slave device responds, after a watchdog timeout, a single entry will be placed in the event log. The watchdog counter will continue to count each time the watchdog timer expires.

IDEC (OpenNet)

Each IDEC (OpenNet) master object is listed in the window.

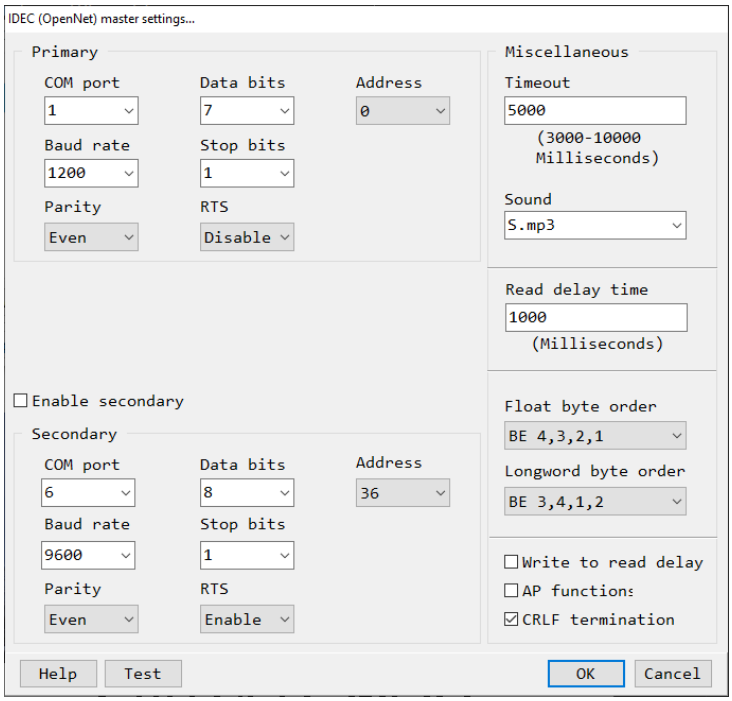


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an IDEC master object select the "Delete" button.

Settings



The serial port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Address

The address of the slave device.

RTS

See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.
If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if

configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sounds delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Float byte order, Longword byte order

Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The register data received from the slave device is in LO-HI byte order. Regular integer data and floating data values can be in the same data read.

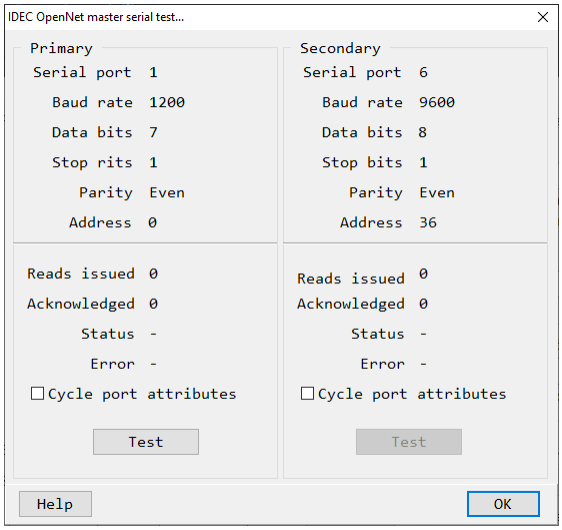
The 4 byte float/longword option is configured on a point level. The integer data is converted into a floating point value at the point level.

When using floating point values the byte order of the slave device might be different than the regular register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

AP functions

See analog functions.

Test button

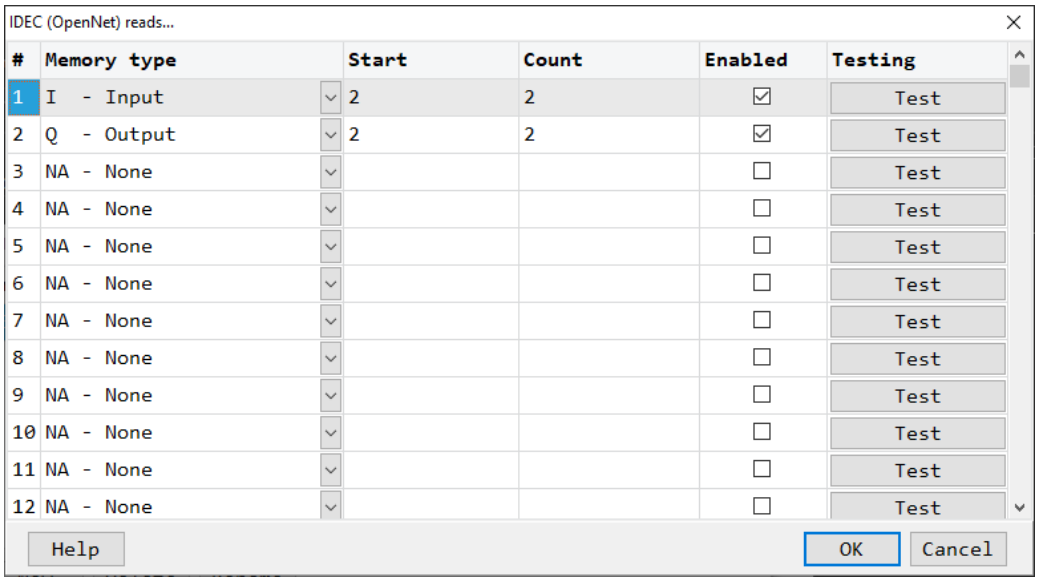


When the test button is selected the program will send an "Enquire" message to the slave device.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads



The address ranges shown may or may not be present in the slave device.
The HMI uses the following addressing for access to the data in the slave.

Example address:
I1, Q74, TP75, CC43, W3

Register Type

The supported registers.

Register Type	Prefix	Data Type
Input	I	Bit
Output	Q	Bit
Internal relay	M	Bit
Link relay	O	Bit
Line register	L	Byte
Shift register	R	Byte
Timer	T	Bit
Counter	C	Bit
Data register	D	Word
Calendar/clock	W	Word
Timer preset	TP	Word
Timer current	TC	Word
Counter preset	CP	Word

Counter current	CC	Word
High speed counter preset	HP	Longword
High speed counter current	HC	Longword

For input, output, internal relay and link relay the last digit of the start must be octal.

Start

The starting address to read. The different memory areas have different counts. Refer to the slave device for legal register values. Enter the starting register value, in octal.

Count

The number of registers to read for the request in decimal. The memory type and slave device determine the maximum number of registers that can be read. The protocol has a maximum of 200 bytes of data.

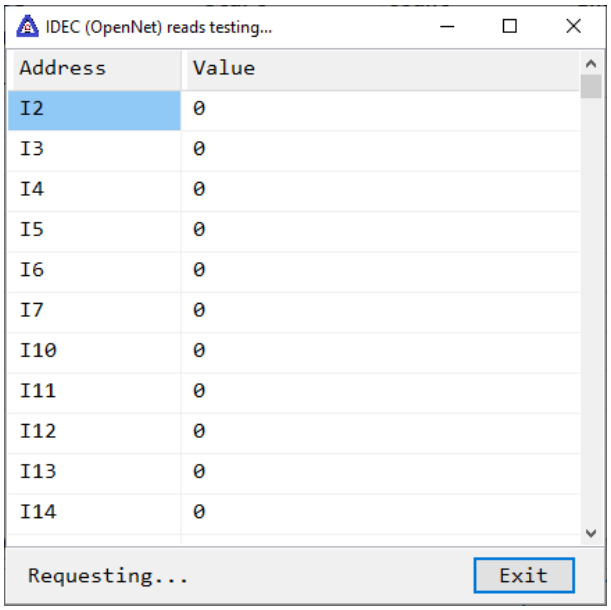
Register Type	**Count**
Input	bytes
Output	bytes
Internal relay	bytes
Link relay	bytes
Line register	bytes
Shift register	bytes
Timer	number of timers, maximum 48
Counter	number of counters, maximum 48
Data register	words - maximum 100 words (200 bytes)
Calendar/clock	7 - the start must be 0 and the count must be 7
High speed counter	0 - the start is the high speed counter number

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

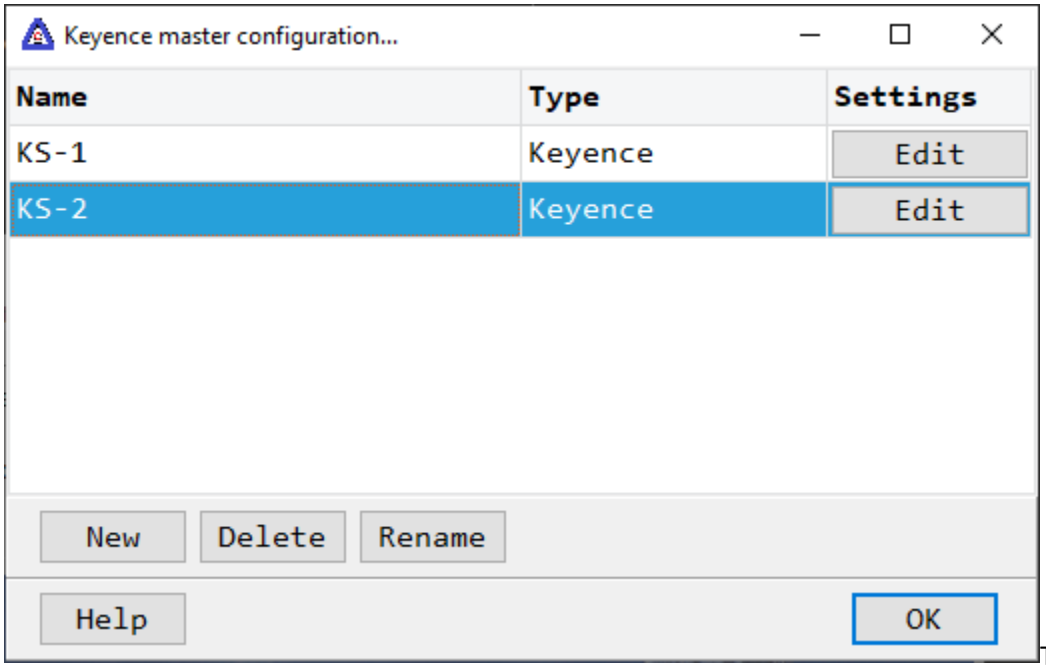
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start not in range for the memory type.
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.
Last digit is not octal	The last digit of the selected memory type must be 0 - 7.

KEYENCE

Each Keyence master object is listed in the window.

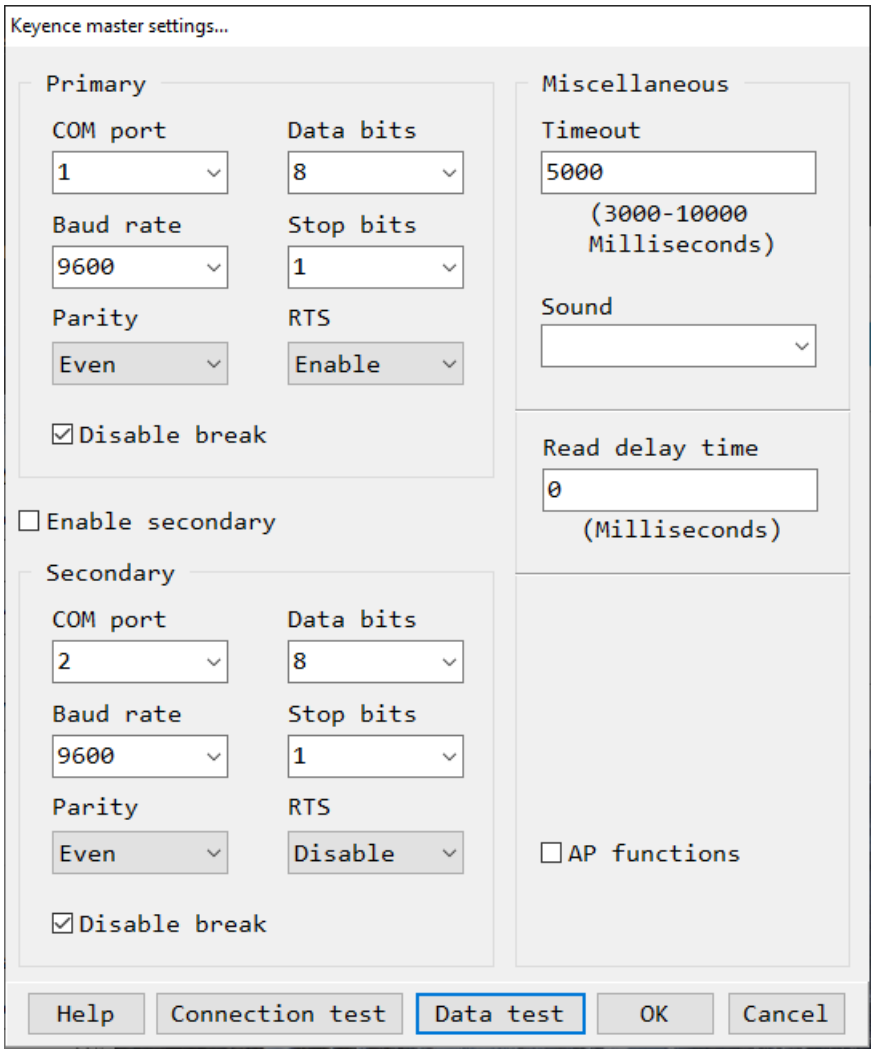


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Keyence master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

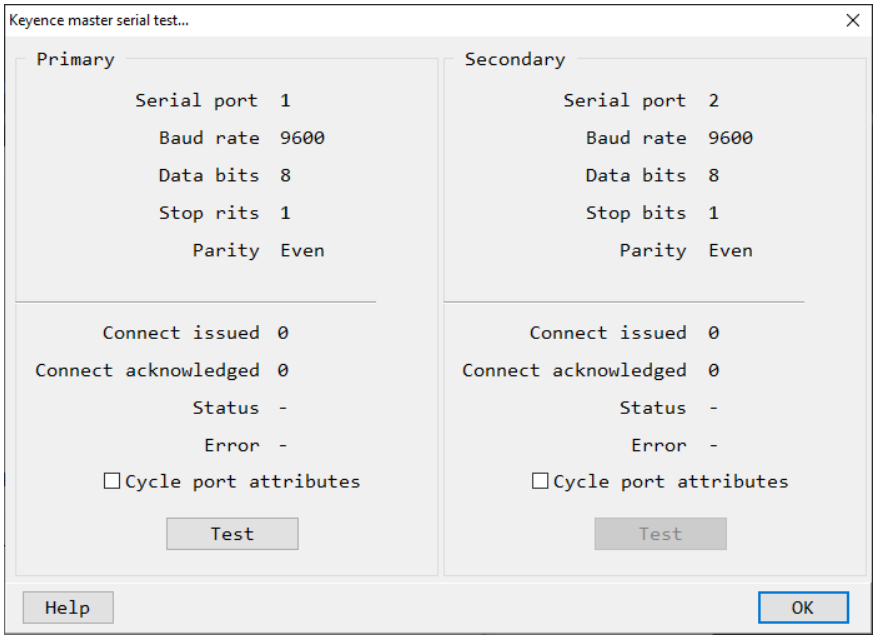
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Connection Test



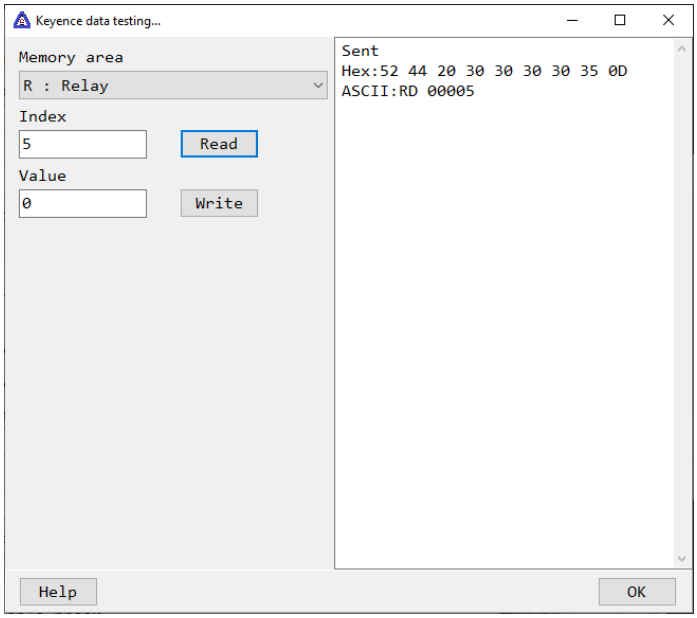
Used to verify the serial ports settings from the PC to the PLC.

When the test button is selected the program will send a "Connect" message to the slave device.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Data Test



Used to verify an address exists in the PLC and can be read.

Holding down the "CTRL" key while selecting the button will command the test code to utilize the secondary port configuration attributes. Otherwise, the primary port configuration attributes will be utilized.

Error messages

E0	Relay number error
E1	Command error
E2	Program unregistered
E3	CPU fault
E4	Write protected
E5	CPU error

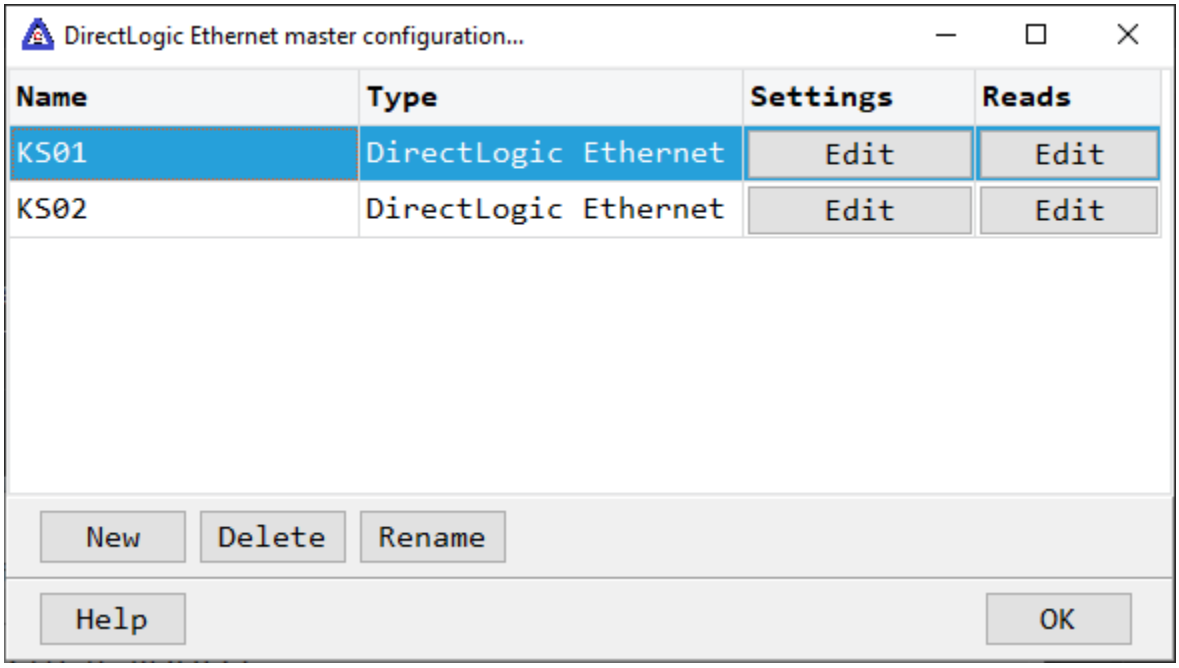
Register types

Supported Keyence registers.

Register Type	Prefix	Data Type
Analog Trimmer	AT	Word
Counter Contact	C	Bit
Counter Current Value	CC	Word
Counter Preset Value	CP	Word
High Speed Counter Comparator Contact	CTC	Bit
High Speed Counter Contact	CTH	Bit
High Speed Counter Comparator Current Value	CTCC	Word
High Speed Counter Comparator Preset Value	CTCP	Word
High Speed Counter Current Value	CTHC	Word
High Speed Counter Preset Value	CTHP	Word
Data Memory	DM	Word
Relay	R	Bit
Timer Contact	T	Bit
Timer Current Value	TC	Word
Temporary Data Memory	TM	Word
Timer Preset Value	TP	Word

K-Sequence (DirectLogic) Ethernet

Each DirectLogic (DL) master object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

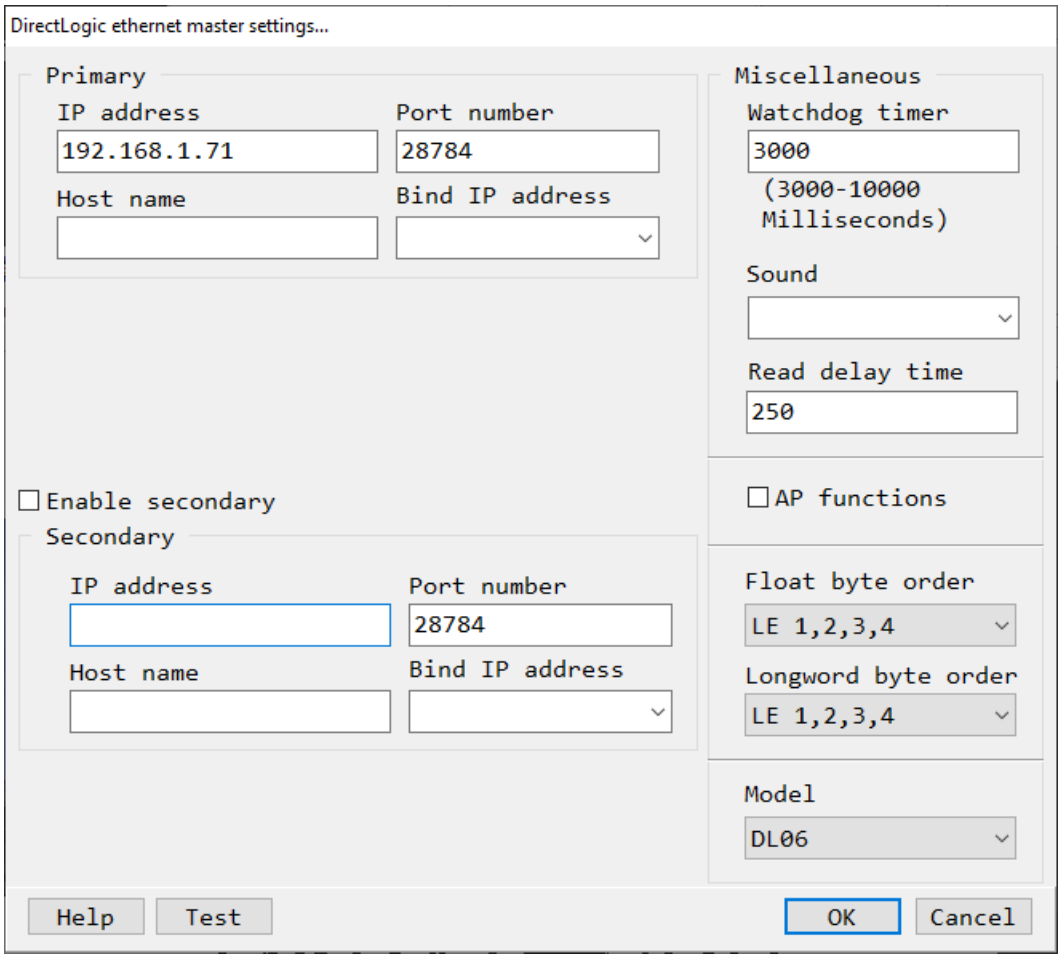
To rename an object select the "Rename" button and supply a new name.

To delete a Direct Logic master object select the "Delete" button.

AD Commands (UDP)

Name	Code
Read	1Eh
Write	20h

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not

replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Note: During testing, using a DL06, we discovered the PLC would sometimes not respond. The PLC supports UDP, a connectionless protocol. Three watchdog timeouts must occur before an error is indicated.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Float byte order, Longword byte order

Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The register data received from the slave device is in LO-HI byte order. Regular integer data and floating data values can be in the same data read.

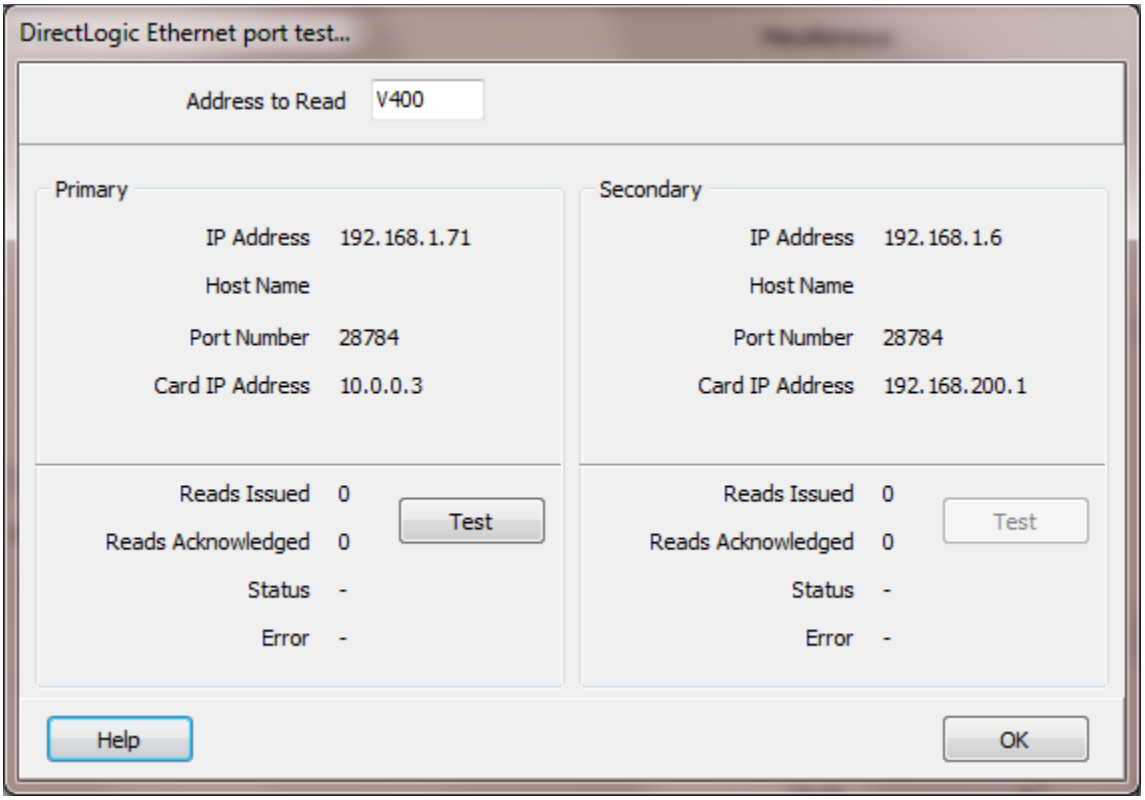
The 4 byte float/longword option is configured on a point level. The integer data is converted into a floating point value at the point level.

When using floating point values the byte order of the slave device might be different than the regular register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

Model

We tested with a DL05, DL06 and a DL205 (CPU DL240). While both PLCs (05/06) use K-Sequence as a protocol they each had slight variations from the specifications. For the DL205 select the DL05 model. If using another model and are not able to communicate using either selection, please contact technical support.

Test button



When the test button is selected the program will attempt to read one word of data from the address in the edit field.

The program will attempt to use the communication parameters configured.

Reads

DirectLogic ethernet reads...

#	Memory type	Start register	Count	Enabled	Testing
1	V - Variable	1	2	☑	Test
2	NA - None			☐	Test
3	NA - None			☐	Test
4	NA - None			☐	Test
5	NA - None			☐	Test
6	NA - None			☐	Test
7	NA - None			☐	Test
8	NA - None			☐	Test
9	NA - None			☐	Test
10	NA - None			☐	Test
11	NA - None			☐	Test
12	NA - None			☐	Test

Help OK Cancel

The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave. All values are in octal

Example address:

V1, TA75, CA43, SW3
V34/6, GX73, X6, Y22

Notes:
1. Do not use leading zeros in the address string.
2. Serial, when modifying a bit in a non-bit register, all other bits in the register are cleared.
3. Ethernet, when modifying a bit register, all other bits in the register are cleared.

Memory type

The supported DirectLogic registers.

Register Type	Prefix	Ref Num (octal)	Data Type
Variable	V	0	Word
Timer accumulator	TA	0	Word
Counter accumulator	CA	1000	Word
System status	SW	7600	Word
Global input	GX	40000	Boolean
Global output	GY	40200	Boolean
Input	X	40400	Boolean
Output	Y	40500	Boolean
Control relays	C	40600	Boolean
Stage status	S	41000	Boolean
Timer status	T	41100	Boolean
Counter status	CT	41140	Boolean
System status	SB	41200	Boolean

If the starting code of the register area is not correct for the slave device or, the register type is not shown, use the "Variable" memory read. This is an "untyped" read.

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values. Enter the starting register value, in octal.

The start register for the 'Boolean' data type must be on a word boundary.

Count

The number of registers to read for the request in decimal. The memory type and slave device determine the maximum number of registers that can be read.

The protocol has a maximum of 128 words of data.

All counts are the number of words to read. For bit data each count value returns 16 bits starting at "start register". For word data, each count value returns 1 word.

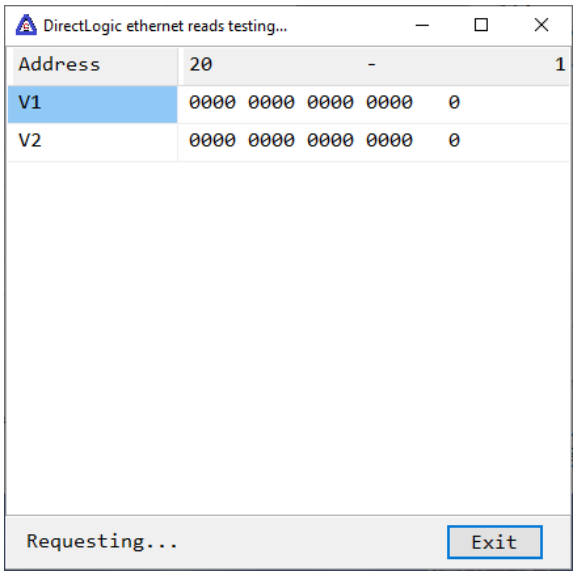
Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be

created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

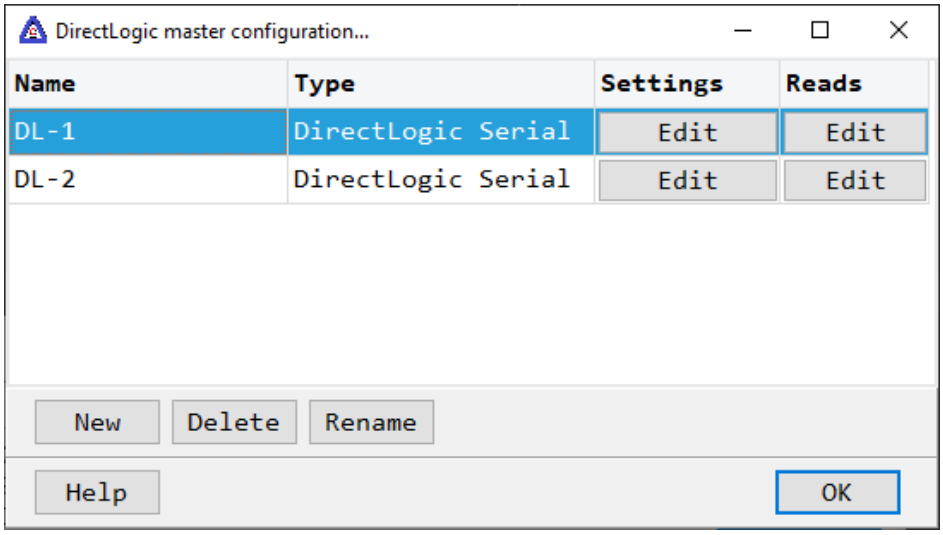
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.
Start register is not octal	All register references are in octal.

K-Sequence (DirectLogic) Serial

Each DirectLogic (DL) master object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

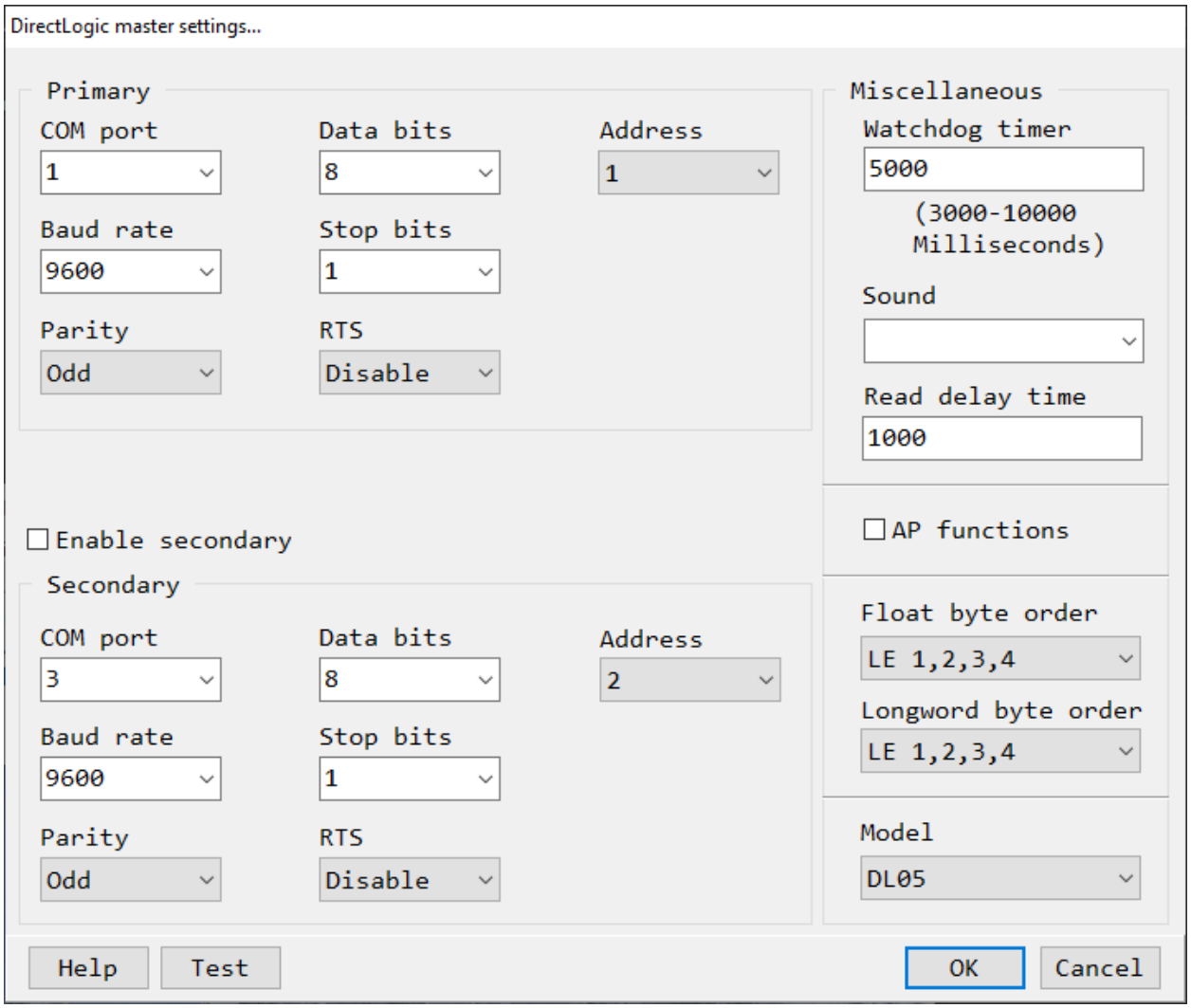
To rename an object select the "Rename" button and supply a new name.

To delete a DL master object select the "Delete" button.

AD Commands (serial)

Name	Task Code
Monitor Data	40h
Write Data to Register Memory	46h
Force On	44h
Force Off	45h

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

RTS

See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Float Byte Order, Longword Byte Order

Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The register data received from the slave device is in LO-HI byte order. Regular integer data and floating data values can be in the same data read.

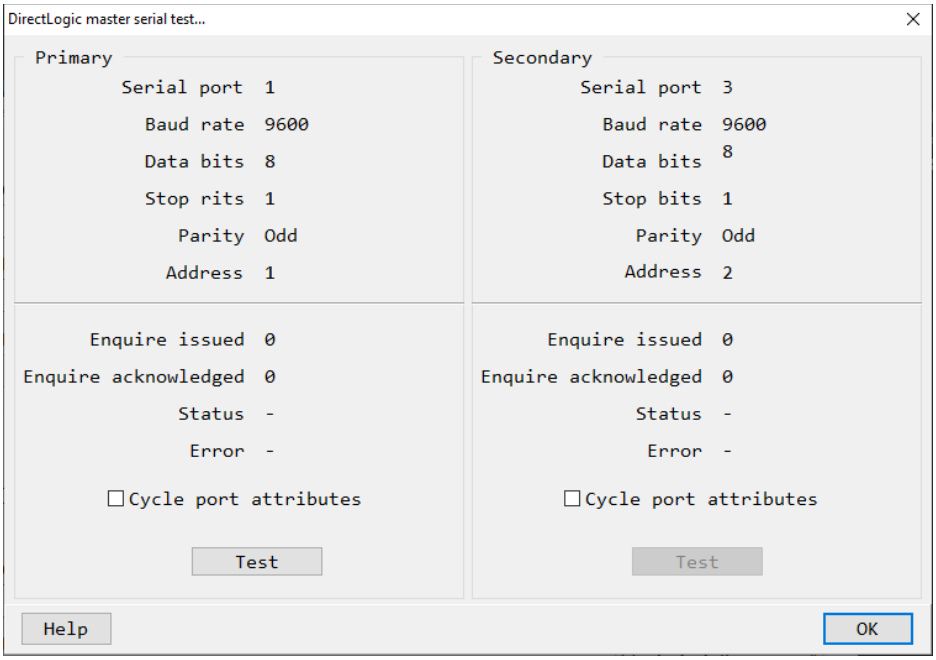
The 4 byte float/longword option is configured on a point level. The integer data is converted into a floating point value at the point level.

When using floating point values the byte order of the slave device might be different than the regular register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

Model

We tested with a DL05, DL06 and a DL205 (CPU DL240). While both PLCs (05/06) use K-Sequence as a protocol they each had slight variations from the specifications. For the DL205 select the DL05 model. If using another model and are not able to communicate using either selection please contact technical support.

Test button

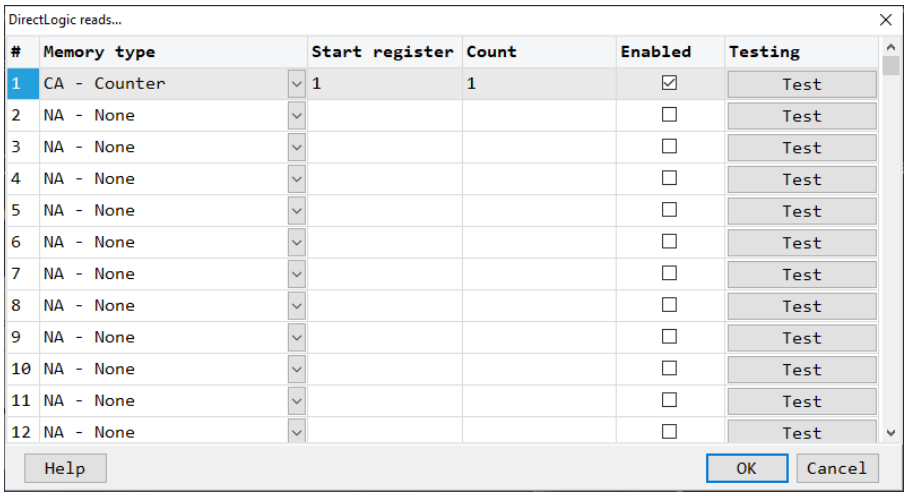


When the test button is selected the program will send an "Enquire" message to the slave device.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave. All values are in octal

Example address:

V1, TA75, CA43, SW3
V34/6, GX73, X6, Y22

Notes:
1. Do not use leading zeros in the address string.
2. Serial, when modifying a bit in a non-bit register, all other bits in the register are cleared.
3. Ethernet, when modifying a bit a register, all other bits in the register are cleared.

Register Type

The supported DirectLogic registers.

Register Type	Prefix	Ref Num (octal)	Data Type
Variable	V	0	Word
Timer Accumulator	TA	0	Word
Counter Accumulator	CA	1000	Word
System Status	SW	7600	Word
Global Input	GX	40000	Boolean
Global Output	GY	40200	Boolean
Input	X	40400	Boolean
Output	Y	40500	Boolean
Control Relays	C	40600	Boolean
Stage Status	S	41000	Boolean

Timer Status	T	41100	Boolean
Counter Status	CT	41140	Boolean
System Status	SB	41200	Boolean

If the starting code of the register area is not correct for the slave device or, the register type is not shown, use the "Variable" memory read. This is an "untyped" read.

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values. Enter the starting register value, in octal.

The start register for the 'Boolean' data type must be on a word boundary.

Count

The number of registers to read for the request in decimal. The memory type and slave device determine the maximum number of registers that can be read.

The protocol has a maximum of 128 words of data.

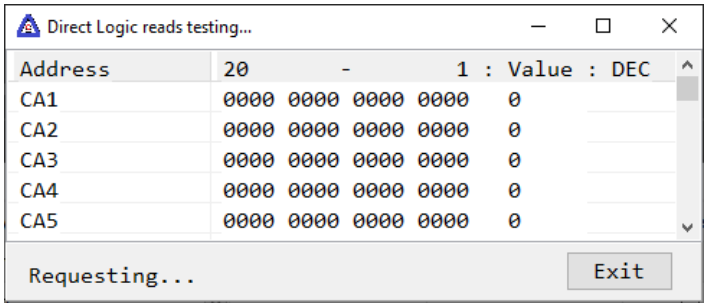
All counts are the number of words to read. For bit data each count value returns 16 bits starting at "start register". For word data, each count value returns 1 word.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

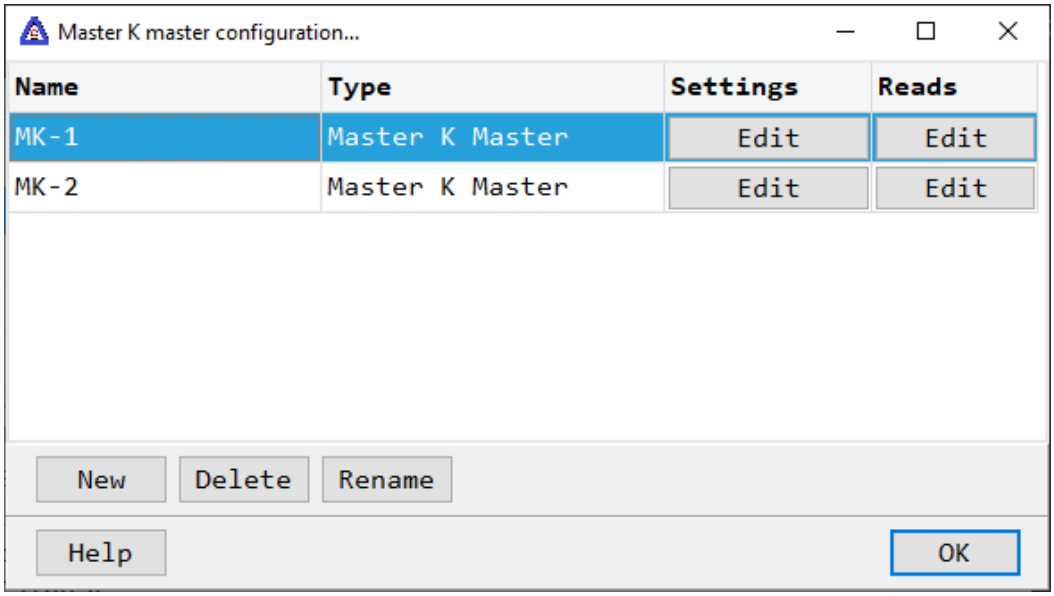
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.
Start register is not octal	All register references are in octal.

MASTER-K

Each Master-K master object is listed in the window.

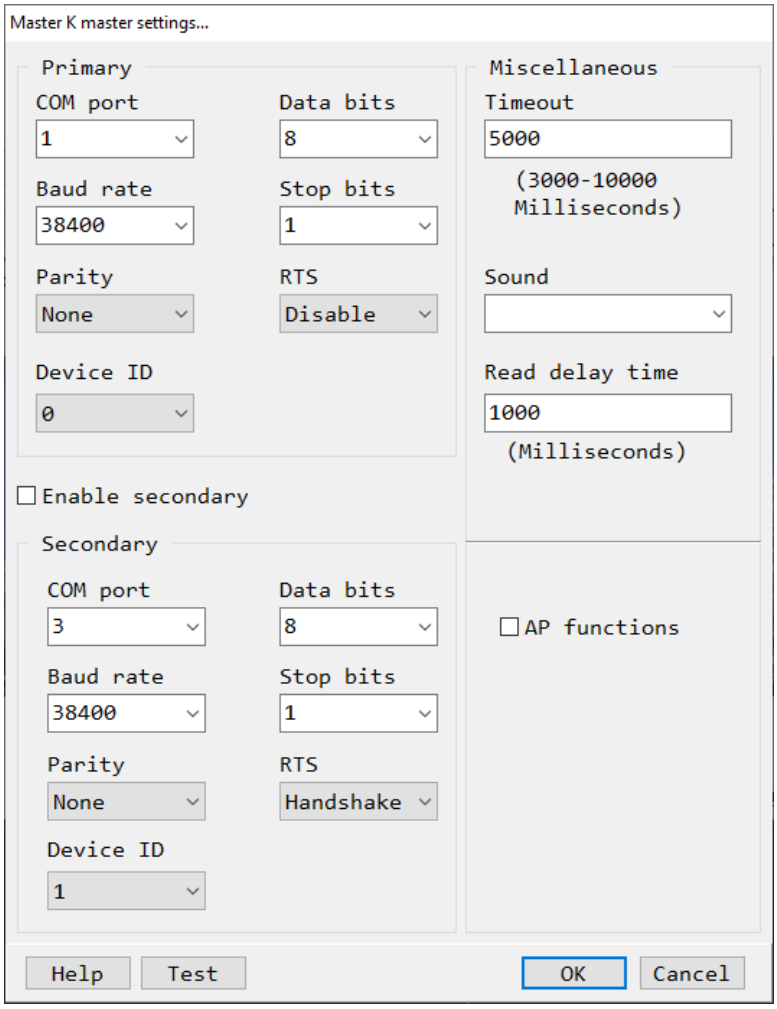


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Master-K master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

RTS

See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if

configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Device ID

The ID of the slave device. (0-31)

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

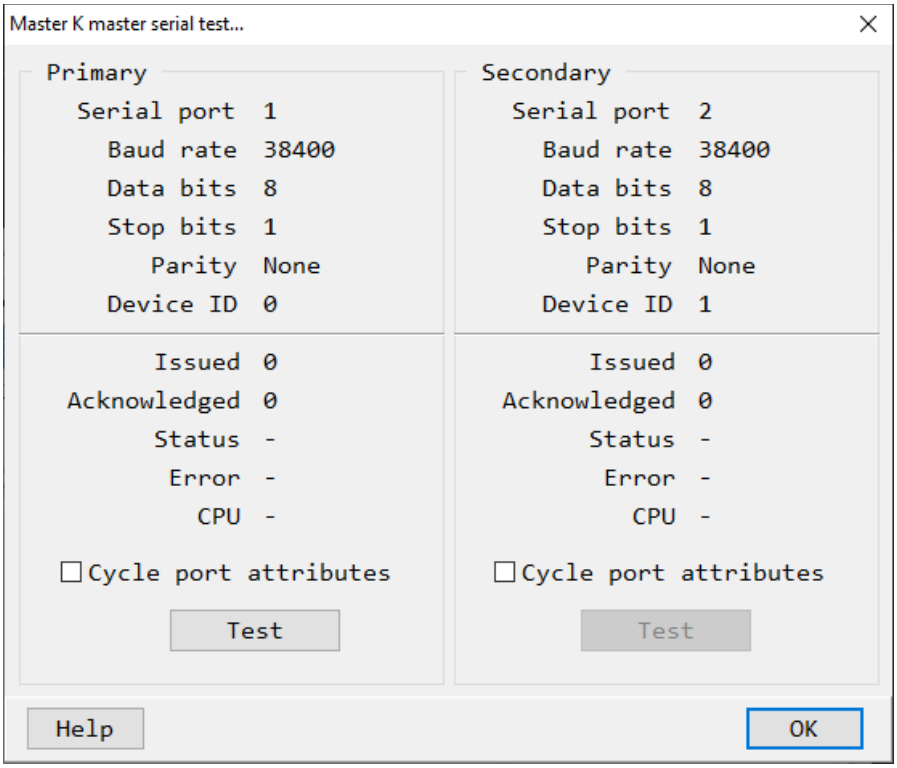
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button

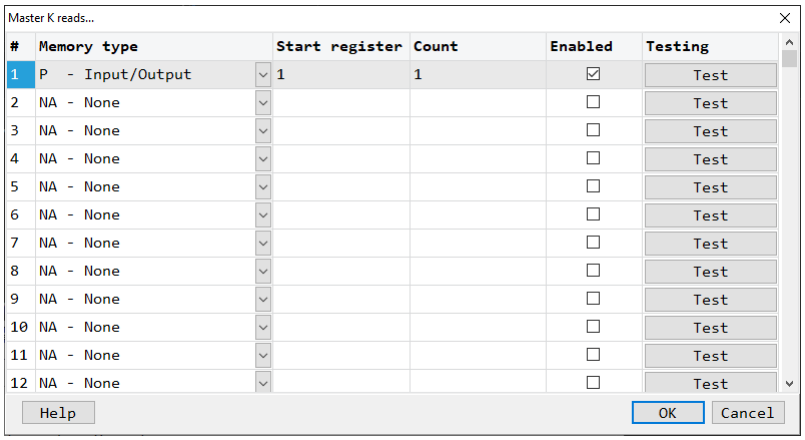


When the test button is selected the program will send an "Enquire" message to the slave device.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the Master-K register device characters. Exception: The '%', 'X', 'B' and 'W' are not used. Bits are 0-15. Bit, word, double word and float are supported.

Addresses in the logic editor are in hexadecimal. Addresses in the HMI are in decimal. For example M001A is a bit reference in the PLC. In the HMI it would be M1.10

Example address:

P0.1	Word 0 bit 1
P9.15	Word 9 bit 15
TV1	Timer 1 elapsed time
T1	Timer 1 complete bit
M9	Memory register 9 (If the point is configured as a 4 byte float or integer M9 and M10 will be used)
M17.4	Word 17 bit 4

Note: The protocol specification does not specify how floats are ordered. Do <u>not</u> use leading zeros in the address. i.e. D0024 should be D24.

Memory type

The supported Master-K defined memory areas.

Register Type	Prefix
Input/Output	P
Auxiliary relay	M
Link relay	L
Keep relay	K
Counter state	C
Timer state	T
Data register	D
Step relay	S
Special relay	F
Timer value	TV
Counter value	CV

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values.

Count

The number of words to read for the request. The memory type and slave device determine the maximum number of registers that can be accessed. For all memory areas except counter and timer state the maximum is 60 register.

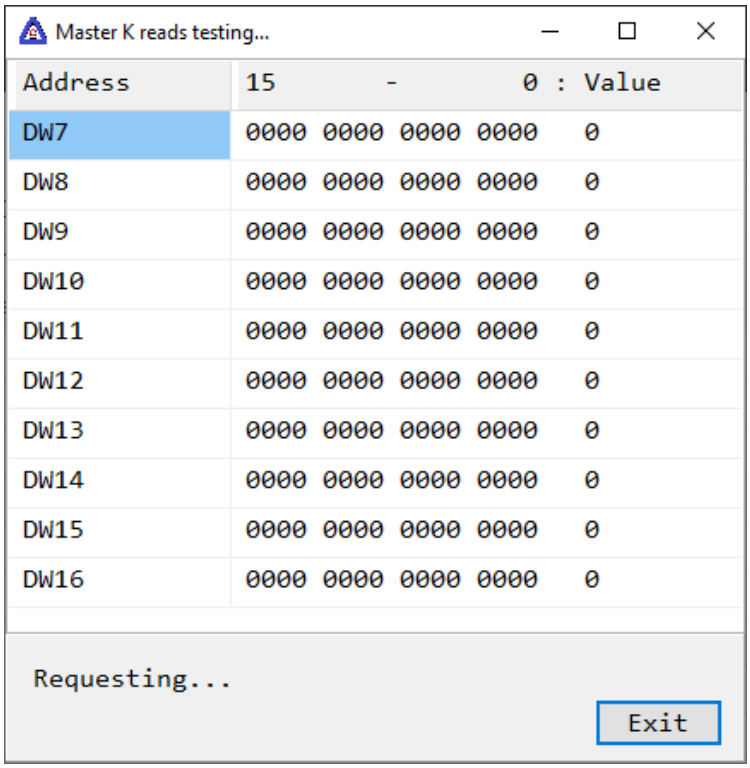
For timer and counter states the maximum is 1. A read of 1 will read the 16 timer/counter states. If reading many timers/counters, it would be more efficient to move the timer state into the 'M' memory area and read the 'M' memory.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



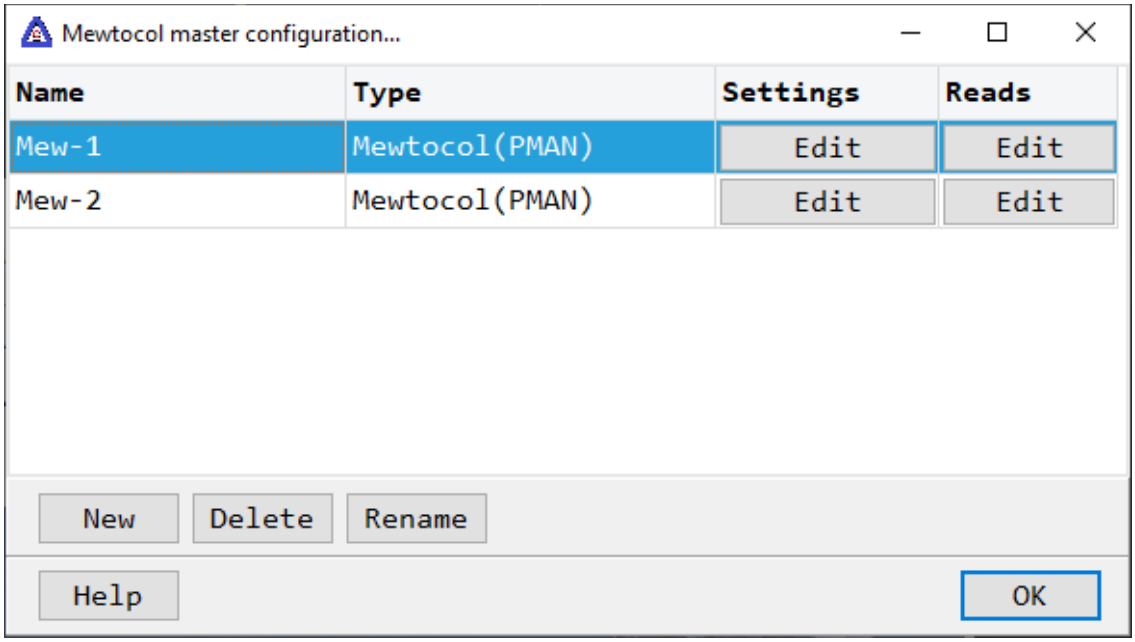
When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 1 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

MEWTOCOL (PANASONIC-MATSUSHITA-AROMAT-NAIS)

Each Mewtocol master object is listed in the window. Panasonic, Matsushita, Aromat and NAIS are some of the PLC brands using the Mewtocol protocol.

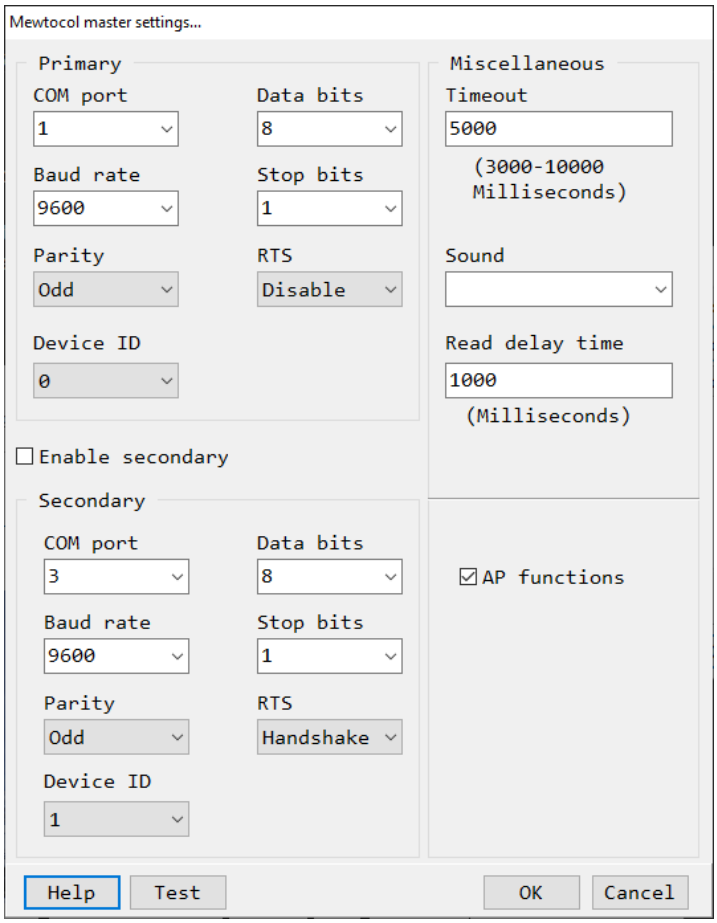


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Mewtocol master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

RTS

See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Device ID

The ID of the slave device. (1-31)

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

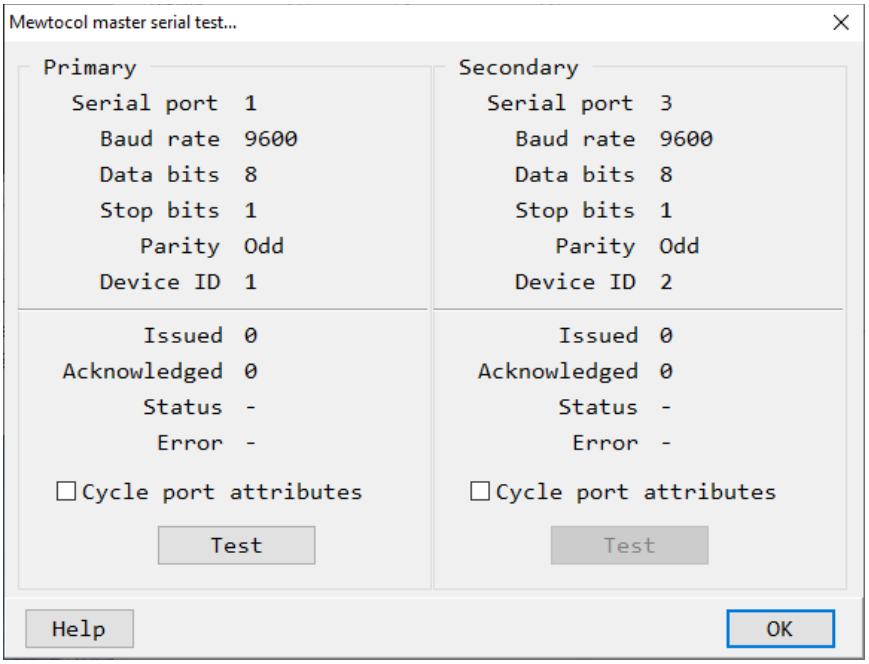
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button



When the test button is selected the program will send an "Enquire" message to the slave device.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads

Mewtocol reads...

#	Memory type	Start register	Count	Enabled	Testing
1	X - External input	1	1	☑	Test
2	DT - Data Register	2	12	☑	Test
3	NA - None			☐	Test
4	NA - None			☐	Test
5	NA - None			☐	Test
6	NA - None			☐	Test
7	NA - None			☐	Test
8	NA - None			☐	Test
9	NA - None			☐	Test
10	NA - None			☐	Test
11	NA - None			☐	Test
12	NA - None			☐	Test

Help OK Cancel

The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave. All values are in decimal

Address examples

X0, X1F
Y0, Y4A
R0, R105, R4A
DT4, DT56

Note: For X, Y, R and L registers the Mewtocol specification states the last digit must be in hex and the leading digits up 3 places must be decimal. 1F is legal, F1 is not. In hexadecimal when 1 is added to 9F the result is A0. A0 would not be legal per the specification. After 9F in the specification is 100.

The HMI accepts all hexadecimal values for addresses. Verify the external device has the address desired.

Memory type

Register Type	Prefix	Data Type	Format
External input	X	Boolean	DDDH (Dec, Dec, Dec, Hex)
External output	Y	Boolean	DDDH
Internal relay	R	Boolean	DDDH
Link relay	L	Boolean	DDDH
Timer state	T	Boolean	Decimal
Counter state	C	Boolean	Decimal
Data Register	DT	Word, double word, float	Decimal
Link Data Register	LD	Word, double word, float	Decimal
File Register	FL	Word, double word, float	Decimal
Timer set	TS	Word	Decimal
Timer elapsed	TE	Word	Decimal
Counter set	CS	Word	Decimal
Counter elapsed	CE	Word	Decimal

Start tegister

The starting register to read. This is the word address of the memory area. The value is entered in decimal. When entering point addresses the value is in hexadecimal or decimal based on the register type.

Count

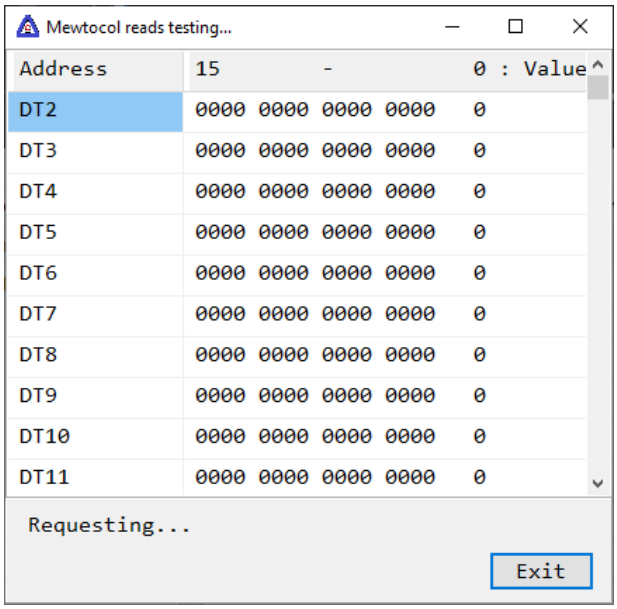
The number of registers to read for the request in decimal. Each register is 16 bits. The protocol has a maximum of 27 words of data per request.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

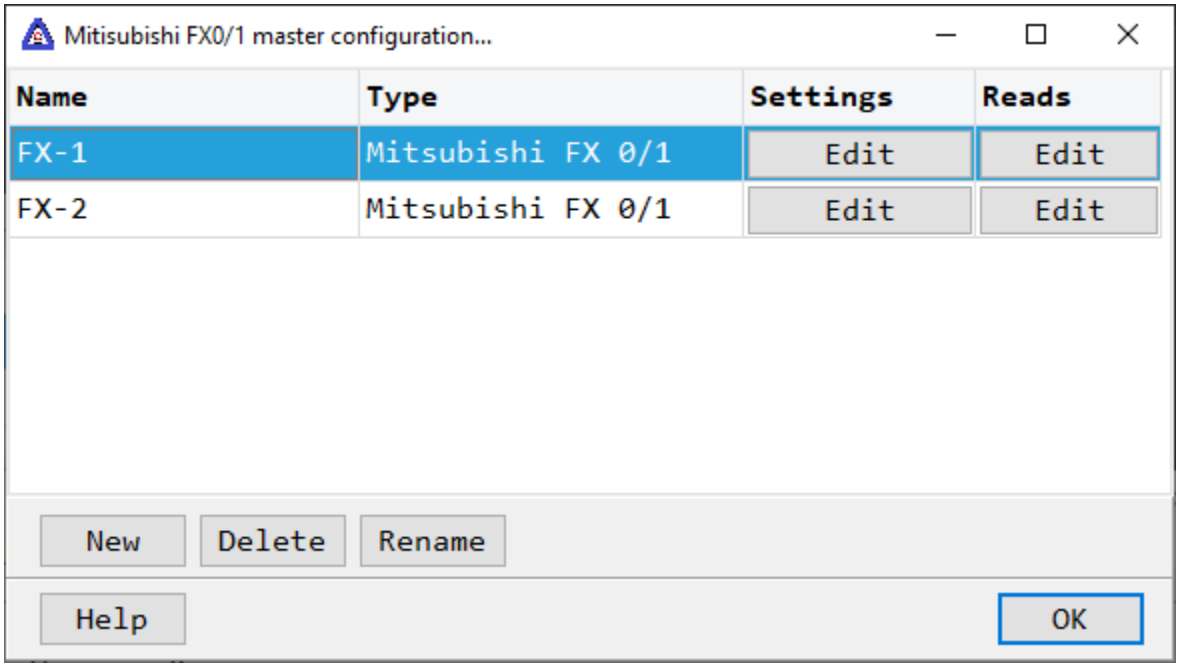
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. See 'Testing error message 'below.

Read configuration error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

MITSUBISHI FX 0/1

Each FX0/1 master object is listed in the window.

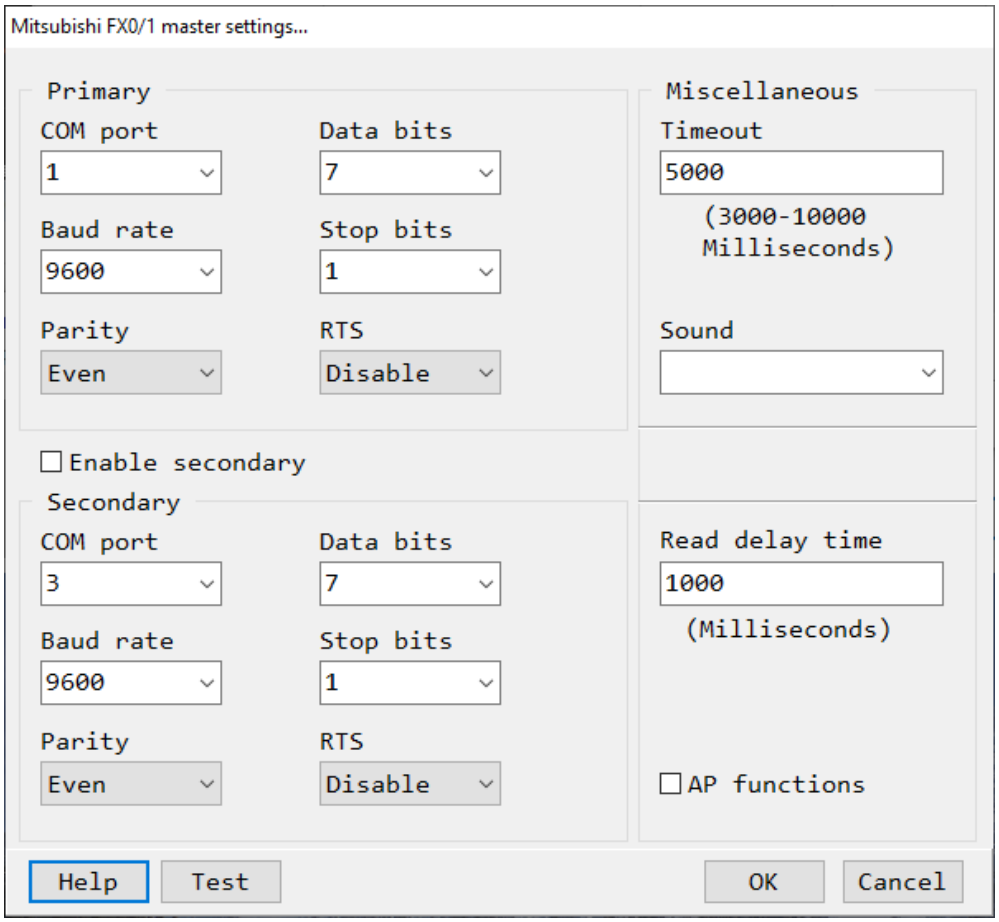


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a FX0/1 master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

RTS

See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

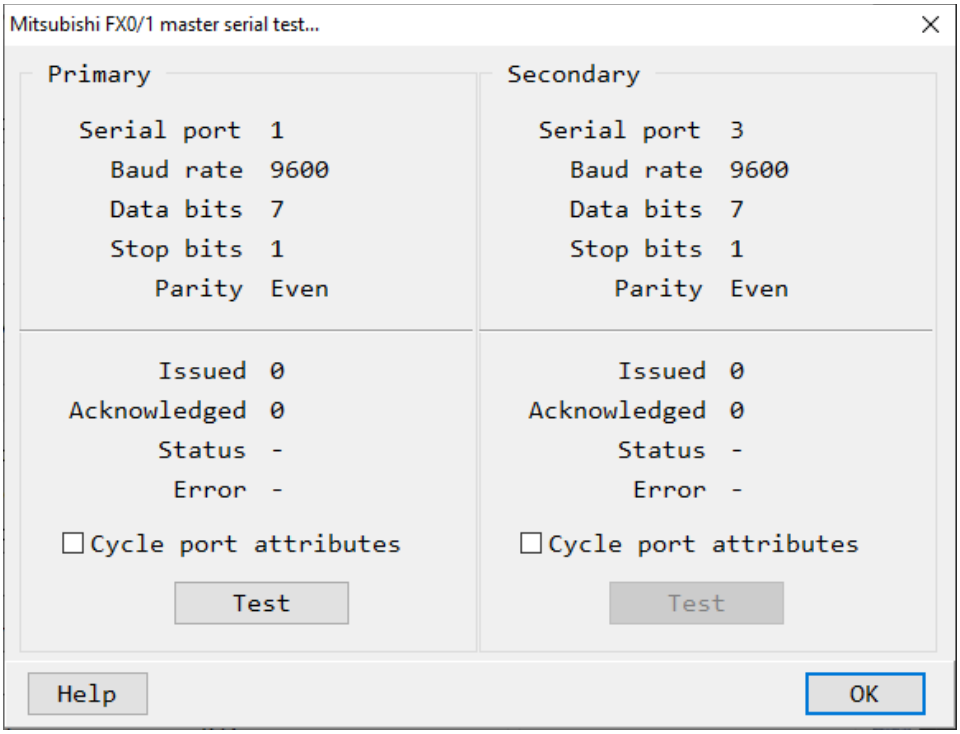
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button



When the test button is selected the program will send an "Enquire" message to the slave device.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads

Mitisubishi FX0/1 reads...

#	Memory type	Start register	Count	Enabled	Testing
1	SR - Special data	8000	10	☑	Test
2	NA - None			☐	Test
3	NA - None			☐	Test
4	NA - None			☐	Test
5	NA - None			☐	Test
6	NA - None			☐	Test
7	NA - None			☐	Test
8	NA - None			☐	Test
9	NA - None			☐	Test
10	NA - None			☐	Test
11	NA - None			☐	Test
12	NA - None			☐	Test

Help OK Cancel

The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave.

Address examples

X0, X105, Y3, Y77	Inputs and outputs are in **octal**.
M2, M90	
MS8000, MS8255	
SR8000	default for this type is word
SR8002	point is configured for longword or float
TS3, TS89	
CS4, CS90	
TR4, TR8	
CR34, CR56	
T4, T67	word access to the timer accumulator
C56, C77	word access to the counter
CX8, CX72	longword access to the counter accumulator
S2, S90	
D1	default for this type is word
D0	point is configured for longword or float

Notes:

1. Do not use leading zeros in the address string.
2. When modifying a bit in a non-bit register, all other bits in the register are cleared.

Memory type

The supported FX0/1 registers.

Register Type	Prefix	Radix	Count	Data Type
Input	X	Octal	Byte	Bit
Output	Y	Octal	Byte	Bit
Auxiliary relays	M	Decimal	Byte	Bit
Special auxiliary relays	MS	Decimal	Byte	Bit
Special data register	SR	Decimal	Word	Word, longword, float
Timer state	TS	Decimal	Byte	Bit
Counter state	CS	Decimal	Byte	Bit
Timer reset	TR	Decimal	Byte	Bit
Counter reset	CR	Decimal	Byte	Bit
Timer	T	Decimal	Word	Word
Counter	C	Decimal	Word	Word
Counter (32 bit)	CX	Decimal	Word	Longword
States	S	Decimal	Byte	Bit
Data register	D	Decimal	Word	Word, longword, float

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values. Enter the starting register value, in octal.

1. Special auxiliary relays minimum start register is 8000.
2. Special data register minimum start register is 8000.

Count

The number of registers to read for the request in decimal. The memory type and slave device determine the maximum number of registers that can be read. The protocol has a maximum of 64 bytes of data.

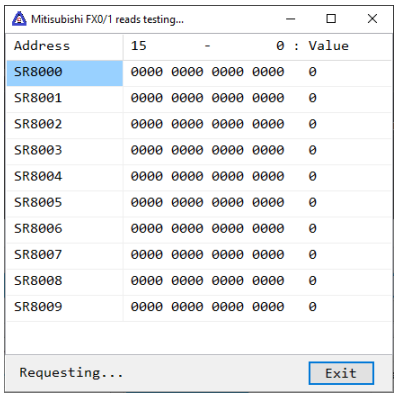
1. For bit data each count value returns 8 bits starting at "start register". The bit address must be byte aligned.
2. For timers and 16 bit counters the count is the number of timers. Each timer/counter is 2 bytes
3. For 32 bit counters the count is the number of counters. Each counter is 4 bytes.
4. When reading data registers and the point data type is longword or float the count must include all bytes. (4 per value)
5. When referencing 32 bit counters apply the start address from the start address for the PLC type. (CX0 = 32 bit start counter in PLC)

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

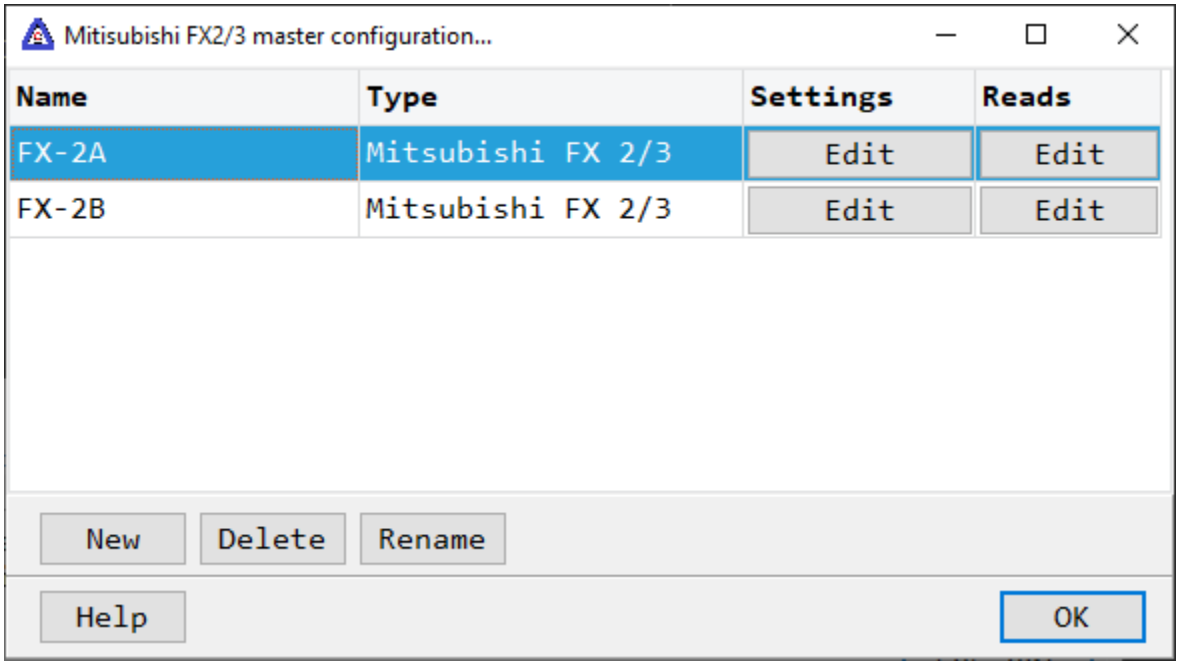
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.
Start register is not octal	All register references are in octal.

MITSUBISHI FX 2/3

Each FX2/3 master object is listed in the window.

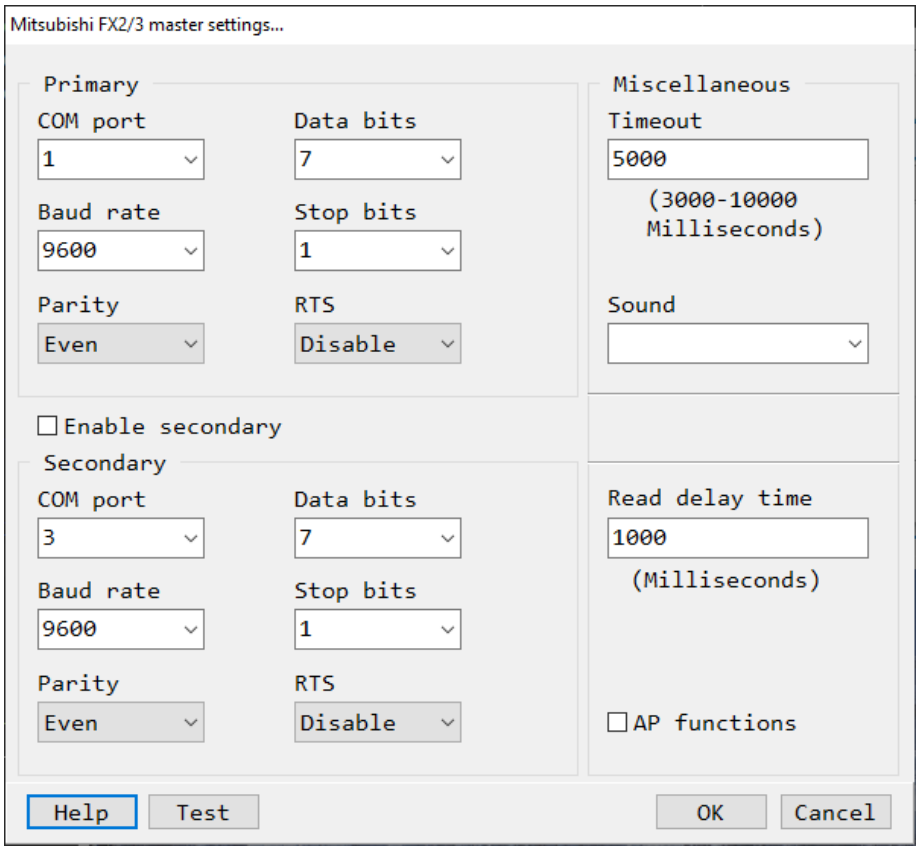


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an FX2/3 master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

RTS

See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

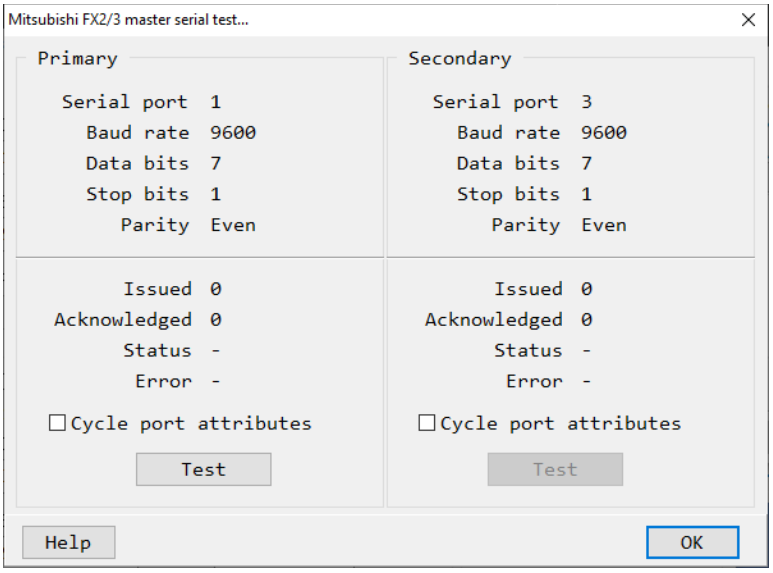
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button



When the test button is selected the program will send an "Enquire" message to the slave device.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads

Mitisubishi FX2/3 reads...

#	Memory type	Start register	Count	Enabled	Testing
1	D - Data register	8000	5	☑	Test
2	NA - None			☐	Test
3	NA - None			☐	Test
4	NA - None			☐	Test
5	NA - None			☐	Test
6	NA - None			☐	Test
7	NA - None			☐	Test
8	NA - None			☐	Test
9	NA - None			☐	Test
10	NA - None			☐	Test
11	NA - None			☐	Test
12	NA - None			☐	Test

Help OK Cancel

The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave.

Address Examples

X0, X105, Y3, Y77	Inputs and outputs are in **octal**.
M2, M90	
MS8000, MS8255	
SR8000	default for this type is word
SR8002	point is configured for longword or float
TS3, TS89	
CS4, CS90	
TR4, TR8	
CR34, CR56	
T4, T67	word access to the timer accumulator
C56, C77	word access to the counter
CX8, CX72	longword access to the counter accumulator
S2, S90	
D1	default for this type is word
D0	point is configured for longword or float

Notes:
1. Do not use leading zeros in the address string.
2. When modifying a bit in a non-bit register, all other bits in the register are cleared.

Memory type

The supported FX2/3 registers.

Register Type	Prefix	Radix	Count	Data Type
Input	X	Octal	Byte	Bit
Output	Y	Octal	Byte	Bit
Auxiliary relays	M	Decimal	Byte	Bit
Special auxiliary relays	MS	Decimal	Byte	Bit
Special data register	SR	Decimal	Word	Word, longword, float
Timer state	TS	Decimal	Byte	Bit
Counter state	CS	Decimal	Byte	Bit
Timer reset	TR	Decimal	Byte	Bit
Counter reset	CR	Decimal	Byte	Bit
Timer	T	Decimal	Word	Word
Counter	C	Decimal	Word	Word
Counter (32 bit)	CX	Decimal	Word	Longword
States	S	Decimal	Byte	Bit
Data register	D	Decimal	Word	Word, longword, float

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values. Enter the starting register value, in octal.

1. Special auxiliary relays minimum start register is 8000.
2. Special data register minimum start register is 8000.

Count

The number of registers to read for the request in decimal. The memory type and slave device determine the maximum number of registers that can be read. The protocol has a maximum of 64 bytes of data.

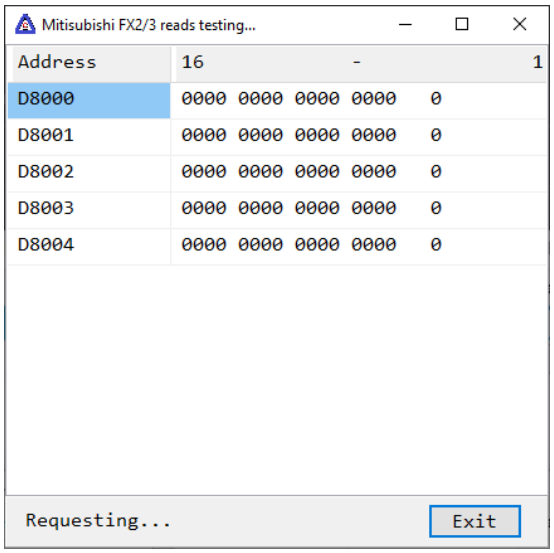
1. For bit data each count value returns 8 bits starting at "start register". The bit address must be byte aligned.
2. For timers and 16 bit counters the count is the number of timers. Each timer/counter is 2 bytes
3. For 32 bit counters the count is the number of counters. Each counter is 4 bytes.
4. When reading data registers and the point data type is longword or float the count must include all bytes. (4 per value)
5. When referencing 32 bit counters apply the start address from the start address for the PLC type. (CX0 = 32 bit start counter in PLC)

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

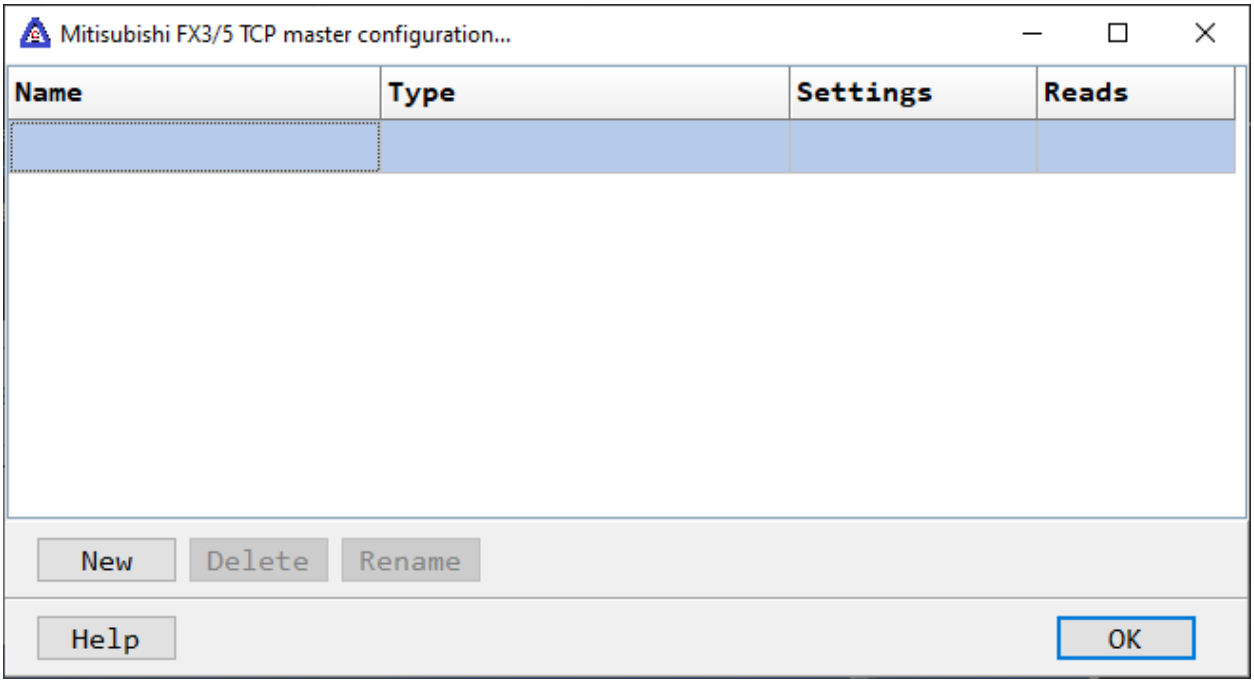
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.
Start register is not octal	The type (X,Y) must be in octal.

Mitsubishi FX3/5 TCP/UDP Ethernet

Each FX3/5 Ethernet master object is listed in the window.
The communications protocol used is MELSEC Communication, A-compatible, 1E frame, subset.
TCP and UDP, Binary and ASCII (FX3 only) are supported.

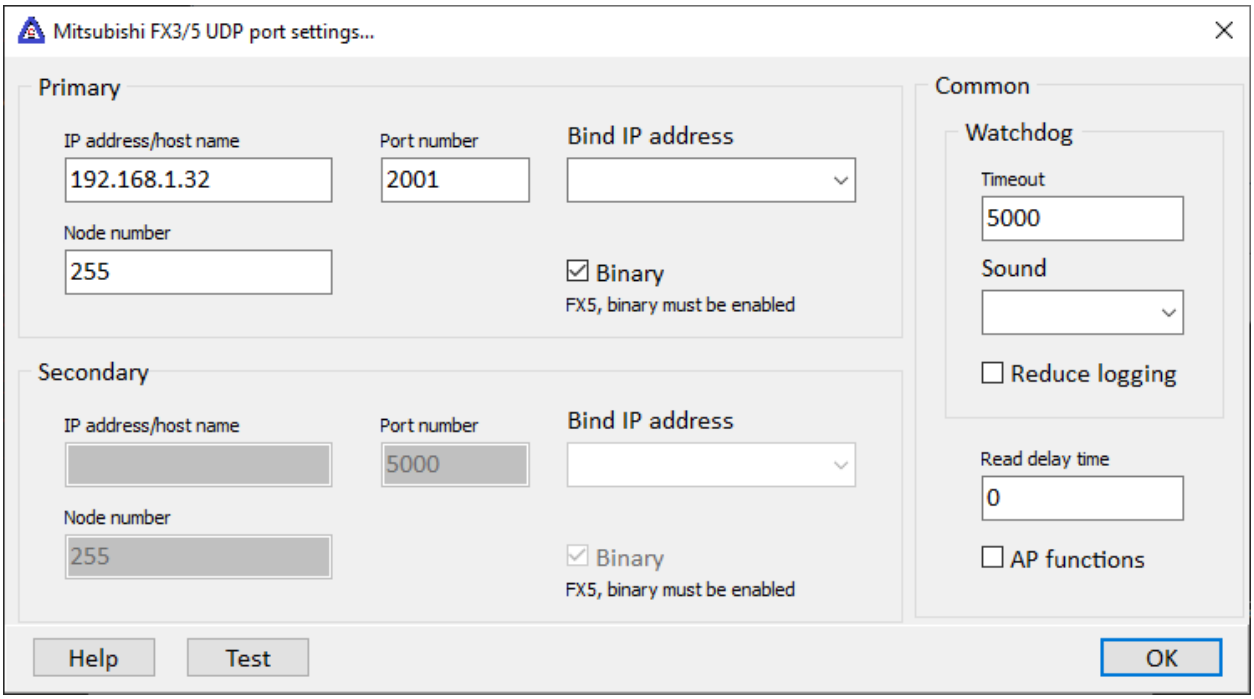


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an FX3/5 Ethernet master object select the "Delete" button.

Settings



Note: Using an FX5, “Binary” mode must be selected.

The port has a primary port and a secondary port. The secondary port is enabled if the “IP address/host name” is supplied and valid. Select the port attributes as is needed.

The “Secondary" provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device.
If the “secondary" is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be placed in the event log and the watchdog sound, if configured, will play. **Note:** Secondary support disabled for initial release.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port, an entry will be placed in the event log and operations will return to the primary port.

Bind IP address See here.

Primary

IP Address	This value can be a IPv4 address or a host name.
Port number	The port number.
Binary	If not checked, ASCII will be used.

Secondary (optional)

IP Address	This value can be a IPv4 address or a host name.
Port number	The port number.
Binary	If not checked, ASCII will be used.

Common

Watchdog

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Sound	If configured, the sound to play when the timer completes.
Reduced logging	If enabled and a watchdog timeout occurs, only one entry will be placed in the event log. The watchdog timeout condition must “reset/clear” and timeout again before another entry is added to the event log.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

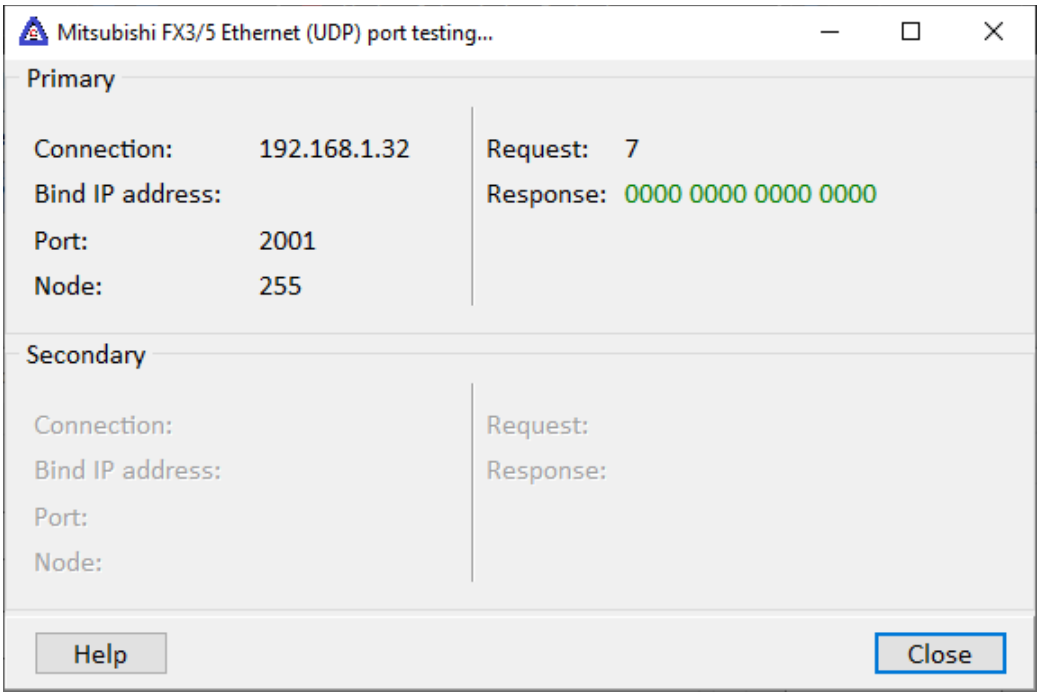
If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

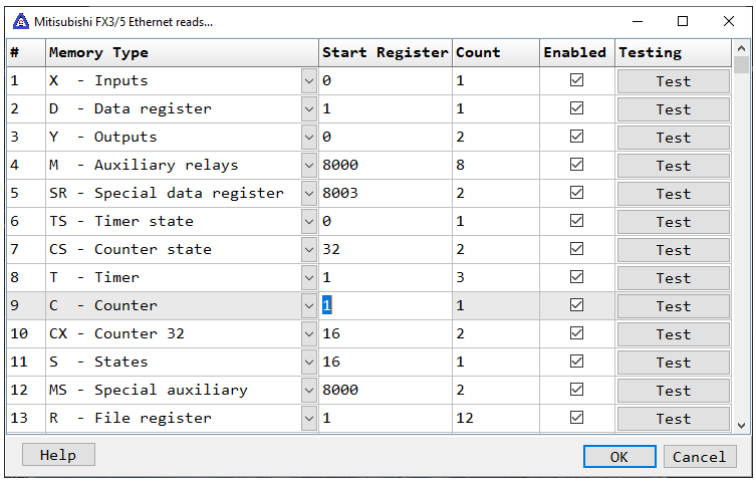
See analog functions.

Test button

The program will attempt to read the first word of input data (X0 – X15) and display the result.



Reads



#	Memory Type	Start Register	Count	Enabled	Testing
1	X - Inputs	0	1	☑	Test
2	D - Data register	1	1	☑	Test
3	Y - Outputs	0	2	☑	Test
4	M - Auxiliary relays	8000	8	☑	Test
5	SR - Special data register	8003	2	☑	Test
6	TS - Timer state	0	1	☑	Test
7	CS - Counter state	32	2	☑	Test
8	T - Timer	1	3	☑	Test
9	C - Counter	1	1	☑	Test
10	CX - Counter 32	16	2	☑	Test
11	S - States	16	1	☑	Test
12	MS - Special auxiliary	8000	2	☑	Test
13	R - File register	1	12	☑	Test

The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for accessing data in the slave.

Address Examples

X0, X105, Y3, Y77	Inputs and outputs are in **octal**.
M2, M90	
MS8000, MS8255	
SR8000	default for this type is word
SR8002	point is configured for longword or float
TS3, TS89	
CS4, CS90	
T4, T67	word access to the timer accumulator
C56, C77	word access to the counter
CX8, CX72	longword access to the counter accumulator
S2, S90	
D1	default for this type is word
D0	point is configured for longword or float
R2, R55	

Notes:

1. Do not use leading zeros in the address string.
2. When modifying a bit in a non-bit register, all other bits in the register are cleared.

Memory type

The supported FX3/5 Ethernet registers.

Register Type	Prefix	Radix	Count	Data Type
Input	X	Octal	Byte	Bit
Output	Y	Octal	Byte	Bit
Auxiliary relays	M	Decimal	Byte	Bit
Special auxiliary relays	MS	Decimal	Byte	Bit
Special data register	SR	Decimal	Word	Word, longword, float
Timer state	TS	Decimal	Byte	Bit
Counter state	CS	Decimal	Byte	Bit
Timer	T	Decimal	Word	Word
Counter	C	Decimal	Word	Word
Counter (32 bit)	CX	Decimal	Word	Longword
States	S	Decimal	Byte	Bit
Data register	D	Decimal	Word	Word, longword, float
File register	R	Decimal	Word	Word, longword, float

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values. Enter the starting register value, in octal.

1. Special auxiliary relays minimum start register is 8000.
2. Special data register minimum start register is 8000.

Count

The number of registers to read for the request in decimal. The memory type and slave device determine the maximum number of registers that can be read. The protocol has a maximum of 64 bytes of data.

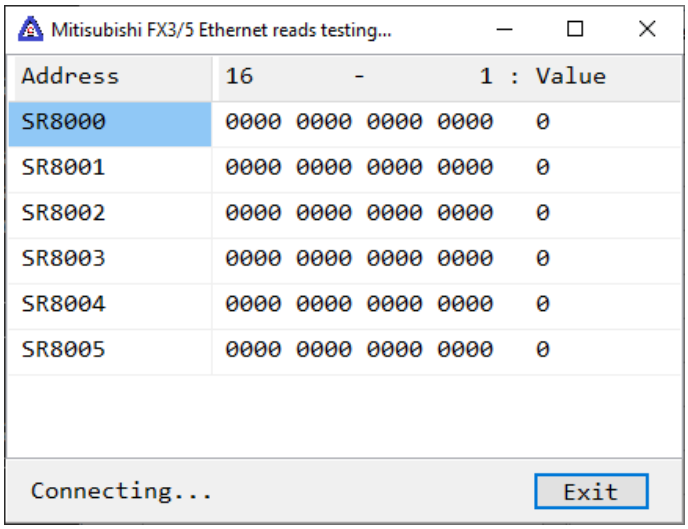
1. For bit data, each count value returns 16 bits starting at "start register". The bit address must be word aligned.
2. For timers and 16 bit counters the count is the number of timers. Each timer/counter is 2 bytes
3. For 32 bit counters the count is the number of counters. Each counter is 4 bytes.
4. When reading data registers and the point data type is longword or float the count must include all bytes. (4 per value)
5. When referencing 32 bit counters apply the start address from the start address for the PLC type. (CX0 = 32 bit start counter in PLC)

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

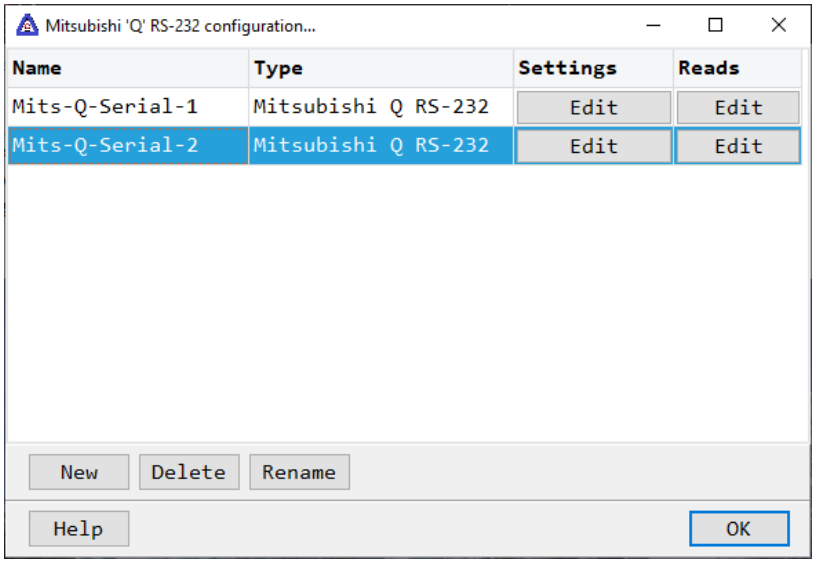
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The count to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.
Start register is not octal	The type (X,Y) must be in octal.

Mitsubishi Q Serial

Each Mitsubishi Q master object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Mitsubishi Q master object select the "Delete" button.

The port uses protocol 3C (ASCII), format 1/4 with checksum enabled/disabled.

We used the following switch settings for the QJ741C24 module, section 4.5.2 of MELSEC System Q Programmable Logic Controllers User's Manual (Basic). **NOTE:** Power must be cycled on the module for changes to the switch settings to become active.

Format 1 with checksum

Switch 1/3 07E6 (hex)

The first byte 07 (hex) is for baud rate of 19200.
The second byte E6 (hex) is for 8 data bits, parity enabled, parity odd, 1 stop bit, sum check code enabled, writing during run, and settings modifications allowed.

Switch 2/4 0001 (hex)

Frame 3C format 1

Switch 5 0000 (hex)

Station 0

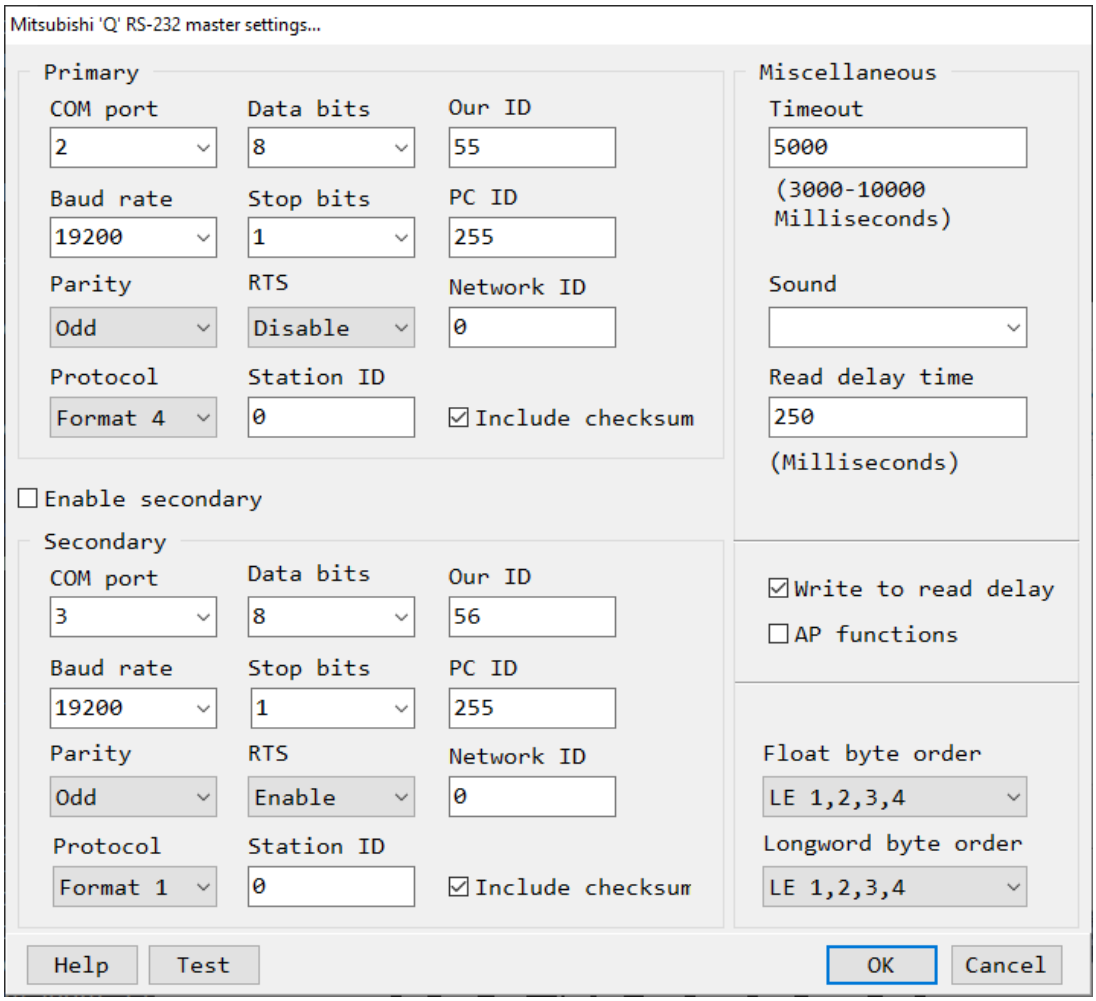
Format 4 with checksum

Switch 2/4 0004 (hex)

Frame 3C format 4

All the other settings are the same as format 1 with checksum, above.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

RTS

See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.
If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not

replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

AP functions

See analog functions.

Float byte order, Longword byte order

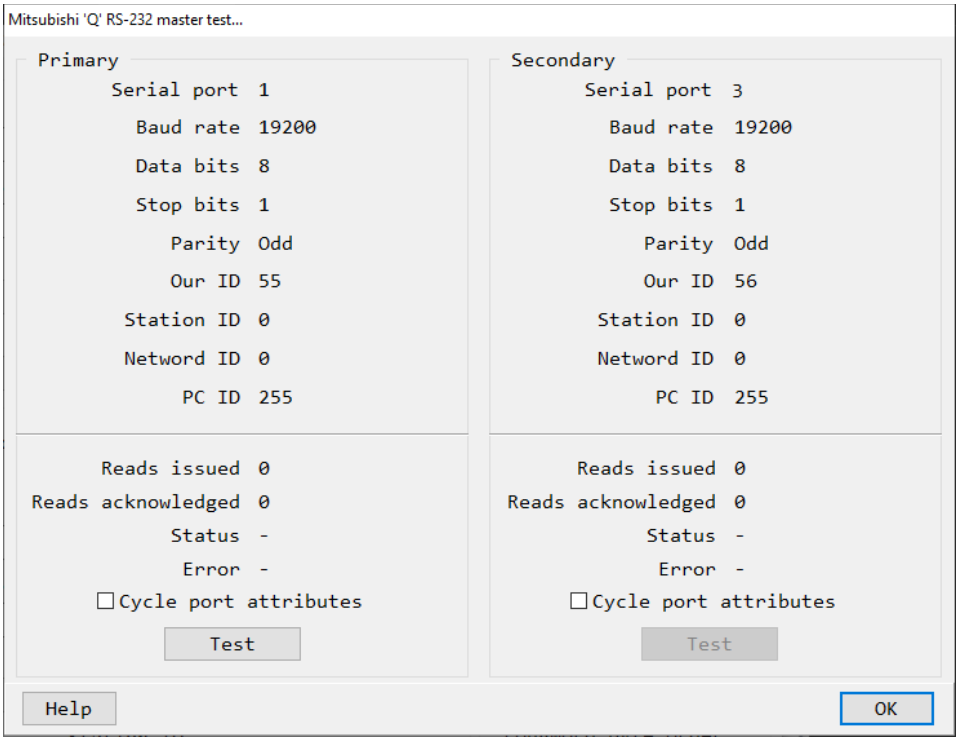
Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The register data received from the slave device is in LO-HI byte order. Regular integer data and floating data values can be in the same data read.

The 4 byte float/longword option is configured on a point level. The integer data is converted into a floating point value at the point level.

When using floating point values the byte order of the slave device might be different than the regular register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

Test button



When the test button is selected the program will send a read request to read one word starting at X0.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads

Mitsubishi 'Q' reads...

#	Memory type	Start register	Count	Enabled	Testing
1	Z - Index register	1	8	☑	Test
2	X - Input	0	2	☑	Test
3	NA - None			☐	Test
4	NA - None			☐	Test
5	NA - None			☐	Test
6	NA - None			☐	Test
7	NA - None			☐	Test
8	NA - None			☐	Test
9	NA - None			☐	Test
10	NA - None			☐	Test
11	NA - None			☐	Test
12	NA - None			☐	Test

Help OK Cancel

The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave.

Address examples

X0, XF1, YA, Y34	Inputs and outputs are in **hexadecimal**.
M2, M90	
M8000, M8255	
TS3, TS89	
CS4, CS90	
TR4, TR8	
CR34, CR56	
T4, T67	word access to the timer accumulator
C56, C77	word access to the counter
S2, S90	
D1	default for this type is word
D0	point is configured for longword or float

Note:

1. Do not use leading zeros in the address string.

Memory type

The supported Mitsubishi Q registers.

Register Type	Prefix	Radix	Data Type
Input	X	Hexadecimal	Boolean
Output	Y	Hexadecimal	Boolean
Internal relay	M	Decimal	Boolean
Latch relay	L	Decimal	Boolean
Link relay	B	Hexadecimal	Boolean
Annunciator	F	Decimal	Boolean
Link special relay	SB	Hexadecimal	Boolean
Edge relay	V	Decimal	Boolean
Step relay	S	Decimal	Boolean
Timer	T	Decimal	Word
Timer reset	TR	Decimal	Boolean
Timer state	TS	Decimal	Boolean
Retentive timer	ST	Decimal	Boolean
Retentive timer reset	RR	Decimal	Word
Retentive timer state	RS	Decimal	Word
Counter	C	Decimal	Word
Counter reset	CR	Decimal	Boolean
Counter state	CS	Decimal	Boolean
Data register	D	Decimal	Word, longword, float
Link register	W	Hexadecimal	Word, longword, float
Link special register	SW	Hexadecimal	Word, longword, float
Special internal relay	SM	Decimal	Boolean
Special internal register	SD	Decimal	Word, longword, float
Index register	Z	Decimal	Word, longword, float
File register	RS	Decimal	Word, longword, float
Direct input	DX	Hexadecimal	Boolean
Direct output	DY	Hexadecimal	Boolean

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for allowable register values. Enter the starting register value in the correct type. See above.

Count

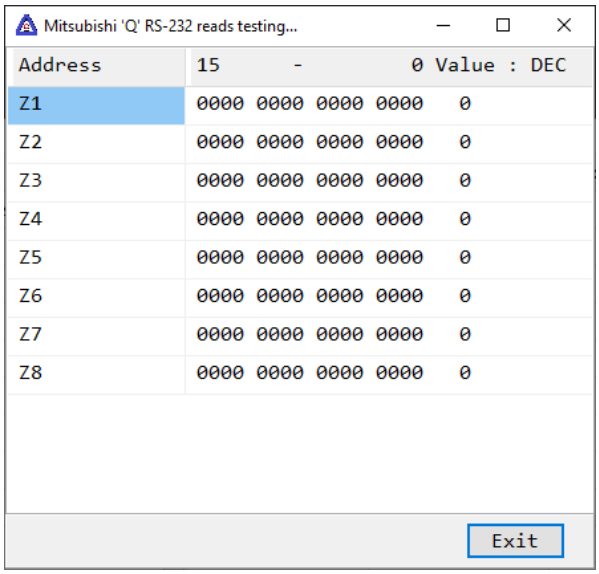
The number of words to read for the request in decimal. The memory type and slave device determine the maximum number of registers that can be read.
1. For Boolean data each count value returns 16 Booleans starting at "start register".
2. When reading registers and the point data type is longword or float the count must include all bytes. (4 per value)
3. Boolean data start register value must be word aligned.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

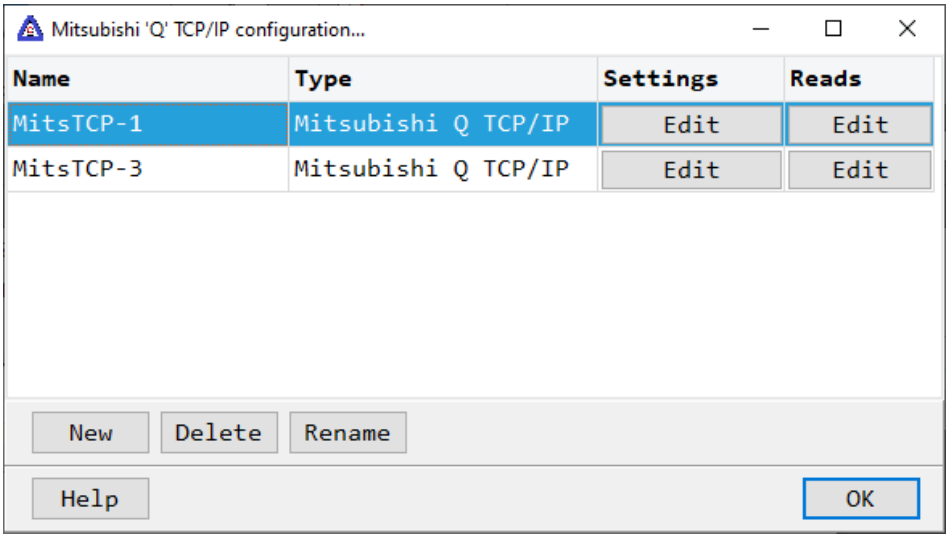
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

MITSUBISHI Q TCP

Each Mitsubishi Q master object is listed in the window.

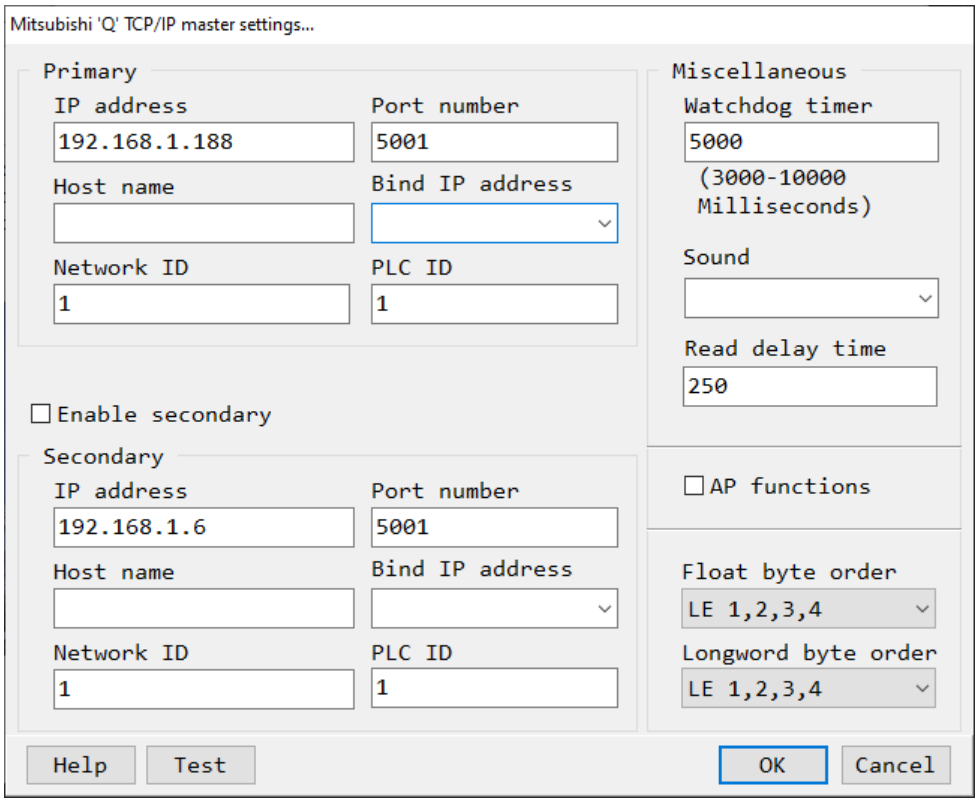


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Mitsubishi Q master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Float Byte Order, Longword Byte Order

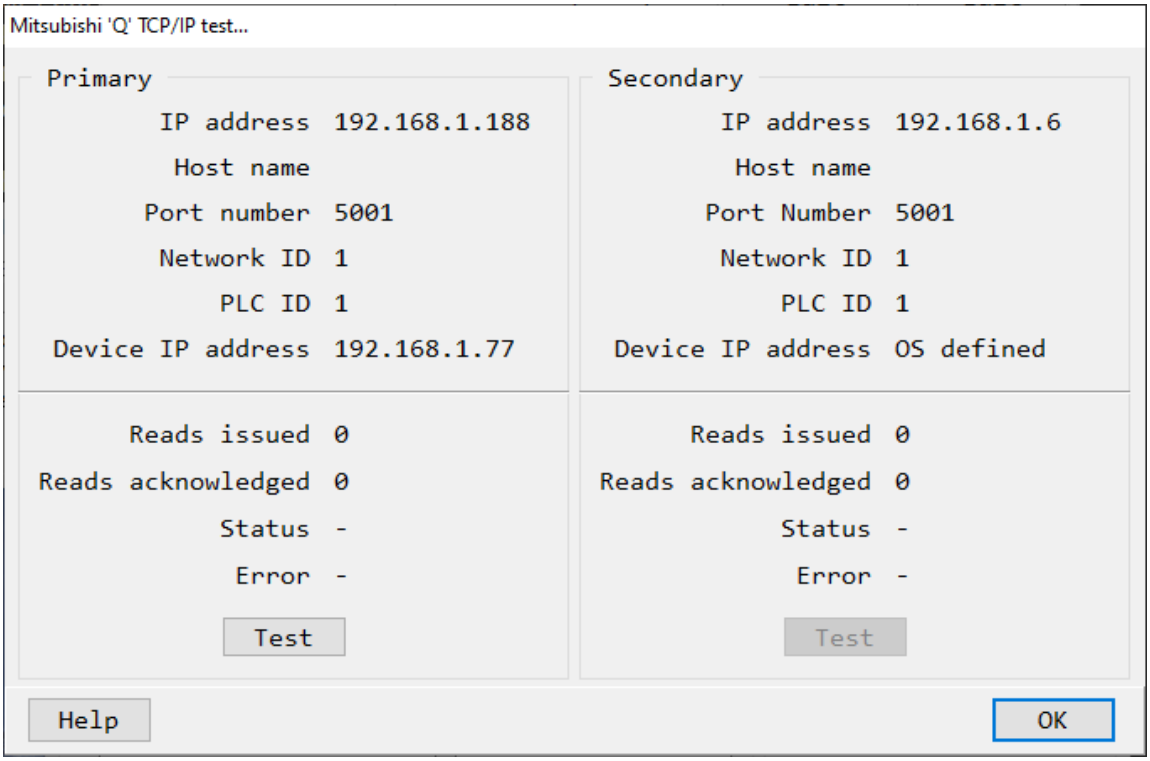
Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The register data received from the slave device is in LO-HI byte order. Regular integer data and floating data values can be in the same data read.

The 4 byte float/longword option is configured on a point level. The integer data is converted into a floating point value at the point level.

When using floating point values the byte order of the slave device might be different from the regular register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

Test button



When the test button is selected the program will send a read request to read one word starting at X0.

Reads

Mitsubishi 'Q' reads...

#	Memory type	Start register	Count	Enabled	Testing
1	RS - File register	0	8	☑	Test
2	X - Input	0	9	☑	Test
3	NA - None			☐	Test
4	NA - None			☐	Test
5	NA - None			☐	Test
6	NA - None			☐	Test
7	NA - None			☐	Test
8	NA - None			☐	Test
9	NA - None			☐	Test
10	NA - None			☐	Test
11	NA - None			☐	Test
12	NA - None			☐	Test

Help OK Cancel

The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave.

Address Examples

X0, XF1, YA, Y34	Inputs and outputs are in **hexadecimal**.
M2, M90	
M8000, M8255	
TS3, TS89	
CS4, CS90	
TR4, TR8	
CR34, CR56	
T4, T67	word access to the timer accumulator
C56, C77	word access to the counter
S2, S90	
D1	default for this type is word
D0	point is configured for longword or float

Note:

1. Do not use leading zeros in the address string.

Memory type

The supported Mitsubishi Q registers.

Register Type	Prefix	Radix	Data Type
Input	X	Hexadecimal	Boolean
Output	Y	Hexadecimal	Boolean
Internal relay	M	Decimal	Boolean
Latch relay	L	Decimal	Boolean
Link relay	B	Hexadecimal	Boolean
Annunciator	F	Decimal	Boolean
Link special relay	SB	Hexadecimal	Boolean
Edge relay	V	Decimal	Boolean
Step relay	S	Decimal	Boolean
Timer	T	Decimal	Word
Timer reset	TR	Decimal	Boolean
Timer state	TS	Decimal	Boolean
Retentive timer	ST	Decimal	Boolean
Retentive timer reset	RR	Decimal	Word
Retentive timer state	RS	Decimal	Word
Counter	C	Decimal	Word
Counter reset	CR	Decimal	Boolean
Counter state	CS	Decimal	Boolean
Data register	D	Decimal	Word, longword, float
Link register	W	Hexadecimal	Word, longword, float
Link special register	SW	Hexadecimal	Word, longword, float
Special internal relay	SM	Decimal	Boolean
Special internal register	SD	Decimal	Word, longword, float
Index register	Z	Decimal	Word, longword, float
File register	RS	Decimal	Word, longword, float
Direct input	DX	Hexadecimal	Boolean
Direct output	DY	Hexadecimal	Boolean

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for allowable register values. Enter the starting register value in the correct type. See above.

Count

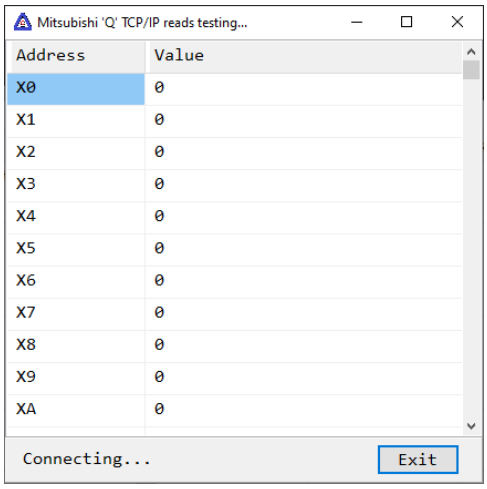
The number of words to read for the request in decimal. The memory type and slave device determine the maximum number of registers that can be read.
1. For Boolean data each count value returns 16 Booleans starting at "start register".
2. When reading registers and the point data type is longword or float the count must include all bytes. (4 per value)
3. Boolean data start register value must be word aligned.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

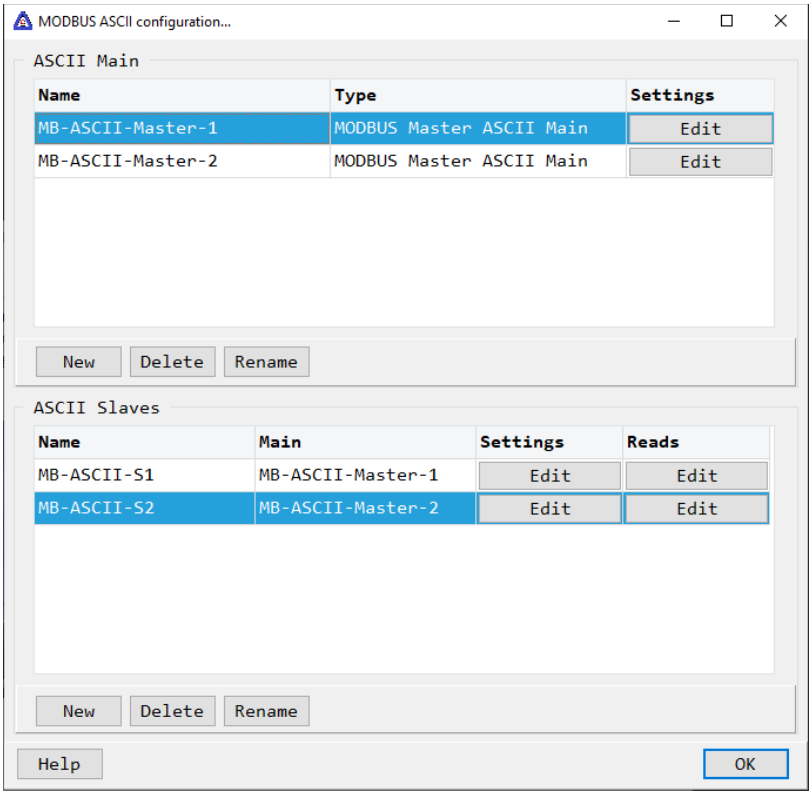
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

MODBUS Master ASCII Serial 232/485

Each MODBUS ASCII master object controls one serial port and can have one or many slave objects. The slaves are configured to address one device.



Note: RS-485 is sensitive to improper wiring and/or terminations. Unpowered units can cause data echoes and other reliability issues. Please follow all RS-485 wiring guidelines.

MODBUS ASCII serial main

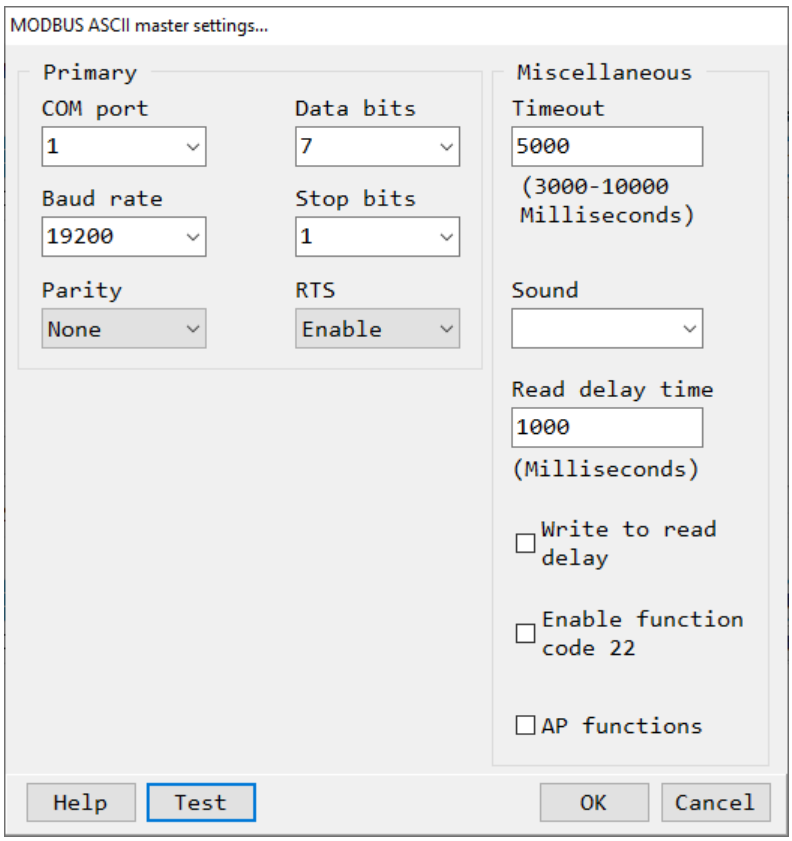
Each MODBUS ASCII master object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS ASCII master object select the "Delete" button.

Settings



Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Note: The default data bit count is 7. Some devices do not support 8 bit data bytes for MODBUS ASCII.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

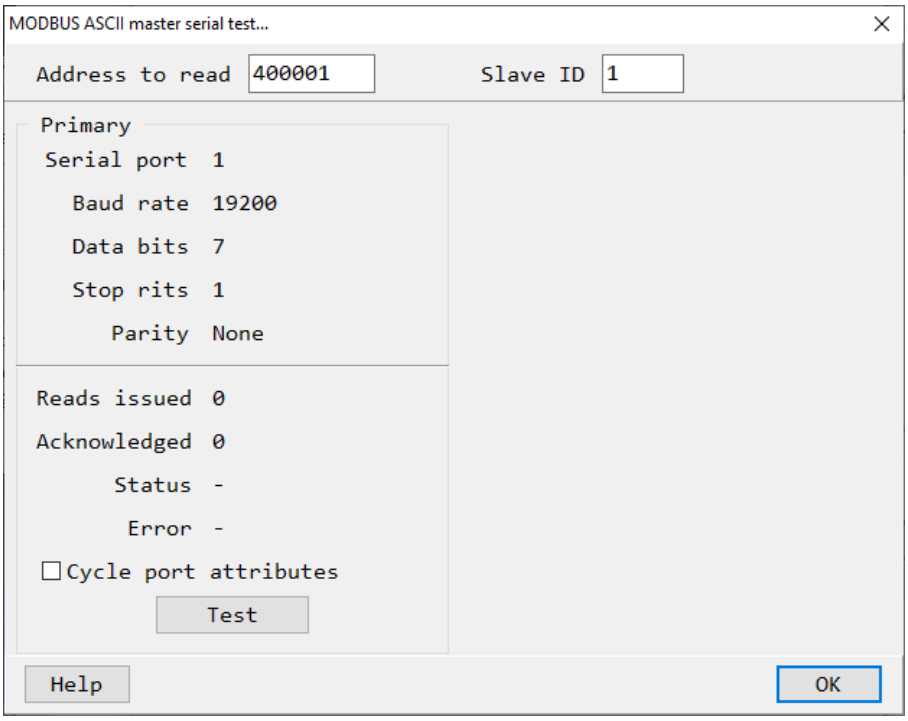
Enable function code 22

When enabled and a bit write to a holding register is requested, function code 22 (Mask write 4X register) will be used to set or clear the bit. If not enabled, function code 6 (Preset single register) will be used. **Note:** Not all devices support function code 22.

AP functions

See analog functions.

Test button



When the test button is selected the program will attempt to read one word of data from the device at the address entered. If the address is a coil (0XXXXX) or an input (1XXXXX), an attempt to read 16 elements (one word) is performed.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

If an input or coil address is entered the program will attempt to read that address plus the following 15 to populate one word of data.

MODBUS ASCII slaves

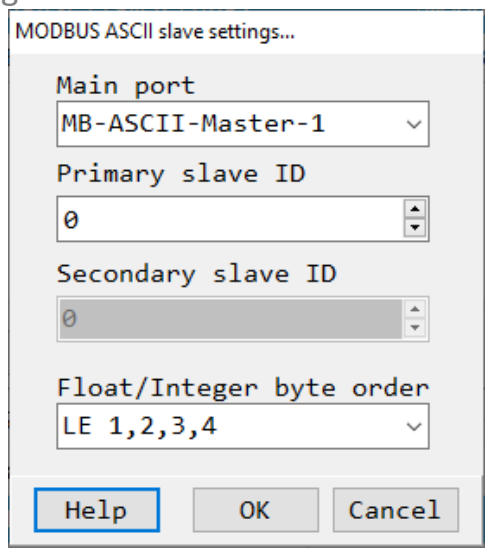
Each MODBUS ASCII slave object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS ASCII slave object select the "Delete" button.

Settings



Main port

This is the master RS-485 port the slave is linked to. A port must be linked to a master for runtime operations.

Primary station number
Secondary station number (Future)

This is the station number of the slave device for the primary and secondary. (1 - 255)

Float/Integer byte order

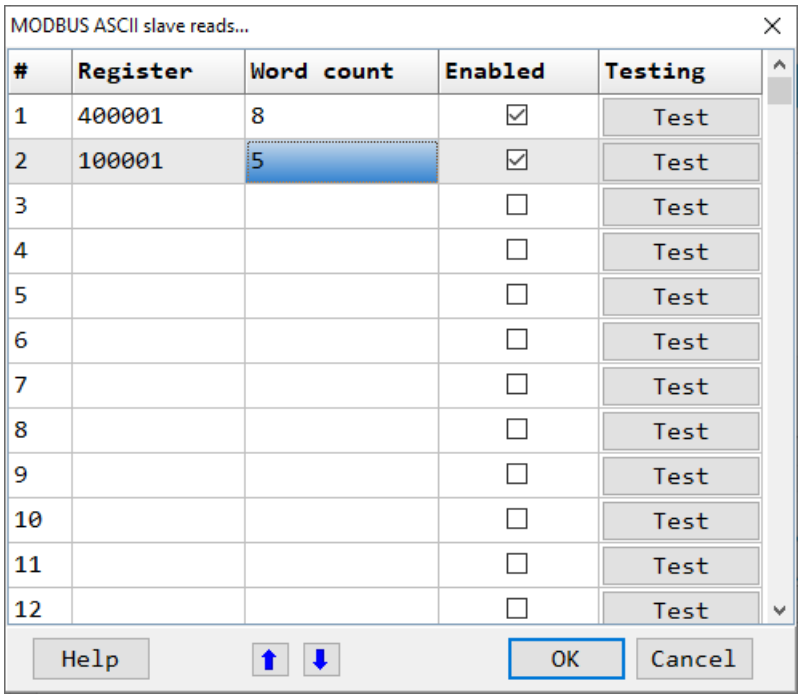
Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The MODBUS holding register data received from the slave device is in HI-LO byte order.

The data bytes are swapped and the result is stored in native format (LO-HI). Regular integer data and floating data values can be in the same data read.

The 4 byte float option is configured on a point level. The integer data is converted into a floating point value at the point level. Because the data bytes have already been swapped the conversion to a floating point value uses method index 3. If the byte swap had not been done at the table read level method index 4 would be the default.
When using floating point values the byte order of the slave device might be different than the regular holding register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the 6 digit MODBUS register address numbering system.

Register

The address must be a valid MODBUS address and not refer to a bit in a holding or input register.

Any of the following addresses are valid:
000001 - 065535
100001 - 165535
300001 - 365535
400001 - 465535

Invalid address would be any that refer to a bit in a word

All of the following addresses are invalid for configuration of the reads. (All are legal for point source addressing.)

400001/01 or 300001/01

Word count

Each read can request up to 125 words of data. For input and holding registers that would be 125 registers.

For coils and inputs that would be 125 * 16 (2000) single bits of data.

For example:

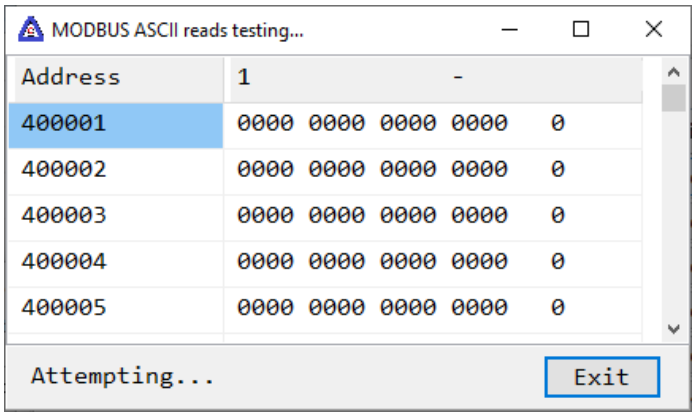
Register	**Count**	**Result**
400001	1	1 Holding register
303521	4	4 Input register
100001	1	16 Inputs
000001	2	32 Coils
000016	125	2000 Coils

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

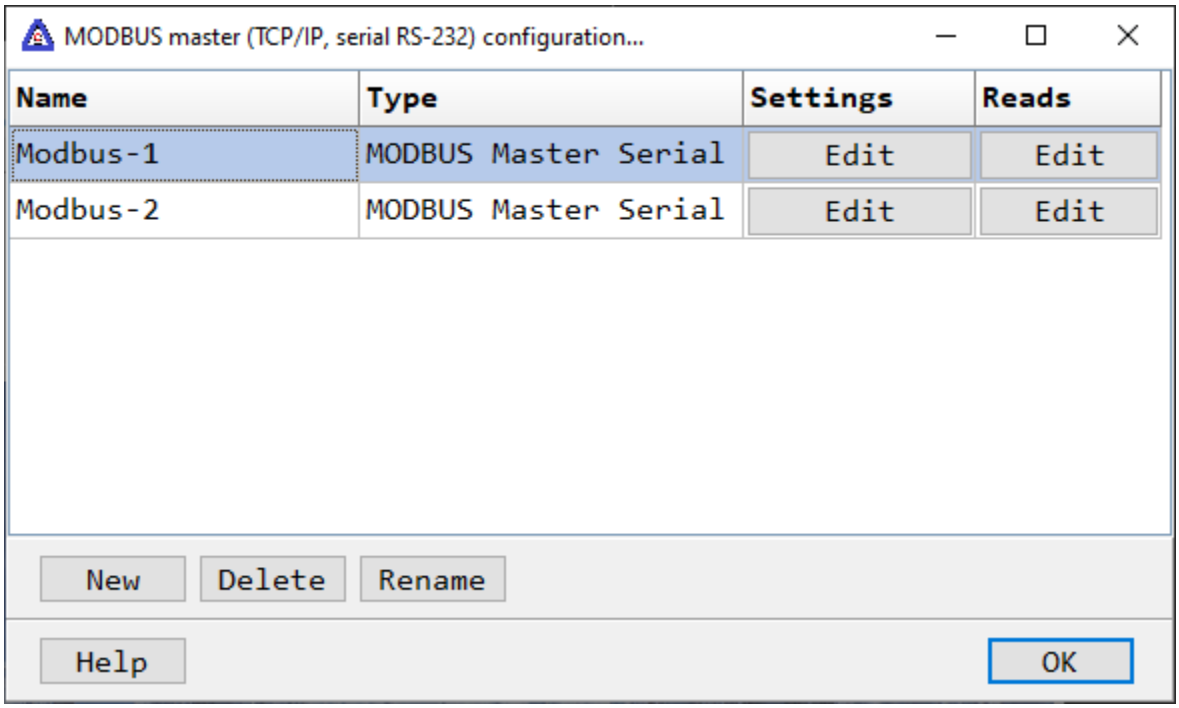
Error messages

No register address	The value in the field is not a MODBUS address.
Invalid count	The count must be between 1 and 125.
Exceeds 65535	The register plus the count exceeds the maximum address range.
Invalid MODBUS address	The value in the field is an invalid MODBUS address.
Bit field not allowed	The value in the field references a bit of a register.

MODBUS Master RTU Serial 232

MODBUS Master TCP

Each MODBUS master object is listed in the window.

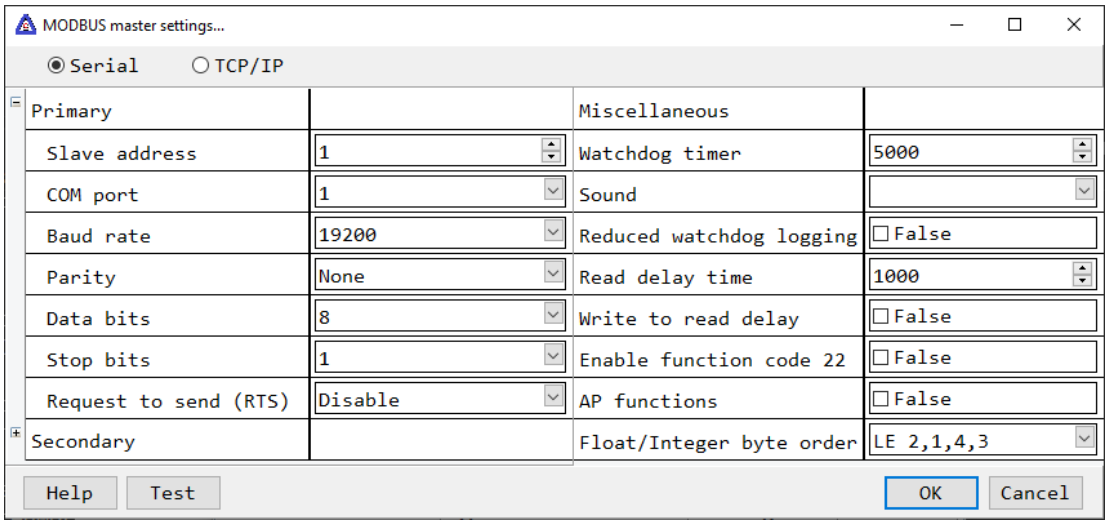


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS master object select the "Delete" button.

Settings



Select either the serial or TCP radio button at the top of the window. Note: Switching from one to the other does not save the settings.

Serial

The serial port has a primary port and if enabled a secondary port. Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Enable secondary

The "Enable secondary" checkbox provides for a second com port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary com port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

TCP

The TCP port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second path to be used as a hot backup. When in run mode the primary configuration is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary configuration. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary configuration, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary configuration an entry will be made in the event log and operations will return to the primary port.

If the computer has two network interface cards (NIC) the program will use the first one detected for the primary and the second one detected for the secondary. If only one NIC is detected the program will use it for the primary and the secondary.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

Enable function code 22

When enabled and a bit write to a holding register is requested, function code 22 (Mask write 4X register) will be used to set or clear the bit. If not enabled, function code 6 (Preset single register) will be used. Note: Not all devices support function code 22.

AP functions

See analog functions.

Float/Integer byte order

Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The MODBUS holding register data received from the slave device is in HI-LO byte order.

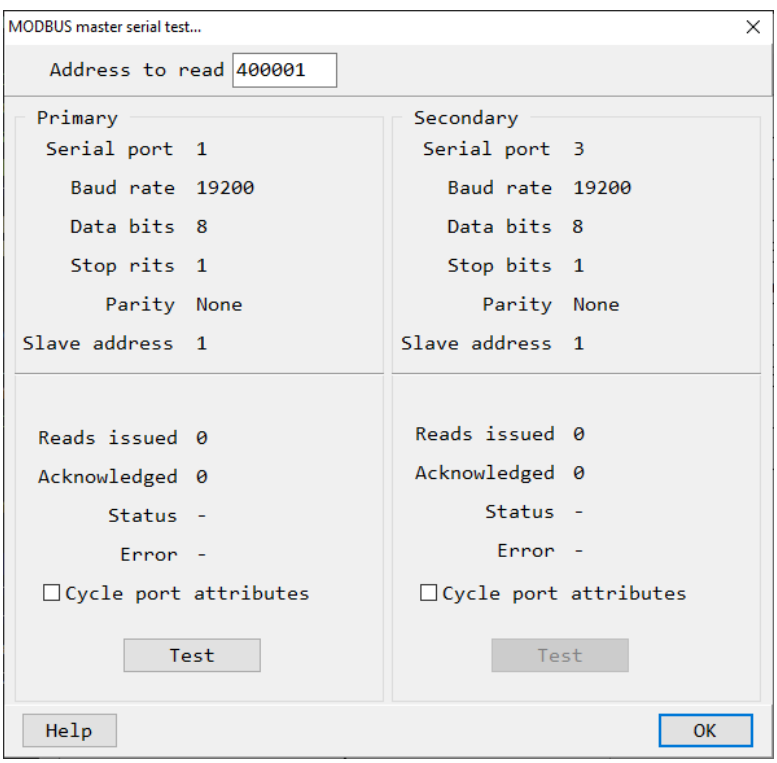
The data bytes are swapped and the result is stored in native format (LO-HI). Regular integer data and floating data values can be in the same data read.

The 4 byte float option is configured on a point level. The integer data is converted into a floating point value at the point level. Because the data bytes have already been swapped the conversion to a floating point value uses method index 3. If the byte swap had not been done at the table read level method index 4 would be the default.

Note: The byte order for a Honeywell HC900 Process Controller is (2), BE 3,4,1,2

Test button

Serial Testing



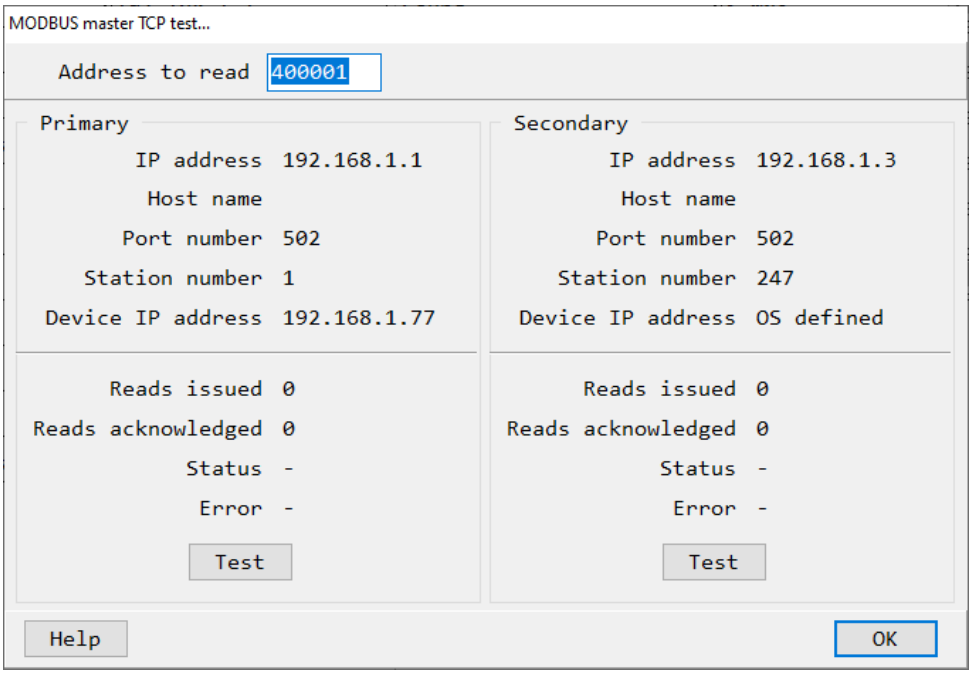
When the test button is selected the program will attempt to read one word of data from the device at the address entered. If the address is a coil (0XXXXX) or an input (1XXXXX), an attempt to read 16 elements (one word) is performed.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

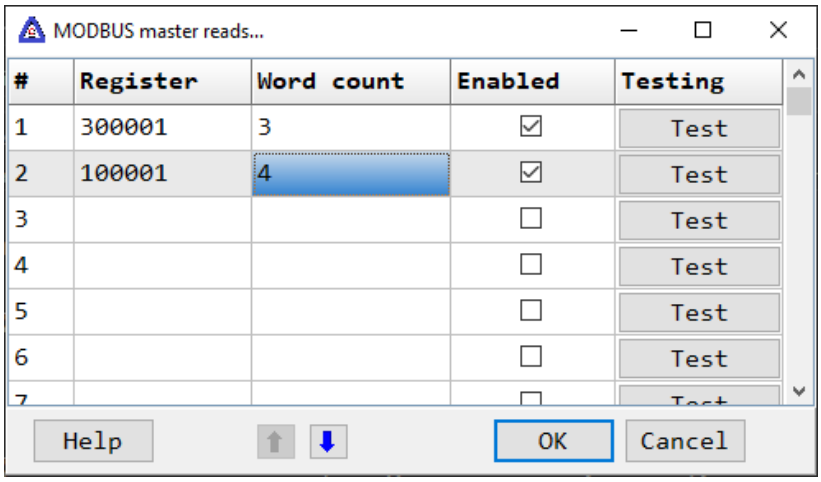
If an input or coil address is entered the program will attempt to read that address plus the following 15 to populate one word of data.

TCP Testing



When the test button is selected the program will attempt to read one word of data from the device at the address entered. If the address is a coil (0XXXXX) or an input (1XXXXX), an attempt to read 16 elements (one word) is performed.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the 6 digit MODBUS register address numbering system.

Register

The address must be a valid MODBUS address and not refer to a bit in a holding or input register.

Any of the following addresses are valid:
000001 - 065535
100001 - 165535
300001 - 365535
400001 - 465535

Invalid address would be any that refer to a bit in a word

All of the following addresses are invalid for configuration of the reads. (All are legal for point source addressing.)

400001/01 or 300001/01

Word count

Each read can request up to 125 words of data. For input and holding registers that would be 125 registers.

For coils and inputs that would be 125 * 16 (2000) single bits of data.

For example:

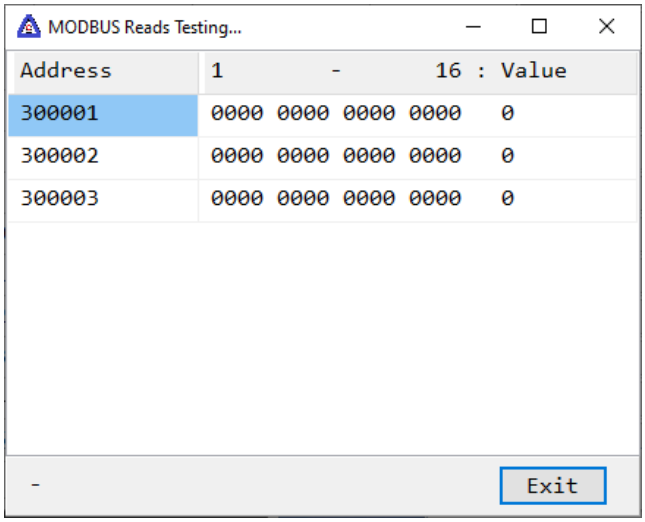
Register	Count	Result
400001	1	1 Holding register
303521	4	4 Input register
100001	1	16 Inputs
000001	2	32 Coils
000016	125	2000 Coils

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



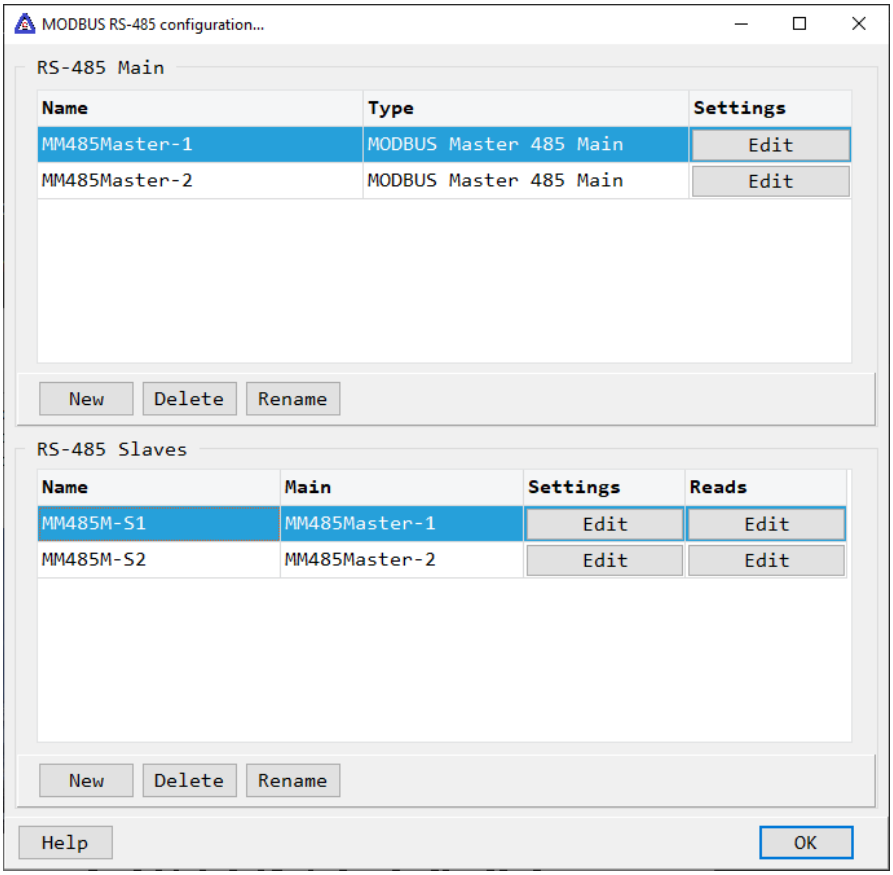
When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Error messages

No register address	The value in the field is not a MODBUS address.
Invalid count	The count must be between 1 and 125.
Exceeds 65535	The register plus the count exceeds the maximum address range.
Invalid MODBUS address	The value in the field is an invalid MODBUS address.
Bit field not allowed	The value in the field references a bit of a register.

MODBUS Masters RTU Serial 485

Each MODBUS RS-485 master object controls one serial port and can have one or many slave objects. The slaves are configured to address one device.



Note: RS-485 is sensitive to improper wiring and/or terminations. Unpowered units can cause data echoes and other reliability issues. Please follow all RS-485 wiring guidelines.

MODBUS RS-485 serial main

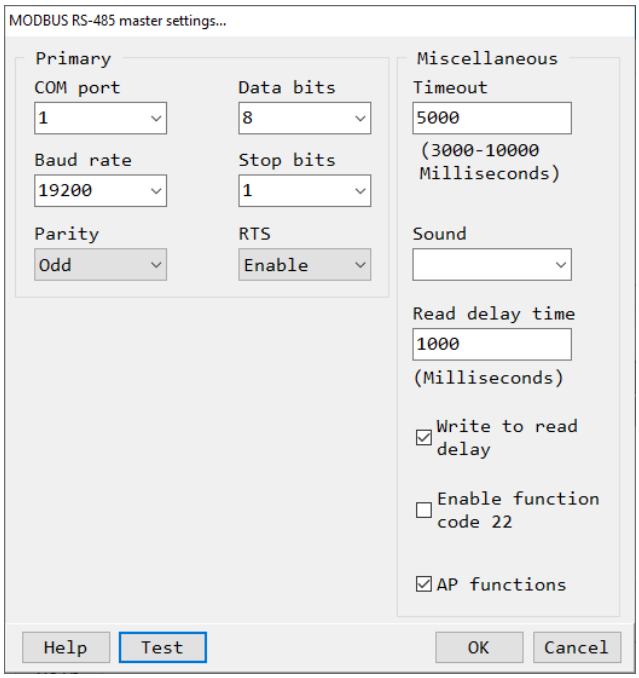
Each MODBUS RS-485 master object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS RS-485 master object select the "Delete" button.

Settings



Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

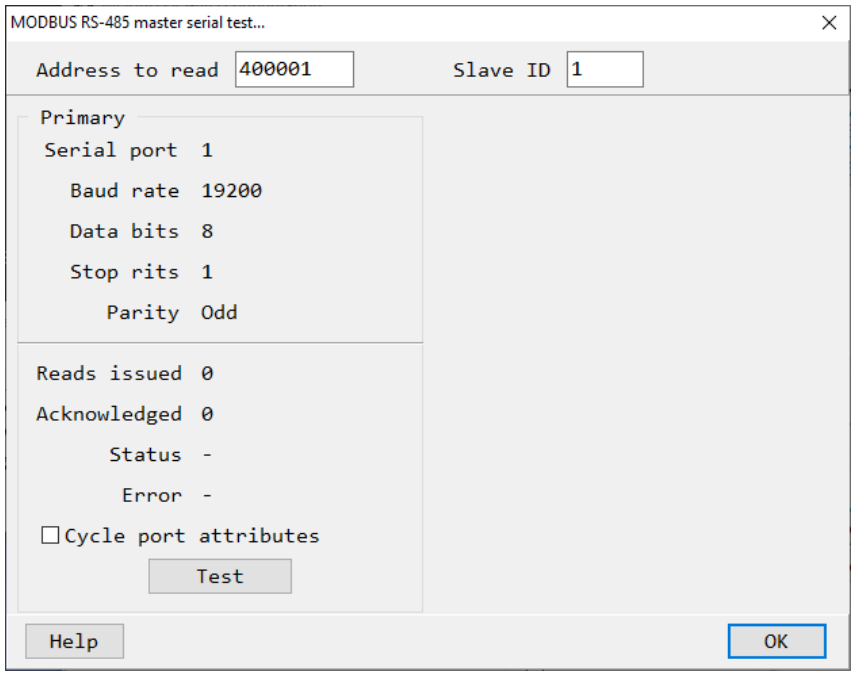
Enable function code 22

When enabled and a bit write to a holding register is requested, function code 22 (Mask write 4X register) will be used to set or clear the bit. If not enabled, function code 6 (Preset single register) will be used. Note: Not all devices support function code 22.

AP functions

See analog functions.

Test button



When the test button is selected the program will attempt to read one word of data from the device at the address entered. If the address is a coil (0XXXXX) or an input (1XXXXX), an attempt to read 16 elements (one word) is performed.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

If an input or coil address is entered the program will attempt to read that address plus the following 15 to populate one word of data.

MODBUS RS-485 slaves

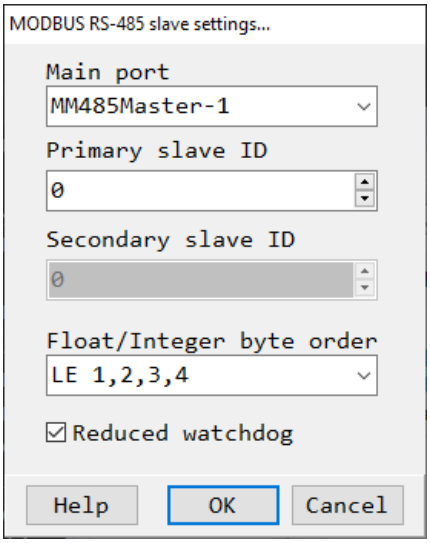
Each MODBUS RS-485 slave object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS RS-485 slave object select the "Delete" button.

Settings



Main port

This is the master RS-485 port the slave is linked to. A port must be linked to a master for runtime operations.

Primary station number
Secondary station number (Future)

This is the station number of the slave device for the primary and secondary. (1 - 255)

Float/Integer byte order

Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The MODBUS holding register data received from the slave device is in HI-LO byte order.

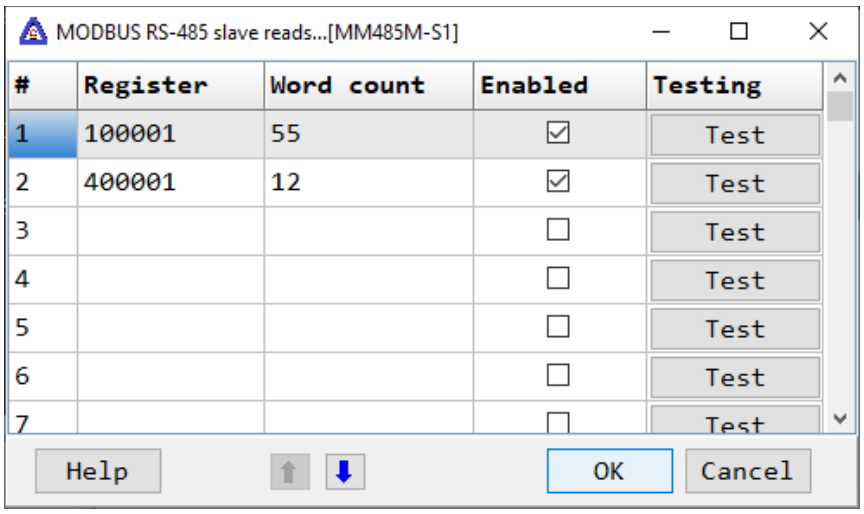
The data bytes are swapped and the result is stored in native format (LO-HI). Regular integer data and floating data values can be in the same data read.

The 4 byte float option is configured on a point level. The integer data is converted into a floating point value at the point level. Because the data bytes have already been swapped the conversion to a floating point value uses method index 3. If the byte swap had not been done at the table read level method index 4 would be the default.
When using floating point values the byte order of the slave device might be different than the regular holding register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

Reduced watchdog logging

When enabled and a watchdog timeout occurs, a single entry will be placed in the event log and the sound will be queued (if configured). When the slave device responds, after a watchdog timeout, a single entry will be placed in the event log. The watchdog counter will continue to count each time the watchdog timer expires.

Reads



#	Register	Word count	Enabled	Testing
1	100001	55	☑	Test
2	400001	12	☑	Test
3			☐	Test
4			☐	Test
5			☐	Test
6			☐	Test
7			☐	Test

The address ranges shown may or may not be present in the slave device.

The HMI uses the 6 digit MODBUS register address numbering system.

Register

The address must be a valid MODBUS address and not refer to a bit in a holding or input register.

Any of the following addresses are valid:
000001 - 065535
100001 - 165535
300001 - 365535
400001 - 465535

Invalid address would be any that refer to a bit in a word

All of the following addresses are invalid for configuration of the reads. (All are legal for point source addressing.)

400001/01 or 300001/01

Word count

Each read can request up to 125 words of data. For input and holding registers that would be 125 registers.

For coils and inputs that would be 125 * 16 (2000) single bits of data.

For example:

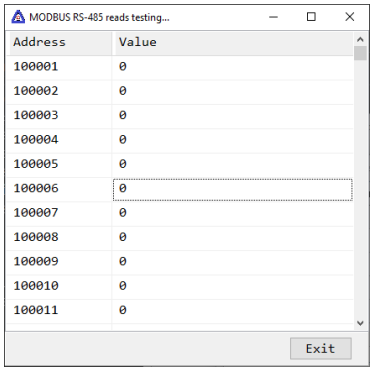
Register	Count	Result
400001	1	1 Holding register
303521	4	4 Input register
100001	1	16 Inputs
000001	2	32 Coils
000016	125	2000 Coils

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Error messages

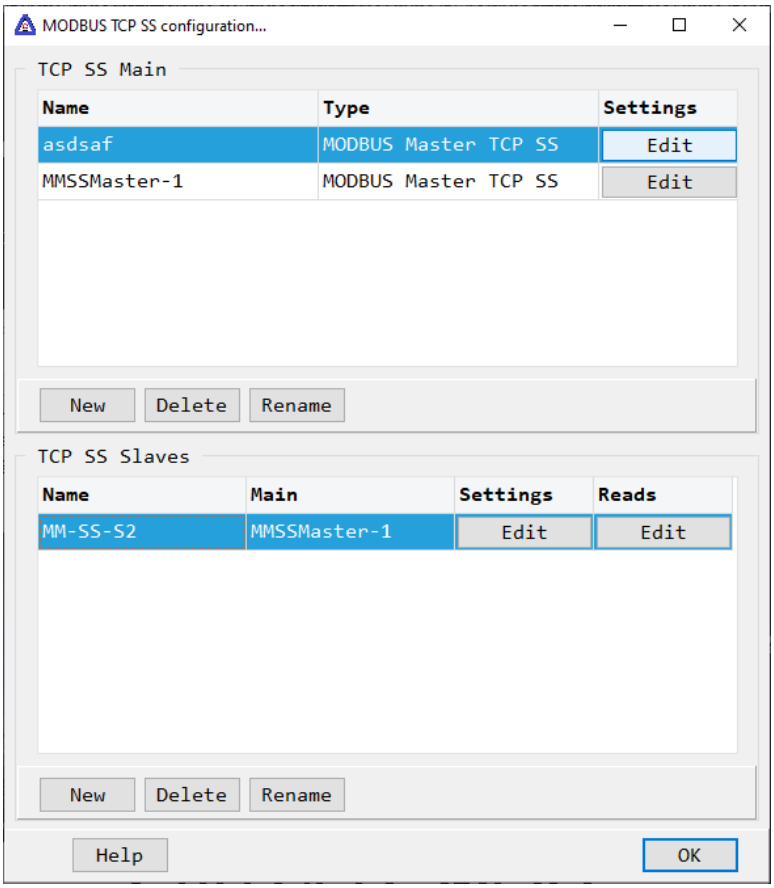
No register address	The value in the field is not a MODBUS address.
Invalid count	The count must be between 1 and 125.
Exceeds 65535	The register plus the count exceeds the maximum address range.
Invalid MODBUS address	The value in the field is an invalid MODBUS address.
Bit field not allowed	The value in the field references a bit of a register.

MODBUS MASTERS TCP (SINGLE SOCKET)

This port type should be used when a single socket is needed to access 2 or more slaves on the same IP address. It is slower in data collection because it accesses the slave devices one at a time reading the data reads configured and moving to the next configured slave.

The regular MODMMSSBUS TCP port type accesses only one slave device. If using 2 or more slaves and each slave has a unique IP address, use the regular MODBUS TCP port.

Each MODBUS TCP SS master object controls one port and can have one or many slave objects. The slaves are configured to address one device.



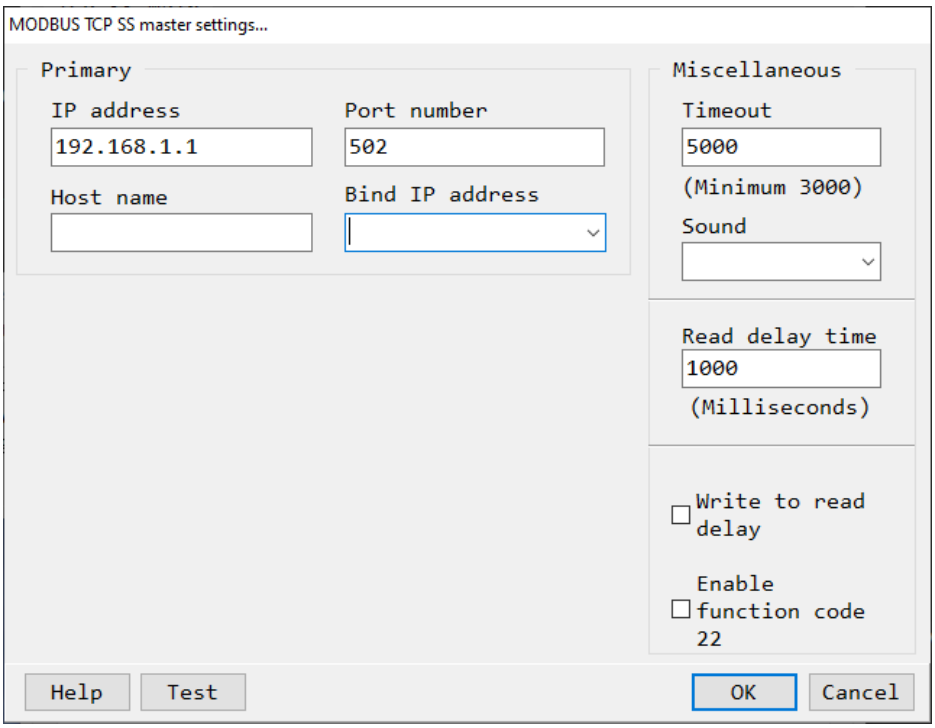
Each MODBUS TCP SS master object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS TCP SS master object select the "Delete" button.

Settings



If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sounds delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

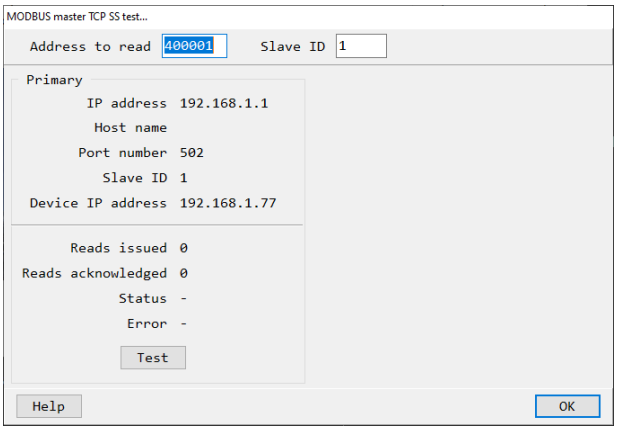
Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

Enable function code 22

When enabled and a bit write to a holding register is requested, function code 22 (Mask write 4X register) will be used to set or clear the bit. If not enabled, function code 6 (Preset single register) will be used. Note: Not all devices support function code 22.

Test button



When the test button is selected the program will attempt to read one word of data from the device at the address entered. If the address is a coil (0XXXXX) or an input (1XXXXX), an attempt to read 16 elements (one word) is performed.

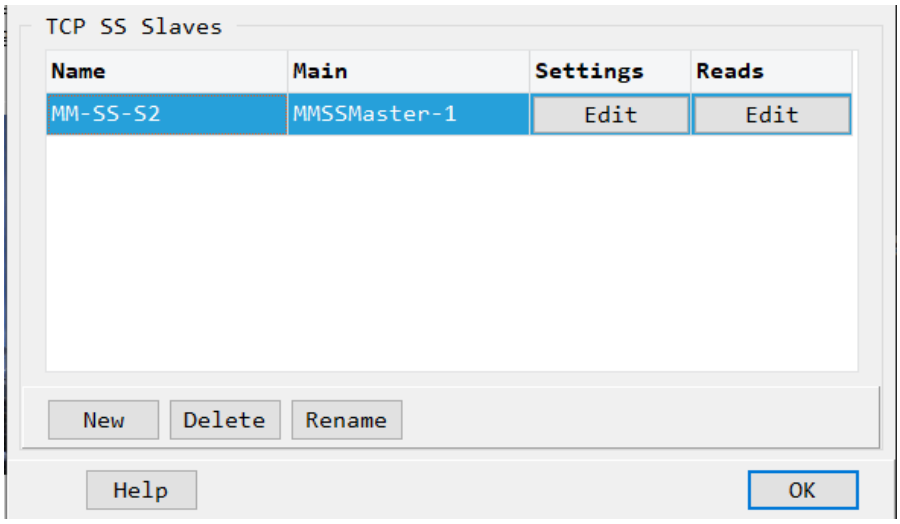
MODBUS TCP SS Slaves

Each MODBUS TCP SS slave object is listed in the window.

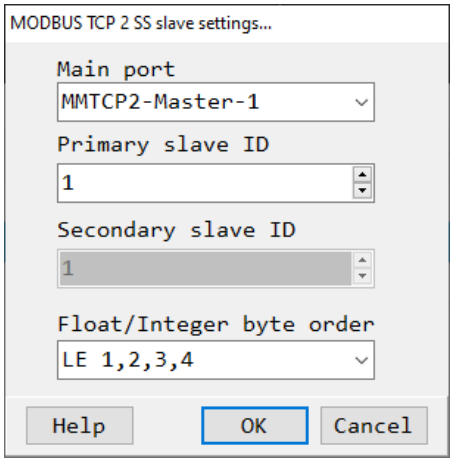
To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS TCP SS slave object select the "Delete" button.



Settings



Main port

This is the master TCP SS port the slave is linked to. A port must be linked to a master for runtime operations.

Primary station number
Secondary station number (Future)

This is the station number of the slave device for the primary and secondary. (1 - 255)

Float/Integer byte order

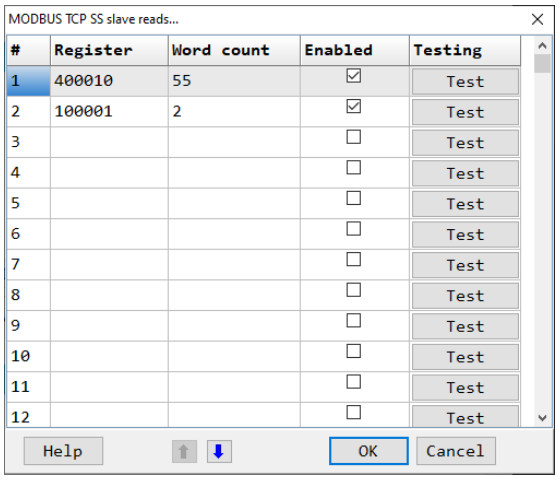
Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The MODBUS holding register data received from the slave device is in HI-LO byte order.

The data bytes are swapped and the result is stored in native format (LO-HI). Regular integer data and floating data values can be in the same data read.

The 4 byte float option is configured on a point level. The integer data is converted into a floating point value at the point level. Because the data bytes have already been swapped the conversion to a floating point value uses method index 3. If the byte swap had not been done at the table read level method index 4 would be the default.
When using floating point values the byte order of the slave device might be different than the regular holding register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the 6 digit MODBUS register address numbering system.

Register

The address must be a valid MODBUS address and not refer to a bit in a holding or input register.

Any of the following addresses are valid:
000001 - 065535
100001 - 165535
300001 - 365535
400001 - 465535

Invalid address would be any that refer to a bit in a word

All of the following addresses are invalid for configuration of the reads. (All are legal for point source addressing.)

400001/01 or 300001/01

Word count

Each read can request up to 125 words of data. For input and holding registers that would be 125 registers.

For coils and inputs that would be 125 * 16 (2000) single bits of data.

For example:

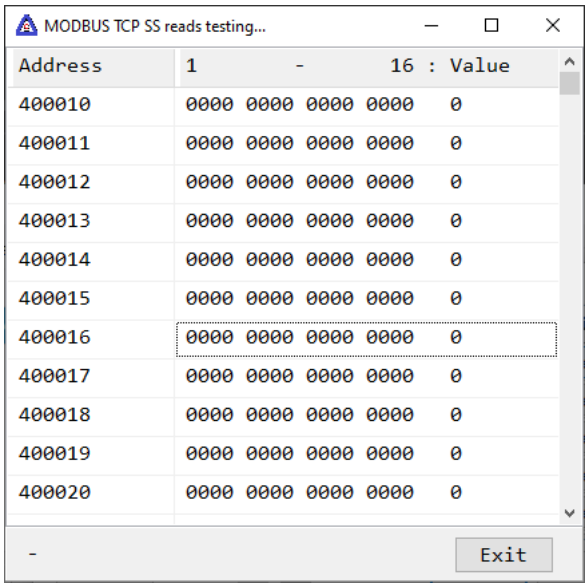
Register	**Count**	**Result**
400001	1	1 Holding register
303521	4	4 Input register
100001	1	16 Inputs
000001	2	32 Coils
000016	125	2000 Coils

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing.

Error messages

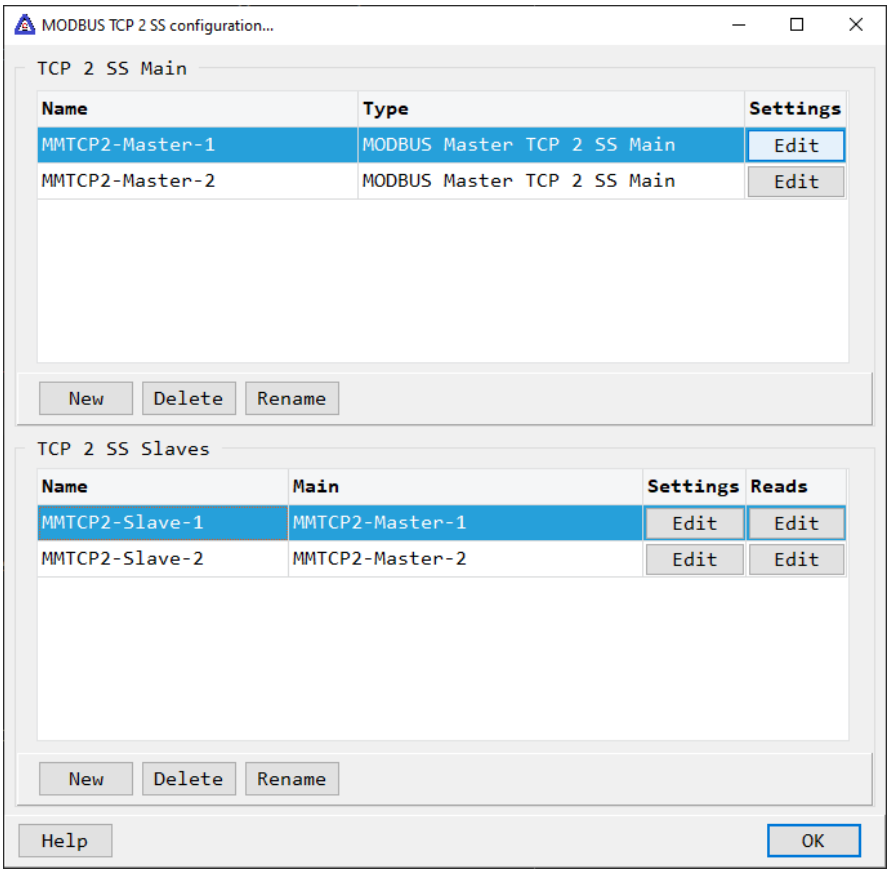
No register address	The value in the field is not a MODBUS address.
Invalid count	The count must be between 1 and 125.
Exceeds 65535	The register plus the count exceeds the maximum address range.
Invalid MODBUS address	The value in the field is an invalid MODBUS address.
Bit field not allowed	The value in the field references a bit of a register.

MODBUS Masters TCP 2 (Single socket)

This port type should be used when a single socket is needed to access 2 or more slaves on the same IP address and the slave ID needs to be two (2) bytes. It is slower in data collection because it accesses the slave devices one at a time reading the data reads configured and moving to the next configured slave.

The regular MODBUS TCP port type accesses only one slave device. If using two (2) or more slaves and slave one has a unique IP address use the regular MODBUS TCP port.

Each MODBUS TCP 2 SS master object controls one port and can have one or many slave objects. The slaves are configured to address one device.

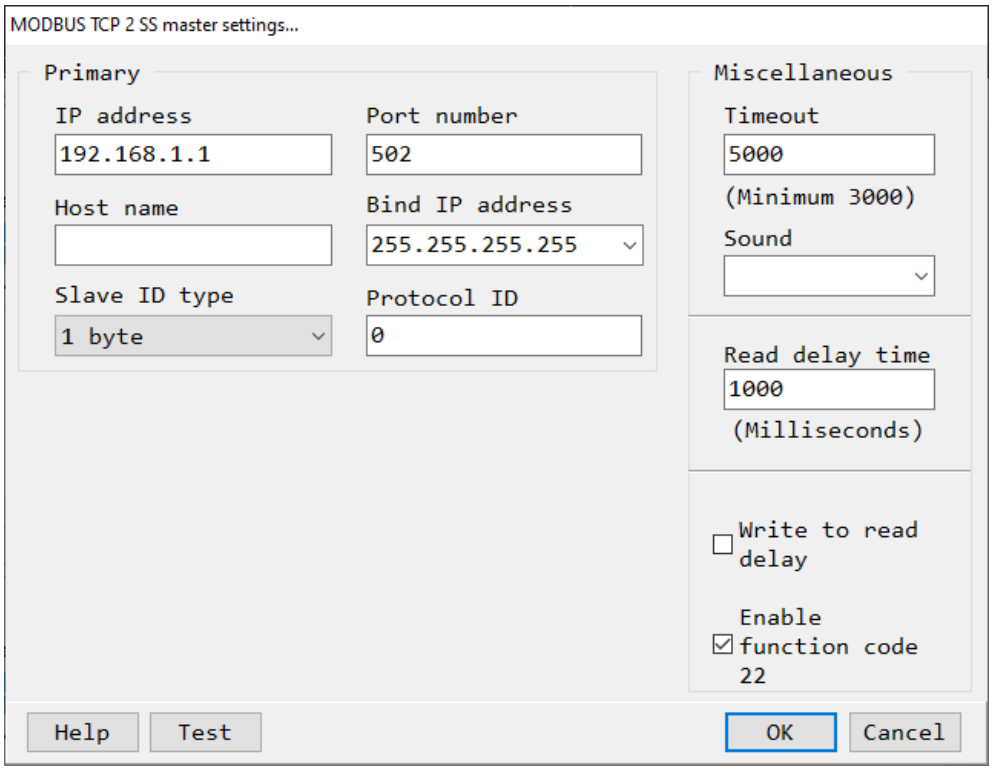


Each MODBUS TCP 2 SS master object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.
To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS TCP 2 SS master object select the "Delete" button.

Settings



If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Slave ID type

The MODBUS protocol defines the slave ID as a single byte field, limiting the maximum number of devices to 255. (0 is reserved for broadcasting)

Selecting “2 hi-lo” or “2 lo-hi” will configure the MODBUS packets to use a 2 byte slave ID. Note: The slave device must support a 2 bytes slave ID.

Protocol ID

The MODBUS protocol defines the protocol identifier (ID) as zero (0). Using a 2 byte slave ID field alters the protocol and the protocol ID zero (0) should be not used. However, the slave ID may or may not rely on a specific protocol ID.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sounds delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

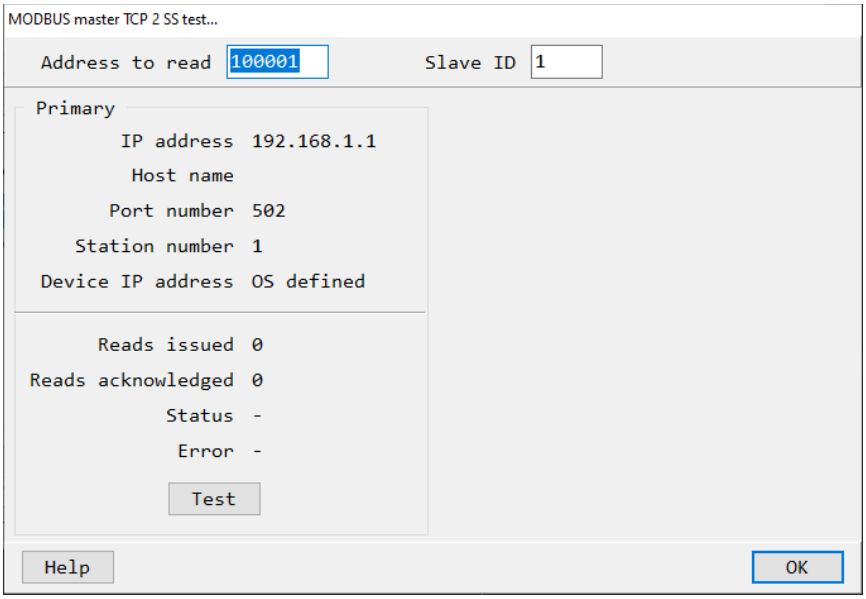
Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

Enable function code 22

When enabled and a bit write to a holding register is requested, function code 22 (Mask write 4X register) will be used to set or clear the bit. If not enabled, function code 6 (Preset single register) will be used. Note: Not all devices support function code 22.

Test button



When the test button is selected the program will attempt to read one word of data from the device at the address entered. If the address is a coil (0XXXXX) or an input (1XXXXX), an attempt to read 1 element (coil or input) is performed.

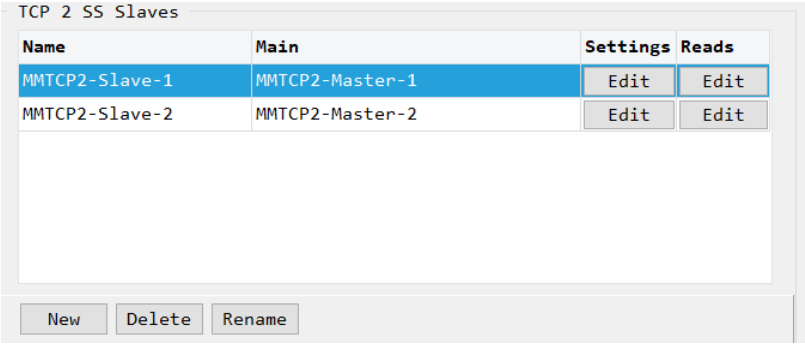
MODBUS TCP SS Slaves

Each MODBUS TCP 2 SS slave object is listed in the window.

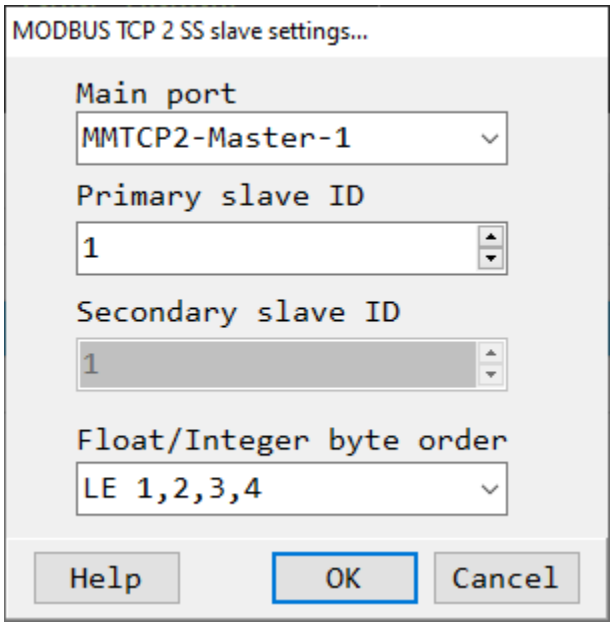
To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS TCP 2 SS slave object select the "Delete" button.



Settings



Main port

This is the master TCP 2 SS port the slave is linked to. A port must be linked to a master for runtime operations.

Primary station number
Secondary station number (Future)

This is the station number of the slave device for the primary and secondary. (1 - 65535)

Float/Integer byte order

Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

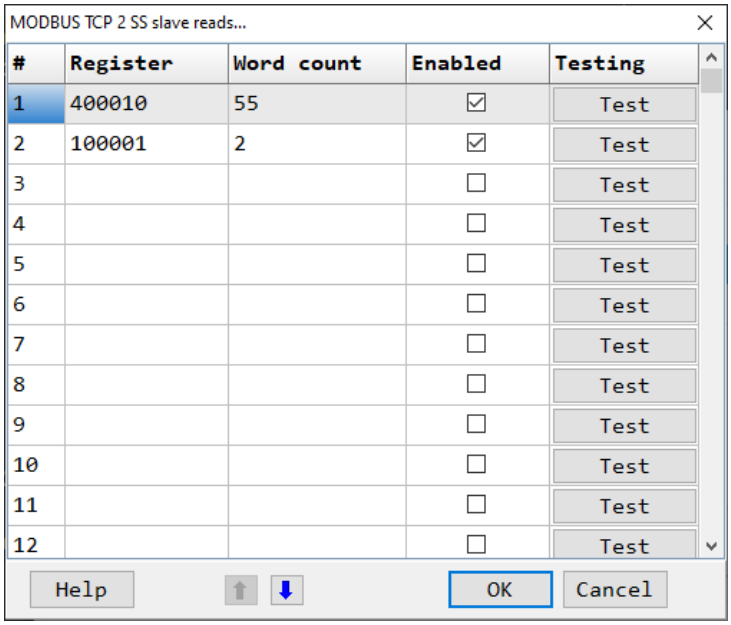
The MODBUS holding register data received from the slave device is in HI-LO byte order.

The data bytes are swapped and the result is stored in native format (LO-HI). Regular integer data and floating data values can be in the same data read.

The 4 byte float option is configured on a point level. The integer data is converted into a floating point value at the point level. Because the data bytes have already been swapped the conversion to a floating point value uses method index 3. If the byte swap had not been done at the table read level method index 4 would be the default.

When using floating point values the byte order of the slave device might be different than the regular holding register byte order. Selecting the correct conversion is required for accurate conversion to floating point.

Reads



#	Register	Word count	Enabled	Testing
1	400010	55	☑	Test
2	100001	2	☑	Test
3			☐	Test
4			☐	Test
5			☐	Test
6			☐	Test
7			☐	Test
8			☐	Test
9			☐	Test
10			☐	Test
11			☐	Test
12			☐	Test

The address ranges shown may or may not be present in the slave device.

The HMI uses the 6 digit MODBUS register address numbering system.

Register

The address must be a valid MODBUS address and not refer to a bit in a holding or input register.

Any of the following addresses are valid:
000001 - 065535
100001 - 165535
300001 - 365535
400001 - 465535

Invalid address would be any that refer to a bit in a word.

All of the following addresses are invalid for configuration of the reads. (All are legal for point source addressing.)

400001/01 or 300001/01

Bits/word Count

Each read can request up to 125 words of data
125 registers for input and holding register or 2000 bits for coils and inputs.

For example:

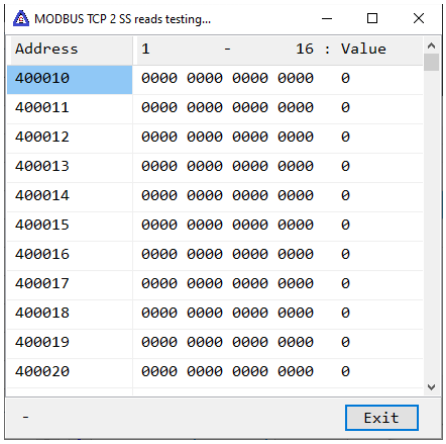
Register	Count	Result
400001	1	1 Holding register
303521	4	4 Input registers
100001	1	1 Input
100001	9	9 Inputs
000001	4	4 Coils
000016	2000	2000 Coils

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



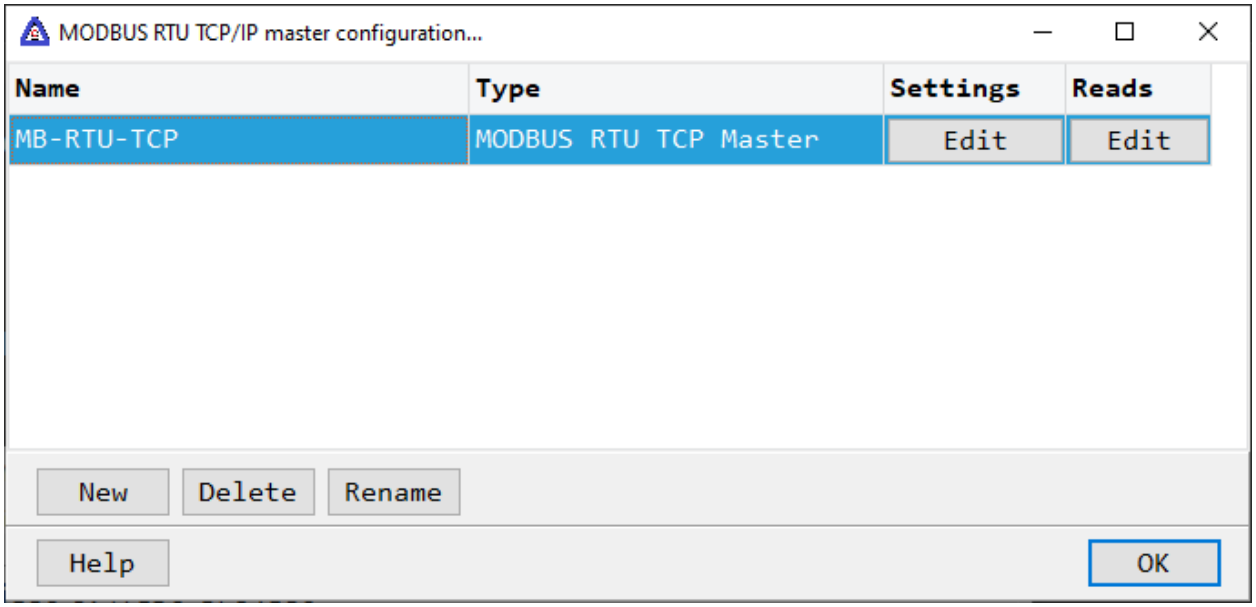
When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Error messages

No register address	The value in the field is not a MODBUS address.
Invalid count	The count must be between 1 and 2000.
Exceeds 65535	The register plus the count exceeds the maximum address range.
Invalid MODBUS address	The value in the field is an invalid MODBUS address.
Bit field not allowed	The value in the field references a bit of a register.

MODBUS RTU TCP MASTER

Each MODBUS RTU TCP/IP master object is listed in the window.

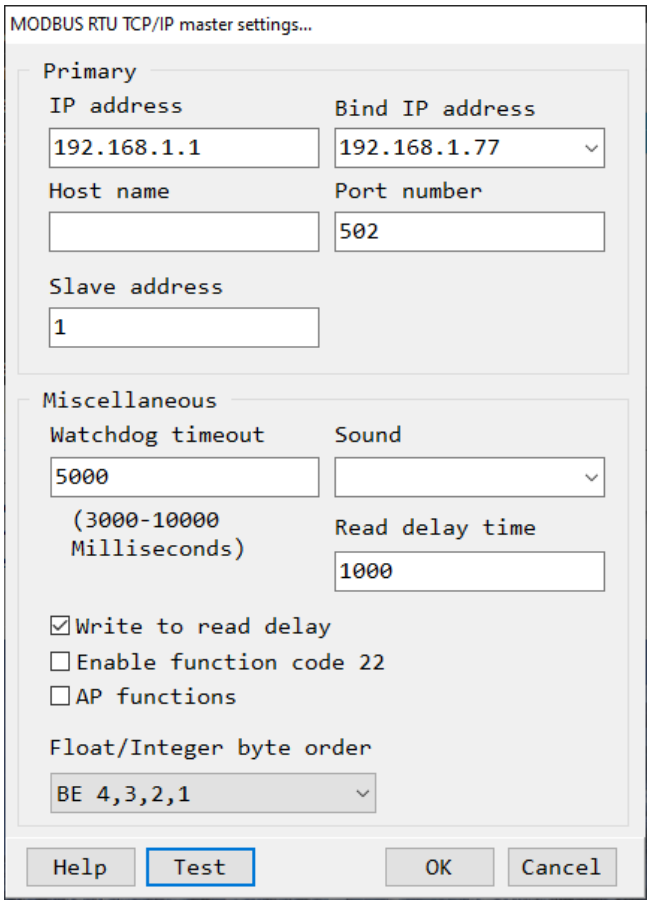


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS RTU TCP/IP master object select the "Delete" button.

Settings



Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sounds delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

Enable function code 22

When enabled and a bit write to a holding register is requested, function code 22 (Mask write 4X register) will be used to set or clear the bit. If not enabled, function code 6 (Preset single register) will be used. Note: Not all devices support function code 22.

AP functions

See analog functions.

Float/Integer byte order

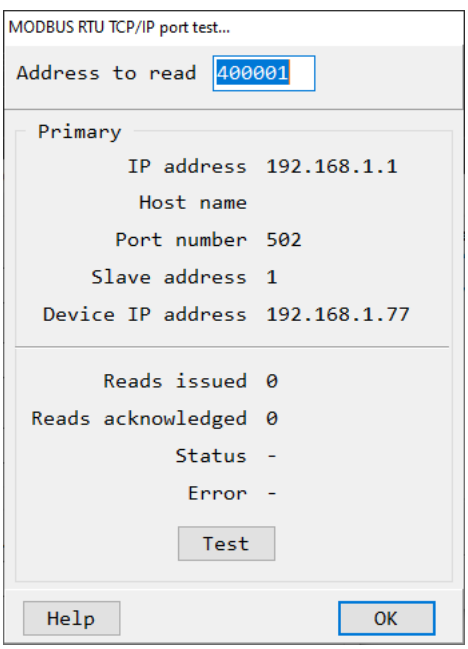
Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

The MODBUS holding register data received from the slave device is in HI-LO byte order.

The data bytes are swapped and the result is stored in native format (LO-HI). Regular integer data and floating data values can be in the same data read.

The 4 byte float option is configured on a point level. The integer data is converted into a floating point value at the point level. Because the data bytes have already been swapped the conversion to a floating point value uses method index 3. If the byte swap had not been done at the table read level method index 4 would be the default.

Test button



When the test button is selected the program will attempt to read one word of data from the device at the address entered. If the address is a coil (0XXXXX) or an input (1XXXXX), an attempt to read 16 elements (one word) is performed.

Reads

Read configuration is the same as MODBUS serial and TCP.

MODBUS Master arrays

All MODBUS master array "port" commands use the same properties and produce the same results.

Commands

Link

The link command "links" a read configuration to an array. Each time the requested data is returned from the external device the data is written to the specified array. The data type of the array determines how the data is processed.

Float	Each two registers in the read data reply are converted to a floating point value, based on the port float configuration order. (See "Float/Integer byte order in the port configuration.) The read request count must be > 1 and an even value.
Integer	Each two registers in the read data reply are converted to a 32 bit integer value, based on the integer configuration order. (See "Float/Integer byte order in the port configuration.) The read request count must be > 1 and an even value.
Boolean	The result is unpacked into consecutive 16 boolean indexes. Each read (count) request 16 booleans. The array must be sized to hold all returned values.
Other types	Small integer, word, the unaltered returned values are placed in the arrays.
Unsupported types	Byte, string, short integer, integer 64, unsigned integer 64
Syntax	values:=UArray('<array name>', 'Port', [<starting dimension index>], ['Link', '<port name>', <read index>]);
Result	values[0] = 0 (zero) or error code values[1] = 0
Example	values:=UArray('Turbine1Pressures', 'Port', [0], ['Link','Turbine1', 2]);

Notes:

1) Only one array can be "linked" with a read. The last called "link" command for a port/index is used.
2) The array must be sized to contain all the data from the read request. The array can be larger than the request.
3) More than one read request can be linked to an array.
4) The register type must be compatible with the array data type. i.e. a boolean read (1xxxxx, 0xxxxx) must be linked with a boolean array, a holding register (4xxxxx) cannot be linked with a boolean array.

Write

The write command "writes" array elements to the port device. The data type of the array determines how the data is written. The write is added to the port write queue and is processed according to the port read/write logic.

Float	Each array element is converted, based on the port float configuration order, and written to two registers. (See "Float/Integer byte order in the port configuration.) Function code 16 (0x10) Preset Multiple Registers is used.
Integer	Each array element is converted, based on the integer configuration order and written to two registers. (See "Float/Integer byte order in the port configuration.) Function code 16 (0x10) Preset Multiple Registers is used.
Boolean	Each array element is packed into consecutive bytes and the bytes are written. Function code 15 (0x0F) Preset Multiple Coils is used.
Other types	Small integer, word, one array element per register. Function code 16 (0x10) Preset Multiple Registers is used.
Unsupported types	Byte, string, short integer, integer 64, unsigned integer 64

Limits	**TCP (Read)**	**TCP (Write)**	**Serial (Read)**	**Serial (Write)**
Coils	2000	1968	2000	2000
Discrete Inputs	2000	N/A	2000	N/A
Holding registers	125	123	125	125
Input register	125	N/A	125	N/A

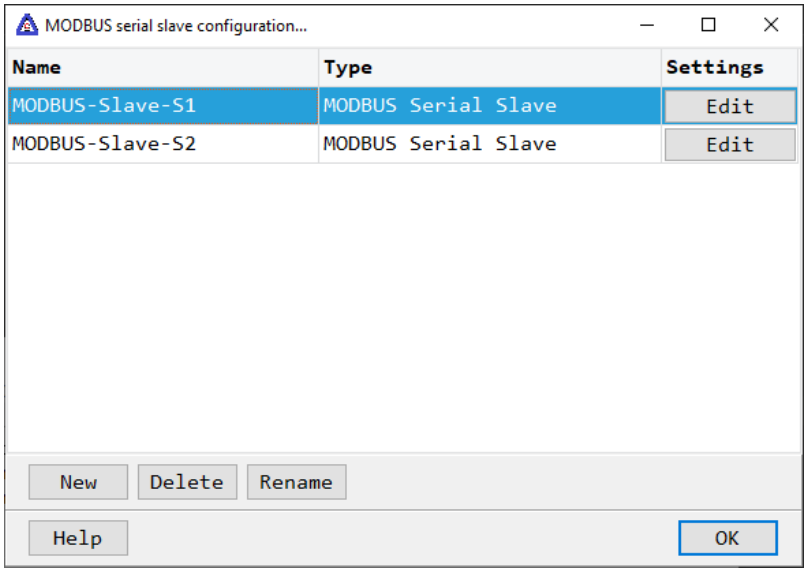
Syntax	values:=UArray('<array name>', 'Port', [<starting dimension index>], ['Write', '<port name>', '<beginning holding address register>', <count array elements to write>]);
Result	values[0] = 0 (zero) or error code values[1] = 0
Example	values:=UArray('Turbine1Limits', 'Port', [2], ['Write','Turbine1', '400001', 7]);

Notes:

1) For double type (8 byte) writes, the float value is converted to a single (4 byte) floating point value. Values outside the range or precision of a single float (32-bit floating point, IEEE-754 standard format) number will be lost.
2) The array must be sized to provide all the data specified in the write request. The array can be larger than the request.
3) The register type must be compatible with the array data type. i.e. a boolean write, coil (0xxxxx) must be linked with a boolean array, a holding register (4xxxxx) cannot be linked with a boolean array.

MODBUS Slave Serial

Each MODBUS slave object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS slave object select the "Delete" button.

The following registers are supported.

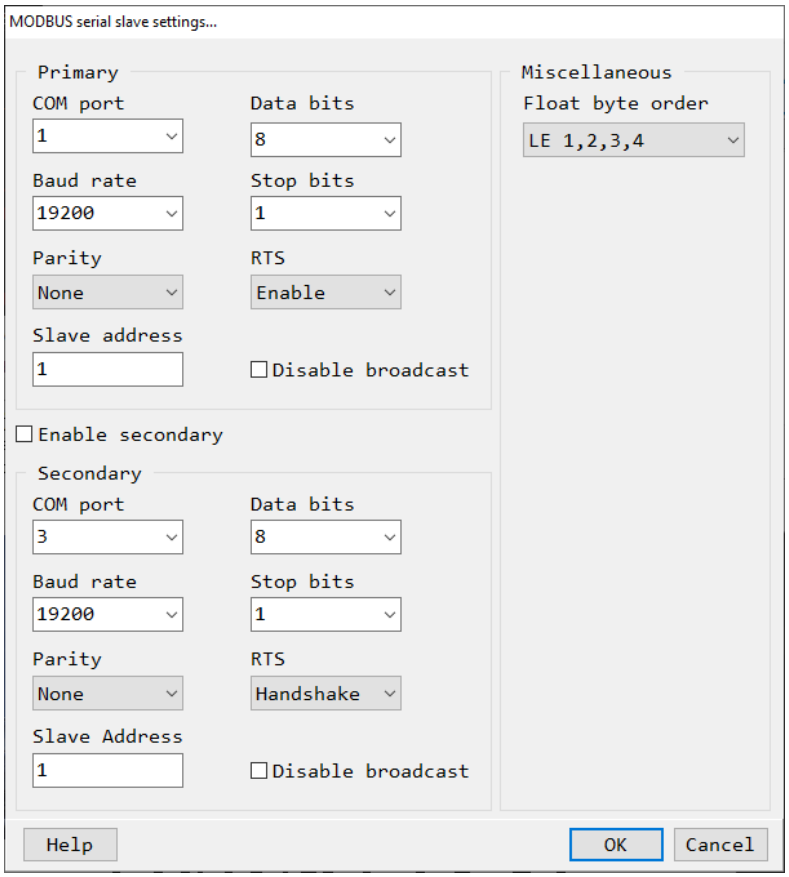
000001-065535 Coil
100001-165535 Input
300001-365535 Input Register
400001-465535 Holding register

Function Codes

The following function codes are supported.

01: Read coil status
02: Read input status
03: Read holding register
04: Read input registers
05: Force single coil
06: Preset single register
15: Force multiple coils
16: Preset multiple registers

Settings



The serial port has a primary port and if enabled a secondary port. Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Slave address

This is the MODBUS slave address of this program. This program only accepts message with a slave address that equals this setting or is 0 (broadcast). The address range is 1-255. Default is 1.

Disable broadcast

If this attribute is enabled, the slave will not accept broadcast messages.

Enable secondary

The "Enable secondary" checkbox provides for a second com port to be used. When in run mode both ports would be active.

Float/Integer byte order

Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

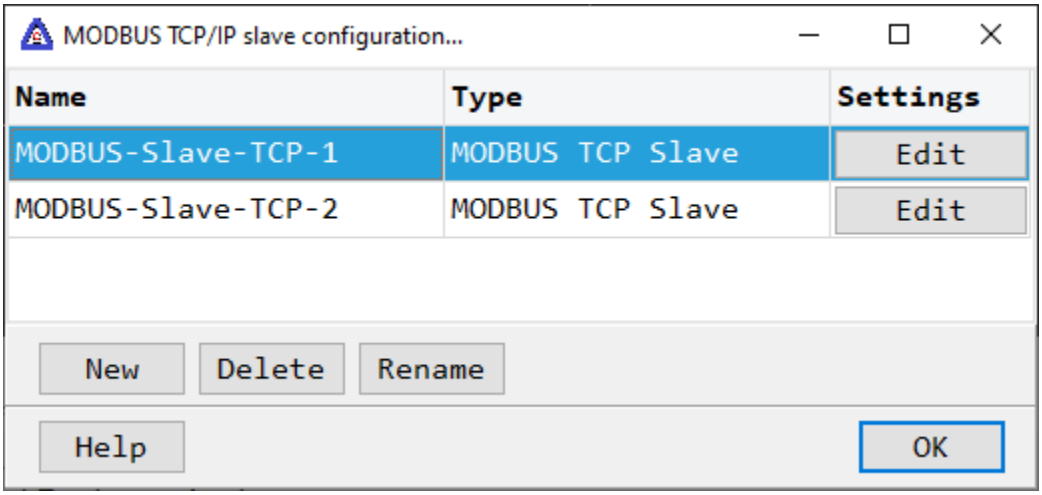
The MODBUS holding register data received from the slave device is in HI-LO byte order.

The data bytes are swapped and the result is stored in native format (LO-HI). Regular integer data and floating data values can be in the same data read.

The 4 byte float option is configured on a point level. The integer data is converted into a floating point value at the point level. Because the data bytes have already been swapped the conversion to a floating point value uses method index 3. If the byte swap had not been done at the table read level method index 4 would be the default.

MODBUS SLAVE TCP/IP

Each MODBUS slave object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MODBUS slave object select the "Delete" button.

The following registers are supported.

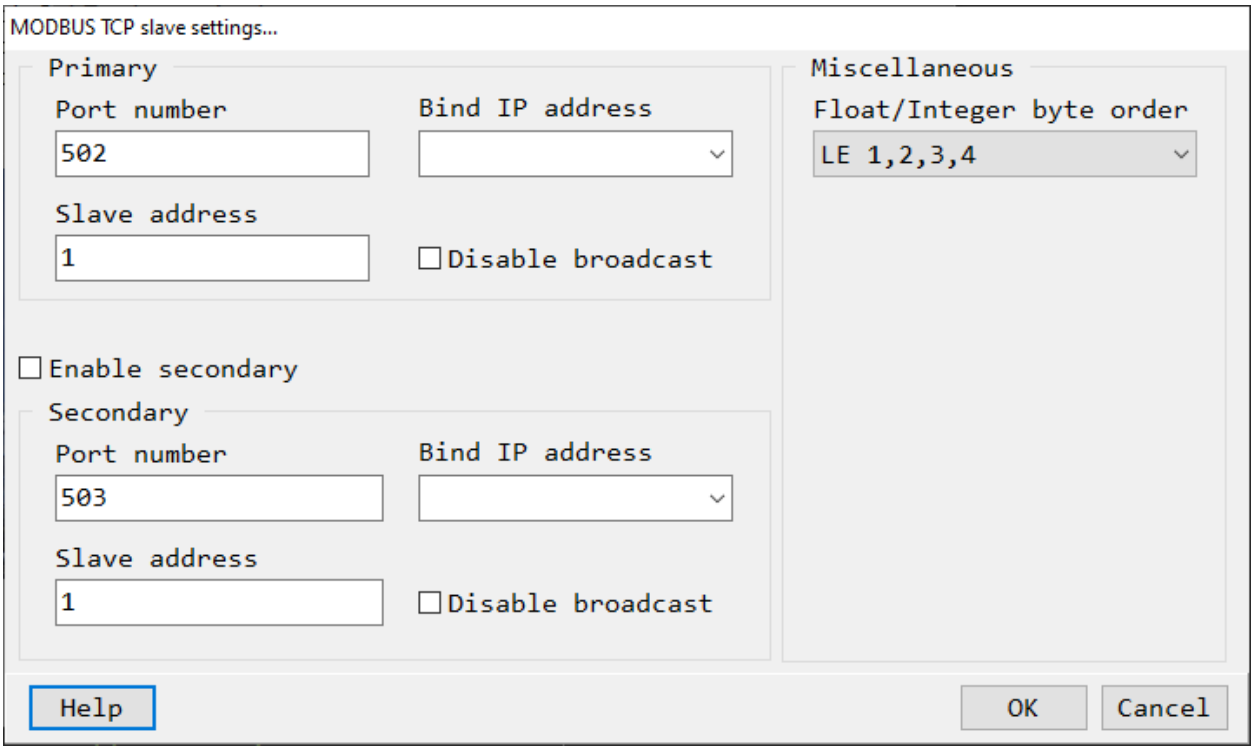
000001-065535 Coil
100001-165535 Input
300001-365535 Input Register
400001-465535 Holding register

Function Codes

The following function codes are supported.

01: Read coil status
02: Read input status
03: Read holding register
04: Read input registers
05: Force single coil
06: Preset single register
15: Force multiple coils
16: Preset multiple registers

Settings



Port number

The TCP/IP port number of the slave device.

Slave address

This is the MODBUS slave address of this program. This program only accepts message with a slave address that equals this setting or is 0 (broadcast). The address range is 1-255. Default is 1.

Bind IP address See here.

Disable broadcast

If this attribute is enabled, the slave will not accept broadcast messages.

Enable secondary

The "Enable secondary" checkbox provides for a second com port to be used. When in run mode both ports would be active.

Float/Integer byte order

Index	Selection Type	Description	Byte Order	Comments
1	BE 4,3,2,1	Big Endian Format	4,3,2,1	
2	BE 3,4,1,2	Big Endian w/ bytes swapped	3,4,1,2	
3	LE 1,2,3,4	Little Endian Format	1,2,3,4	Default
4	LE 2,1,4,3	Little Endian w/ bytes swapped	2,1,4,3	

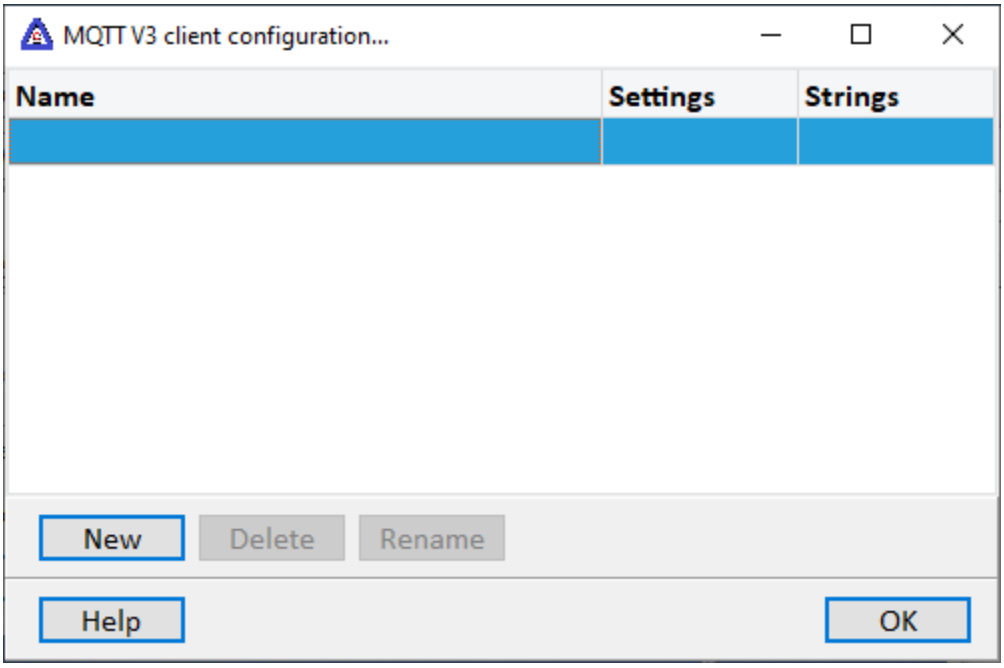
The MODBUS holding register data received from the slave device is in HI-LO byte order.

The data bytes are swapped and the result is stored in native format (LO-HI). Regular integer data and floating data values can be in the same data read.

The 4 byte float option is configured on a point level. The integer data is converted into a floating point value at the point level. Because the data bytes have already been swapped the conversion to a floating point value uses method index 3. If the byte swap had not been done at the table read level method index 4 would be the default.

MQTT CLIENT V3

Each MQTT (Message Queuing Telemetry Transport) V3 client object is listed in the window.



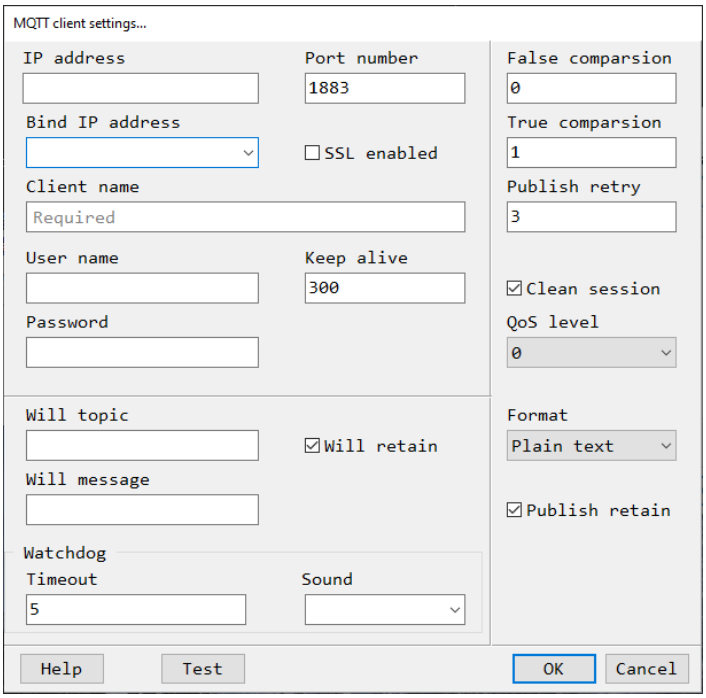
To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MQTT client object select the "Delete" button.

Note: The source address (topic name) **is case sensitive**.

Settings



IP address number

The TCP/IP address of the MQTT broker.

Port number

The TCP/IP port number of the MQTT broker.
1883 is the default port number for MQTT
8883 is the default port number for MQTT when using SSL

Bind IP address See here.

SSL enabled (Secure Socket Layer)

If enabled, SSL will be used for TCP/IP encryption.

Client name

This is also called the “Client identifier.” The name must be unique across all clients connected to the broker.
Note: If a client with the same client name is currently connected to the broker/server, the connected client will be disconnected by the broker/server.

Keep alive (seconds)

This value sets the frequency, in seconds, a “Ping” message is sent to the broker. If another data related message is sent to the broker before the timer expires, the timer is reset.

User name/password (optional)

These fields are optional. The password is ignored if a user name is not supplied.

Publish retry

This value is the number of times to attempt to send a “Publish” message. If the message is not acknowledged after the retry count is reached, the message is deleted and an entry is placed in the event log. This value only applies to “QoS” (Quality of Service) levels above zero (0).

QoS level (Quality of Service)

The QoS level for “will”, “publish” and “subscriptions”.

Format (payload)

The HMI supports two payload formats. If another payload format is required, contact support. Strings can be used to fetch and parse the payloads in a script. See “Action” for more parsing options.

Plain text — The payload is only the value of the topic name. For example, true, 23, 17.9

Weintek — Weintek wraps the data in JSON notation. For example the payload for a topic name “HMI/machineOne/stats/Speed”

```
{
  "d" : {
    "Speed" : [ 0 ],
    "TankLevel" : [ 88 ]
  },
  "ts" : "2016-05-07T23:04:57.747138"
}
```

Weintek allows for the grouping of topic names. In the HMI only one variable can be processed per topic name. For the above example select:

“HMI/machineOne/stats/Speed” for the speed value

or

“HMI/machineOne/stats/TankLevel” for the tank level value

A source address of “HMI/machineOne/stats” would be an error. The result could not evaluate to a single value.

Note: Each variable should be in a single topic name. If not, writes from the HMI will not function properly.

False/true comparison

When a point is created it is analog or digital. For an analog point, the source type defines the format and the payload parser will convert the text to the correct format. For digital points a source type property is not used. The “False/True comparison” properties define the text for “False” and “True”. The text could be “0”, “False”, “false”, “F”, “1”, “True”, “TRUE”, etc. Set the property to the text for false/true.
If the payload text matches the false property, the point is set to false.
If the payload text matches the true property, the point is set to true.
If the payload text does not match the false or true property, the point is set to false.

Note: For the Weintek payload format the false comparison is “ false “ and the true comparison is “ true “; a space character before and after the word.

Clean session

If enabled the broker must discard earlier information concerning the client and begin a “clean” session.

Publish retain

When the HMI sends a “publish” message to the broker and the “retain” is enabled the broker will retain the value. If a client subscribes to the topic, the value will be sent upon subscription. If “retain” is not enabled, the broker will not send a value when a new subscription occurs.

Will retain

If enabled the broker will retain the ”Will” message and publish the ”Will” message to clients subscribing to a client that disconnects without a disconnect message. If enabled, the “Will topic” and “Will message” must be supplied.

Will topic/message

The topic and message the broker will send to clients of a client that disconnects without sending a disconnect message.

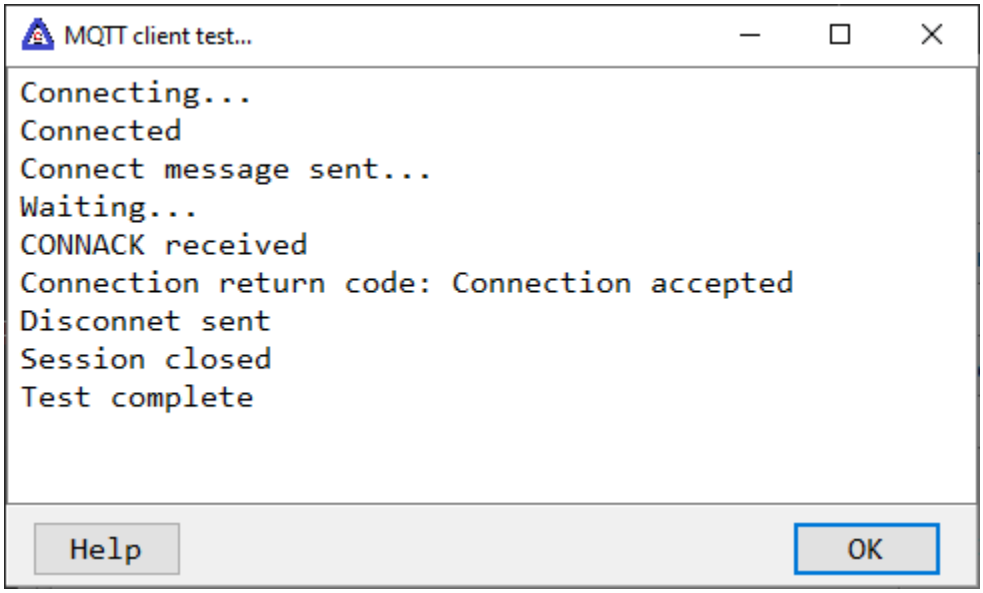
Watchdog (seconds)

The timer starts when a message is sent to the broker and a response is expected. The watchdog timer is reset when data is received or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

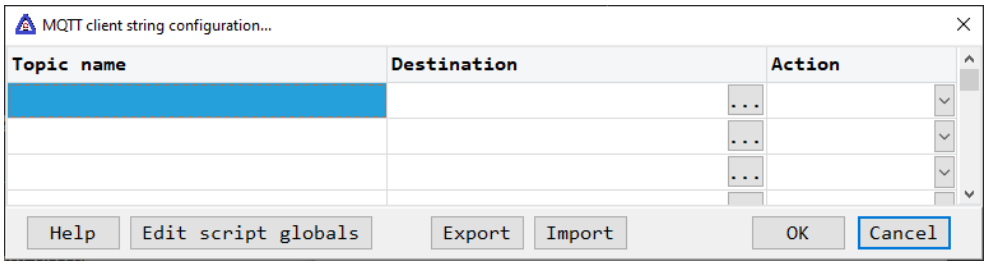
Testing

Test the configured parameters. (iot.coreflux.cloud was used for the test)



Strings

This feature can be used to subscribe a string payload.



Topic name

The complete topic name in the broker.

Notes:
1) The topic name is case sensitive.
2) The string is not formatted or parsed based on the payload format selection.

Destination (optional)

If desired, select a script global location and the string/payload will be copied to the location. If this option is used or not the string may still be accessed via the "StringGet" and "StringSet" script commands. If this is configured and access to the string, in a script, is desired use the "GlobalGet" or "StringGet" to copy the string. The format of the destination is: <section>.<name>.
Note: Writing to the script global does not write the value to the MQTT broker. "StringSet" must be used to write a string to the MQTT broker.

Action

The payload can be processed with several methods.

Default

If a section and item are defined the payload is written to the script global. If the section and item are not defined, no action is performed.

Script global

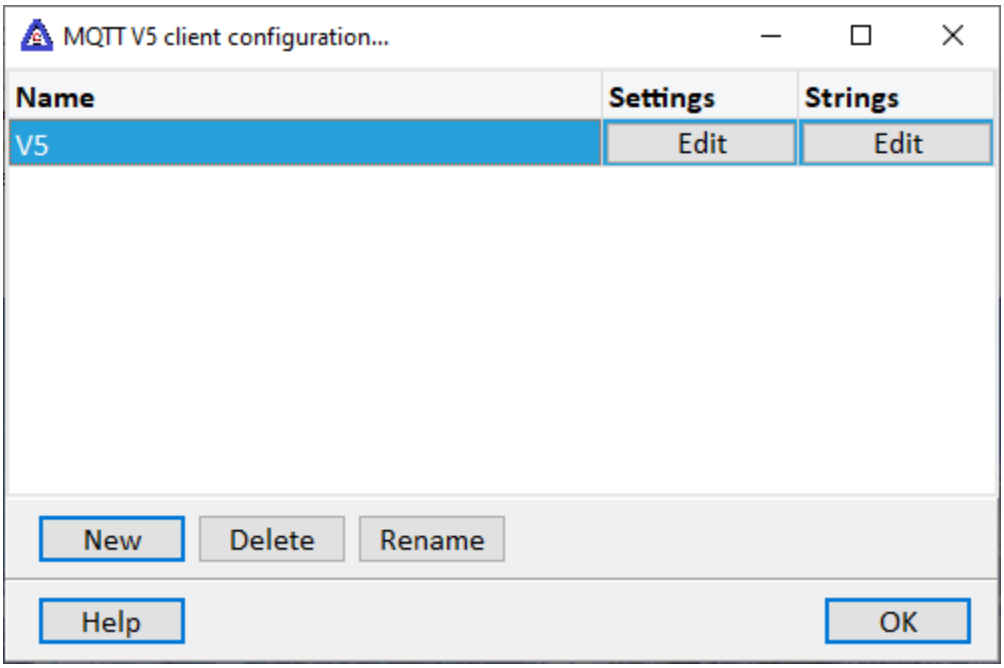
If a section is defined the payload is parsed using the JSONToScriptGlobal script command.

Host points

If a section is defined the payload is parsed using the JSONToHostPoints script command.

MQTT CLIENT V5

Each MQTT (Message Queuing Telemetry Transport) V5 client object is listed in the window.



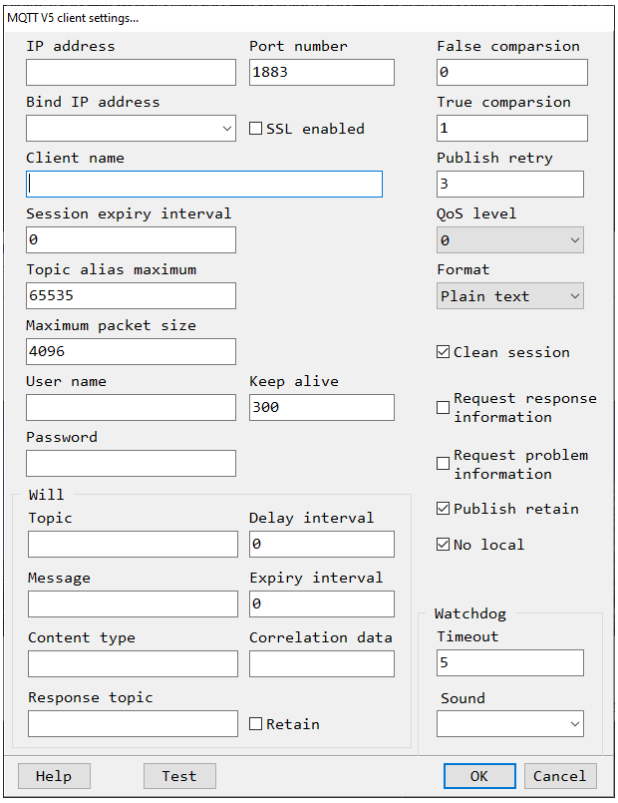
To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a MQTT client object select the "Delete" button.

Note: The source address (topic name) **is case sensitive**.

Settings



Note: Fields not defined here are the same as MQTT V3 above. The MQTT specification can provide more information.

Bind IP address See here.

Session expiry interval

The value is sent to the server but the timer is not implemented in the HMI. Use with “clean session”.

Maximum packet size

This maximum packet size the HMI will accept.

Request response information, Request problem information

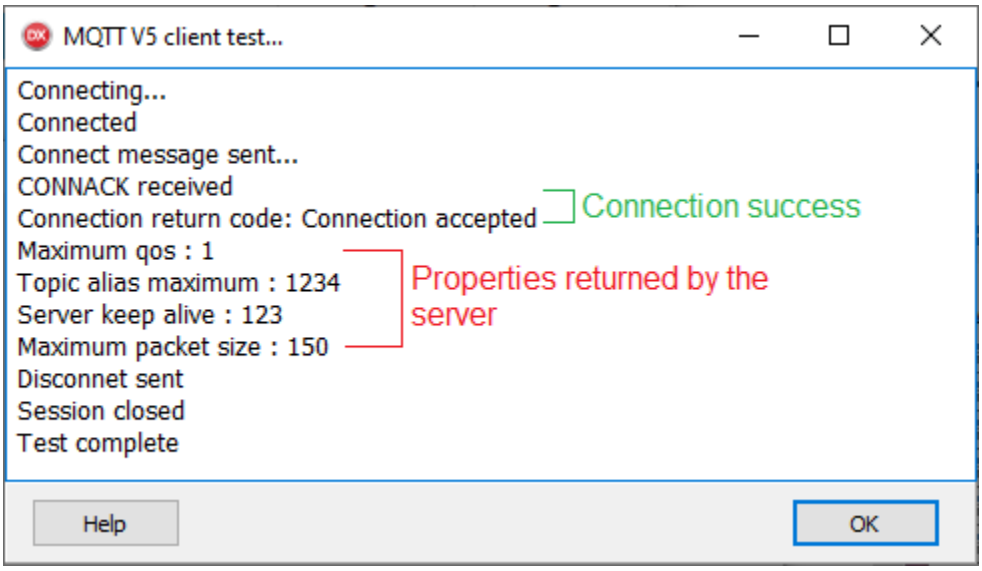
See the MQTT specification.

No local

Instructs the server to not send a message back when a publish message is transmitted from the HMI for a monitored topic name.

Test button

Select this button to attempt to connect to a server.

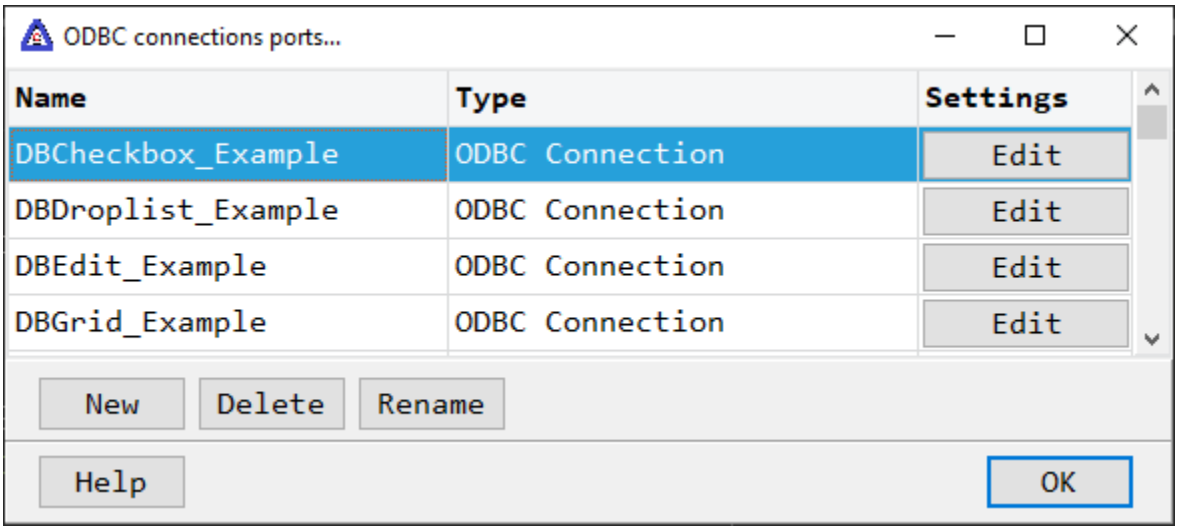


ODBC Connections

The “ODBC Connections” port type is used to connect graphic elements, which support database connections, with a database. The other ODBC port types, ODBC Data Logger, ODBC In, ODBC In/Out and ODBC Out provide for connection settings and do not use this port type.

The graphic elements that support database connections are listed here. The script command is here.

Each ODBC Connection object is listed in the window.

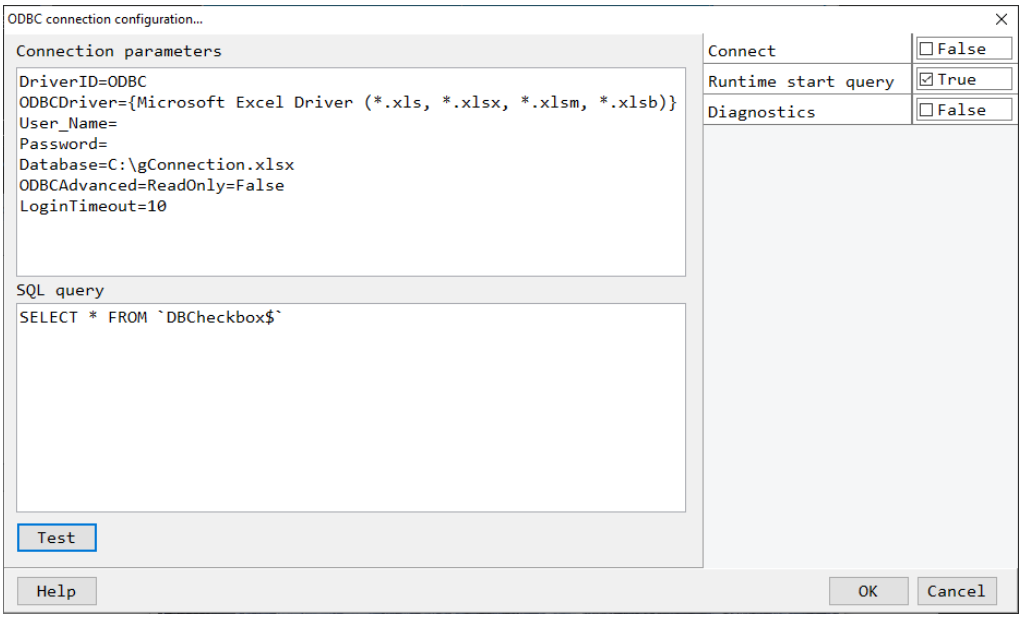


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an ODBC Connections object select the "Delete" button.

Settings



Connection parameters

Refer to the section below.

SQL query

This query is used if “Runtime start query” is enabled. Note: If the connection is to Excel, the sheet name (table name) must begin and end with a “backtick” character (`), also called the ‘grave accent’. On US English keyboards, the key is located to the left of the one (1) key. The SQL query above is for Excel. A query must be active for any visible database graphic elements to function.

Connect

If enabled, when runtime monitoring begins, a connection to the database will be established. The connection must be maintained for any visible database graphic elements to function. The connection can be ended or established via scripting.

Runtime start query

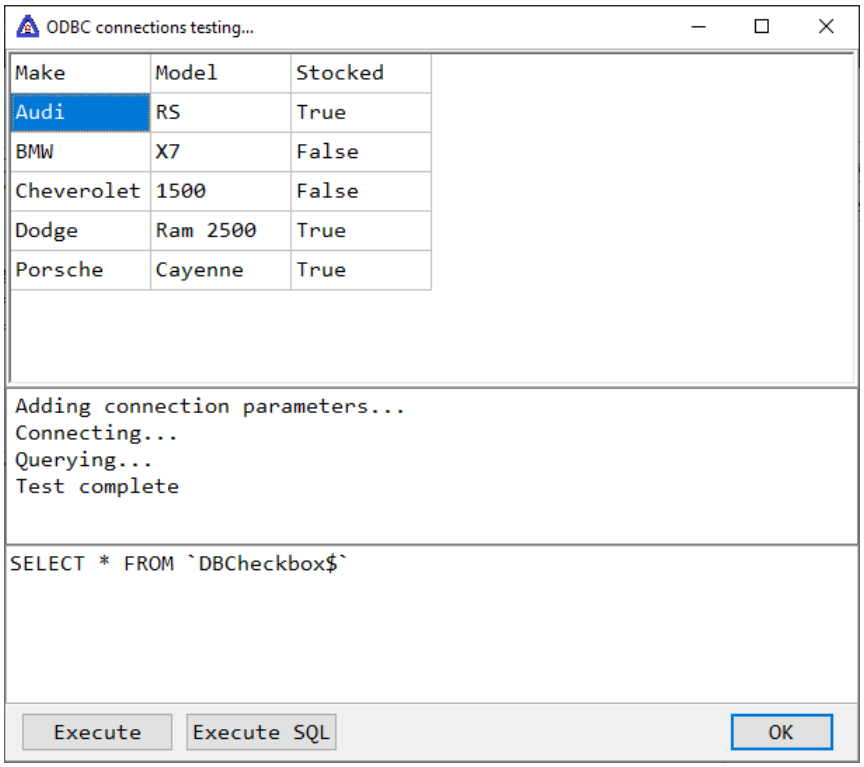
If this property is enabled, the internal “Query” request object will be populated with data from the “SQL query” field.
If the “connect” property (above) is also enabled, an “SQL query” will be initiated at runtime start. The query can be modified and initiated via scripting.

Diagnostics

If this property is enabled, the port will log commands executed, and other miscellaneous events to a log. **Note:** If this data is not required, disable this property. If this property is enabled and not required, it is a waste of resources, processing time, etc..

Test

This button will open the test window, establish a connection to the database and issue the query.
The top pane is the result of the query.
The middle pane is diagnostic information.
The bottom pane is the query string and can be altered for testing.
The “Execute” button will initiate a query using the string in the bottom pane.

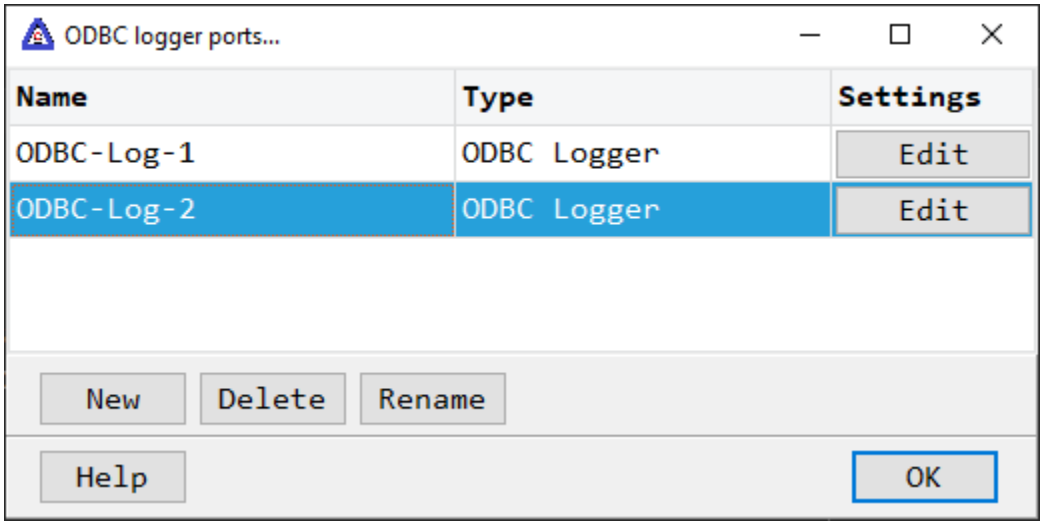


ODBC Data logger

Note: It is suggested that all the fields in the database be set to 'Text'.

ODBC (Open Database Connectivity) provides a method to connect with many different databases via one interface type.

Each ODBC logger object is listed in the window.



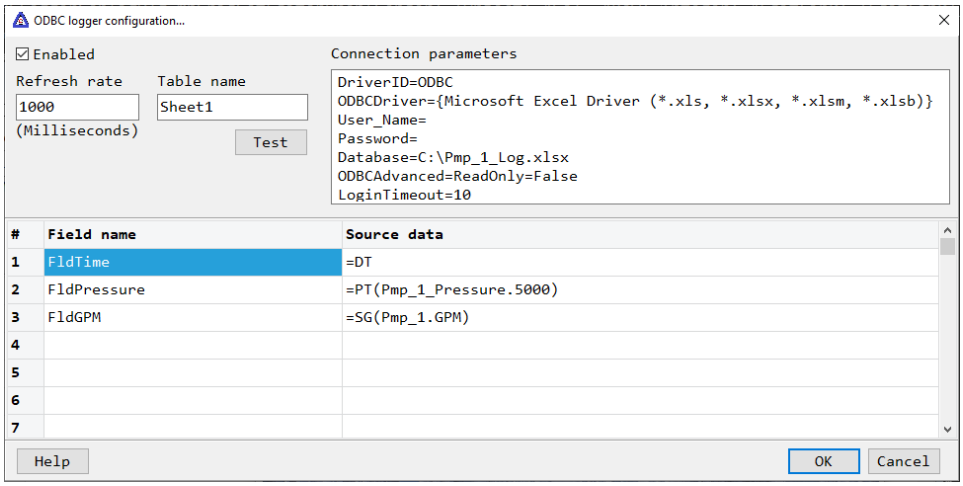
To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an ODBC logger object select the "Delete" button.

A single connection is used to log the data to the file/database. Each new record is added to the end of the table at the configured refresh rate. If the refresh rate is zero (0), records are added via scripting or mouse commands.

Settings



The database must exist prior to access. The format of the database is unknown so the program cannot create the database. This feature logs a new row/record to the database on each command, via user control or at the configured “Refresh rate”. The maximum number of fields(columns) per row/record is 64.

Enabled

If enabled, at runtime startup a connection will be made to the database.

Refresh rate

If the refresh rate is 0 (zero) a record will only be inserted via a script command or mouse command. If the refresh rate is 100 milliseconds or greater a record will be inserted at the configured refresh rate.

Table name

The name of the table in the database. For Excel this is the sheet name.

Connection parameters

Refer to the section below.

Field name

The name of the field to place the source data. For Excel this is the text in the first row of each column.

The column/field names cannot be any word reserved by the database. (e.g. Date, Time, Add, Counter)

Source data

This is the source for the data to insert into the database. The format is the same used for reports.

Right click the mouse in a cell and select a menu item to display the cell formatter dialog. Or type in the correct configuration.

Prefix	**Operation**
PT	Collects the value of a point.item
SG	Collects the value of a script global
DT	Date time display

Points

The value of the point.item will be placed in the cell.
=PT(Tank.Process Variable Analog)

The value of the point.item will be placed in the cell with 3 decimal places. The default is 2 and does not need to be present.
=PT(Tank.Process Variable Analog~D3)

The value of the point.item will be placed in the cell using the supplied strings. The default is True/False and does not need to be present.
=PT(Tank.Process Variable Digital~TOpen~FClosed)

Script globals

The value of the script global will be placed in the cell. (Section.item)
=SG(Logged on.User)

The value of the script global will be placed in the cell and limited to the first ten characters. Default is all characters and does not need to be present.
=SG(Logged on.User~L10)

Note: If the string contains any spaces, the string must be enclosed in single quotes. Example 'some text'.

Date/Time

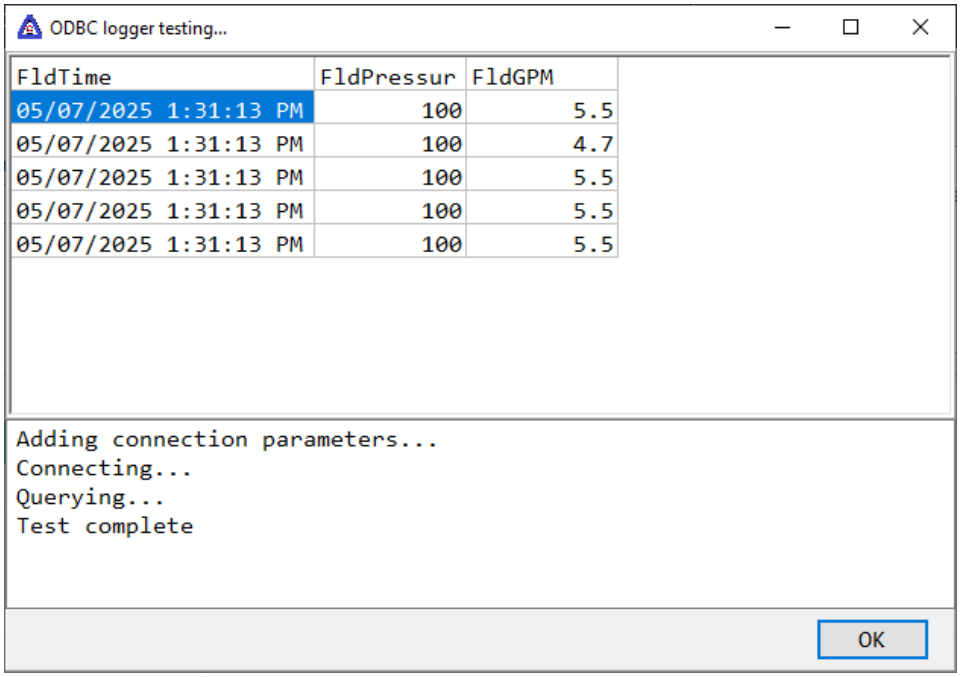
The date and/or time will be placed in the cell. This command uses the same format as the graphic engine. No format specified will use the system default.

Note: This is a 'text' field type.

=DT	// as defined by the OS
=DT(h:n:s d/m/yy)	// 13:6:36 24/5/11
=DT(mm/dd/yyyy hh:mm:ss)	// 05/24/2011 13:06:36
=DT(mm-dd-yyyy hh:mm:ss)	// 05-24-2011 13:06:36
=DT(hh:mm:ss)	// 13:06:36

Note: It is suggested tow use a 24 hour clock format.

Test button



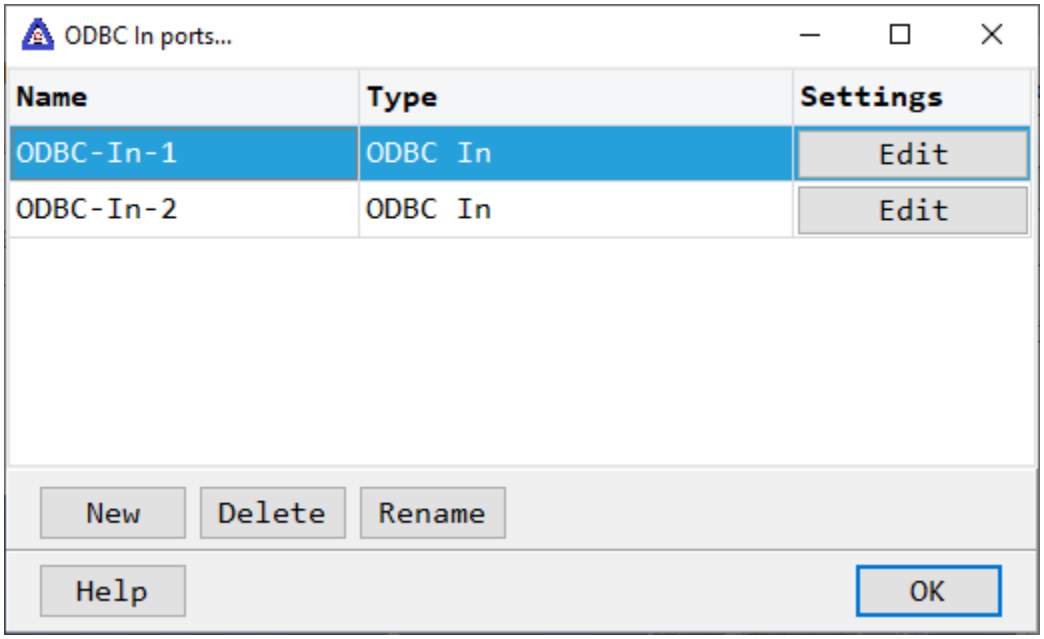
The program will attempt to connect to the database and perform a query. It will not write to the database.

ODBC In

Note: It is suggested that all the fields in the database be set to 'Text'.

ODBC (Open Database Connectivity) provides a method to connect with many different databases via one interface type.

Each ODBC In object is listed in the window.



The HMI divides connections to write point status (out) from connections to collect point data (in).

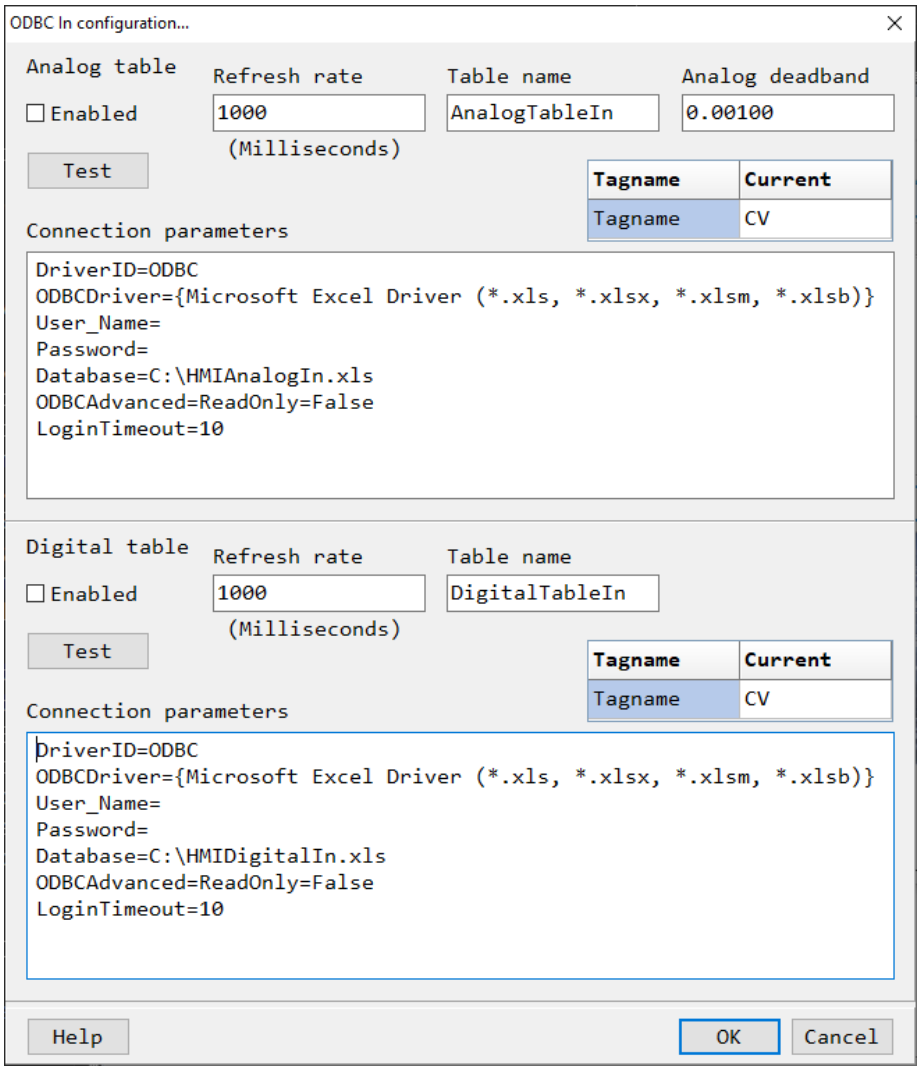
The ODBC In provides a connection to write data into analog and/or digital points.

The number of database connections is unlimited. This allows for the data to be segmented. For analog and digital points only the tagname and current value are required.

Some example files/database are in the '..\Database Templates\' directory.

WARNING: Do not change the order or value of the first column in the database after runtime monitoring has started.

Settings



'ODBC In' is really a read/write port type. The points are defined in the 'Points Editor' and the values are collected (read) from the database. Depending on the point access rights the values may be written to the database - just like a PLC or other device that contains memory locations with data. The source address in the 'Points Editor' is used for the tagname column in the database.

Analog and Digital table

Each table has several fields. Each column must have a name and it must be unique for the table.

The column/field names cannot be any word reserved by the database. (e.g. Date, Time, Add, Counter)
The column/field names can not contain any space characters.

Analog table

Field name	Type
Tagname	string
Current value	float

Digital table

Field name	Type
Tagname	string
Current value	Boolean

Enabled

If enabled at runtime a connection will be made to the database.

Table name

The name of the table in the database. For Excel this is the sheet name.

Analog deadband

This value is the minimum difference between the old value and current value to cause the change logic to execute.

Floating point numbers are represented in binary fractions and therefore are always an approximation of a decimal fraction.

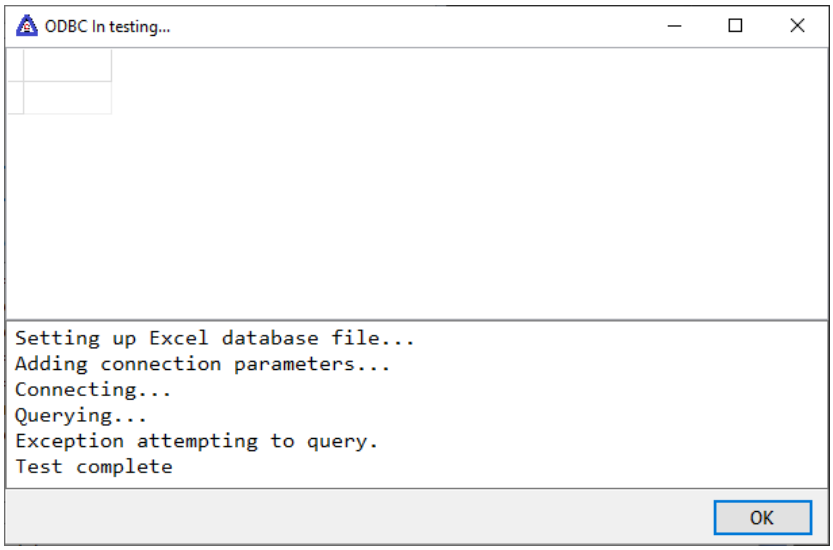
That is why occasionally a value like 1.980000000001 is displayed for something that would be expected to have only a few decimal places of precision. When comparing two floating point values, the comparison should be on how close the two values are to each other rather than testing for equality.

AD = 0.001 (default)
Change = ((a - b) >= AD)

Connection parameters

Refer to the section below.

Test



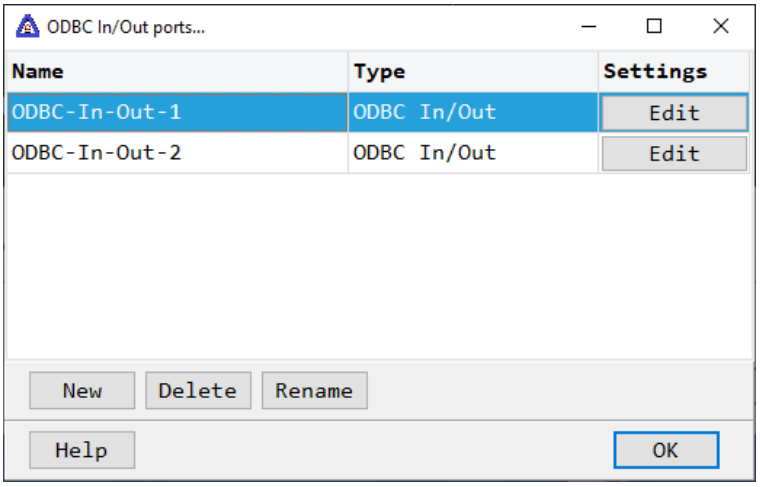
Selecting this button will attempt to create a connection to the database, set up the table and write the configured tagnames to the database. WARNING: Verify the database file is not use before selecting the 'Test' button.

ODBC In/Out

Note: It is suggested that all the fields in the database be set to 'Text'.

ODBC (Open Database Connectivity) provides a method to connect with many different databases via one interface type.

Each ODBC In/Out object is listed in the window.



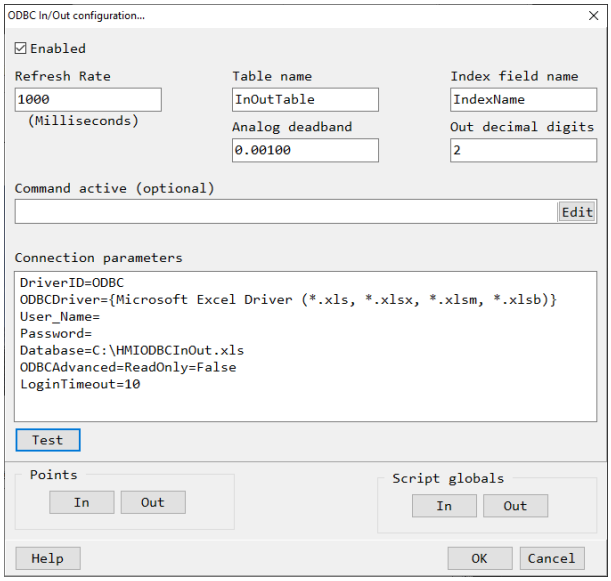
A single connection is used for access. Points are written to and read from using a single connection (file/database).

A read list is used to provide the source row and column in the database and the destination tag in the HMI database. The type of the point will determine the data type.

A write list is used to provide the destination row and column in the database and the source point.item in the HMI database. The type of the point will determine the data type written to the database. The first method writes the point data and some additional tag data. This method writes a single point.item per write.

This method can also be controlled via scripting.

Settings



Enabled

If enabled at runtime a connection will be made to the database.

Table name

The name of the table in the database.
For Excel this is the sheet name and cannot be **Sheet1**, **Sheet2** or **Sheet3**.

Index field name

This is the name of the field (column) to search for the row name.

The program will access the row that contains the name configured in the in/out list.

Then the value will be collected from/written to the row under the column name configured in the in/out list.
The row search can be based on any field (column). Supply the name of the field (column) in this attribute.

The column/field names cannot be any word reserved by the database. (e.g. Date, Time, Add, Counter)
The column/field names can not contain any space characters.

For Excel

1) The first row of the sheet contains the column names.
2) Row names, not row numbers are used to locate the destination cell.

For example: Row 1 could contain Unit, Pressure, Temperature. These are the column names.

Setting the 'Index field name' to 'Unit' would configure the program to search the 'Unit' column.

Column 1 (Unit) could then contain all the unit names after the first row. e.g. Crusher, Shredder, Mixer, etc.

In the In/out points the 'Row' is the name to search for in the 'Index field name'. In this example, 'Unit'.

Analog deadband

This value is the minimum difference between the old value and current value to cause the change logic to execute.

Floating point numbers are represented in binary fractions and therefore are always an approximation of a decimal fraction.

That is why occasionally a value like 1.980000000001 is displayed for something that would be expected to have only a few decimal places of precision. When comparing two floating point values, the comparison should be on how close the two values are to each other rather than testing for equality.

AD = 0.001 (default)
Change = ((a - b) >= AD)

Out decimal digits

The decimal digit count for analog out points. (0-7)

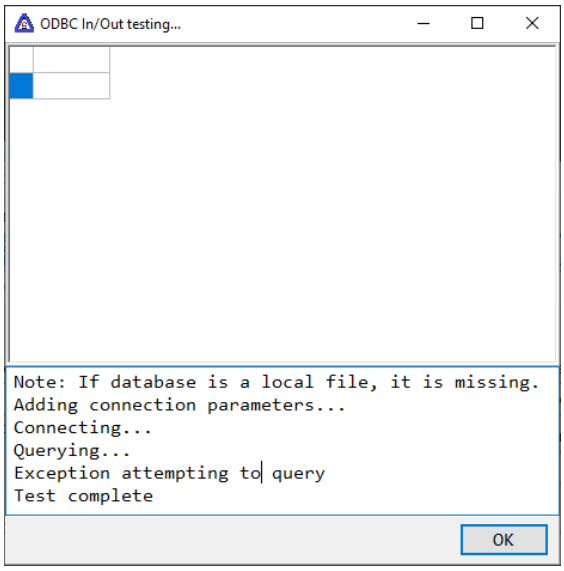
Command active (optional)

This attribute can point to a digital host point. When a command is accepted and execution has begun this point is set to true. Then the command completes the point is set to false. **Note:** Some commands execute very fast. The on to off period may be very short.

Connection parameters

Refer to the section below.

Test



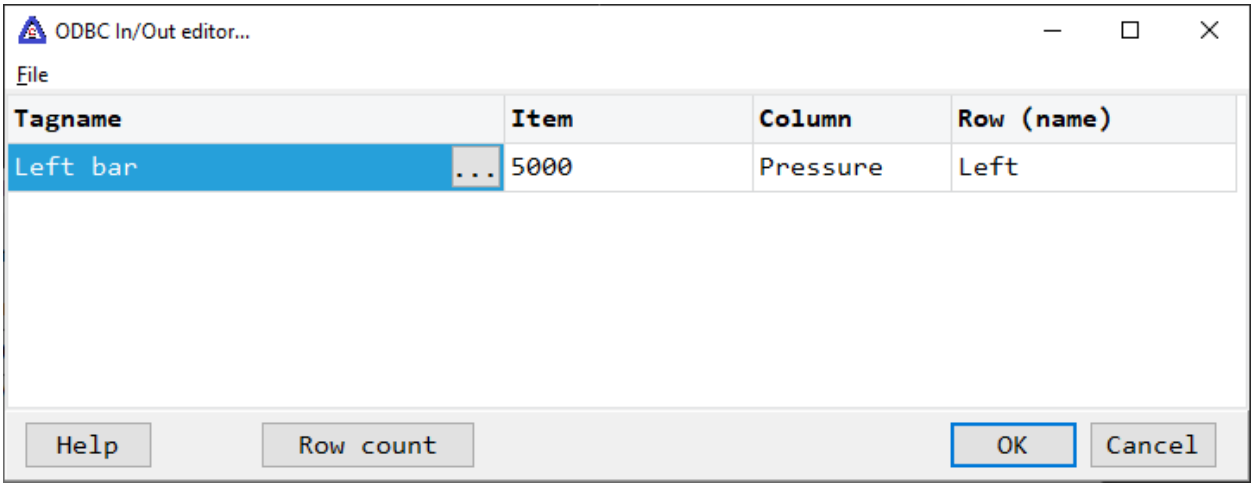
Selecting this button will attempt to create a connection to the database, set up the table and write the configured tagnames to the database. WARNING: Verify the database file is not use before selecting the 'Test' button.

Point/script globals In/Out

These are the list of commands to read and write to the database.

Note: For script globals, set the field type to 'text' for all the fields in the table that are referenced. Failure to do so may cause reading and/or writing to fail.

ODBC In/Out list editor



This window allows for the editing of the in (reads) and out (writes) command list.

When importing a CSV or XLS file the first row of the data is ignored.

Selecting the button in column 1 will display the selection tool for the correct type.

For Excel
1) The first row of the sheet contains the column names.
2) Row names, not row numbers are used to locate the destination cell.

For example: Row 1 could contain Unit, Pressure, Temperature. These are the column names.

Setting the 'Index field name' to 'Unit' would configure the program to search the 'Unit' column.
Column 1 (Unit) could then contain all the unit names after the first row. e.g. Crusher, Shredder, Mixer, etc.

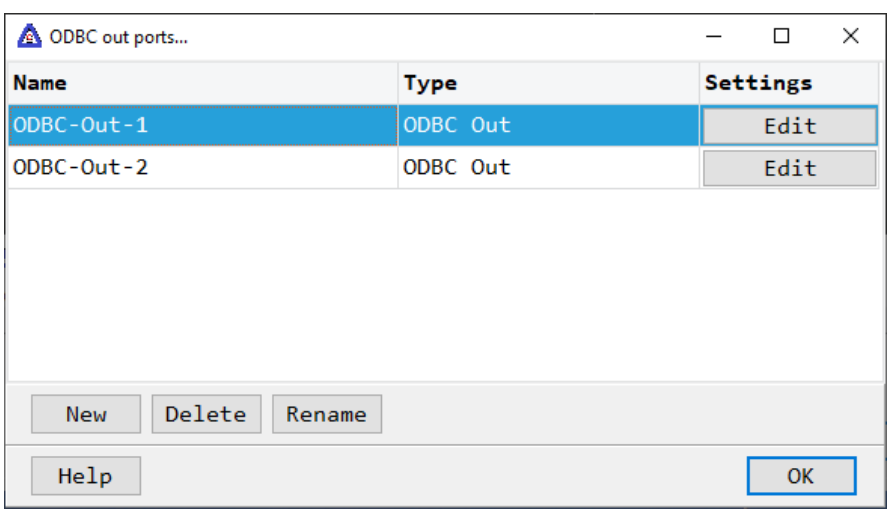
In the In/out points the 'Row' is the name to search for in the 'Index field name'. In this example, 'Unit'.

ODBC Out

Note: It is suggested that all the fields in the database be set to 'Text'.

ODBC (Open Database Connectivity) provides a method to connect with many different databases via one interface type.

Each ODBC Out object is listed in the window.



The HMI divides connections to write point status (out) from connections to collect point data (in).

The ODBC Out provides a connection to write the dynamic values of analog points to one database and digital points to a separate database. All points can be written or only selected points. The number of database connections is not limited. This allows for the data to be segmented or shared.

For analog points the current value, percent full scale and alarm points are written to the database.

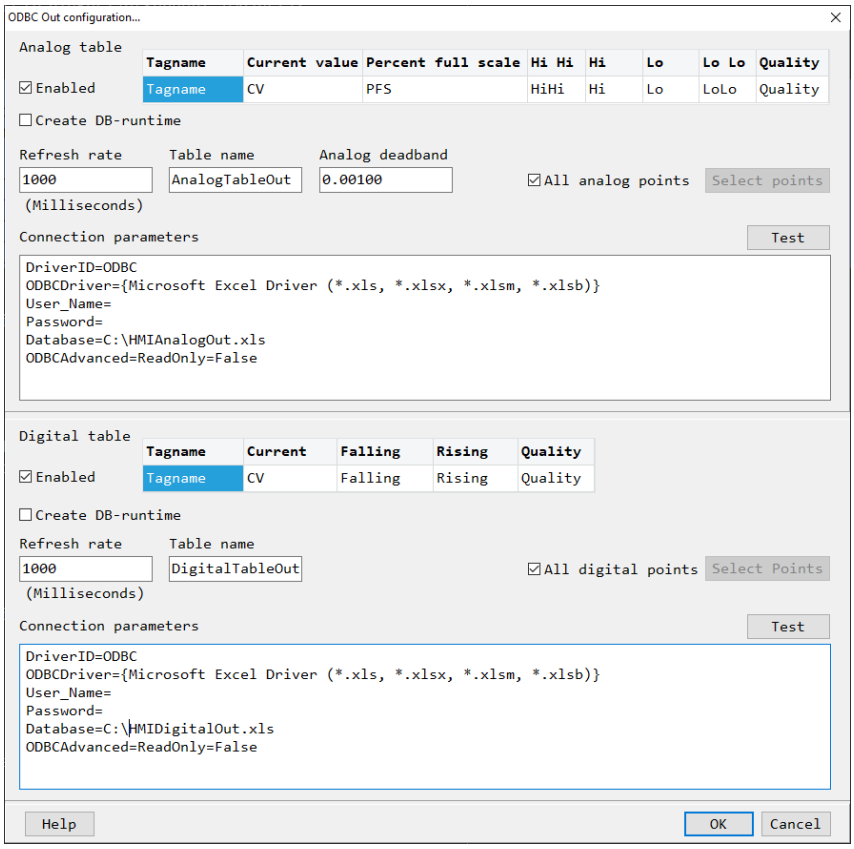
For digital points the tagname, current value, point quality and alarm points are written to the database.

The number of database connections is not limited. This allows for the data to be segmented. For analog and digital points only the tagname and current value are required.

Some example files/database are in the '..\Database Templates\' directory.

WARNING: Do not change the order or value of the first column in the database after runtime monitoring has started.

Settings



Analog and Digital table

Each table has several fields. Each column must have a name and it must be unique for the table.

The column/field names cannot be any word reserved by the database. (e.g. Date, Time, Add, Counter)

Analog table

Field name	Type
Tagname	string
Current value	float
Percent full scale	float
Hi Hi	Boolean
Hi	Boolean
Lo	Boolean
Lo Lo	Boolean
Quality	Boolean

Digital table

Field name	Type
Tagname	string
Current value	Boolean
Falling	Boolean
Rising	Boolean
Quality	Boolean

Enabled

If enabled at runtime, a connection will be made to the database.

Table name

This is the “table name” in the database. For Excel, this is the sheet name and cannot be “Sheet1, “Sheet2” or “Sheet3”.

All analog points/All digital points

Analog/digital point status will be monitored and written to the database upon a change of value. If enabled all of the points will be monitored. If not enabled. the point(s) to monitor must be selected via the "Select Points" button.

Create DB-runtime

Excel: If enabled the database file will be deleted, if present, and a new one created with the correct column headings. Excel is not a database. It has some of the features

of a database. The database file should only contain the header and table name definition. The HMI will not save any other data to the file at configuration or runtime. The “table name” cannot be “Sheet1, “Sheet2” or “Sheet3”.

For ODBC In, other programs should not modify the database or database definition.

WARNING: Excel does not allow the editing of sheets via ODBC while the file is open in Excel. The error messages returned by Excel will not indicate the correct solution - quit Excel.

Microsoft Access:
No records are deleted. The table should only contain the desired points as configured. Rows with non-configured data are, not efficient.

Analog deadband

This value is the minimum difference between the old value and current value to cause the change logic to execute.

Floating point numbers are represented in binary fractions and therefore are always an approximation of a decimal fraction.

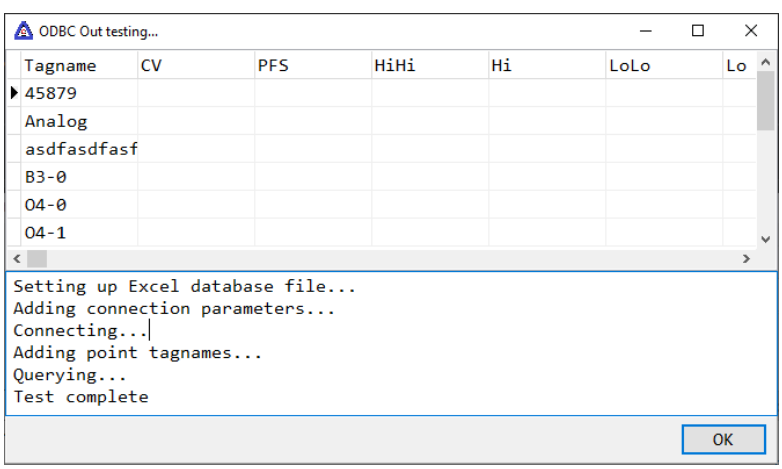
That is why occasionally a value like 1.980000000001 is displayed for something that would be expected to have only a few decimal places of precision. When comparing two floating point values, the comparison should be on how close the two values are to each other rather than testing for equality.

AD = 0.001 (default)
Change = ((a - b) >= AD)

Connection parameters

Refer to the section below.

Test



Selecting this button will attempt to create a connection to the database, set up the table and write the configured tagnames to the database. WARNING: Verify the database file is not use before selecting the 'Test' button.

Excel: An attempt is made to delete any file with the database name and recreate the file with the correct headers.

Connection Parameters

Each database may have additional connection parameters. Please refer to the database supplier for required information.

The required connection parameters vary by connection type.

Direct connect to the data base

At least these three parameters must be present for direct connection to a database via ODBC.

DriverID
ODBCDriver
Database

Connect via DSN (Data source name) entry to database (ODBC Data Source Administrator)

DataSource
The format for each line is a name/value pair and the format is name=value

Note:
1. The Database file path is an example.

Examples:

Excel

DriverID=ODBC
ODBCDriver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *.xlsb)}
User_Name=
Password=
Database=C:\HMIAnalogOut.xls
ODBCAdvanced=ReadOnly=False

Notes for Excel
1. The data will be placed on the first sheet.

2. Generally, Excel does not allow the editing of sheets via ODBC while the file is open in Excel. The error messages returned by Excel will not indicate the correct solution - quit Excel.
3. When running the HMI and the spread sheet opens or becomes visible, a copy of Excel is running. Use the 'Task Manager' to stop all Excel processes.
4. If an error appears: 'Data source name not found and no default driver specified.' the Microsoft OBDC driver is not installed. This link might work: https://www.microsoft.com/en-US/download/details.aspx?id=13255

Access

DriverID=ODBC
ODBCDriver={Microsoft Access Driver (*.mdb, *.accdb)}
User_Name=
Password=
Database=Y:\HMIAnalogOut.mdb
ODBCAdvanced=ReadOnly=False

Notes for Access
1. Files with extension "mdb" are for Access 2003 and files with extension "accdb" are for Access 2007 or later
2. Example using Universal Naming Convention (UNC) format with file sharing enabled - Database=\\192.168.1.3\Testing\HMIAnalogOut.accdb
3. Using a mapped drive with file sharing enabled - Database=Y:\HMIAnalogOut.accdb

MySql

Local

DriverID=ODBC
ODBCDriver=MySQL ODBC 5.1 Driver
User_Name=HMILocal
Password=local
Database=hmidb

Remote

DriverID=ODBC
ODBCDriver=MySQL ODBC 5.1 Driver
User_Name=HMIRemote
Password=remote
Database=hmidb
ODBCAdvanced=SERVER=192.168.1.3

Notes for MySQL
1. The user name and password are case sensitive.

Microsoft SQL Express

These examples used Windows Authentication.

Local

DriverID=ODBC
ODBCDriver=SQL Server
Database=hmidb
ODBCAdvanced=SERVER=S2003\SQLEXPRESS

Remote

DriverID=ODBC
ODBCDriver=SQL Server
Database=hmidb
LoginTimeout=10
ODBCAdvanced=SERVER=192.168.1.68\SQLEXPRESS

These examples used SQL Authentication mode.

Local

DriverID=ODBC
ODBCDriver=SQL Server
Database=hmidb
User_Name=HMILocal
Password=local
ODBCAdvanced=SERVER=S2003\SQLEXPRESS

Remote

DriverID=ODBC
ODBCDriver=SQL Server
Database=hmidb
User_Name=HMIRemote
Password=remote
LoginTimeout=10
ODBCAdvanced=SERVER=192.168.1.68\SQLEXPRESS

Notes for Microsoft SQL Express

1. In the example, S2003 is the computer name and SQLEXPRESS is the SQL server name (SQL instance name).
2. If using the "Windows Authentication" to authorize access do not supply a User_Name or Password.

ODBC In/Out commands

Notes:
#1 Unless noted these commands operate on 'in memory' data only. Changes are not saved to the project.
#2 These commands only work on a port that is not enabled. Ports that are enabled are always connected at runtime issuing read/writes. The underlying data structure cannot be changed while the port is enabled.

ODBCAddToInList

This adds a read to the 'in' list of commands.

ODBCAddToOutList

This adds a write to the 'out' list of commands.

ODBCClearInList

This clears the 'in' list of commands. All reads from the database to points and global strings are deleted.

ODBCClearOutList

This clears the 'out' list of commands. All writes to the database from points and global strings are deleted.

ODBCIssueRead

This will connect to the database and read all the commands in the 'in' list.

ODBCIssueWrite

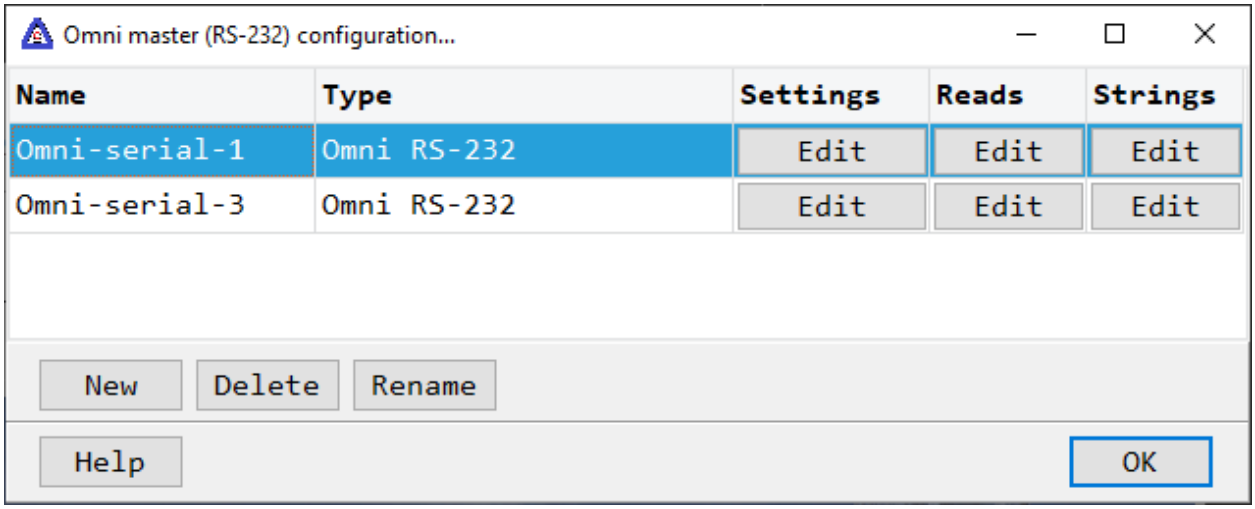
This will connect to the database and write all the commands in the 'out' list.

ODBCSetTableName

This is used to change the table to access in the database.

Omni Serial

Each Omni serial master object is listed in the window.

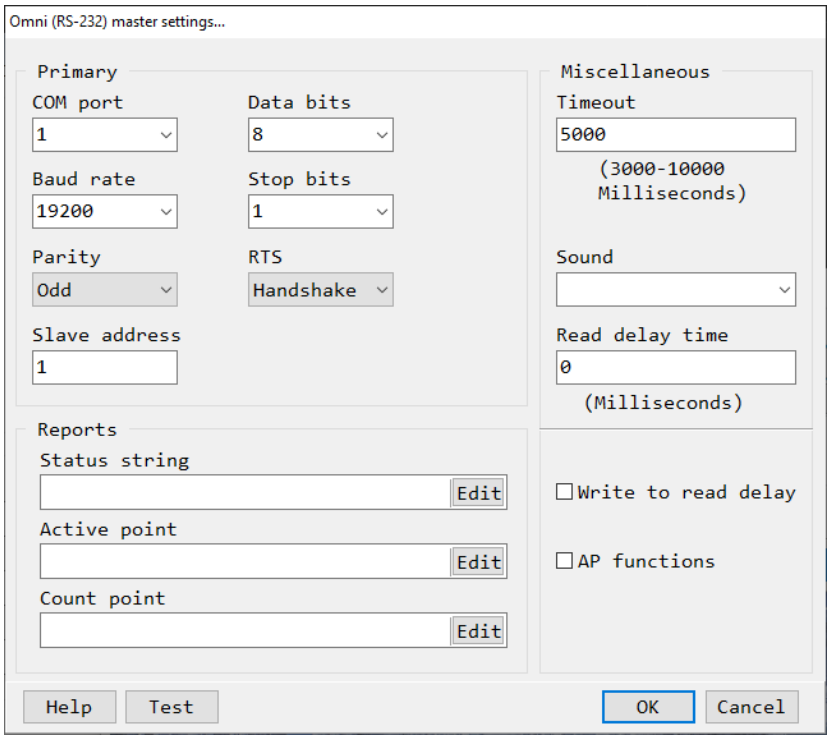


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an Omni serial master object select the "Delete" button.

Settings



Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Reports

Status string (optional)

If a script global is supplied it will be updated with various strings indicating an action while a report is being collected.

Active point (optional)

If a digital point is defined it will be set true when report collection is initiated and set false when the collection ends. If more than one report is queued, the flag will stay true while the reports are collected.

Count (optional)

The 'Historical Alarm/Audit Trail' report retrieves 'count' records from the flow computer. If an analog point is defined its current value is used. The alarm record count is limited to 500 records. The audit trails record count is limited to 150 records.

If the count is zero (0) or the point is not defined, 25 records will be retrieved.

If then count is less than zero (0) or greater than the report limit the report will not be retrieved.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

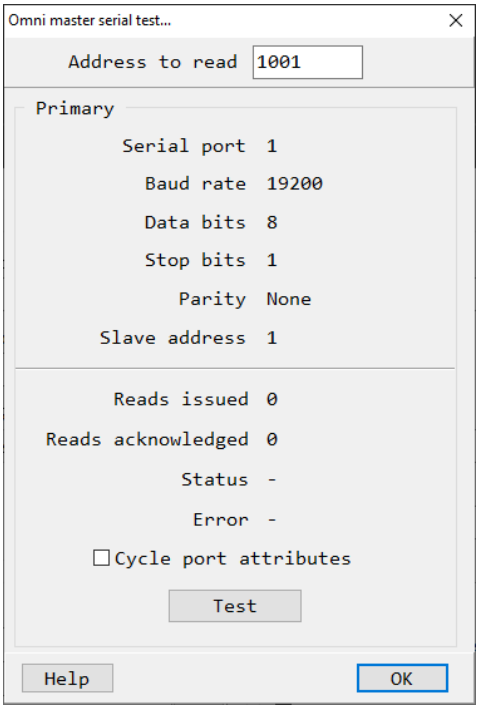
Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

AP functions

See analog functions.

Test button

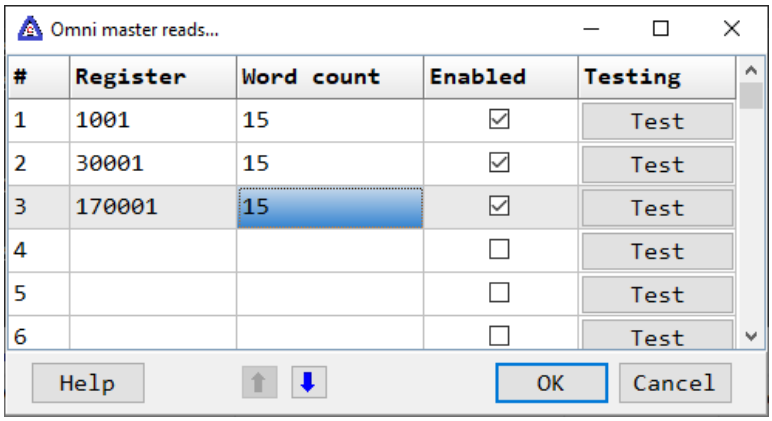


When the test button is selected the program will attempt to read one element of data from the device at the address entered.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads



The address ranges shown may or may not be present in the slave device.

Register

The address must be a valid Enron address.
Any of the following addresses are valid:

Register	Data type
1001 - 1999	Booleans
2001 - 2999	Booleans
3001 - 3999	16 bit short integers
4001 - 4999	8 character strings
5001 - 5999	32 bit long integers
6001 - 6999	32 bit floats
7001 - 7999	32 bit floats
8000 - 8999	32 bit floats
13001 - 13999	16 bit short integers
14001 - 14999	16 character strings
15001 - 15999	32 bit long integers
17001 - 17999	32 bit floats
18001 - 18999	32 bit floats
19001 - 19999	32 bit floats

Note: Do not configure reads that overlap. Two reads that contain the same address will cause the second address to be ignored.

Count

The count is the number of a 'type' to read. Each read can request up to 125 words of data. Each data type has a limit.

Data type	Max count per request
Booleans	2000
16 bit short integers	125
8 character strings	31 (Each element contains 2 characters)
32 bit long integers	62

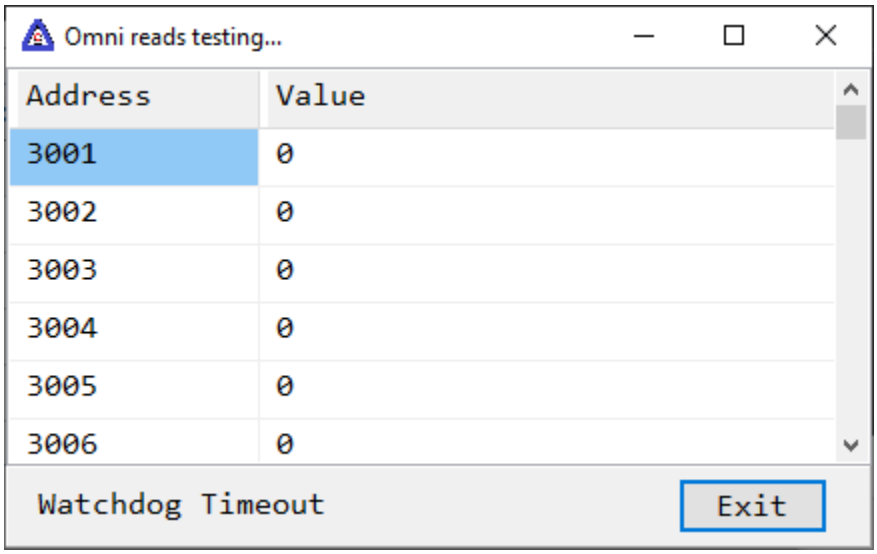
32 bit floats	62
16 character strings	15 (Each element contains 2 characters)

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test

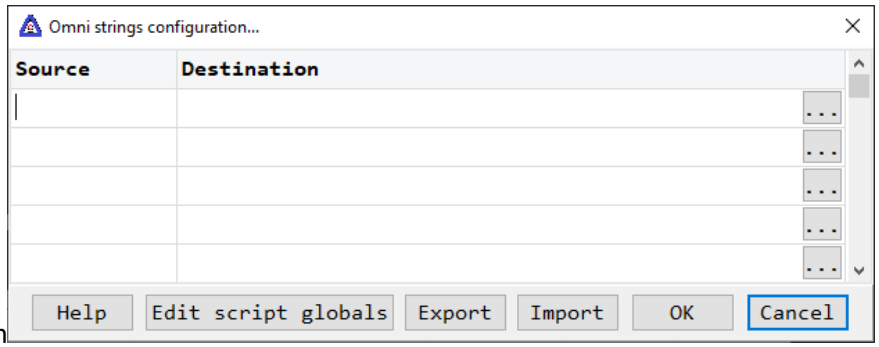


When the button is selected the program will attempt to access the data in the device using the communication parameters configured.

Error messages

No register address	The value in the field is not a Enron address.
Invalid count	The count must be between 1 and 2000.
Exceeds 999	The register plus the count exceeds the maximum address range.
Invalid Enron address	The value in the field is an invalid Enron address.

Strings



Om

Strings are not regular points. They do not have alarms, printing, etc. Strings may be displayed in any window via the "Script Global" animation. Any other actions, parsing, comparing, etc. must be done in scripts.

Source

This is the register address in the PLC.

Destination (optional)

If desired, select a script global location and the string will be copied to the location when the string value is returned by the external device. If this option is used or not the string may still be accessed via the "StringGet" and "StringSet" script commands. If this is configured and access to the string, in a script, is desired use the "GlobalGet" or "StringGet" to copy the string. The format of the destination is: <section>.<name>.
Note: Writing to the script global does not write the value to the PLC. "StringSet" must be used to write a string to the PLC.

Export

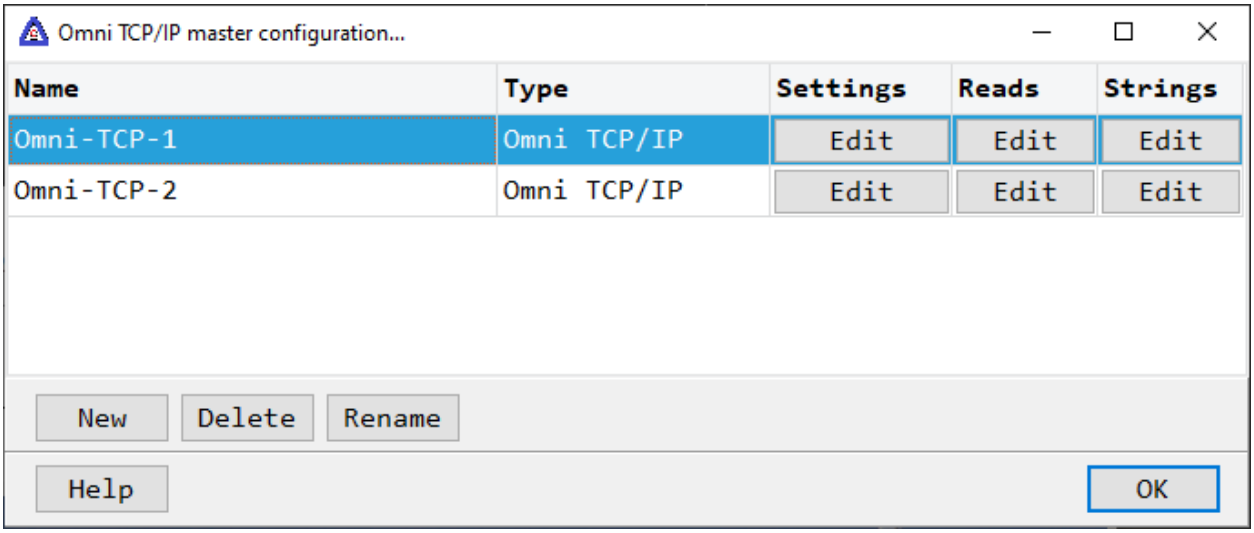
This is used to export the configuration to an Excel worksheet.

Import

This is used to import string configurations from an Excel worksheet. The contents of the grid are completely replaced by the contents of the file.

Omni TCP/IP

Each Omni TCP/IP master object is listed in the window.

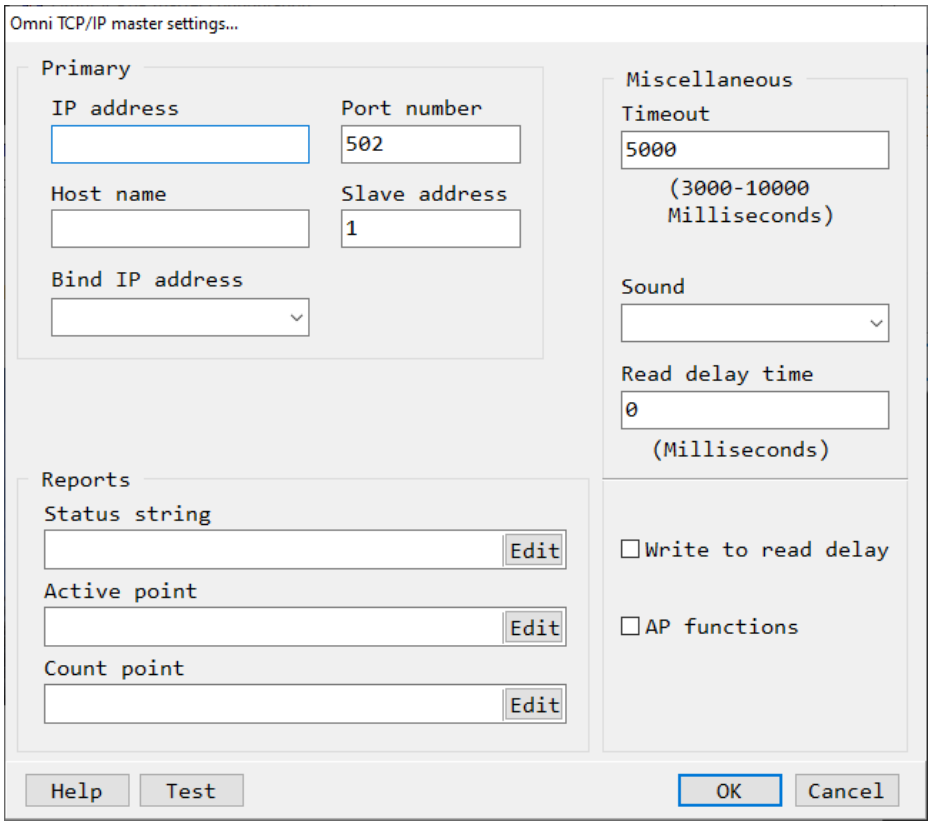


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an Omni TCP/IP master object select the "Delete" button.

Settings



Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Reports

Status string (optional)

If a script global is supplied it will be updated with various strings indicating an action while a report is being collected.

Active point (optional)

If a digital point is defined it will be set true when report collection is initiated and set false when the collection ends. If more than one report is queued, the flag will stay true while the reports are collected.

Count (optional)

The 'Historical Alarm/Audit Trail' report retrieves 'count' records from the flow computer. If an analog point is defined its current value is used. The alarm record count is limited to 500 records. The audit trails record count is limited to 150 records.

If the count is zero (0) or the point is not defined, 25 records will be retrieved.

If then count is less than zero (0) or greater than the report limit the report will not be retrieved.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

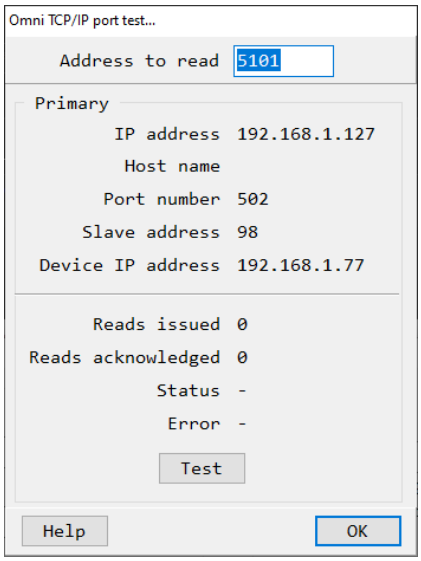
Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

AP functions

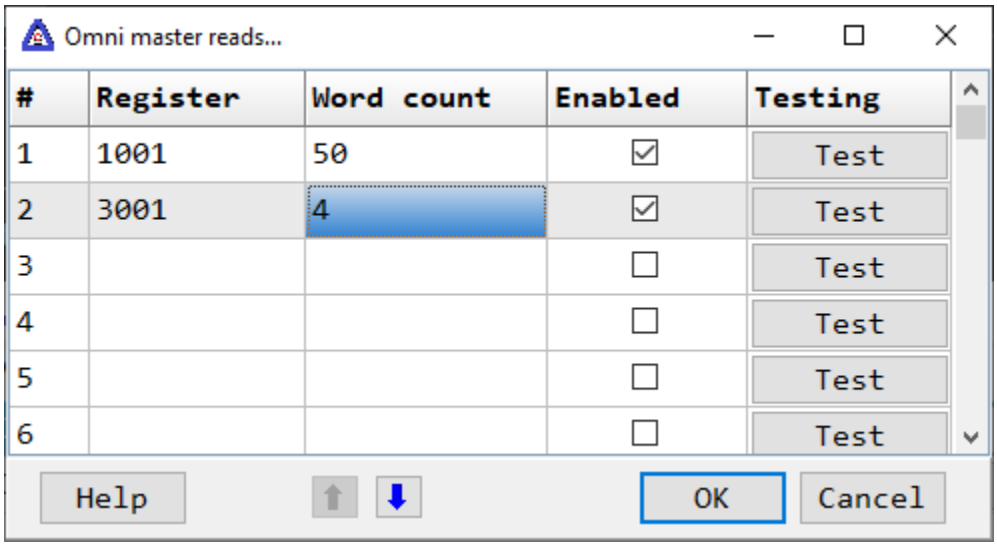
See analog functions.

Test button



When the test button is selected the program will attempt to read one element of data from the device at the address entered.

Reads



The address ranges shown may or may not be present in the slave device.

Register

The address must be a valid Enron address.
Any of the following addresses are valid:

Register	Data type
1001 - 1999	Booleans
2001 - 2999	Booleans
3001 - 3999	16 bit short integers
4001 - 4999	8 character strings
5001 - 5999	32 bit long integers
6001 - 6999	32 bit floats
7001 - 7999	32 bit floats
8000 - 8999	32 bit floats
13001 - 13999	16 bit short integers
14001 - 14999	16 character strings
15001 - 15999	32 bit long integers
17001 - 17999	32 bit floats
18001 - 18999	32 bit floats
19001 - 19999	32 bit floats

Note: Do not configure reads that overlap. Two reads that contain the same address will cause the second address to be ignored.

Count

The count is the number of a 'type' to read. Each read can request up to 125 words of data. Each data type has a limit.

Data type	Max count per request

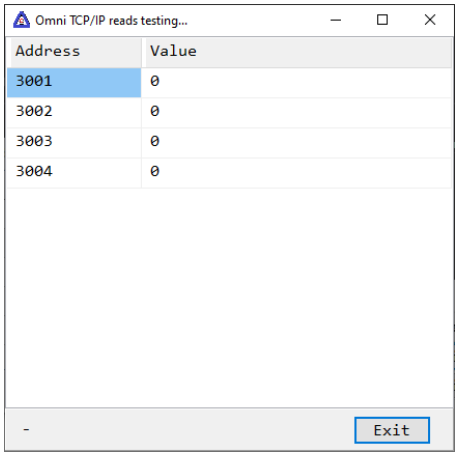
Booleans	2000
16 bit short integers	125
8 character strings	31 (Each element contains 2 characters)
32 bit long integers	62
32 bit floats	62
16 character strings	15 (Each element contains 2 characters)

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test

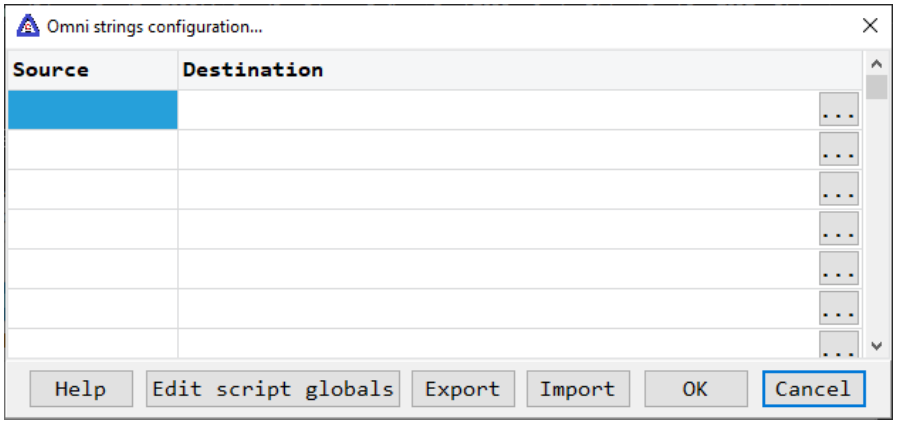


When the button is selected the program will attempt to access the data in the device using the communication parameters configured.

Error messages

No register address	The value in the field is not a Enron address.
Invalid count	The count must be between 1 and 2000.
Exceeds 999	The register plus the count exceeds the maximum address range.
Invalid Enron address	The value in the field is an invalid Enron address.

Strings



Strings are not regular points. They do not have alarms, printing, etc. Strings may be displayed in any window via the "Script Global" animation. Any other actions, parsing, comparing, etc. must be done in scripts.

Source

This is the register address in the PLC.

Destination (optional)

If desired, select a script global location and the string will be copied to the location when the string value is returned by the external device. If this option is used or not the string may still be accessed via the "StringGet" and "StringSet" script commands. If this is configured and access to the string, in a script, is desired use the "GlobalGet" or "StringGet" to copy the string. The format of the destination is: <section>.<name>.
Note: Writing to the script global does not write the value to the PLC. "StringSet" must be used to write a string to the PLC.

Export

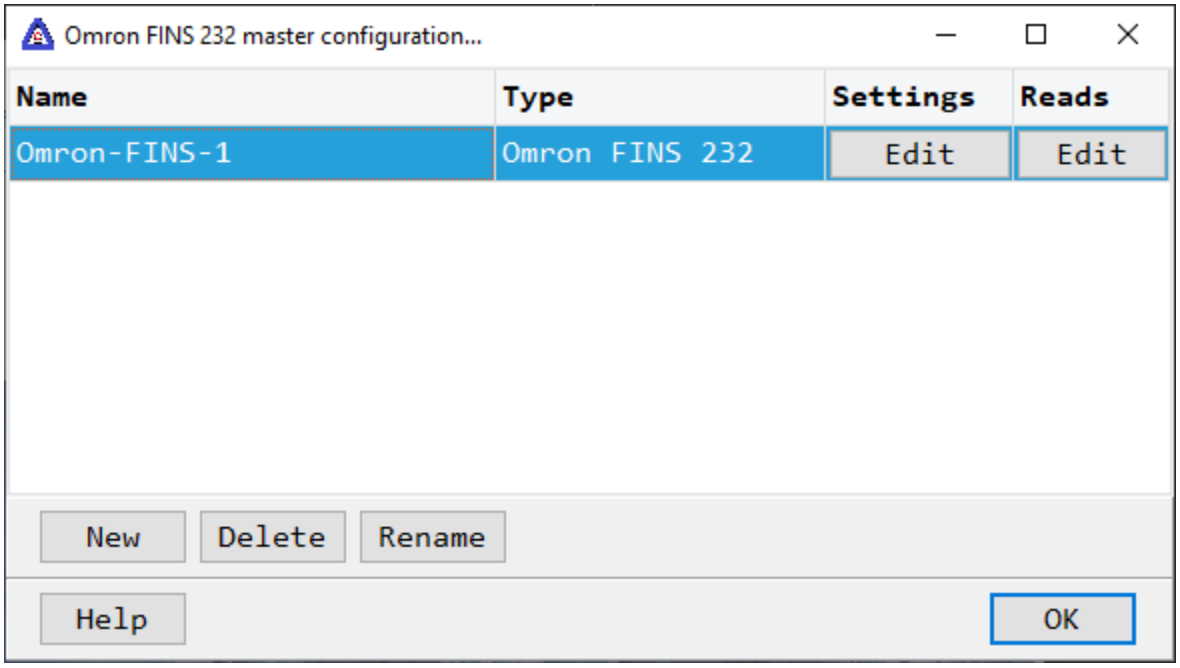
This is used to export the configuration to an Excel worksheet.

Import

This is used to import string configurations from an Excel worksheet. The contents of the grid are completely replaced by the contents of the file.

Omron FINS Serial

Each Omron FINS serial master object is listed in the window.

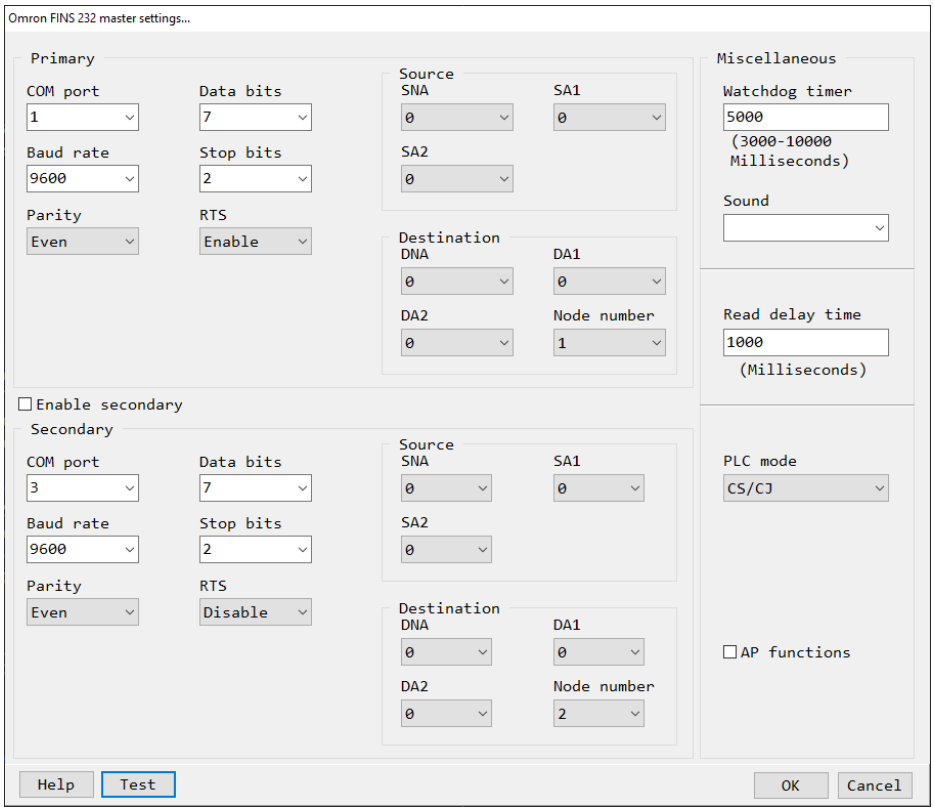


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an Omron FINS serial master object select the "Delete" button.

Settings



The serial port has a primary port and if enabled, a secondary port. Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Enable secondary

The "Enable secondary" checkbox provides for a second com port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary com port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Source/Destination

These are normally all set to zero and adjusted by the PLC if needed to route the message.

SNA	Source network address
SA1	Source node number
SA2	Source unit number
DNA	Destination network address
DA1	Destination number
DA2	Destination unit number
Node number	The node number of the destination. The value is decimal.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

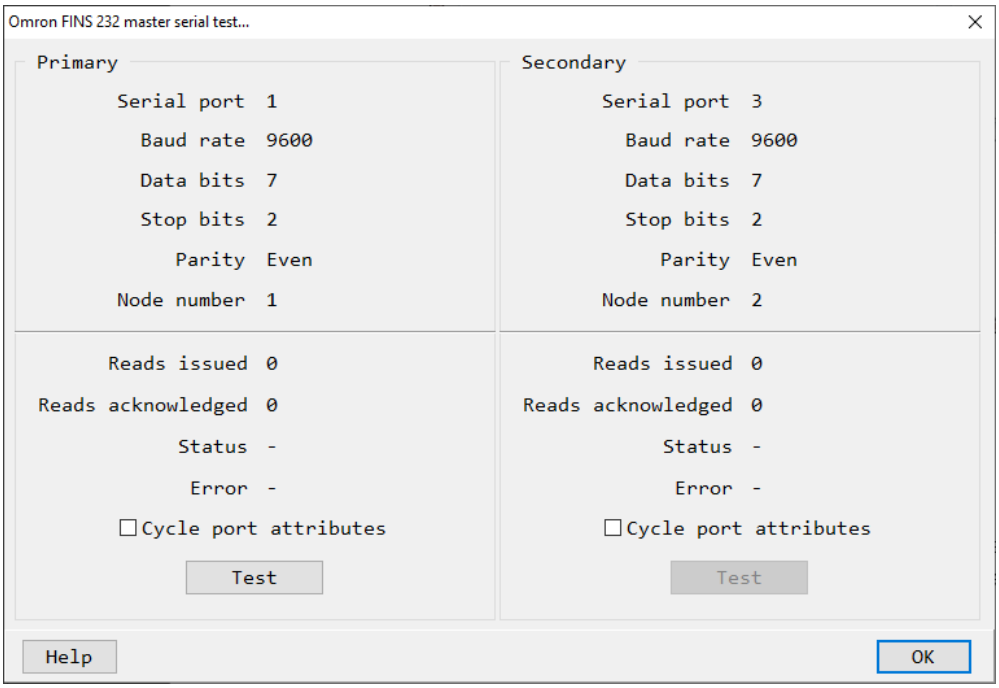
PLC mode

The driver supports 'C', 'CV' and 'CS/CJ' modes.

AP functions

See analog functions.

Test button

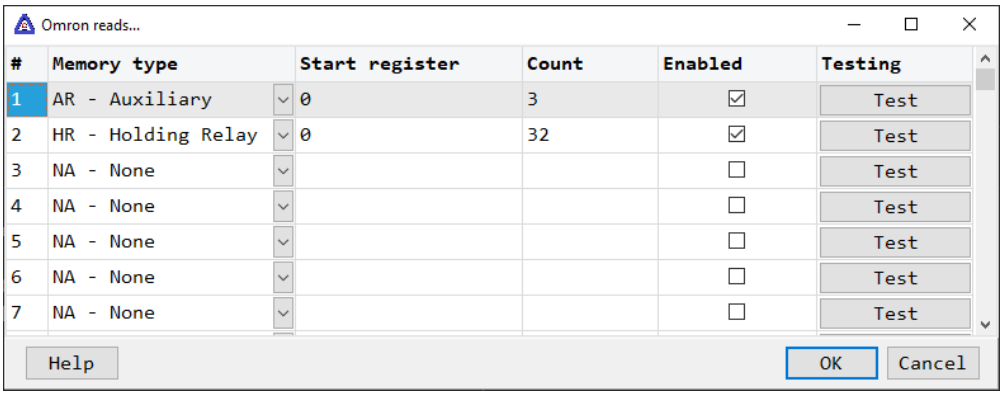


When the test button is selected the program will send a read request to read one word starting at CIO 0.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave. All values are in decimal.

Register Type

Register Name	Prefix	Examples	Data Type
Action Flags	AC	ACxxx	Boolean
Auxiliary Area	A	Axxx, Axxx.yy	Word, double word, float, Boolean
Auxiliary Relay	AR	ARxxx, ARxxx.yy	Word, double word, float, Boolean
Bus Link	G	Gxxx, Gxxx.yy	Word, double word, float, Boolean
Core I/O Area	CIO	CIOxxx, CIOxxx.yy	Word, double word, float, Boolean
Counter	C	Cxxx	Boolean
Counter PV	CV	CVxxx	Word
Data Memory	DM	DMxxx, DMxxx.yy	Word, double word, float, Boolean
Data Register	DR	DRxxx, DRxxx.yy	Word, double word, float, Boolean
Expansion Data Memory	E	Eaa:xxx, Eaa:xxx.yy	Word, double word, float, Boolean
Holding Relay	HR	HRxxx, HRxxx.yy	Word, double word, float, Boolean
Index Register	IR	IRxxx, IRxxx.yy	Word, double word, float, Boolean
Link Relay	LR	LRxxx, LRxxx.yy	Word, double word, float, Boolean
Step Timer	ST	STxxx	Boolean
Step Timer PV	SV	SVxxx	Word
Task Flag	TK	TKxxx.yy	Boolean
Temporary Relay	TR	TRxxx, TRxxx.yy	Word, double word, float, Boolean
Timer	T	Txxx	Boolean
Timer PV	TV	TVxxx	Word
Transition Area	TN	TNxxx	Boolean
Work Area	WR	WRxxx, WRxxx.yy	Word, double word, float, Boolean

Notes:

1. Not all devices support all addressing types, register types or PLC mode.

2. Omron does not provide for single bit writes in some register types e.g. Data memory. A bit write will set the desired bit and clear all other bits in the register. Only using the true state for commands and grouping all commands in a word or words not containing status bits overcomes this limitation. Index register does not support bit writes.
3. Each bit reference must use 2 decimal places. For bits 0 to 9 use a leading 0. i.e. 01, 02, 09

Start register

The starting register to read. This is the word address of the memory area.

Count

The number of registers to read for the request in decimal. Each register is 16 bits. The protocol has a maximum of 29 words of data per request.

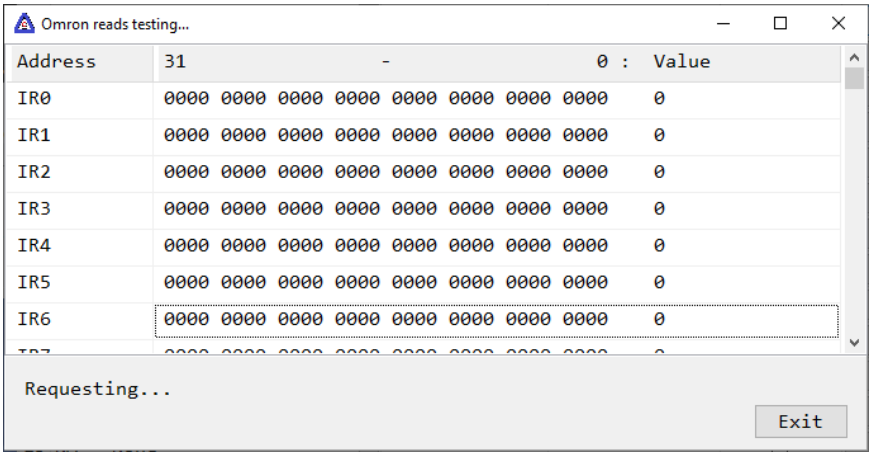
The timer/counter state is limited to 3 words (48 bits). Each state bit is returned as one byte.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default, the primary configuration

is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. See 'Testing error messages' below.

Read configuration error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.
Timer/Counter limit 3 words (48 status bits)	Timer/Counter status bits are returned in one byte each timer/counter.

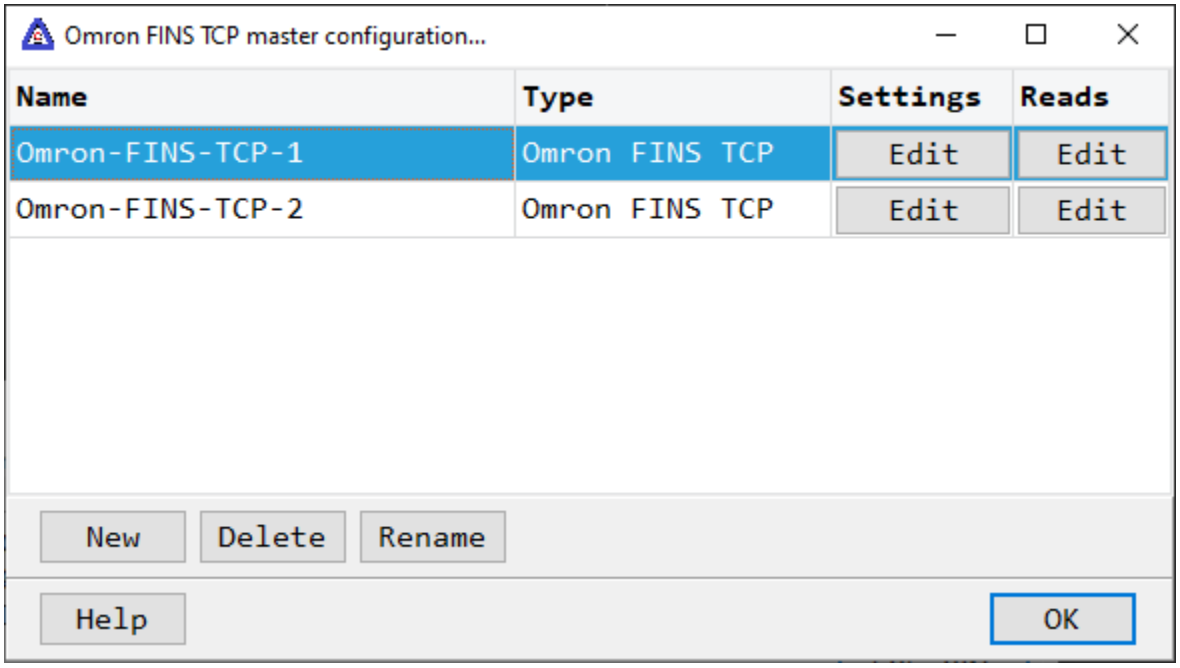
Testing error messages

Service was interrupted
Local node not part of Network
Token time-out, node number too large
Number of transmit retries exceeded
Maximum number of frames exceeded
Node number setting error (range)
Node number duplication error
Destination node not part of Network
No node with the specified node number
Third node not part of Network/Broadcasting was specified
Busy error, destination node busy
Response time-out, message packet was corrupted by noise/Response time-out, response watchdog timer interval too short/Frame lost in transmission
Error occurred in the communications controller, ERC indicator is lit
CPU error occurred in the PC at destination node
A controller error has prevented normal response from being returned.
Node number setting error
An undefined command has been used.
Cannot process command because the specified unit model or version is wrong.
Destination node number is not set in the routing table.
Routing table isn't registered.
Routing table error
The maximum number of relay nodes (2) was exceeded in the command.
The command is longer than the max. permissible length.
The command is shorter than min. permissible length.
The designated number of data items differs from the actual number.
An incorrect command format has been used.

An incorrect header has been used. (The local node's relay table or relay node's local network table is wrong.)
A correct memory area code has not been used or Expansion Data Memory is not available.
The access size specified in the command is wrong, or the first address is an odd number.
The first address is in an inaccessible area.
The end of specified word range exceeds the acceptable range.
A non-existent program no. has been specified.
The sizes of data items in the command block are wrong.
The IOM break function cannot be executed because it is already being executed.
The response block is longer than the max. permissible length.
An incorrect parameter code has been specified.
The data is protected. An attempt was made to download a file that is being uploaded.
The registered table does not exist or is incorrect./Too many files open.
The corresponding search data does not exist.
A non-existing program no. has been specified.
A non-existing file has been specified.
A verification error has occurred.
The specified area is read-only or is write-protected.
The data is protected. An attempt was made to simultaneously download and upload a file. The data link table cannot be written manual because it is set for automatic generation.
The number of files exceeds the maximum permissible. Too many files open.
A non-existing program no. has been specified.
A non-existent file has been specified.
The specified file already exists.
Data cannot be changed.
The mode is wrong (executing). Data links are active.
The mode is wrong (stopped). Data links are active.
The PC is in the PROGRAM mode.
The PC is in the DEBUG mode.
The PC is in the MONITOR mode.
The PC is in the RUN mode.
The specified node is not the control node.
The mode is wrong and the step cannot be executed.
A file device does not exist where specified.
The specified memory does not exist.
The data link table either hasn't been created or is incorrect.

Omron FINS TCP

Each Omron FINS TCP master object is listed in the window.

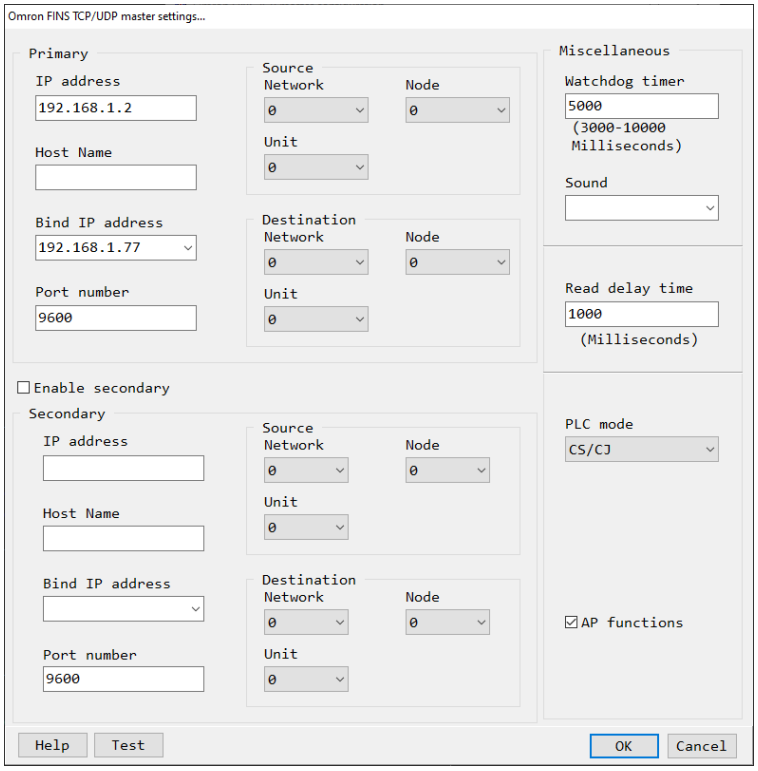


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an Omron FINS TCP master object select the "Delete" button.

Settings



If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second com port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary com port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Node

The node is normally the same as the last digit of the IP address. Some devices allow zero (0) for automatic node assignment. The HMI uses the node value returned with the connection handshake.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

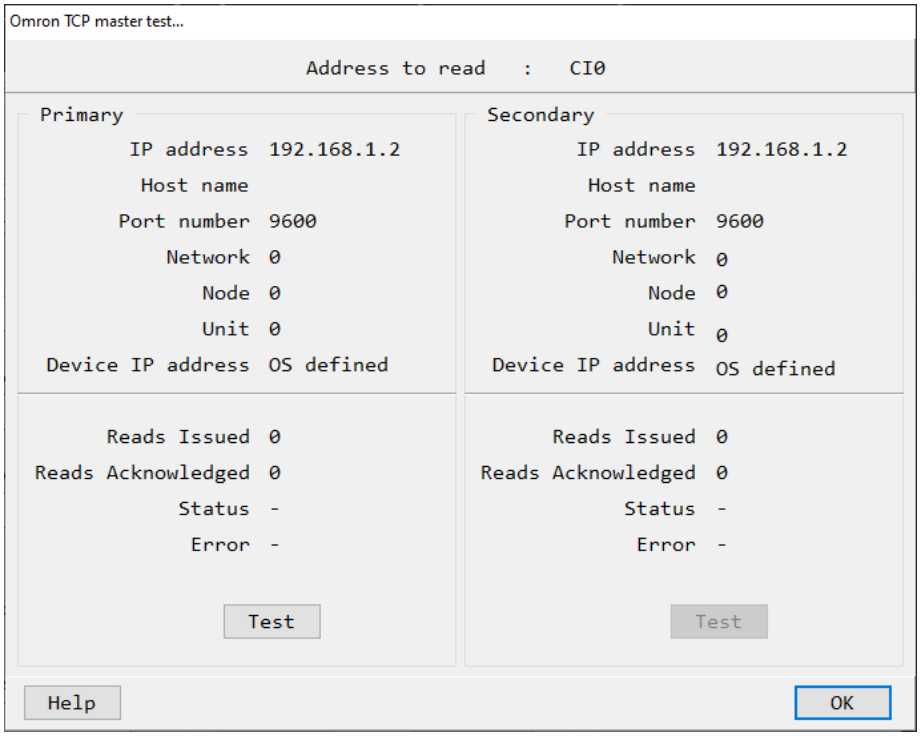
AP functions

See analog functions.

PLC mode

The driver supports 'C', 'CV' and 'CS/CJ' modes.

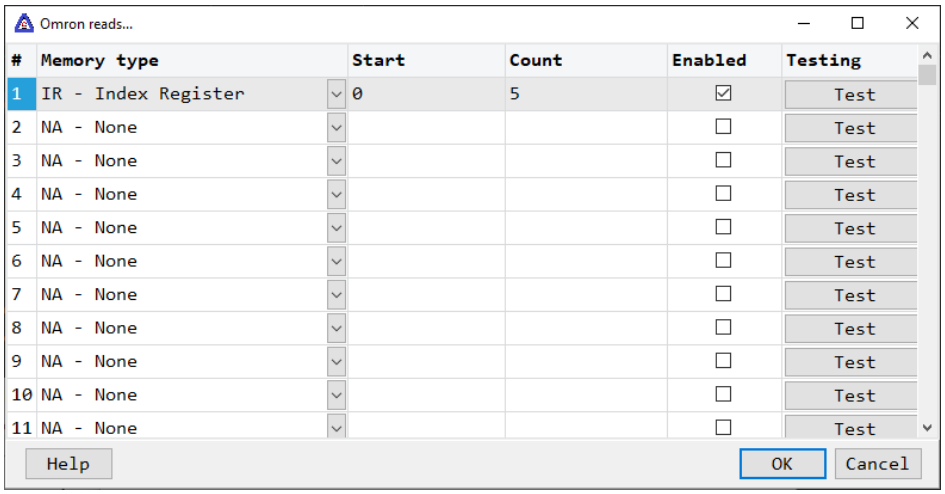
Test button



When the test button is selected the program will send a read request to read one word starting at CIO 0.

The program will attempt to use the communication parameters configured.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave. All values are in decimal.

Register Type

Register Name	Prefix	Examples	Data Type
Action Flags	AC	ACxxx	Boolean
Auxiliary Area	A	Axxx, Axxx.yy	Word, double word, float, Boolean
Auxiliary Relay	AR	ARxxx, ARxxx.yy	Word, double word, float, Boolean
Bus Link	G	Gxxx, Gxxx.yy	Word, double word, float, Boolean
Core I/O Area	CIO	CIOxxx, CIOxxx.yy	Word, double word, float, Boolean
Counter	C	Cxxx	Boolean
Counter PV	CV	CVxxx	Word
Data Memory	DM	DMxxx, DMxxx.yy	Word, double word, float, Boolean
Data Register	DR	DRxxx, DRxxx.yy	Word, double word, float, Boolean
Expansion Data Memory	E	Eaa:xxx, Eaa:xxx.yy	Word, double word, float, Boolean
Holding Relay	HR	HRxxx, HRxxx.yy	Word, double word, float, Boolean
Index Register	IR	IRxxx, IRxxx.yy	Word, double word, float, Boolean
Link Relay	LR	LRxxx, LRxxx.yy	Word, double word, float, Boolean
Step Timer	ST	STxxx	Boolean
Step Timer PV	SV	SVxxx	Word
Task Flag	TK	TKxxx.yy	Boolean
Temporary Relay	TR	TRxxx, TRxxx.yy	Word, double word, float, Boolean
Timer	T	Txxx	Boolean
Timer PV	TV	TVxxx	Word
Transition Area	TN	TNxxx	Boolean
Work Area	WR	WRxxx, WRxxx.yy	Word, double word, float, Boolean

Notes:

1. Not all devices support all addressing types, register types or PLC mode.
2. Omron does not provide for single bit writes in some register types e.g. Data memory. A bit write will set the desired bit and clear all other bits in the register. Only using the true state for commands and grouping all commands in a word or words not containing status bits overcomes this limitation. Index register does not support bit writes.
3. Each bit reference must use 2 decimal places. For bits 0 to 9 use a leading 0. i.e. 01, 02, 09

Start register

The starting register to read. This is the word address of the memory area.

Count

The number of registers to read for the request in decimal. Each register is 16 bits. The protocol has a maximum of 29 words of data per request.

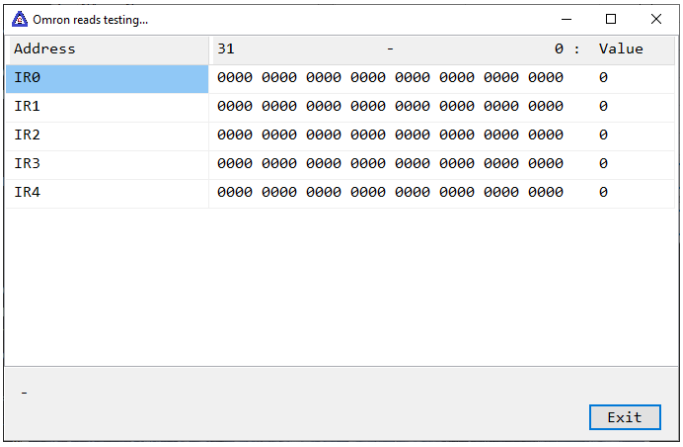
The timer/counter state is limited to 3 words (48 bits). Each state bit is returned as one byte.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration

is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. See 'Testing error message' below.

Read configuration error messages

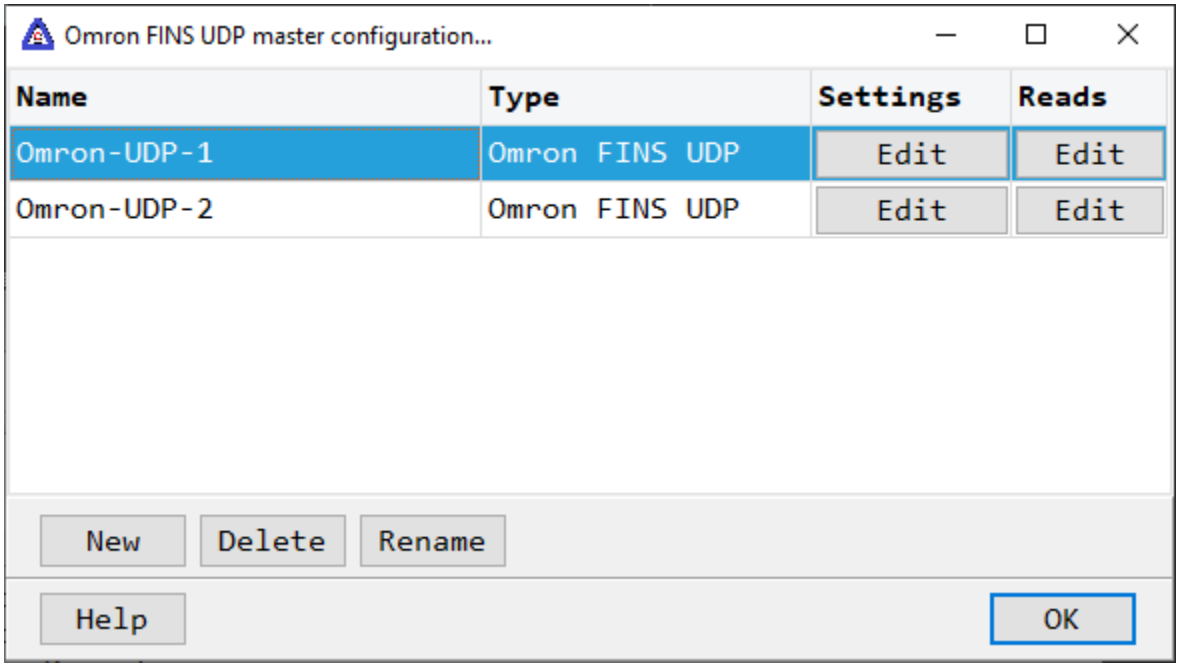
No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

Testing error messages

See here

Omron FINS UDP

Each Omron FINS UDP master object is listed in the window.

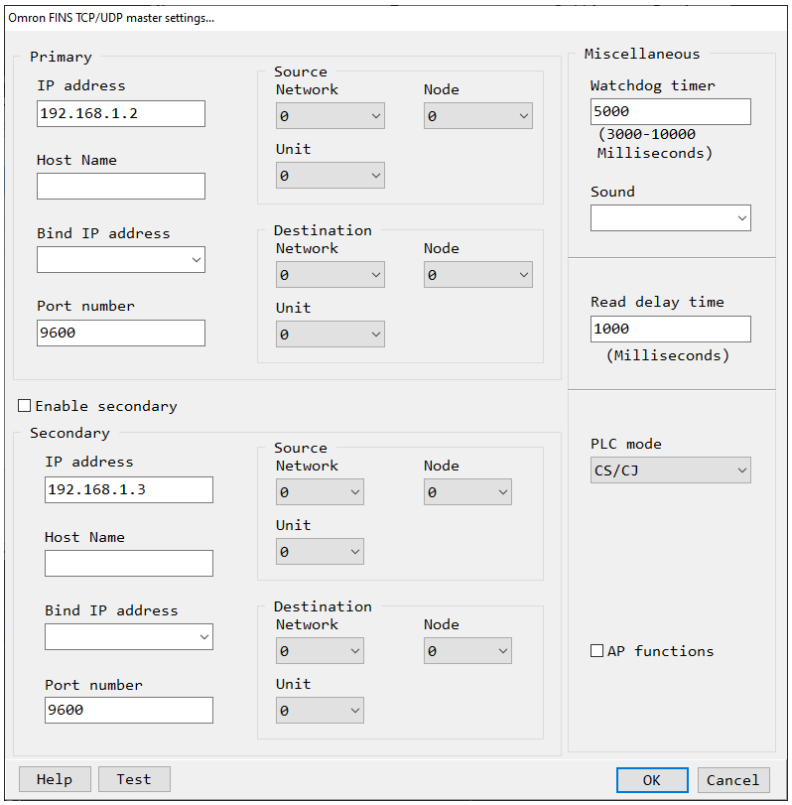


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an Omron FINS UDP master object select the "Delete" button.

Settings



If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second com port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary com port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Node number

The node is normally the same as the last digit of the IP address. Some devices allow zero (0) for automatic node assignment. The HMI uses the node value returned with the connection handshake.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

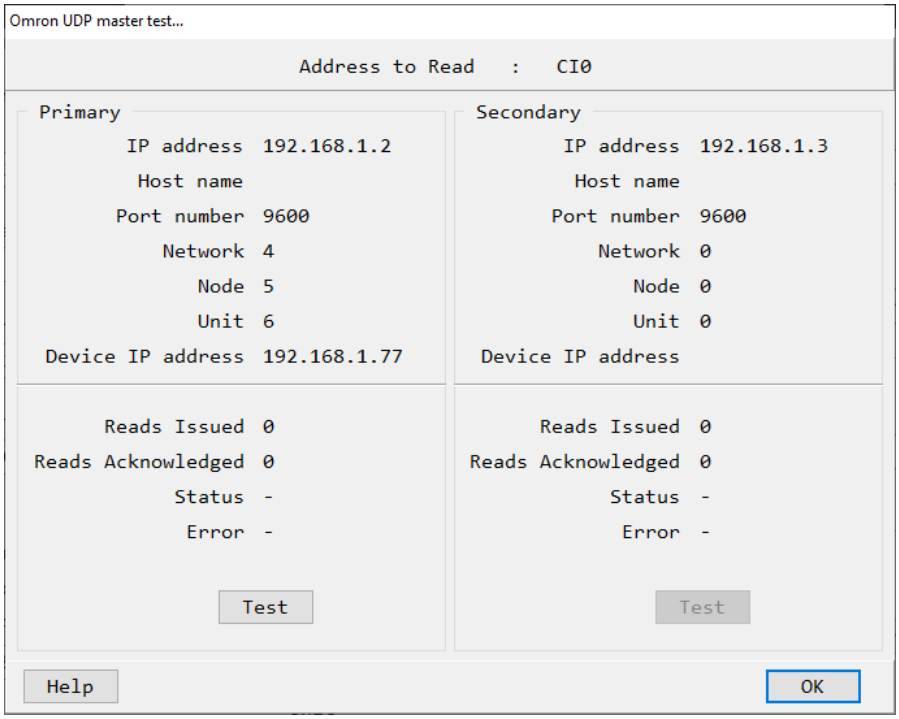
AP functions

See analog functions.

PLC mode

The driver supports 'C', 'CV' and 'CS/CJ' modes.

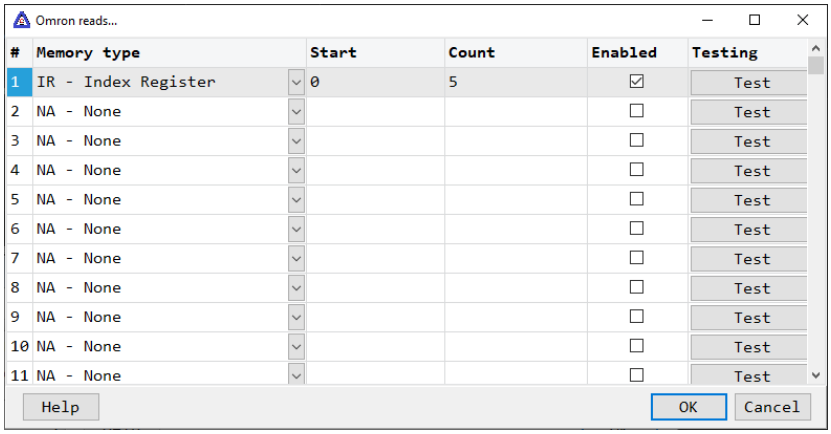
Test button



When the test button is selected the program will send a read request to read one word starting at CIO 0.

The program will attempt to use the communication parameters configured.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave. All values are in decimal.

Register Type

Register Name	Prefix	Examples	Data Type
Action Flags	AC	ACxxx	Boolean
Auxiliary Area	A	Axxx, Axxx.yy	Word, double word, float, Boolean
Auxiliary Relay	AR	ARxxx, ARxxx.yy	Word, double word, float, Boolean
Bus Link	G	Gxxx, Gxxx.yy	Word, double word, float, Boolean
Core I/O Area	CIO	CIOxxx, CIOxxx.yy	Word, double word, float, Boolean
Counter	C	Cxxx	Boolean
Counter PV	CV	CVxxx	Word
Data Memory	DM	DMxxx, DMxxx.yy	Word, double word, float, Boolean
Data Register	DR	DRxxx, DRxxx.yy	Word, double word, float, Boolean
Expansion Data Memory	E	Eaa:xxx, Eaa:xxx.yy	Word, double word, float, Boolean
Holding Relay	HR	HRxxx, HRxxx.yy	Word, double word, float, Boolean
Index Register	IR	IRxxx, IRxxx.yy	Word, double word, float, Boolean
Link Relay	LR	LRxxx, LRxxx.yy	Word, double word, float, Boolean
Step Timer	ST	STxxx	Boolean
Step Timer PV	SV	SVxxx	Word
Task Flag	TK	TKxxx.yy	Boolean
Temporary Relay	TR	TRxxx, TRxxx.yy	Word, double word, float, Boolean
Timer	T	Txxx	Boolean
Timer PV	TV	TVxxx	Word
Transition Area	TN	TNxxx	Boolean
Work Area	WR	WRxxx, WRxxx.yy	Word, double word, float, Boolean

Notes:

1. Not all devices support all addressing types, register types or PLC mode.
2. Omron does not provide for single bit writes in some register types e.g. Data memory. A bit write will set the desired bit and clear all other bits in the register. Only using the true state for commands and grouping all commands in a word or words not containing status bits overcomes this limitation. Index register does not support bit writes.
3. Each bit reference must use 2 decimal places. For bits 0 to 9 use a leading 0. i.e. 01, 02, 09

Start register

The starting register to read. This is the word address of the memory area.

Count

The number of registers to read for the request in decimal. Each register is 16 bits. The protocol has a maximum of 29 words of data per request.

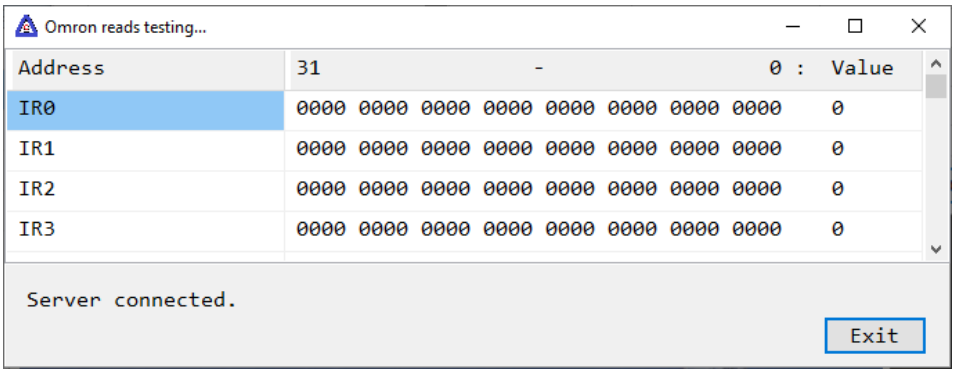
The timer/counter state is limited to 3 words (48 bits). Each state bit is returned as one byte.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. See 'Testing error message 'below.

Read configuration error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.

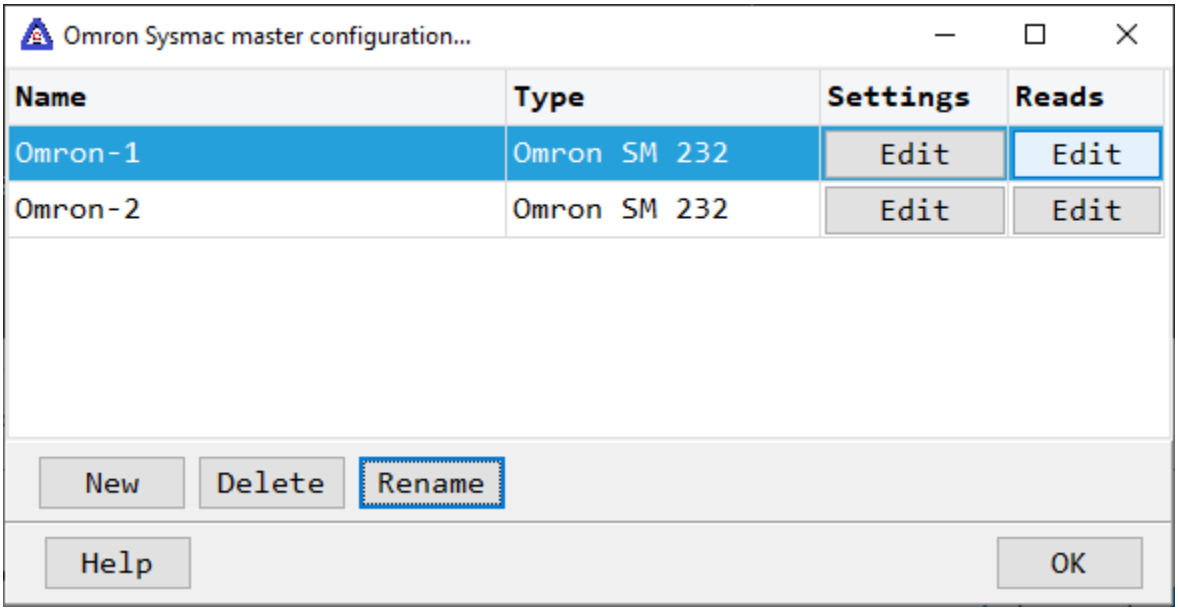
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

Testing error messages

See here

Omron Sysmac Serial

Each Omron SYSMAC serial master object is listed in the window.

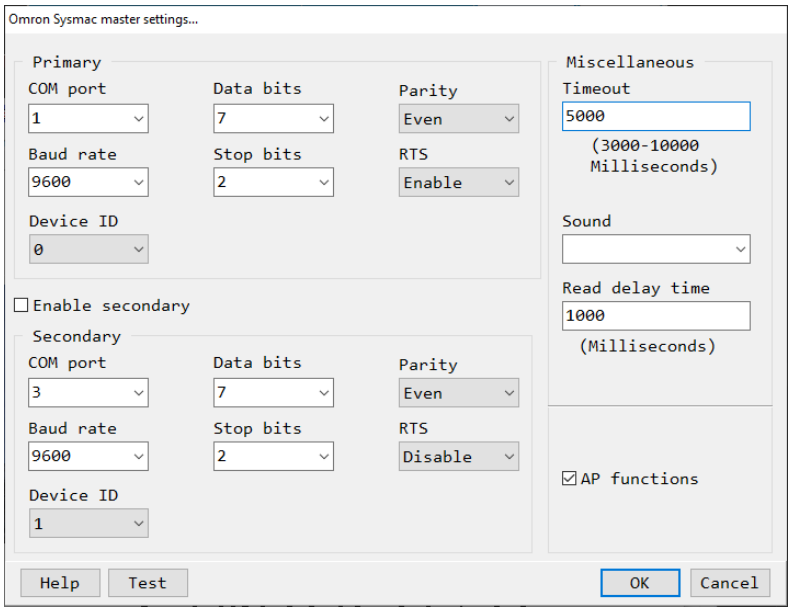


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an Omron SYSMAC serial master object select the "Delete" button.

Settings



The serial port has a primary port and if enabled a secondary port. Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Enable secondary

The "Enable secondary" checkbox provides for a second com port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary com port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Device ID

The address of the PLC.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

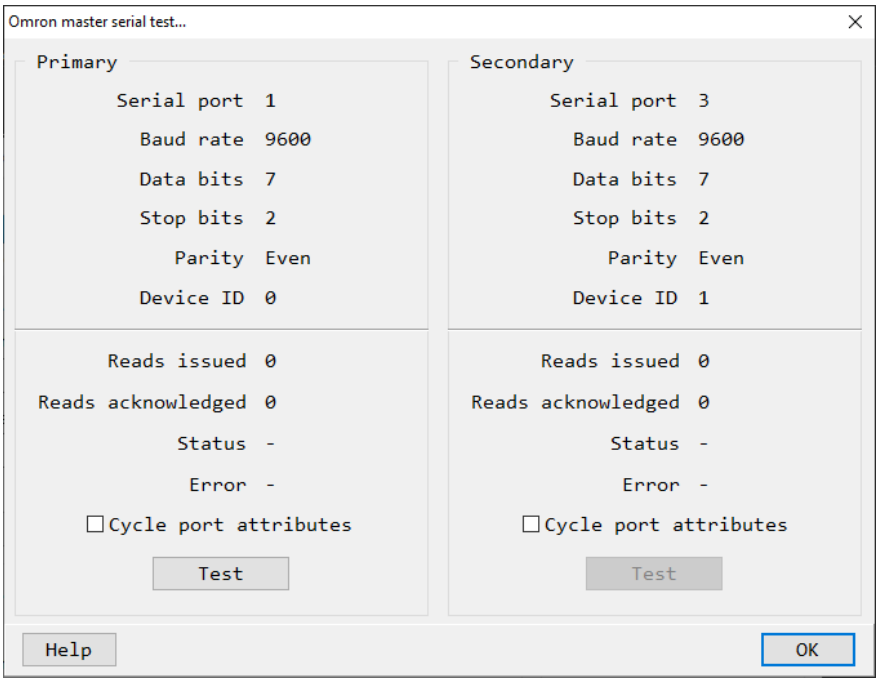
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button

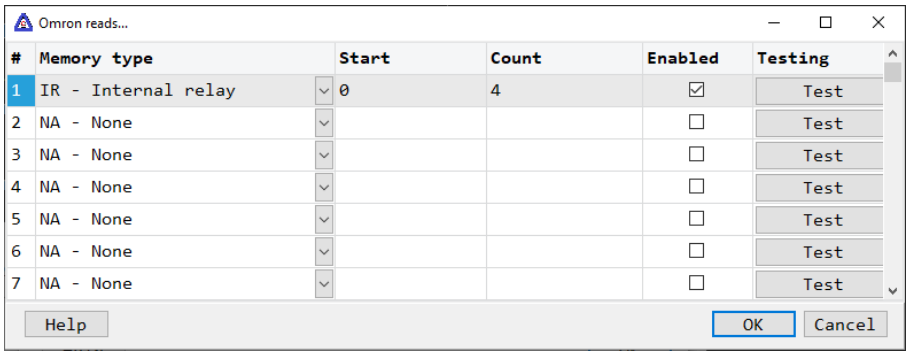


When the test button is selected the program will send an "Enquire" message to the slave device.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads



The address ranges shown may or may not be present in the slave device.

The HMI uses the following addressing for access to the data in the slave. All values are in decimal.

Address examples

AR0, AR105, AR1.02, AR1.15
IR0, IR105, IR1.02, IR1.15
HR0, HR105, HR1.02, HR1.15
LR0, LR105, LR1.02, LR1.15
DM0, DM105, DM1.02, DM1.15
TP0, TP45, TP99 word data type **Note:** This might be expressed in BCD.
TS0, TS45, TS99 Boolean data type

Notes:
1. Omron does not provide for single bit writes. A write to a digital data type will clear all other bits in the register. Only using the true state for commands and grouping all commands in a word or words not containing status bits overcomes this limitation.
2. Each bit reference must use 2 decimal places. For bits 0 to 9 use a leading 0. i.e. 01, 02, 09

Register Type

Register Type	Prefix	Radix	Data Type
Auxiliary relays	AR	Decimal	Word, float, Boolean
Internal relay	IR	Decimal	Word, float, Boolean
Holding relay	HR	Decimal	Word, float, Boolean
Link relay	LR	Decimal	Word, float, Boolean
Data memory	DM	Decimal	Word, float, Boolean
Timer/Counter PV	TP	Decimal	Word, Boolean **Note:** This might be expressed in BCD.
Timer/Counter state TS	Decimal	Boolean	

Start register

The starting register to read. This is the word address of the memory area.

Count

The number of registers to read for the request in decimal. Each register is 16 bits. The protocol has a maximum of 29 words of data per request.

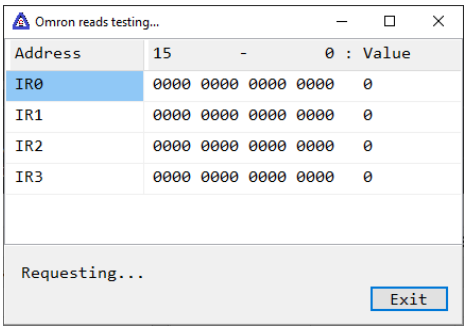
The timer/counter state is limited to 3 words (48 bits). Each state bit is returned as one byte.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string. See 'Testing error message' below.

Read configuration error messages

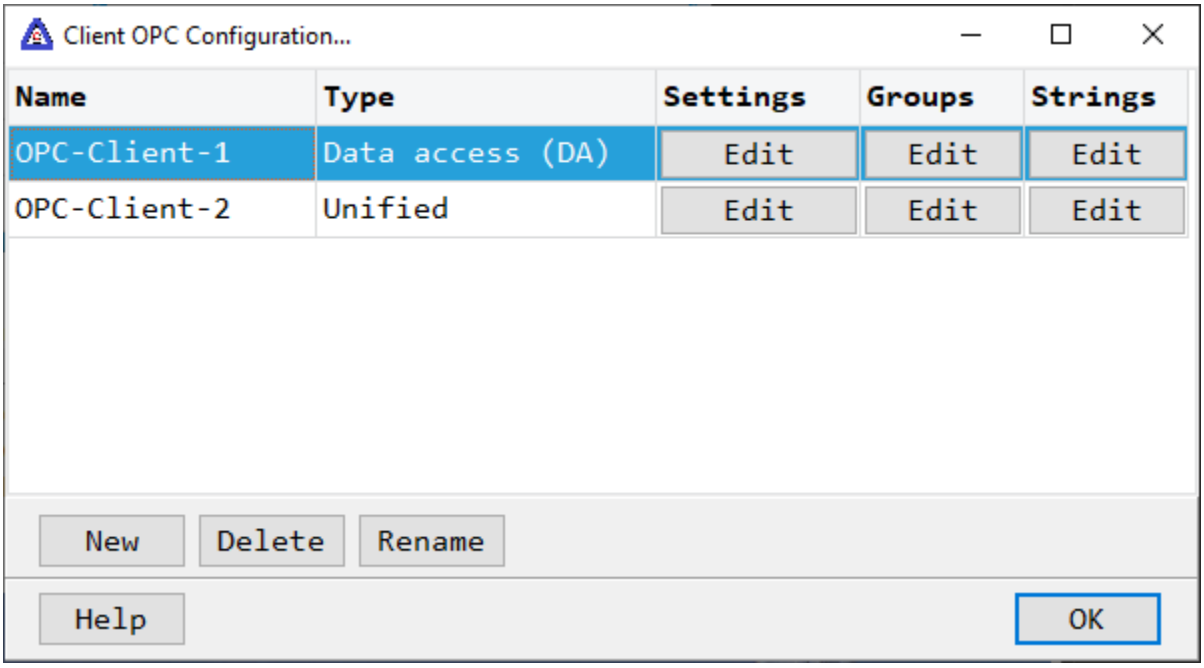
No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.
Timer/Counter limit 3 words (48 status bits)	Timer/Counter status bits are returned in one byte each timer/counter.

Testing error messages

See here

OPC Clients

Each OPC Client (Data Access or Unified Architecture) object is listed in the window.

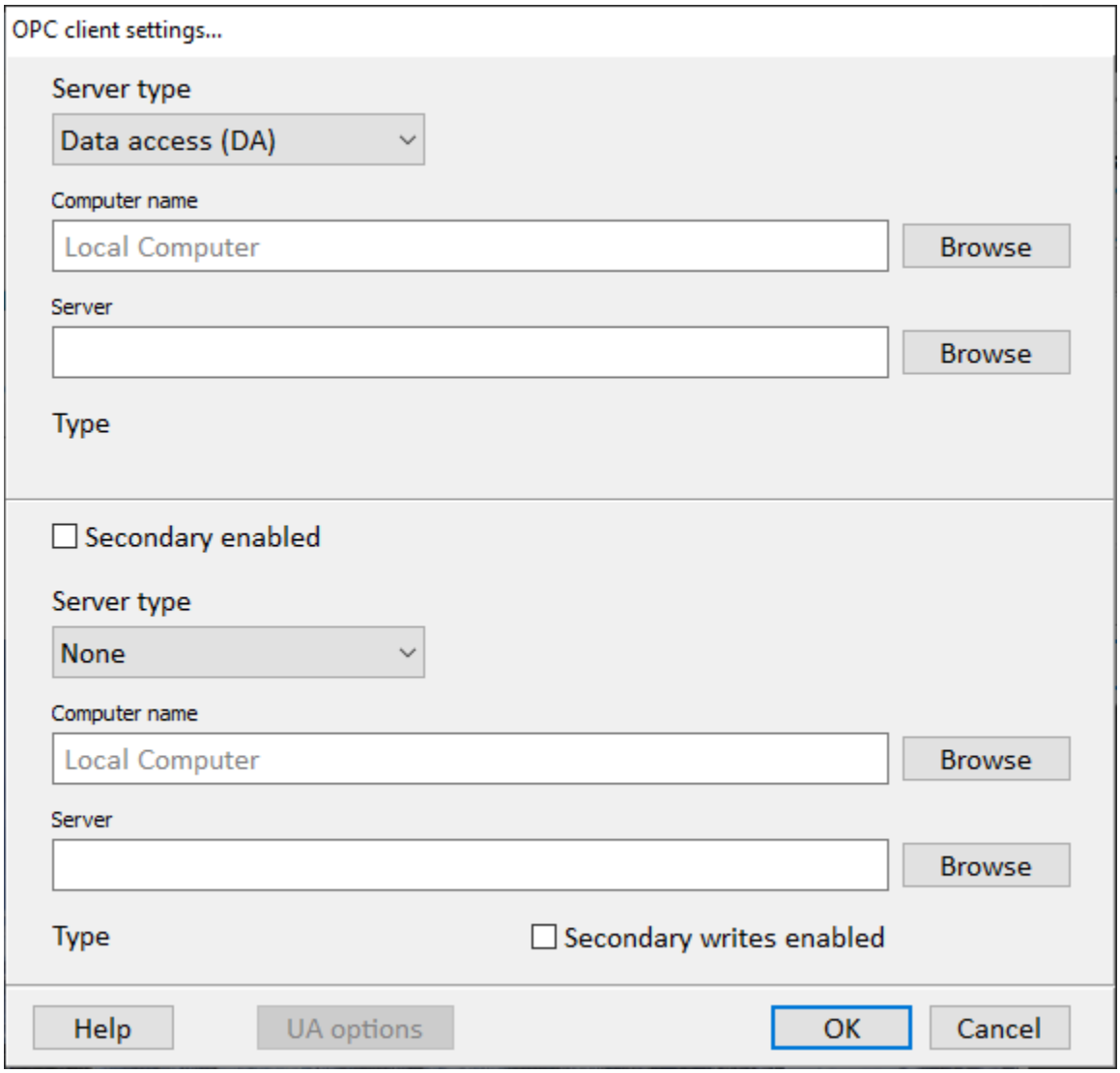


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an OPC Client master object select the "Delete" button.

Settings



Server type

None
Data Access (2.x, 3.x or XML 1/x)
Unified Architecture

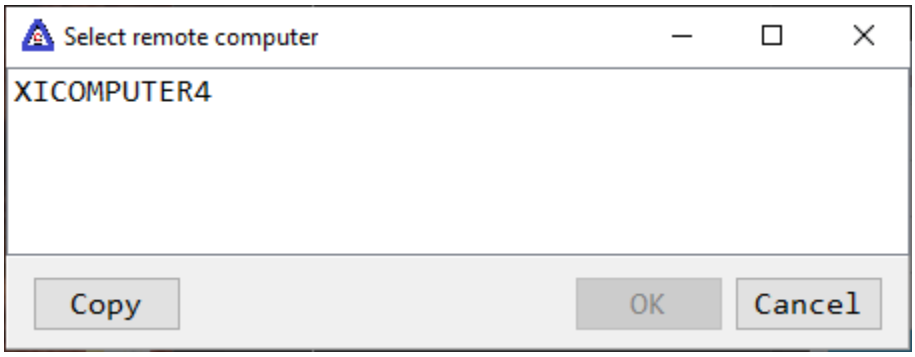
Discovery server.

Leave blank for searching the local computer.
Enter the computer name for searching remote computers.

Unified Architecture
The discovery server URL. If the computer only has one server installed, the URL can be the same as the server. The “Browse” button can search for computers but will not provide a qualified URL.

Browse (Discovery server / computer)

Selecting the "Browse" button will display a dialog allowing for the browsing of the network. Depending on network configuration, size, speed, etc., it can take several minutes to open individual branches. Note: Selecting the browse command can take many minutes to search the network.



Server

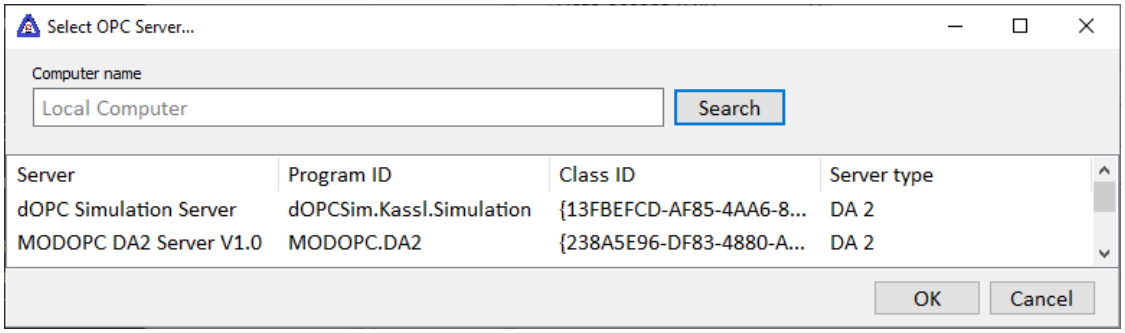
This is the name of the OPC server. For UA the computer URL.

Note: Testing with the “Prosys OPC UA Simulation Server”, version 5.0.0-217, the address/URL is shown: opc.tcp://<computer name>.lan:53530/OPCUA/SimulationServer
We could not connect to that URL. The ‘**.lan**’ was not needed.
opc.tcp://<computer name>:53530/OPCUA/SimulationServer worked as well as:
opc.tcp://<ip address>:53530/OPCUA/SimulationServer
e.g. opc.tcp://192.168.1.5:53530/OPCUA/SimulationServer

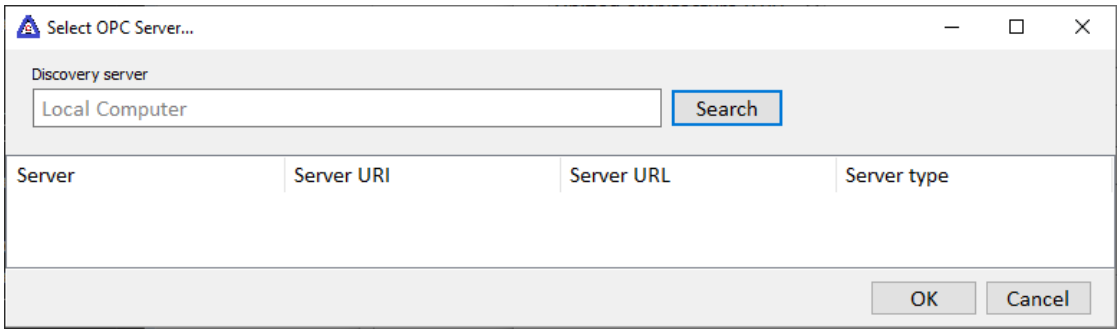
Browse (Server)

Selecting the "Browse" button will display a dialog to allow browsing of available servers on the computer.

Data Access



Unified Architecture



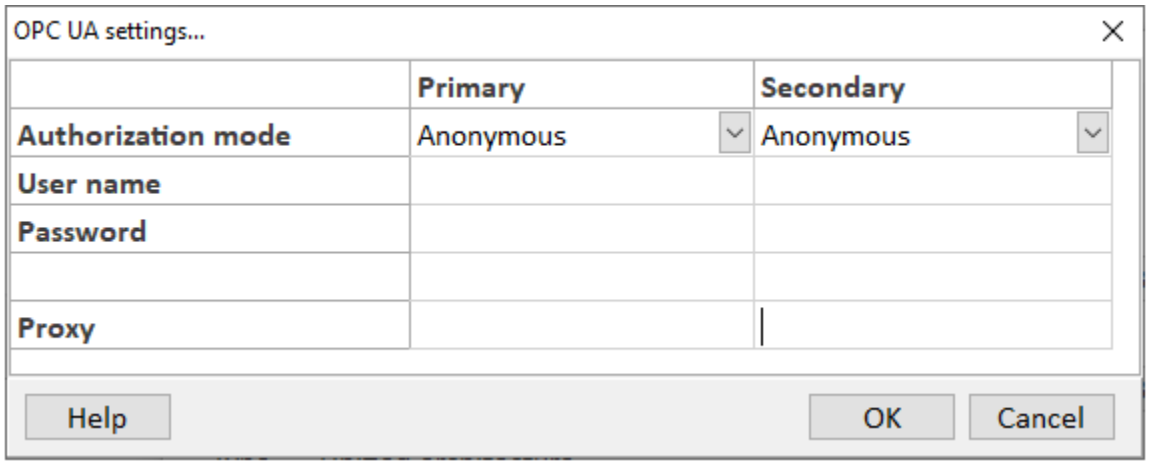
Secondary enabled

A connection to a second OPC server can be configured. This connection will operate in "Hot Standby". When runtime monitoring is initiated the groups and items configured will be added to the secondary server. When the secondary server reports a data change and the primary server is connected the data is discarded. If the primary server is not connected the data will be processed.

Secondary writes enabled

When the primary server is connected and a write command is sent to the primary server, if this attribute is enabled, a write of the same data point and value is sent to the secondary server. If the primary server is disconnected all writes are transmitted to the secondary server regardless of this configuration attribute.

UA options (Unified Architecture)



Authorization mode

Anonymous, no user name or password.
User name, a user name and password.

Proxy

When a proxy server is used supply the proxy settings.
Format: **urn://user:password@proxyIP:portNumber**
Examples:
urn://@192.168.10.254:3128 //no user name and password
urn://Operations:123ABC@192.168.10.254:3128

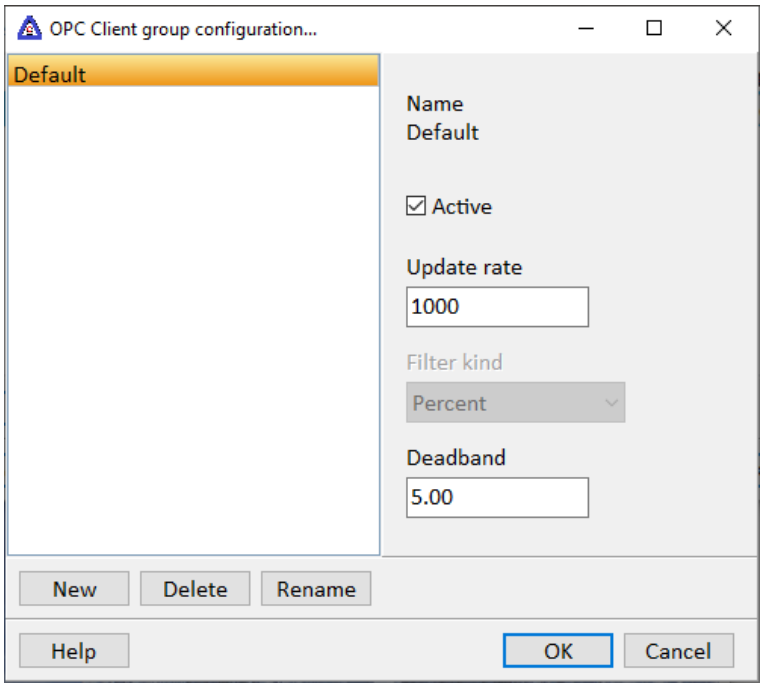
Note:

The OPC UA specification changed the value of several status codes compared to OPC DA

Description	OPC DA	OPC UA
Good quality	192	0
Bad quality	0	2,147,483,648
Not Connected	8	2,156,527,616
Quality Uncertain	64	1,073,741,824

To maintain continuity, the OPC DA values will be used, in the HMI, for the above status codes. All other status code values are determined on the protocol used, OPC DA or OPC UA.

Groups



Active

Sets the active state of the group at runtime start. Control at runtime is provided via scripting.

Update rate

All the update rate indicates is that (a) callbacks from the server should happen no faster than this and (b) the cache should be updated at least this rate.

The update rate is a "request" from the client. The server should respond with an update rate that is as close as possible to that requested.

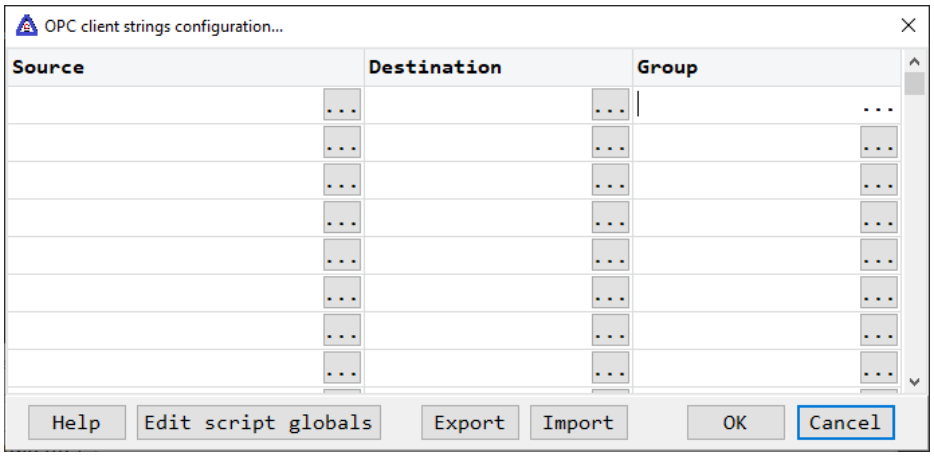
Filter kind (deadband)

Deadband only applies to analog type items. “None”, “Absolute” or “Percent”. Data Access only supports “Percent”. For no filtering set “Deadband” to 0 (zero). Unified Architecture, the first release only supports “None” and “Percent”.

Deadband

Deadband only applies to analog type items. If the change between the last cached value and the current value is greater than the deadband the cache will be updated.

Strings



Strings are not regular points. They do not have alarms, printing, etc. Strings may be displayed in any window via the "Script Global" animation. Any other actions, parsing, comparing, etc. must be done in scripts.

Source

This is the OPC in the PLC. It must be a valid tagname with a datatype "string". Select the button to browse the server. (See browse window below)
Data Access: The item ID.
Unified Architecture: The namespace and item id.
<namespace> ?= <item id>
Example: http://www.someUAserver.com/OPCUA/StaticNodes?=GUIDDataItem

Destination (optional)

If desired, select a script global location and the string will be copied to the location when the string value is returned by the external device. If this option is used or not the string may still be accessed via the "StringGet" and "StringSet" script commands. If this is configured and access to the string, in a script, is desired use the "GlobalGet" or "StringGet" to copy the string. The format of the destination is: <section>.<name>.
Note: Writing to the script global does not write the value to the PLC. "StringSet" must be used to write a string to the PLC.

Group

Each OPC string must belong to a group. This is a group in the server. Select which group this string is a member.

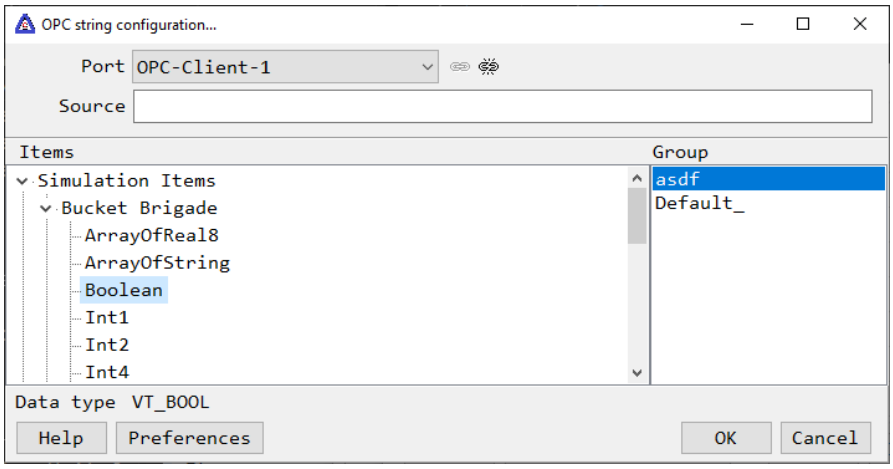
Export

This is used to export the configuration to an Excel worksheet.

Import

This is used to import string configurations from an Excel worksheet. The contents of the grid are completely replaced by the contents of the file.

String browse window



Port

This is the OPC port assigned to the string.

Next to the "Port" combo box are two small icons. The left one icon is to connect to the server selected in the combo box and the right icon is to disconnect from the server if connected. Depending on the preferences connection to the server may occur when the window is opened and disconnected when the window is closed.

Source

The item ID in the server. The is the full item ID used to identify what item in the server provides the data.

Data Access: The item ID.
Unified Architecture: The namespace and item id.
<namespace> ?= <item id>
Example: http://www.someUAserver.com/OPCUA/StaticNodes?=GUIDDataItem

Items

When a connection to the server is established a command to browse all items in the server will be executed. The result will be displayed in the items tree view.

Group

Each OPC point must belong to a group. This is a group in the server. Select which group this point is a member.

Data type

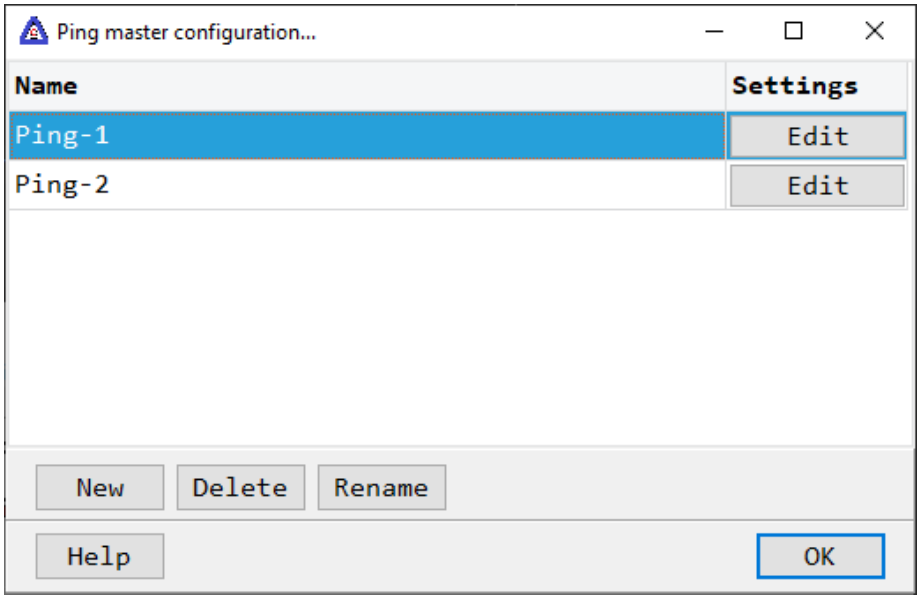
This shows the data type of the item selected when the mouse button is clicked on an item in the items tree view. Only string types are allowed.

Preferences

See OPC Client Configuration Preferences.

PING

Each Ping object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

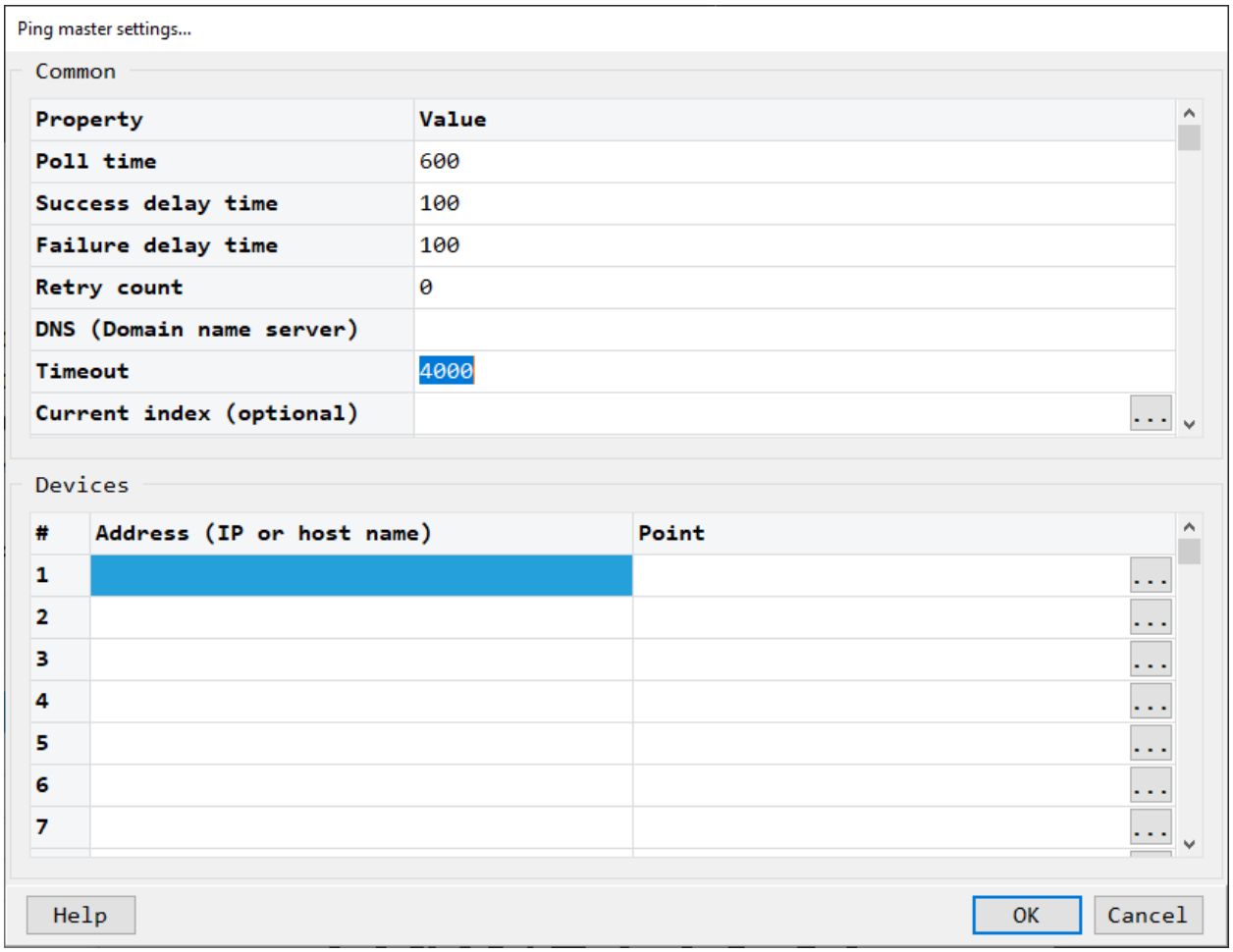
To rename an object select the "Rename" button and supply a new name.

To delete a Ping master object select the "Delete" button.

Notes:

1) Some routers have ICMP disabled by default. If ICMP is disabled, anywhere in the data route to the device, “Ping” will fail, regardless of the state of the device.
2) Some devices do not support the ping command.

Settings



Note:

It is recommend to **not** use any point that references an external device for the optional counter fields OR the required devices "point" fields. Depending on the timer settings, the pinging logic can execute very fast and fill up the write buffer of the port or overwhelm external devices with write commands. Use host analog and host digital points.

Common

Poll time

On runtime monitoring start the program will begin "pinging" all the devices in the order configured. After the last device in the list is tested, this value is the amount of time (seconds) to wait before restarting the testing process from the first device in the list. (0 - 100,000)

Success delay time

After a successful test of a device, this is the amount of time to wait (milliseconds) before testing the next device. (0 - 100,000)

Failure delay time

After a failure testing a device, this is the amount of time to wait (milliseconds) before testing the next device. (0 - 100,000)

Retry count

This is the number of times to retry (after the initial failure) the test before marking the test of the device/address a failure. (0-10)

DNS (Domain name server)

The name or address of the DNS used to resolve device addresses that contain a host name and not an IP address. If all the device addresses are IP address (e.g. 192.168.1.1) this field is not utilized.

Notes:
1) If using a URL in any device address, the DNS field must be an IP address or the name of the server.
2) To determine the DNS for the computer;
   A) Launch a command window.
   B) At the prompt type “nslookup” and press the 'Enter/Return' key.
   C) The response will be: Default Server: <server name> Address: <IP address>
   D) Type the IP address in the DNS field of the HMI.
   E) Close the command window.
3) The device digital point will be set false on a DNS lookup failure.

Timeout

The maximum time to wait (milliseconds) for a response to the device ping. A router could return a failure message before the maximum time is reached. (0 - 100,000)
Note: A timeout less than 4000 milliseconds is not recommended.

Current Index (optional)

An analog point indicating the current device under ping test. A value of 0 (zero) indicates the logic is waiting for the poll timer to complete.

Completed test cycles (optional)

This value indicates the count of times all addresses have been tested. The "Success" and "Failure" count will be equal to the total device count after one complete cycle of

the device list. After the first complete scan of all the addresses and when this value rolls over, it will roll over to 1 (one). 0 (zero) can be used as a first scan indicator. (0 - 4,294,967,295)

Success count (optional)

This value indicates the count of addresses that passed the ping test.

Failure count (optional)

This value indicates the count of addresses that failed the ping test.

Devices

Address

The address can be an IP address or a host name. Note: If a host name is used the DNS server field (above) must be valid.

Point

A digital point that stores the result of the test. (The item ID is the process variable digital.)

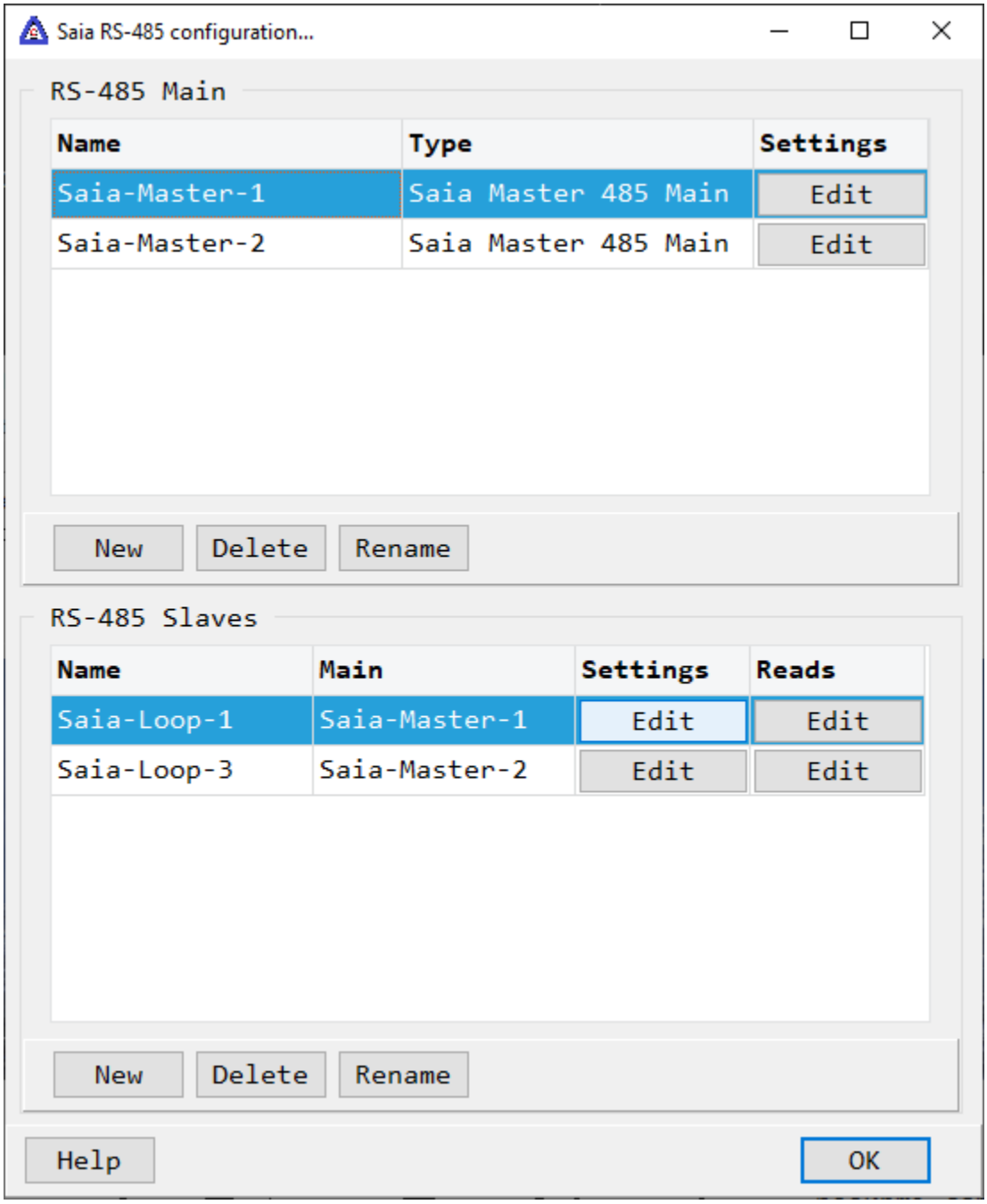
If the point value is **false**:
A ping test has not been performed (the list is on the first scan)
OR
The ping test was successful.

If the point value is **true**, the ping test failed.

Note: Set the digital host initial value to true. Then use the "falling" alarm to signal a ping failure on the first and subsequent test.

SAIA SERIAL 232/485

Each Saia master object controls one serial port and can have one or many slave objects. The slaves are configured to address one device.



Note: RS-485 is sensitive to improper wiring and/or terminations. Unpowered units can cause data echoes and other reliability issues. Please follow all RS-485 wiring guidelines.

Saia serial main

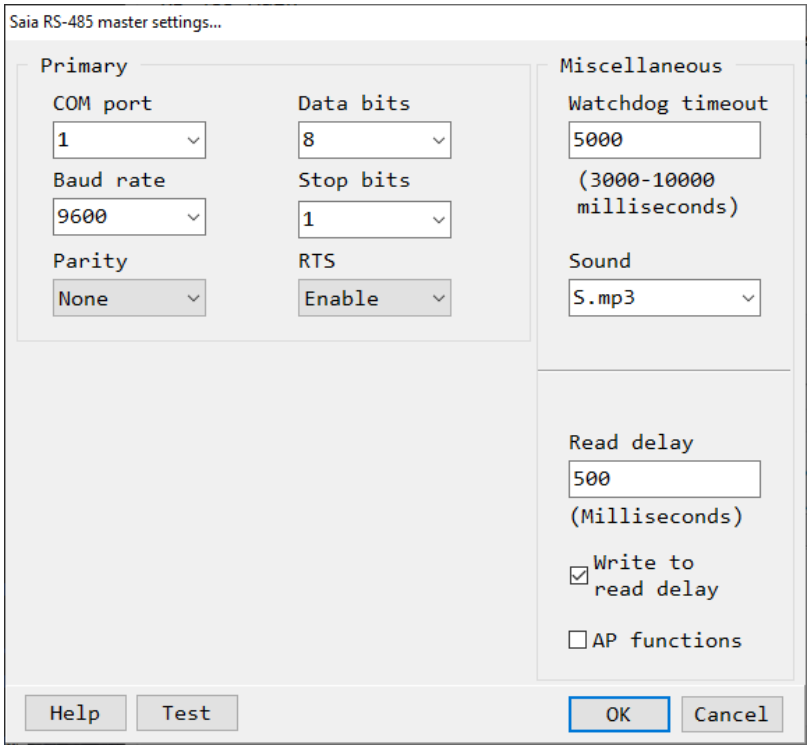
Each Saia master object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Saia master object select the "Delete" button.

Settings



Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

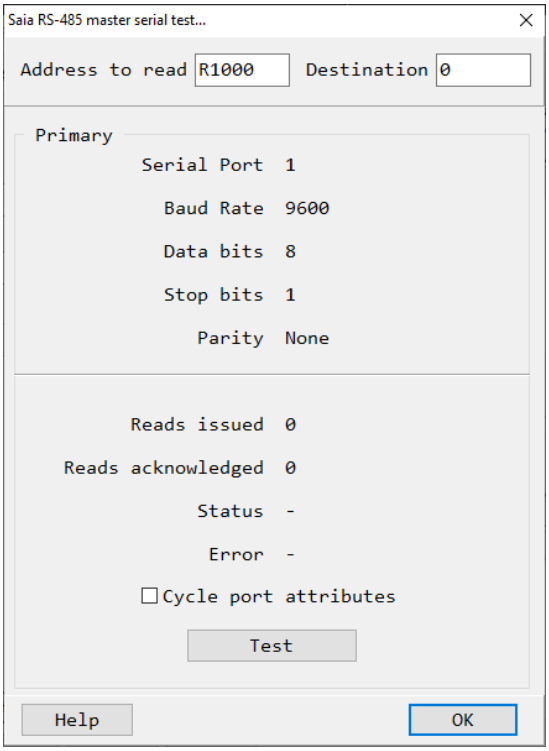
Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

AP functions

See analog functions.

Test button



When the test button is selected the program will attempt to read one point of data from the device at the address entered.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Saia serial slaves

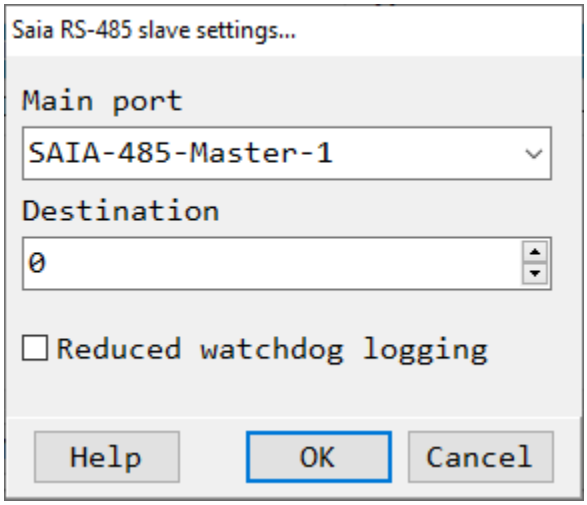
Each Saia slave object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Saia slave object select the "Delete" button.

Settings



Main port

This is the master RS-485 port the slave is linked to. A port must be linked to a master for runtime operations.

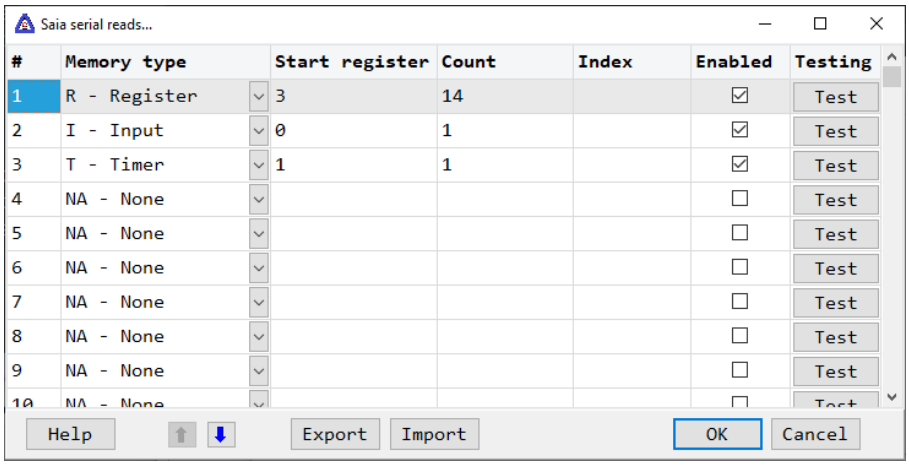
Destination

This is the destination number of the slave device. (0 - 255)

Reduced watchdog logging

When enabled and a watchdog timeout occurs, a single entry will be placed in the event log and the sound will be queued (if configured). When the slave device responds, after a watchdog timeout, a single entry will be placed in the event log. The watchdog counter will continue to count each time the watchdog timer expires.

Reads



The address ranges shown may or may not be present in the slave device.

Memory type

Address	**Examples**
R1	Register 1
R283	Register 283
C12	Counter 12
T37	Timer 37
F21	Flag 21
I3	Input 3
O17	Output 17
DB21.1	Data block 21 index 1
R283.5	Register 283 bit 5
DB21.1.7	Data block 21 index 1 bit 7

Point configuration source field for analog points.

Source Type	**Range**
None	-2,147,483,648 .. 2,147,483,647
Float	IEEE-754 standard format
Unsigned integer	0 .. 4,294,967,295

Notes:

1) Using the 'StringSet' script command is limited to 32 registers. Using format 0, 1, or 2 allows for a maximum of 64 characters. Using format 3 or 4 allows for a maximum of 32 characters.

2) The HMI float range is: minimum 1.5×10^{-45}..... maximum 3.4×10^{38}.

The float range of Saia is: minimum 5.42101070 x 10^{-20}..... maximum 9.223372773.4 x 10^{18}.

Start register or data block number

The starting register to read or the data block number. This is the word address of the memory area.

Count

This is the number of words/items to return. The per read limit is 32 registers/timers/counters/data blocks and 128 flags/inputs/outputs

Index (Data blocks only)

This is the index into the data block.

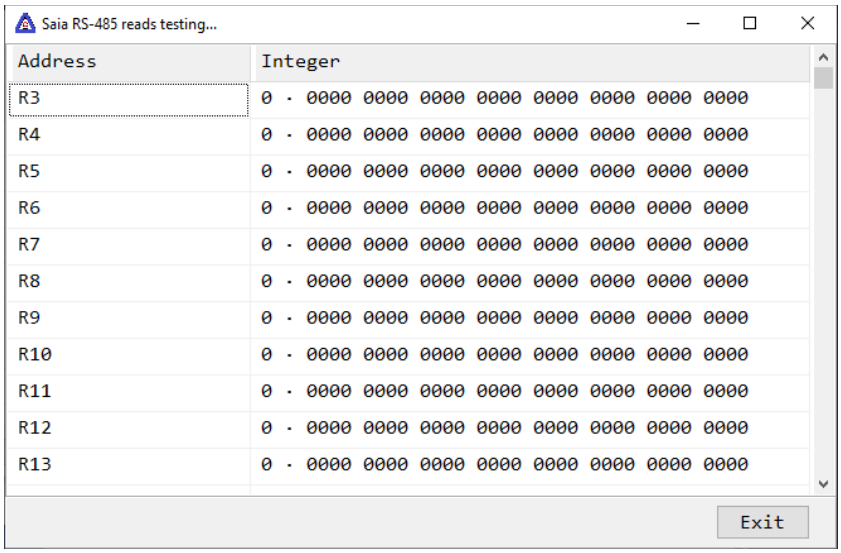
Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test

When the button is selected the program will attempt to access the data in the device using the communication parameters configured.

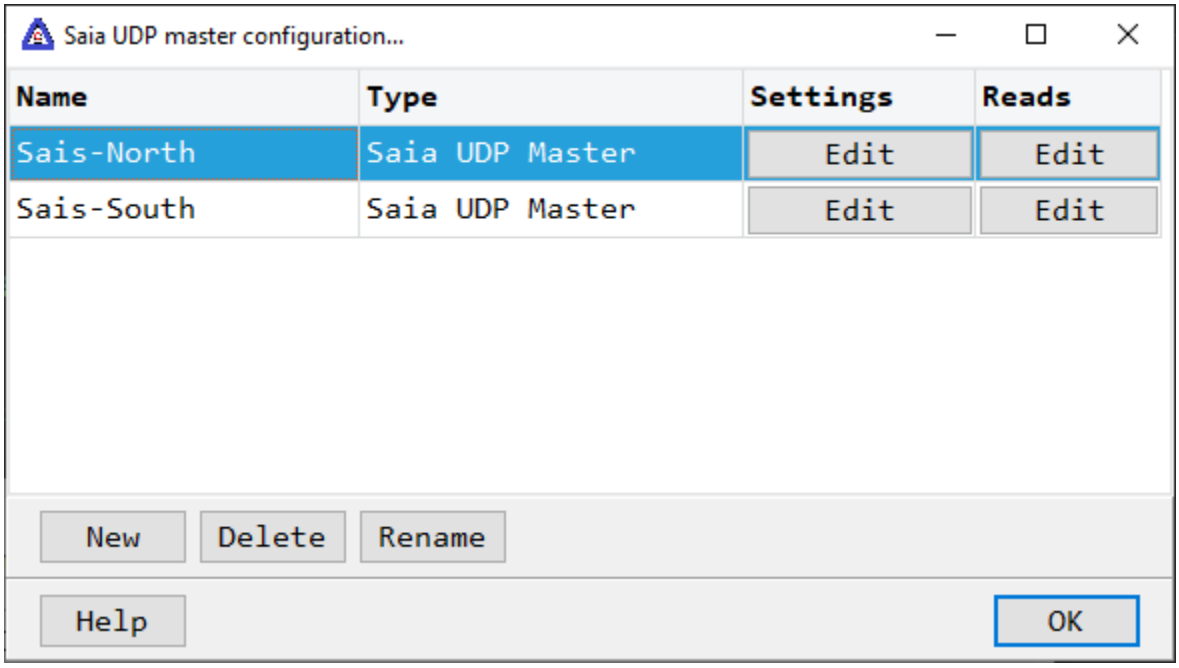


Error messages

No memory type selected	The memory type has not been selected.
Start out of range	The value is out of range for the type.
Count exceeds limit	The count must be 32/128 or less.
Start + count exceeds register limit	The register plus the count exceeds the maximum address range.
Count < 1	The line must read at least one register.
Index out of range	Data block index out of range.

SAIA-UDP

Each Saia-UDP master object is listed in the window.

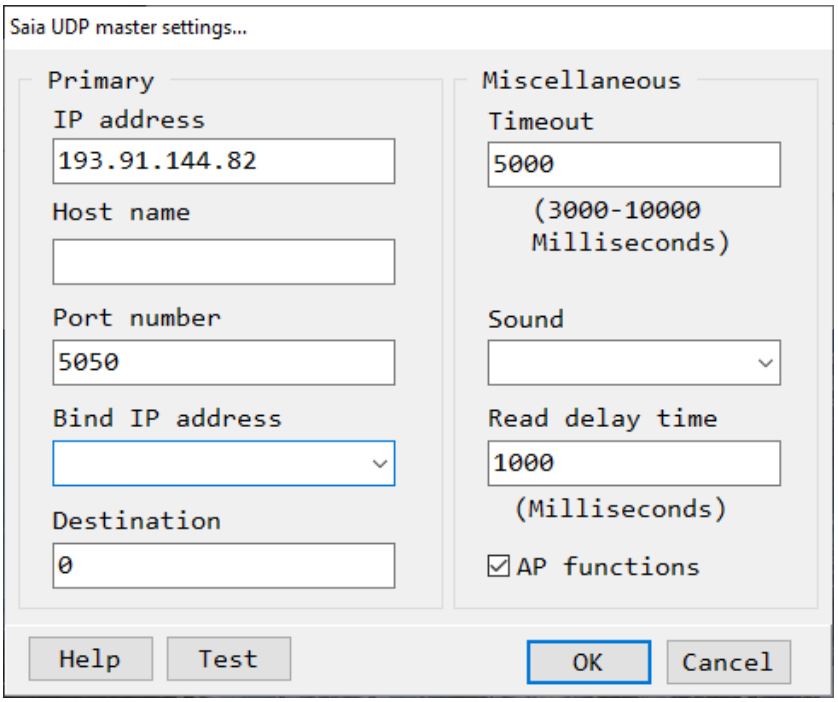


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Saia-UDP master object select the "Delete" button.

Settings



If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Destination

The address of the device.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

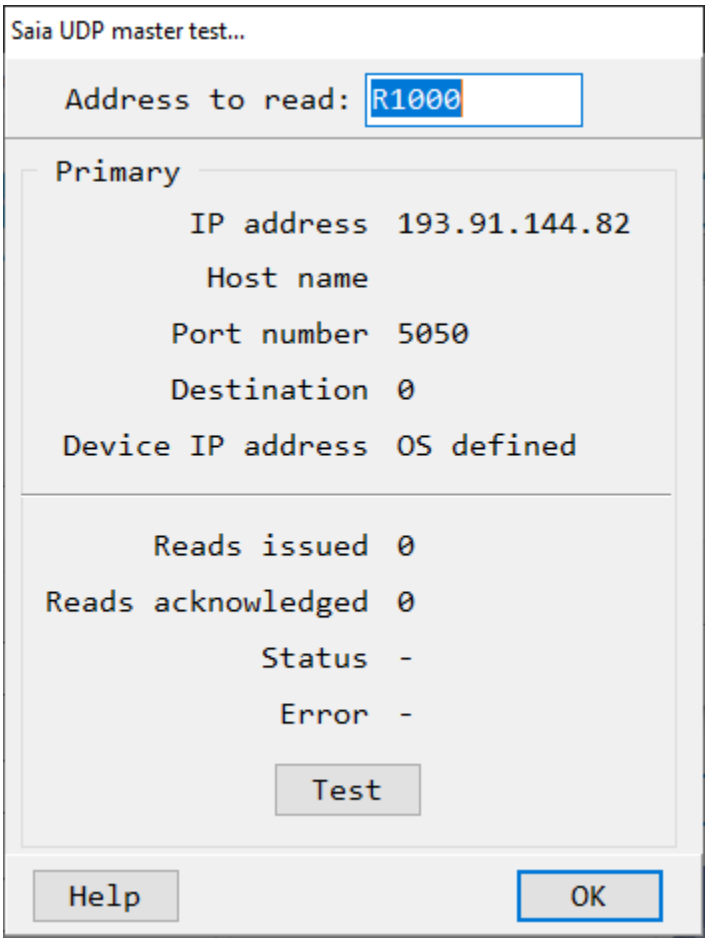
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

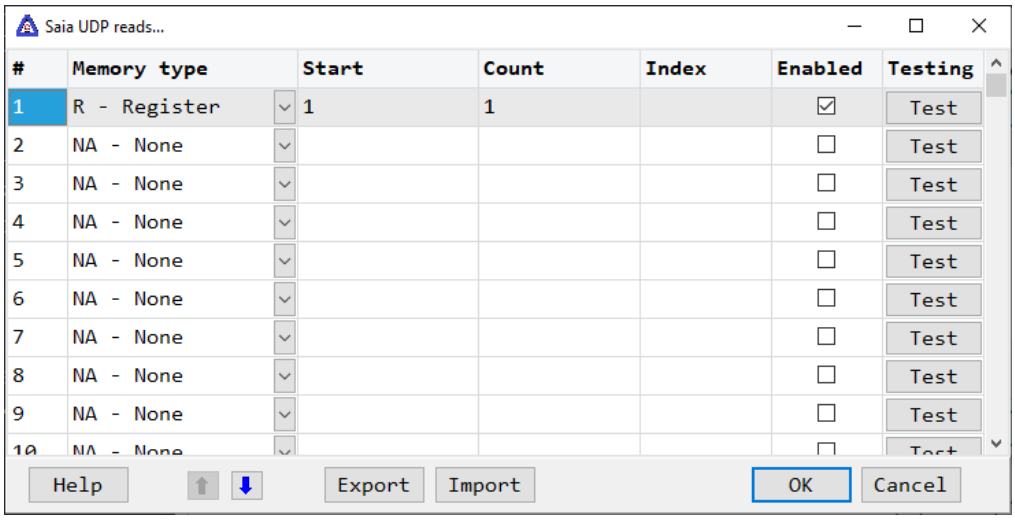
Test button



When the test button is selected the program will attempt to read one element using the address supplied.

The program will attempt to use the communication parameters configured.

Reads



The address ranges shown may or may not be present in the device.

Memory type

Address	Examples
R1	Register 1
R283	Register 283
C12	Counter 12
T37	Timer 37
F21	Flag 21
I3	Input 3
O17	Output 17
DB21.1	Data block 21 index 1
R283.5	Register 283 bit 5
DB21.1.7	Data block 21 index 1 bit 7

Point configuration source field for analog points.

Source Type	Range
None	-2,147,483,648 .. 2,147,483,647
Float	IEEE-754 standard format
Unsigned integer	0 .. 4,294,967,295

Notes:

1) Using the 'StringSet' script command is limited to 32 registers. Using format 0, 1, or 2 allows for a maximum of 64 characters. Using format 3 or 4 allows for a maximum of 32 characters.
2) The HMI float range is: minimum 1.5×10^{-45}..... maximum 3.4×10^{38}.
The float range of Saia is: minimum $5.42101070 \times 10^{-20}$..... maximum $9.223372773.4 \times 10^{18}$.

Start register or data block number

The starting register to read or the data block number. This is the word address of the memory area.

Count

This is the number of words/items to return. The per read limit is 32 registers/timers/counters/data blocks and 128 flags/inputs/outputs

Index (Data blocks only)

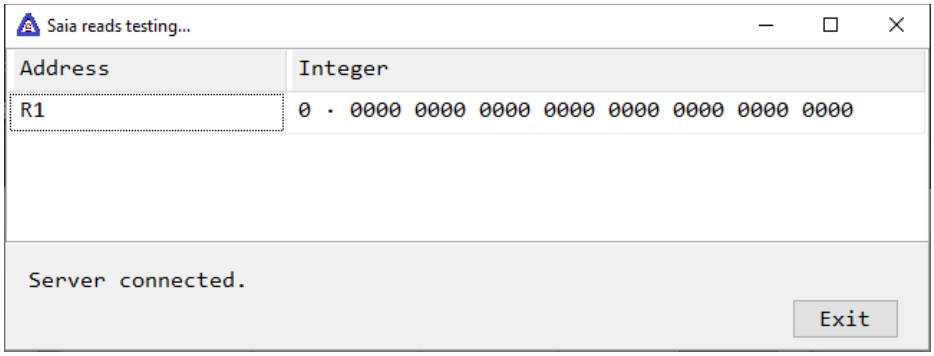
This is the index into the data block.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



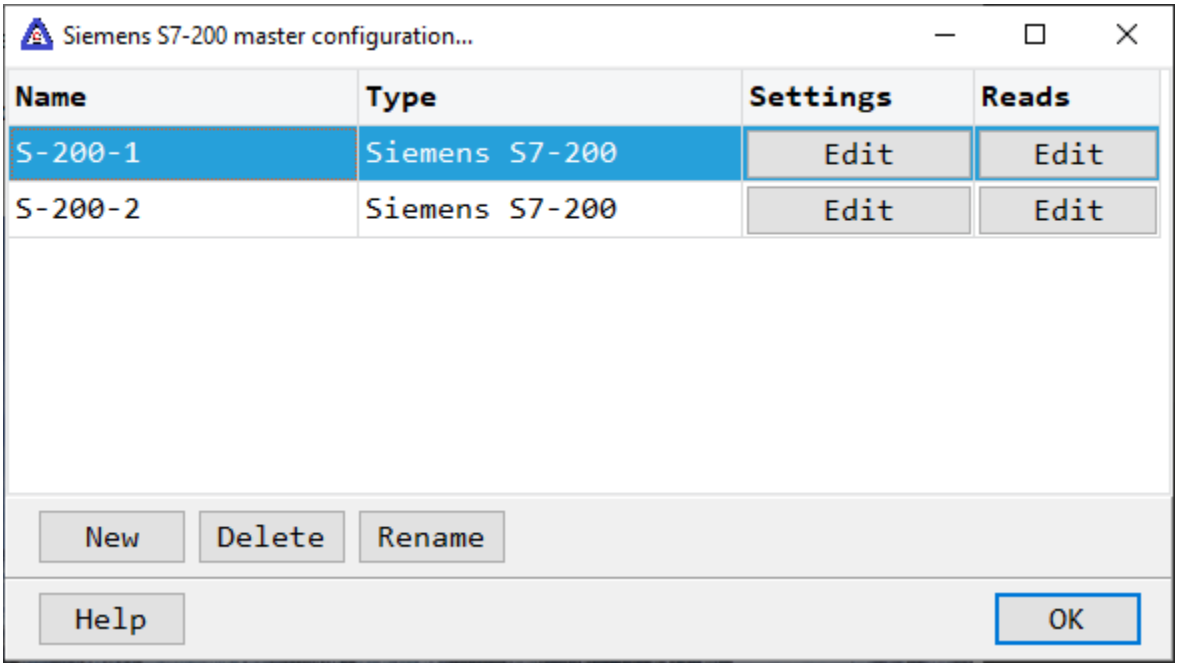
When the button is selected the program will attempt to access the data in the device using the communication parameters configured.

Error messages

No memory type selected	The memory type has not been selected.
Start out of range	The value is out of range for the type.
Count exceeds limit	The count must be 32/128 or less.
Start + count exceeds register limit	The register plus the count exceeds the maximum address range.
Count < 1	The line must read at least one register.
Index out of range	Datablock index out of range.

Siemens S7-200 (PPI) Serial

Each S7-200 master object is listed in the window.

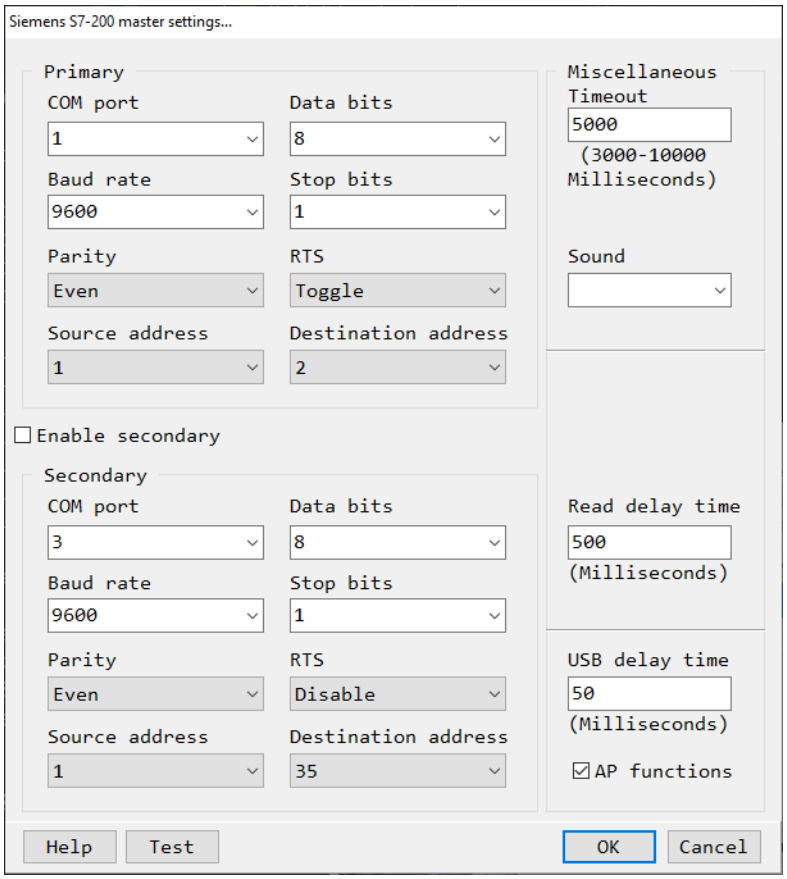


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an S7-200 master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

RTS

See here.

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if

configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Destination address

The address of the slave device.

Source Address

The address of the HMI program.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

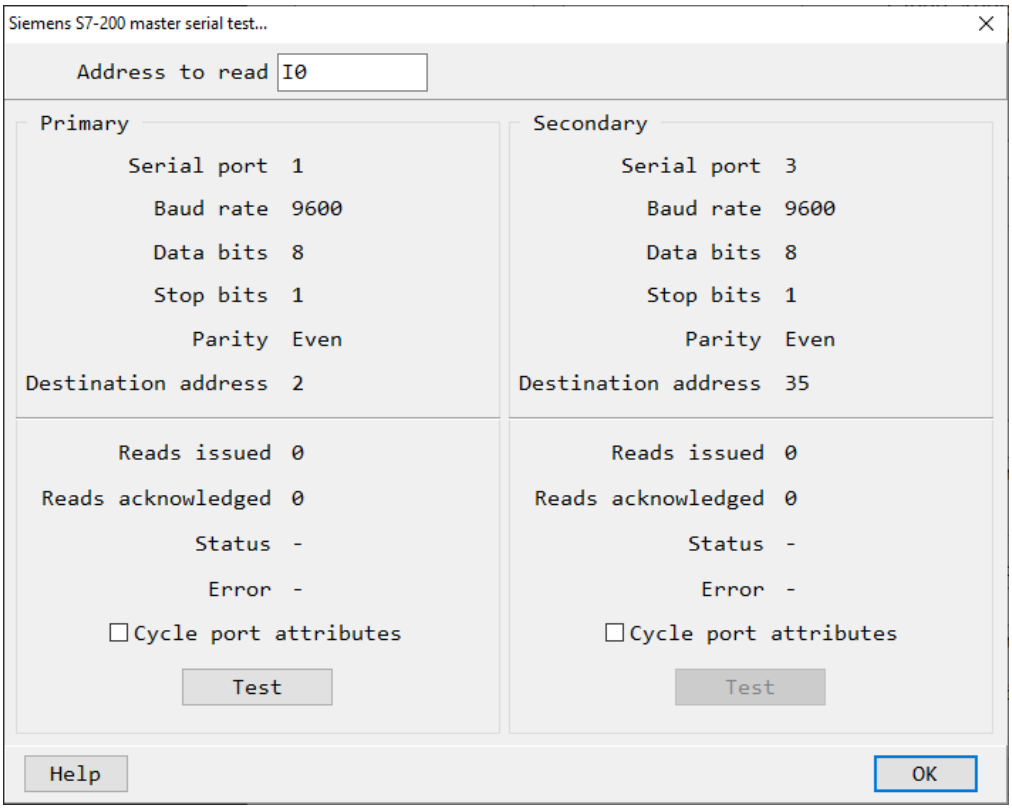
AP functions

See analog functions.

USB delay time

In testing it was found that the PC would receive a packet and transmit the response faster than the converter/PLC could handle it. This was observed to occur most often when a USB to serial converter was used. It has been witnessed using a conventional serial port. A value of zero (0) will disable the delay. The smaller the value the faster the turnaround time.

Test button

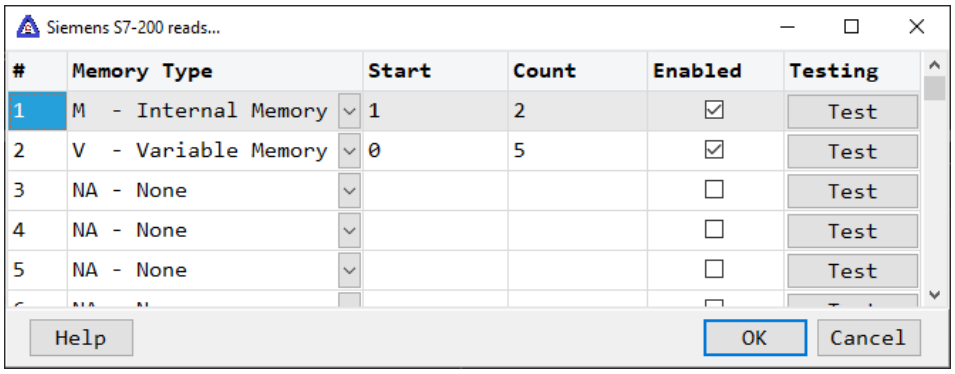


When the test button is selected the program will attempt to read one element of data from the device at the address entered.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads



The address ranges shown may or may not be present in the slave device.

Register type

The following addresses are used for access to the data in the slave.

Register Type	Prefix	Information	
Inputs	I, IW, ID	I0.7	Bit 7 of byte 0
		I1.0	Bit 0 of byte 1
		I5	Byte 5
		IW4	Word 4
		ID8	Double word 8
Outputs	Q, QW, QD	Q0.1	Bit 1 of byte 0
		Q1.3	Bit 3 of byte 1
		Q9	Byte 9
		QW6	Word 6
		QD12	Double word 13
Internal Memory	M, MW, MD	M0.1	Bit 1 of byte 0
		M1.3	Bit 3 of byte 1
		M14	Byte 14
		MW6	Word 6
		MD12	Double word 13
Special Memory	SM, SMW, SMD	SM0.1	Bit 1 of byte 0
		SM1.3	Bit 3 of byte 1
		SM3	Byte 3
		SMW6	Word 6
		SMD12	Double word 13
Variable	V, VW, VD	V0.1	Bit 1 of byte 0
		V1.3	Bit 3 of byte 1
		V222	Byte 222
		VW6	Word 6
		VD12	Double word 13
Timer	T	T0	Timer 0

		T12	Timer 12

If the point type is digital, the address is the done bit of the timer.
If the point type is analog, the address is the word value of the timer accumulator.

Counter	C	C1	Counter 1
		C4	Counter 4

If the point type is digital, the address is the done bit of the counter.
If the point type is analog, the address is the word value of the counter accumulator.

Analog Input	AI	AI4	Analog input 4
		AI0	Analog input 0

Analog inputs are word values.

Analog Output	AQ	AQ4	Analog output 4
Write Only		AQ0	Analog output 0

Analog outputs are word values.

Caution: Siemens addressing has overlap.
IB0 is byte 0.
IB1 is byte 1.
IW0 is byte 0 and 1.
IW1 is byte 1 and 2.
IW2 is byte 2 and 3.

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values. Enter the starting register value.

Count

The number of registers to read for the request. The memory type and slave device determine the maximum number of registers that can be read. The protocol has a maximum of 222 bytes of data.

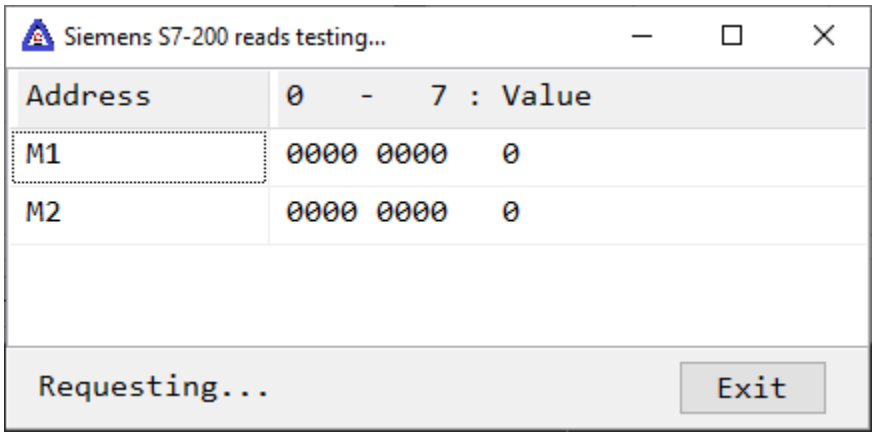
Register Type	**Data Size**	**Maximum Count**
Inputs, Outputs, Internal Memory, Special Memory, Variable	Byte	222
Timer	Structure	44
Counter	Structure	74
Analog In	Word	111

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

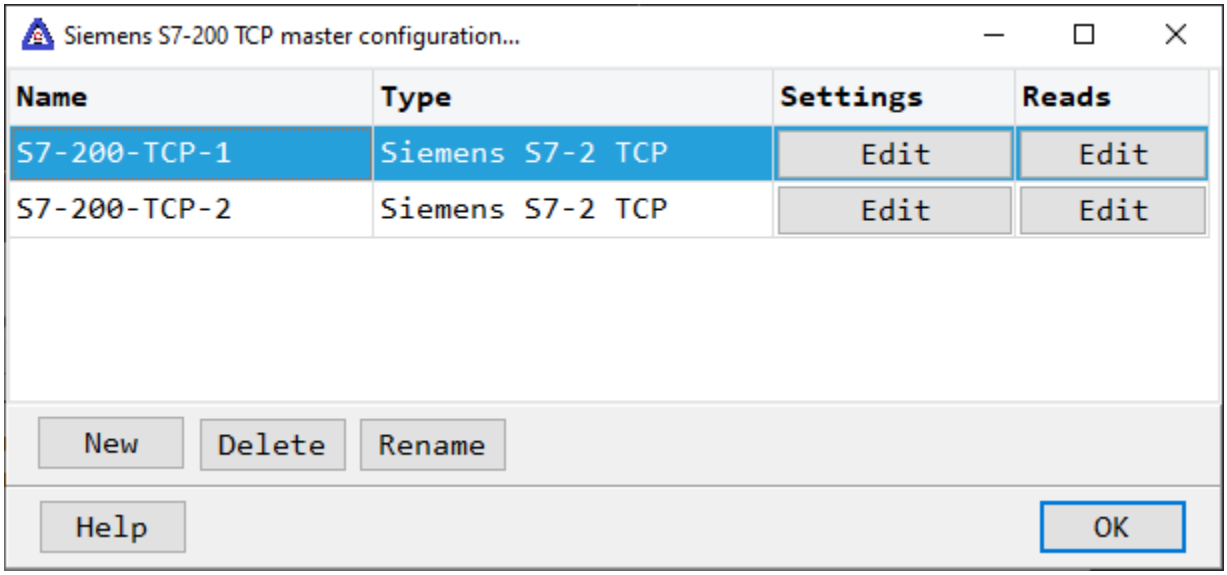
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

Siemens S7-200 TCP

Each S7-200 TCP master object is listed in the window.

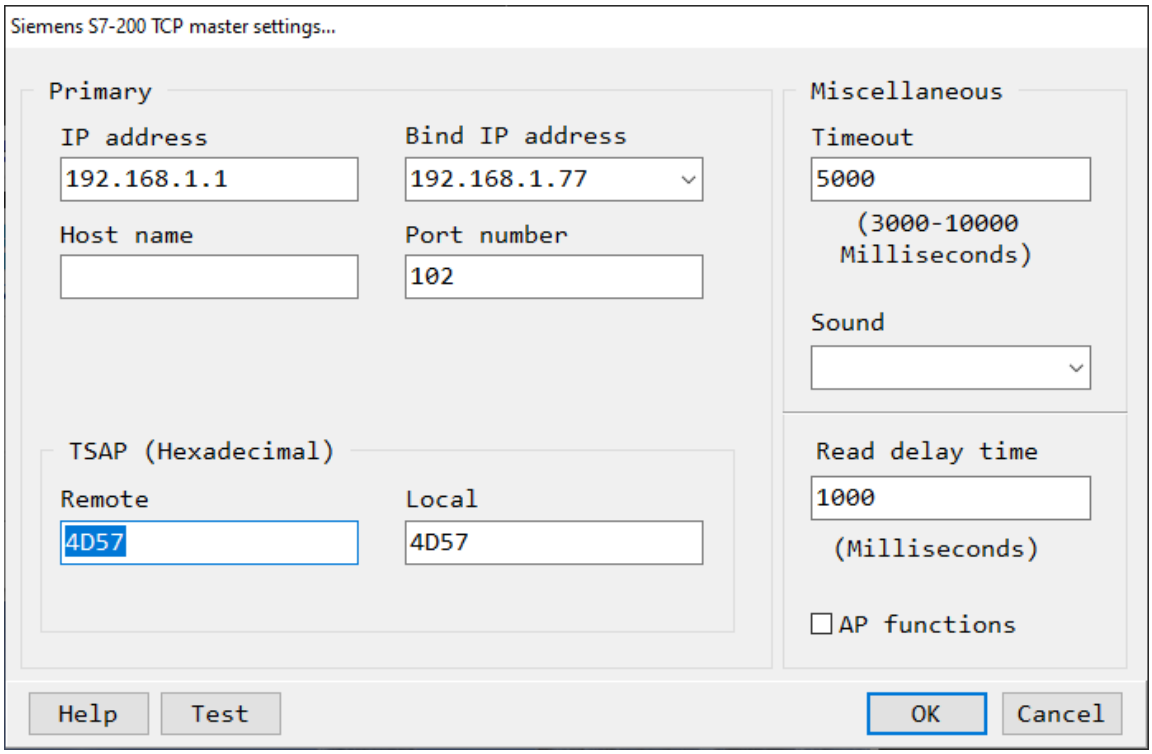


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an S7-200 TCP master object select the "Delete" button.

Settings



Select the port attributes as is needed.

If a "host name" is entered, the IP address is ignored.

TSAP (Hexadecimal) Remote/Local

This is the “Transport Service Access Point” number.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

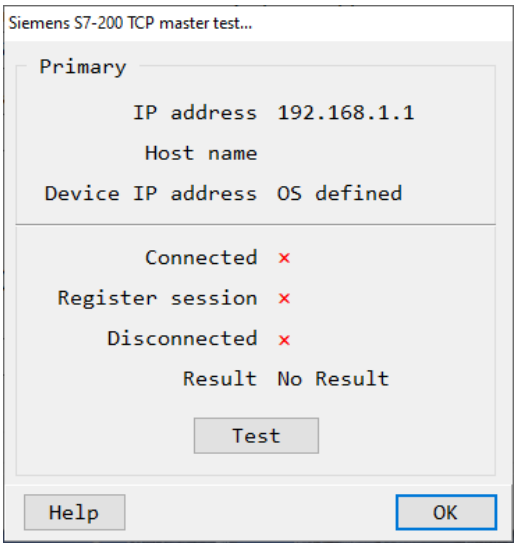
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button



When the test button is selected the program will send a connect message to the device.

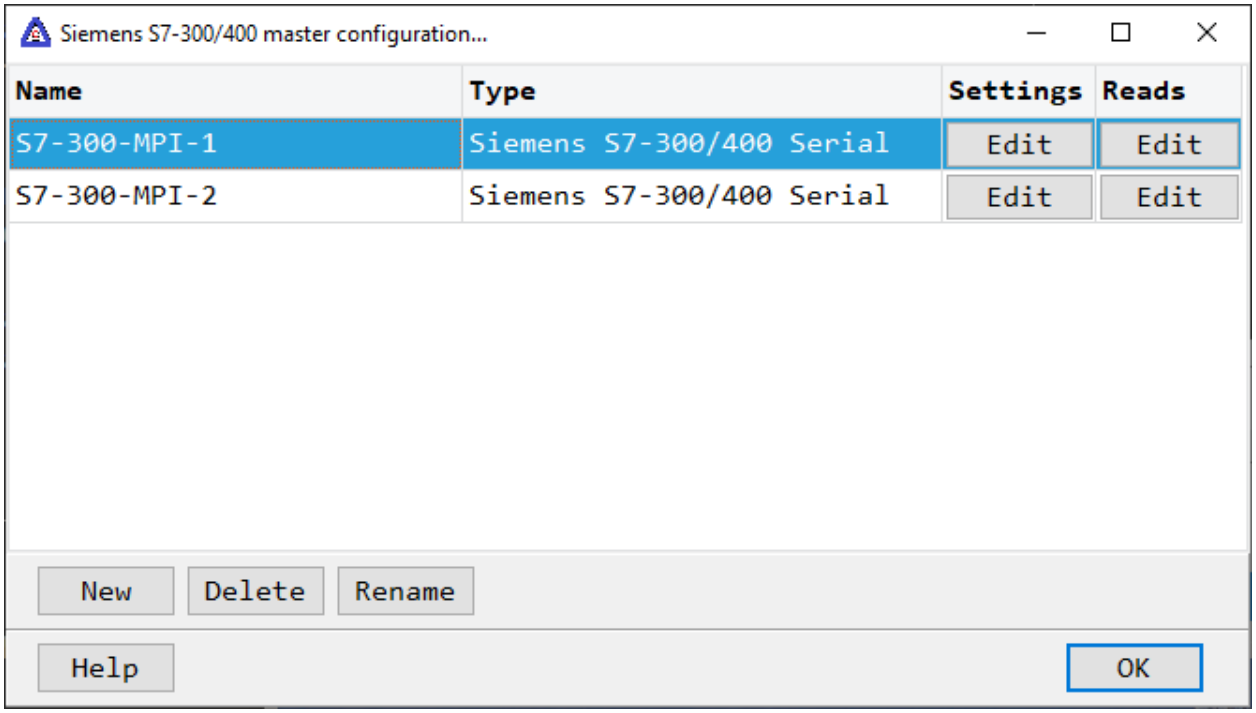
The program will attempt to use the communication parameters configured.

Reads

Read configuration for TCP is the same as serial.

Siemens S7-300/400 (MPI) Serial

Each S7-300/400 master object is listed in the window.

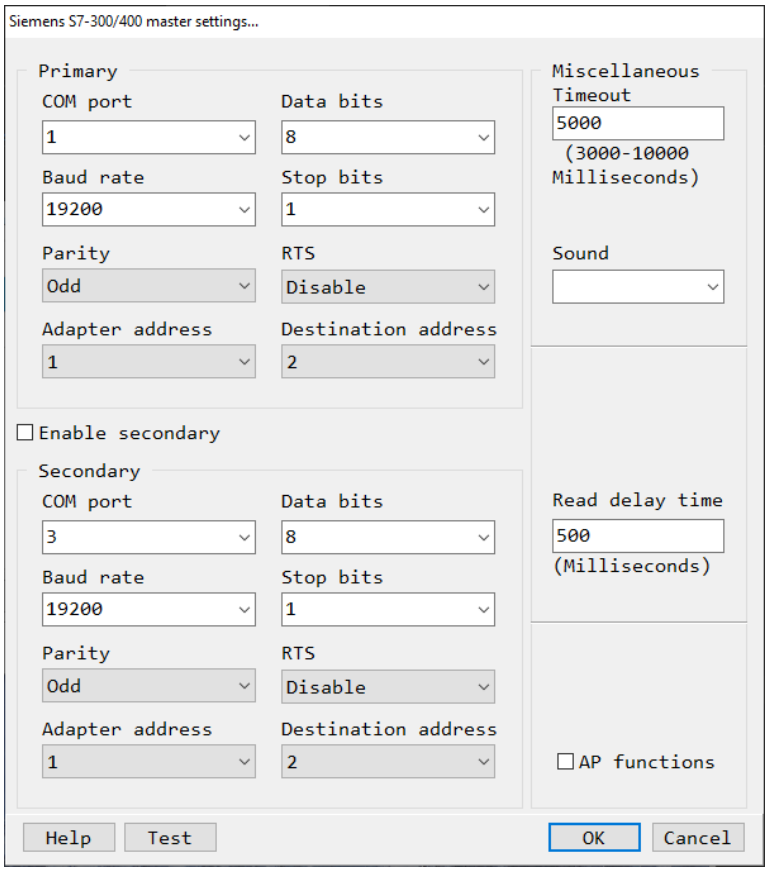


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an S7-300/400 master object select the "Delete" button.

Settings



The port has a primary port and if enabled a secondary port. Select the port attributes as is needed.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Enable secondary

The "Enable secondary" checkbox provides for a second communication port to be used as a hot backup. When in run mode the primary port is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary communication port. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary port, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not

replay. When communications is reestablished with the device on the primary port an entry will be made in the event log and operations will return to the primary port.

Destination address

The address of the slave device.

Source Address

The address of the HMI program.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

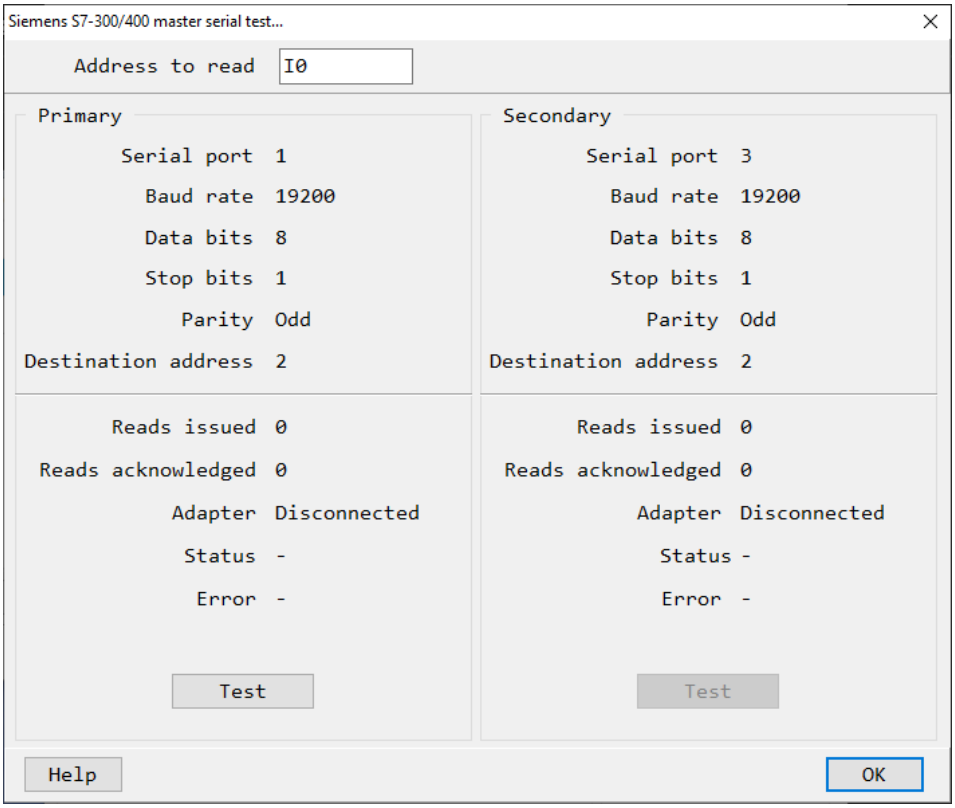
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

Test button

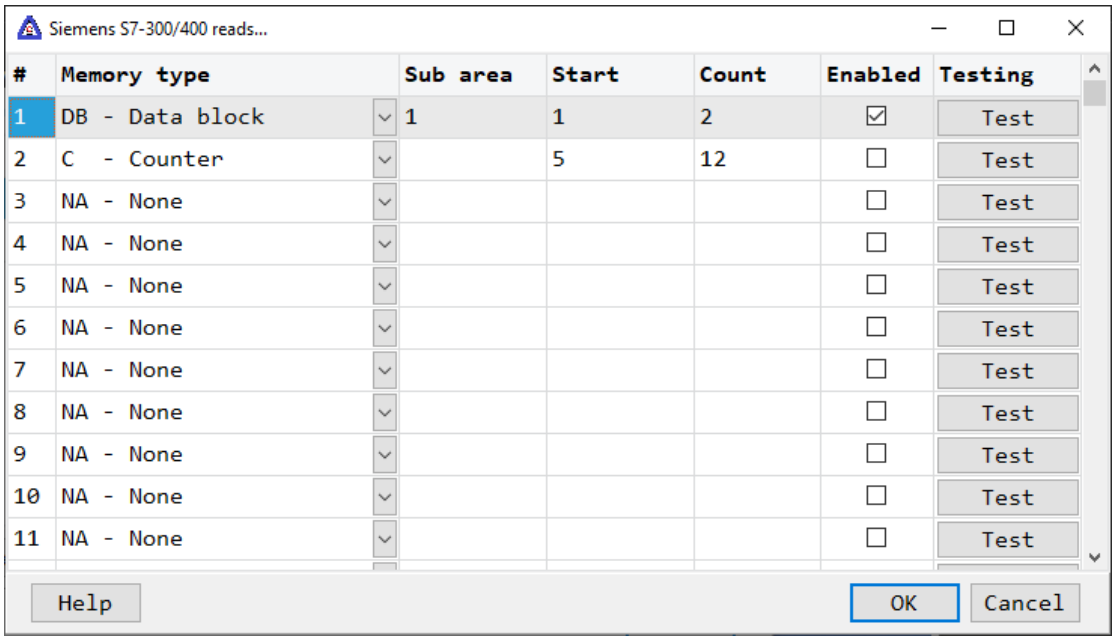


When the test button is selected the program will send an "Enquire" message to the slave device.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads



The address ranges shown may or may not be present in the slave device.

Register type

The following addresses are used to access data in the slave.

Register type	Prefix	Information	
Inputs	I, IW, ID	I0.7	Bit 7 of byte 0
		I1.0	Bit 0 of byte 1
		I5	Byte 5
		IW4	Word 4
		ID8	Double word 8
Outputs	Q, QW, QD	Q0.1	Bit 1 of byte 0
		Q1.3	Bit 3 of byte 1
		Q9	Byte 9
		QW6	Word 6
		QD12	Double word 13
Internal Memory	M, MW, MD	M0.1	Bit 1 of byte 0
		M1.3	Bit 3 of byte 1
		M14	Byte 14
		MW6	Word 6
		MD12	Double word 13
Timer	T	T0	Timer 0
		T12	Timer 12

If the point type is digital, the address is the done bit of the timer.
If the point type is analog, the address is the word value of the timer accumulator.

Counter	C	C1	Counter 1
		C4	Counter 4

If the point type is digital, the address is the done bit of the counter.
If the point type is analog, the address is the word value of the counter accumulator.

Data blocks

Source	Source type	Data type
DB127.0	Default	Byte
DB127.D0	Integer	Double word
DB127.W1	Default	Word
DB127.D4	Float	Floating point
DB127.10.0	N/A	Boolean (digital point)

Caution: Siemens addressing has overlap, example:
IB0 is byte 0.
IB1 is byte 1.
IW0 is byte 0 and 1.
IW1 is byte 1 and 2.
IW2 is byte 2 and 3.

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values. Enter the starting register value.

Sub area

For data blocks, this is the data block number (1-65535). Otherwise, this field is ignored.

Count

The number of registers to read for the request. The memory type and slave device determine the maximum number of registers that can be read. The protocol has a maximum of 222 bytes of data.

Register Type	Data Size	Maximum Count
Inputs, Outputs, Internal Memory	Byte	128
Timer	Word	111
Counter	Word	111

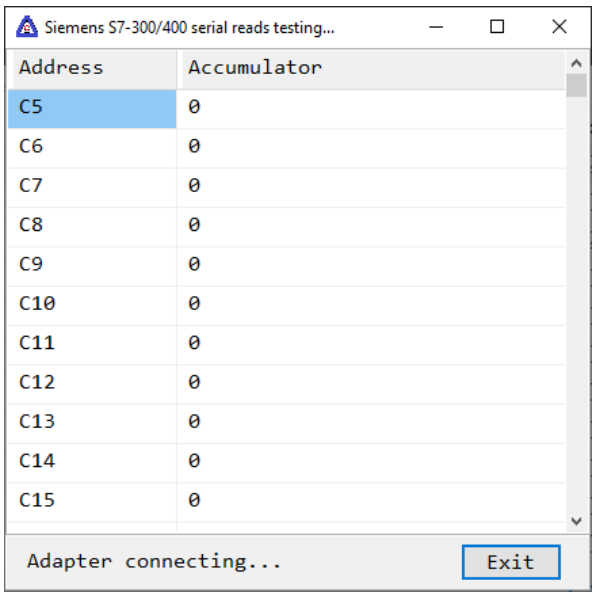
Note:
Data blocks are also limited by the size of the data block.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

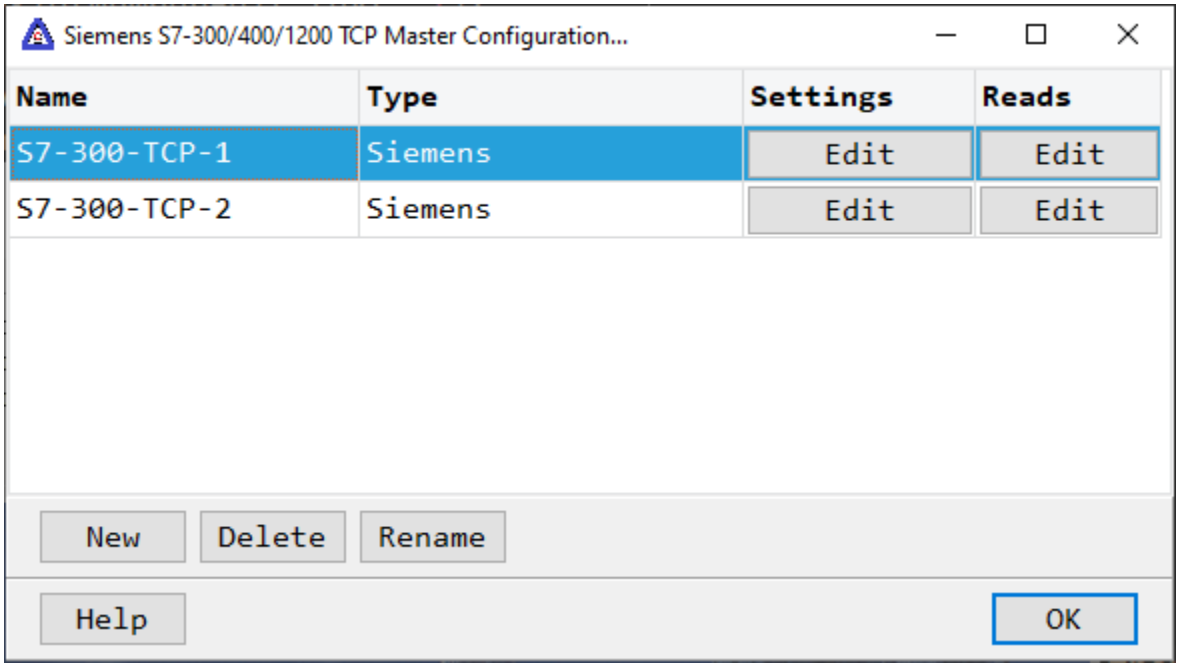
Error code	
5	The count was too large for the device.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

SIEMENS S7-300/400/1200 TCP

Each S7-300/400/1200 TCP master object is listed in the window.



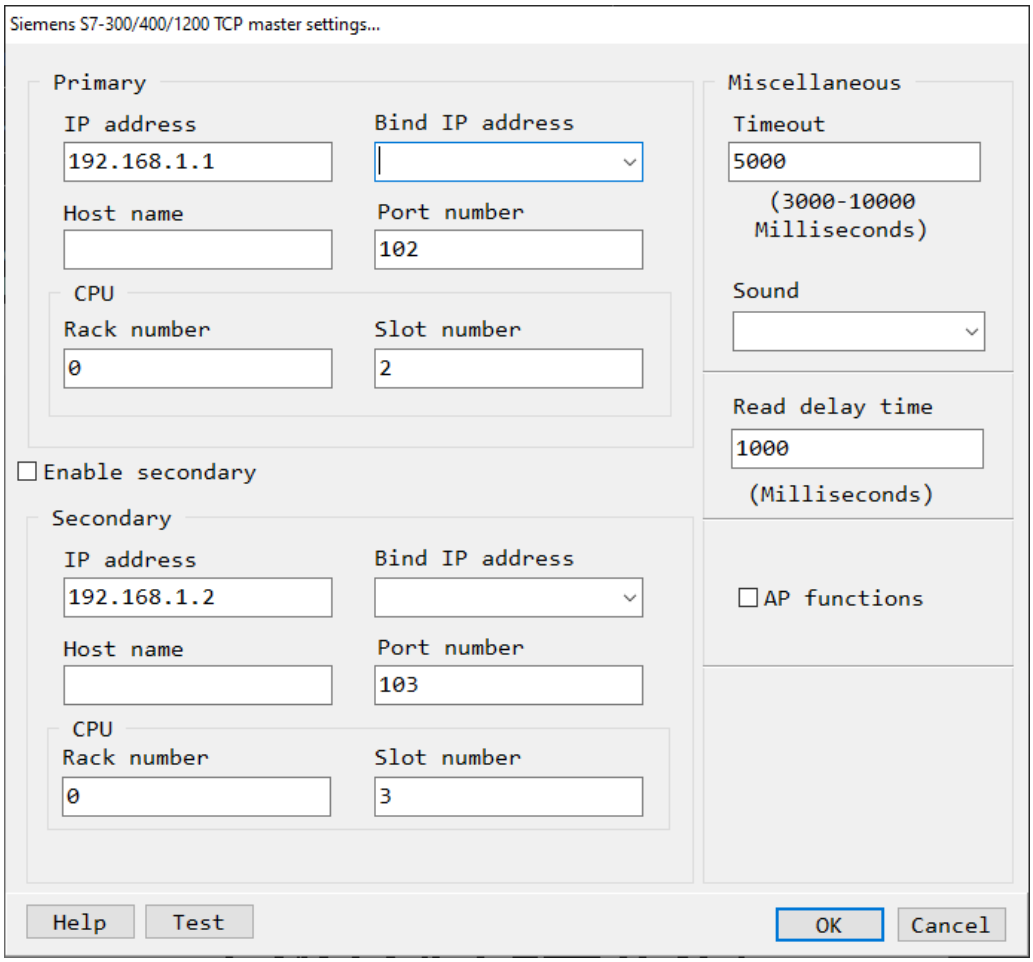
To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an S7-300/400/1200 TCP master object select the "Delete" button.

Note: This applies to the 1200 PLC. The PLC project must have the **"Connection mechanisms: Permit access with PUT/GET communication from remoted partner (PLC, HMI, OPC ...)"** enabled. The setting is configured via the "General/Protection" group. The pane might need to be scrolled up to see the setting.

Settings



The TCP port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Enable secondary

The "Enable secondary" checkbox provides for a second path to be used as a hot backup. When in run mode the primary configuration is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary configuration. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary configuration, if configured. An entry will be made in the event log and the watchdog sound, if

configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary configuration an entry will be made in the event log and operations will return to the primary port.

If the computer has two network interface cards (NIC) the program will use the first one detected for the primary and the second one detected for the secondary. If only one NIC is detected the program will use it for the primary and the secondary.

Rack number

The rack number of the CPU.

Slot number

The slot number of the CPU. (Slot 1 for S7-1200).

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:
The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

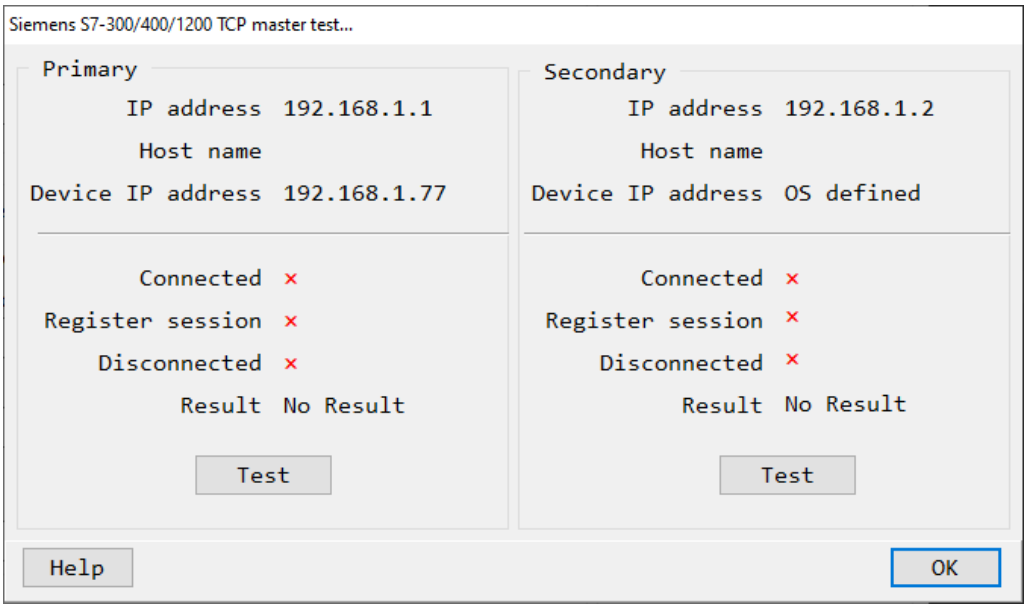
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

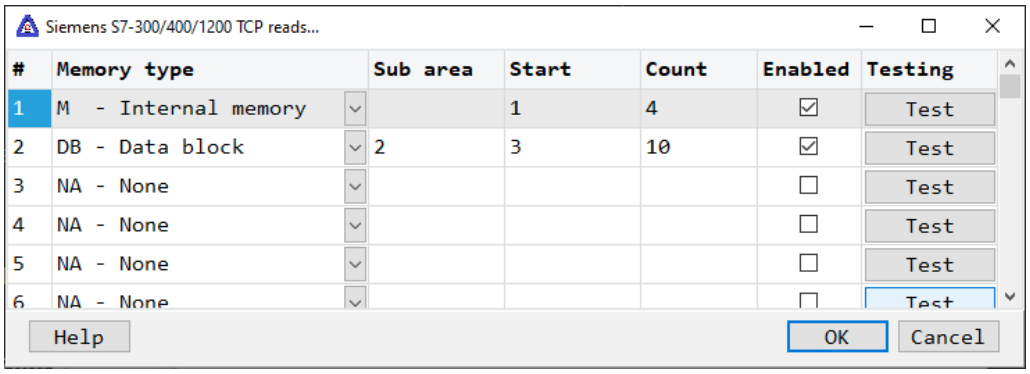
Test button



When the test button is selected the program will send a connect message to the device.

The program will attempt to use the communication parameters configured.

Reads



The address ranges shown may or may not be present in the slave device.

Register type

The following addresses are used for access to the data in the slave.

Register Type	Prefix	Information	
Inputs	I, IW, ID	I0.7	Bit 7 of byte 0
		I1.0	Bit 0 of byte 1
		I5	Byte 5
		IW4	Word 4
		ID8	Double word 8
Outputs	Q, QW, QD	Q0.1	Bit 1 of byte 0
		Q1.3	Bit 3 of byte 1
		Q9	Byte 9
		QW6	Word 6
		QD12	Double word 13
Internal Memory	M, MW, MD	M0.1	Bit 1 of byte 0
		M1.3	Bit 3 of byte 1
		M14	Byte 14
		MW6	Word 6
		MD12	Double word 13
Timer	T	T0	Timer 0
		T12	Timer 12

If the point type is digital, the address is the done bit of the timer.
If the point type is analog, the address is the word value of the timer accumulator.

Counter	C	C1	Counter 1
		C4	Counter 4

If the point type is digital, the address is the done bit of the counter.
If the point type is analog, the address is the word value of the counter accumulator.

Data blocks

Source	Source type	Data type
DB127.0	Default	Byte
DB127.D0	Integer	Double word
DB127.W1	Default	Word
DB127.D4	Float	Floating point
DB127.10.0	N/A	Boolean (digital point)

Caution: Siemens addressing has overlap, example:
IB0 is byte 0.
IB1 is byte 1.
IW0 is byte 0 and 1.
IW1 is byte 1 and 2.
IW2 is byte 2 and 3.

Start register

The starting register to read. The different memory areas have different register counts. Refer to the slave device for legal register values. Enter the starting register value.

Sub area

For data blocks this is the data block number (1-65535). Otherwise, this field is ignored.

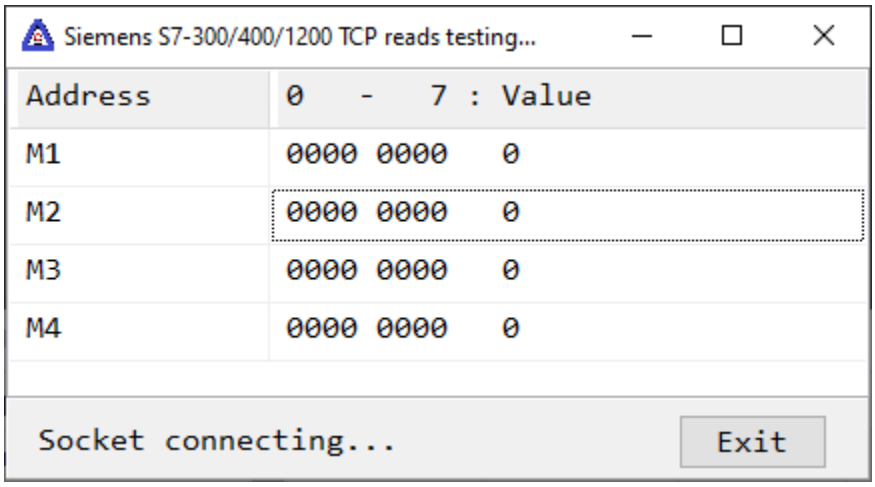
Count

The number of registers to read for the request. The memory type and slave device determine the maximum number of registers that can be read. The protocol has a maximum of 222 bytes of data.

Register Type	Data Size	Maximum Count
Inputs, Outputs, Internal Memory	Byte	128
Timer	Word	111
Counter	Word	111

Notes:
1. Data blocks are also limited by the size of the data block.
2. When testing reads an error code '5' is returned when reading too many bytes/words.

Test



When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

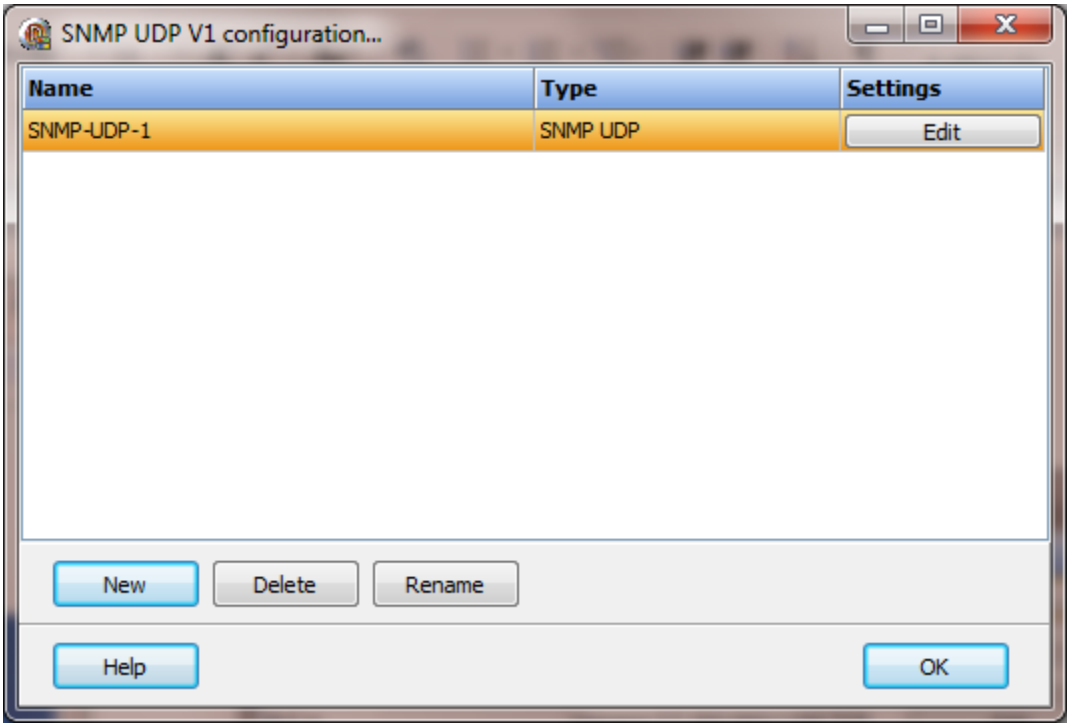
Located at the bottom left of the window is a label. If communication is all normal the label will indicate an increasing number. The label may display an error string.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

SNMP UDP

Each SNMP UDP master object is listed in the window.

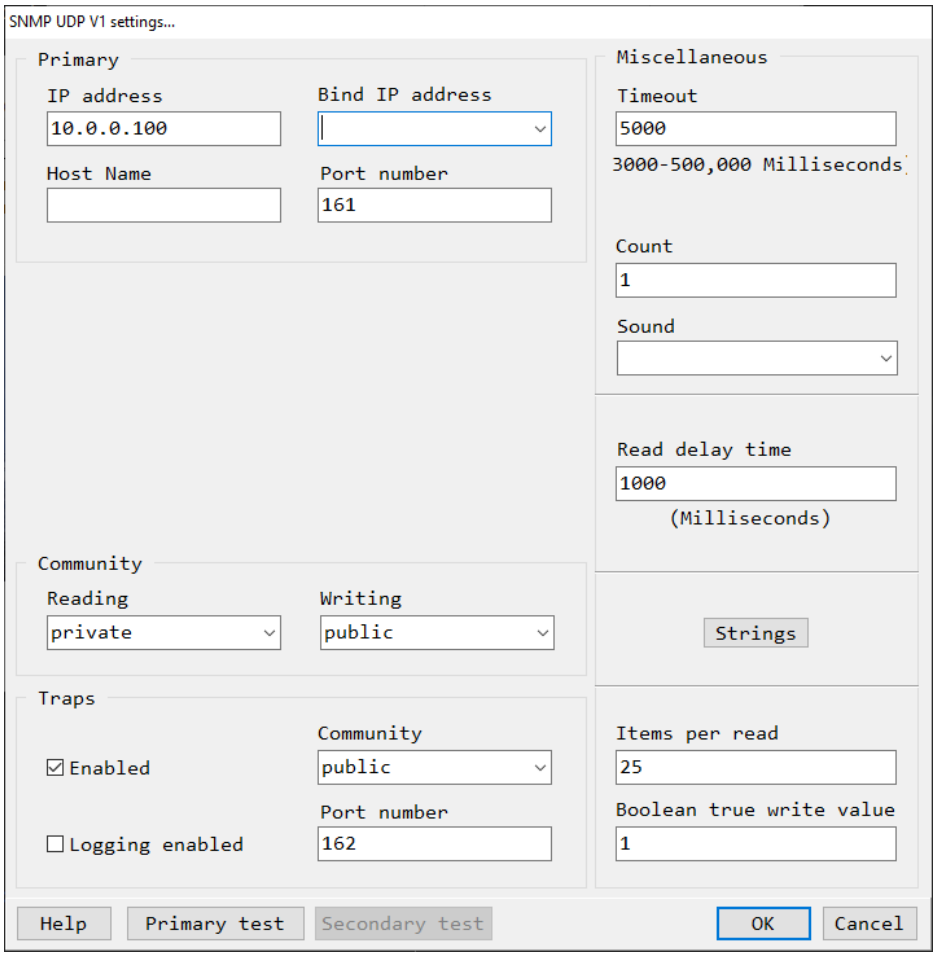


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete an SNMP UDP master object select the "Delete" button.

Settings



The UDP port has a primary configuration and if enabled a secondary configuration. Select the configuration attributes as is needed.

If a "host name" is entered, the IP address is ignored.

Enable secondary

The "Enable secondary" checkbox provides for a second path to be used as a hot backup. When in run mode the primary configuration is used for reading and writing to the connected device. If the "Enable secondary" checkbox is enabled, periodically, the run time program will attempt to communicate with the device through the secondary configuration. If it fails to communicate an entry will be made in the event log and the watchdog sound, if configured, will play.

If the watchdog timer expires the program will switch to the secondary configuration, if configured. An entry will be made in the event log and the watchdog sound, if configured, will play. If communication is established then the watchdog timer will not replay. When communications is reestablished with the device on the primary

configuration an entry will be made in the event log and operations will return to the primary port.

If the computer has two network interface cards (NIC) the program will use the first one detected for the primary and the second one detected for the secondary. If only one NIC is detected the program will use it for the primary and the secondary.

Miscellaneous

Timeout

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

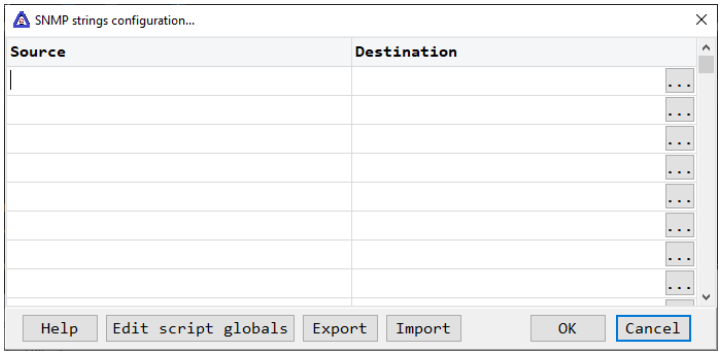
When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Count

UDP is a connectionless protocol. A client can respond to a message and the response can get lost. The HMI can only detect a communication failure when the client does not respond to a request. When a watchdog timeout occurs a counter is incremented. When the counter equals this property, the actions described above in, “Timeout”, will occur.

On each watchdog timer timeout the logic will issue the next request for data in the queue.

Strings



Strings are not regular points. They do not have alarms, printing, etc. Strings may be displayed in any window via the "Script Global" animation. Any other actions, parsing, comparing, etc. must be done in scripts.

Source

This is the object ID in the device.

Destination (optional)

If desired, select a script global location and the string will be copied to the location when the string value is returned from the external device. If this option is used or not the string may still be accessed via the "StringGet" and "StringSet" script commands. If this is configured and access to the string, in a script, is desired, use the "GlobalGet" or "StringGet" to copy the string. The format of the destination is: <section>.<name>.
Note: Writing to the script global does not write the value to the PLC. "StringSet" must be used to write a string to the device.

The format type for the "StringSet" command must be:

Name	Value
ASN1_OCTSTR	0
ASN1_IPADDR	1
ASN1_OPAQUE	2
ASN1_OBJID	3
ASN1_NULL	4

If the format type is not 0-4, 0 (ASN1_OCTSTR) will be used.

Export

This is used to export the configuration to an Excel worksheet.

Import

This is used to import string configurations from an Excel worksheet. The contents of the grid are completely replaced by the contents of the file.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:
The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Items per read

Some devices allow the reading of multiple items per read. This might need to be adjusted down to communicate with the device. Some devices only respond with a single item per read.

Boolean true write value

SNMP does not have a Boolean data type. The HMI simulates a Boolean data type, for digital points, by defining true as not equal (<>) to zero (0) and false as zero (0). Writing a digital point to a device requires the HMI to write an integer to the device. The value can be 1 (default) 255 or -1 for true and will be zero (0) for false.

Community reading/writing

The two common communities 'public' and 'private' are listed. Enter a different community name as required. The field must contain a valid name. The most common initial cause for write failures is the community name is incorrect.

Traps

The HMI can accept traps if enabled.

Community

Only traps from the community selected will be processed. Leave the field blank to process ALL traps.

Logging enabled

Traps are logged to a file. Each master port has a separate log file.
Each line has seven columns and each column is separated with a 'TAB' character.
Line format:<Local time> TAB <Community> TAB <Enterprise> TAB <IP address> TAB <Trap code> TAB <Specific trap code> TAB <Time stamp> CRLF(carriage return + line feed)

Error messages/codes

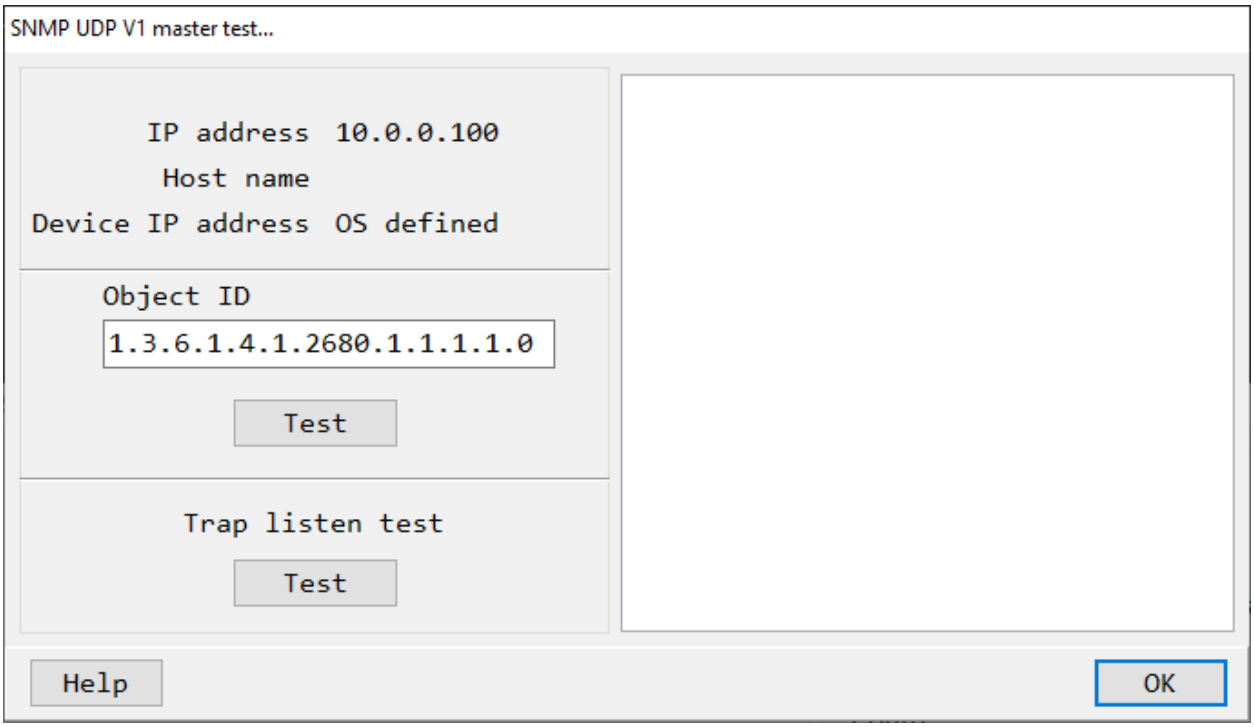
If the error message is a general message it will be a number.
If the error is associated with a request item (OID) the format is:
<read index> : <err code> : <error index>.
Example 1 : 2 : 3 : = read index 1, error code 2, item 3. Use port diagnostics to determine the point associated with the read index and item number.

Codes

0	None
1	The agent could not place the results of the requested SNMP operation in a single SNMP message.
2	The requested SNMP operation identified an unknown variable.
3	The requested SNMP operation tried to change a variable but it specified either a syntax or value error.
4	The requested SNMP operation tried to change a variable that was not allowed to change, according to the community profile of the variable.
5	An error other than one of those listed here occurred during the requested SNMP operation.
6	The specified SNMP variable is not accessible.
7	The value specifies a type that is inconsistent with the type required for the variable.
8	The value specifies a length that is inconsistent with the length required for the variable.
9	The value contains an Abstract Syntax Notation One (ASN.1) encoding that is inconsistent with the ASN.1 tag of the field.
10	The value cannot be assigned to the variable.
11	The variable does not exist, and the agent cannot create it.
12	The value is inconsistent with values of other managed objects.

13 Assigning the value to the variable requires allocation of resources that are currently unavailable.
14 No validation errors occurred, but no variables were updated.
15 No validation errors occurred. Some variables were updated because it was not possible to undo their assignment.
16 An authorization error occurred.
17 The variable exists but the agent cannot modify it.
18 The variable does not exist; the agent cannot create it because the named object instance is inconsistent with the values of other managed objects.

Primary/Secondary test button



When the test button is selected the program will display the 'SNMP Master Test...' dialog. Enter a valid object ID and select the 'Test' button. The program will attempt to issue a SNMP 'Get Request' command to the agent configured

The request packet and the response packet are displayed in the memo field. The agent must respond within three seconds or the attempt is aborted.

Addressing

Source address is the "direct code" addressing.

Example: 1.3.6.1.2.1.1.6

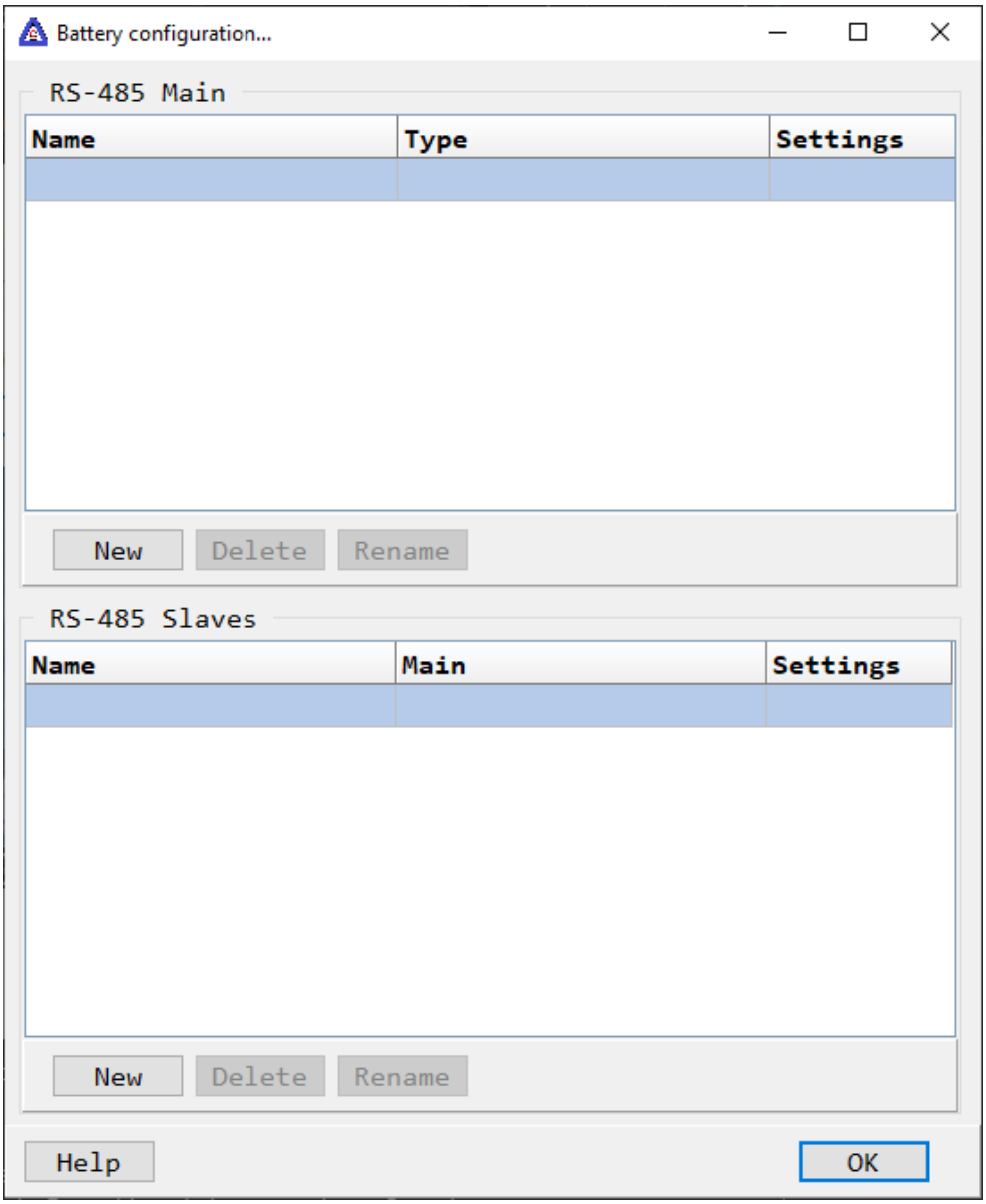
Solar

Inverter

Battery

Serial

Each battery serial master object controls one serial port and can have one or many slave objects (batteries). The slaves are configured to address one device. The port can be RS-232 or RS-485 or any port type that simulates an MS Windows com port.



Note: RS-485 is sensitive to improper wiring and/or terminations. Unpowered units can cause data echoes and other reliability issues. Please follow all RS-485 wiring guidelines.

Battery serial main

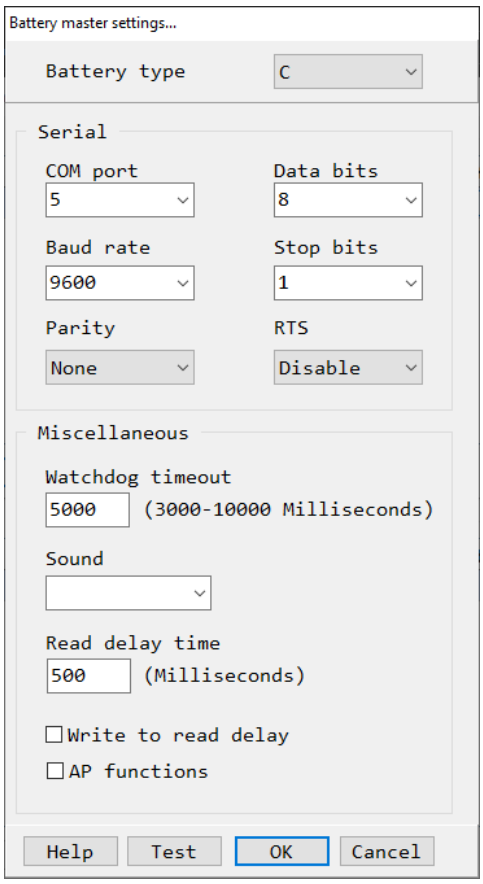
Each Battery serial master object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Battery serial master object select the "Delete" button.

Settings



Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Battery type

Type A: Tian protocol. Tested with a Trophy battery (48V100E-2UL).
Type B: Pylon 485 protocol. Tested with Sungold (SG48100P). *2
Type C: Pylon 485 protocol. Tested with RUiXU (RX-LFP48100). *1
Type D: Tested with EG4 LifePower4. (Modified Pylon protocol)

Notes:

1) The RUiXU battery addressing is dynamic. Default is zero (0). We tested with six batteries on one RS-485 loop and addressed the batteries 0 -5.
2) The battery address(s) must be 2-15 and RS-485B or RS-485C used. Use B & C to daisy chain batteries.

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

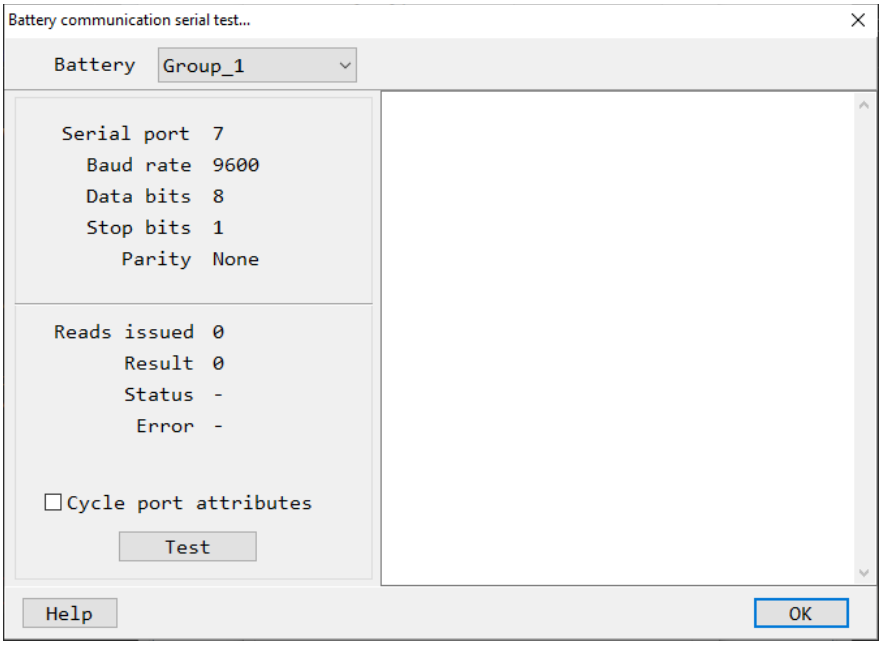
A read request is issued. When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

If the watchdog timer times out, it fires the read processing logic.

AP functions

See analog functions

Test button



When the test button is selected the program will attempt to read the “analog data from the selected slave.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

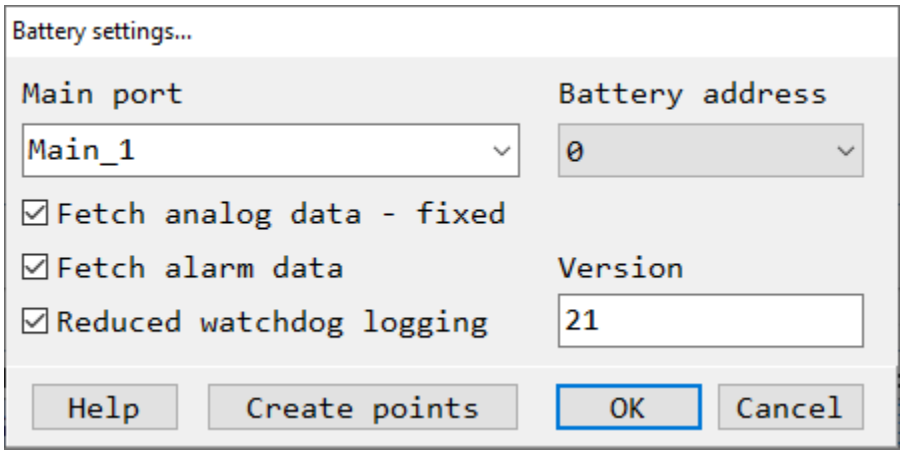
Battery serial slaves

Each Battery slave object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Battery slave object select the "Delete" button.



Main port

This is the master RS-485 port the slave is linked to. A port must be linked to a master for runtime operations.

Battery address

This is the address of the battery. (0 - 255)

Fetch analog data – fixed

This request the battery to return the analog data. e.g. voltage, current, temperatures, etc..

Fetch alarm data

This request the battery to return the alarm/status/event data. Not all batteries support this request. Contact support if assistance is needed to interpret the results.

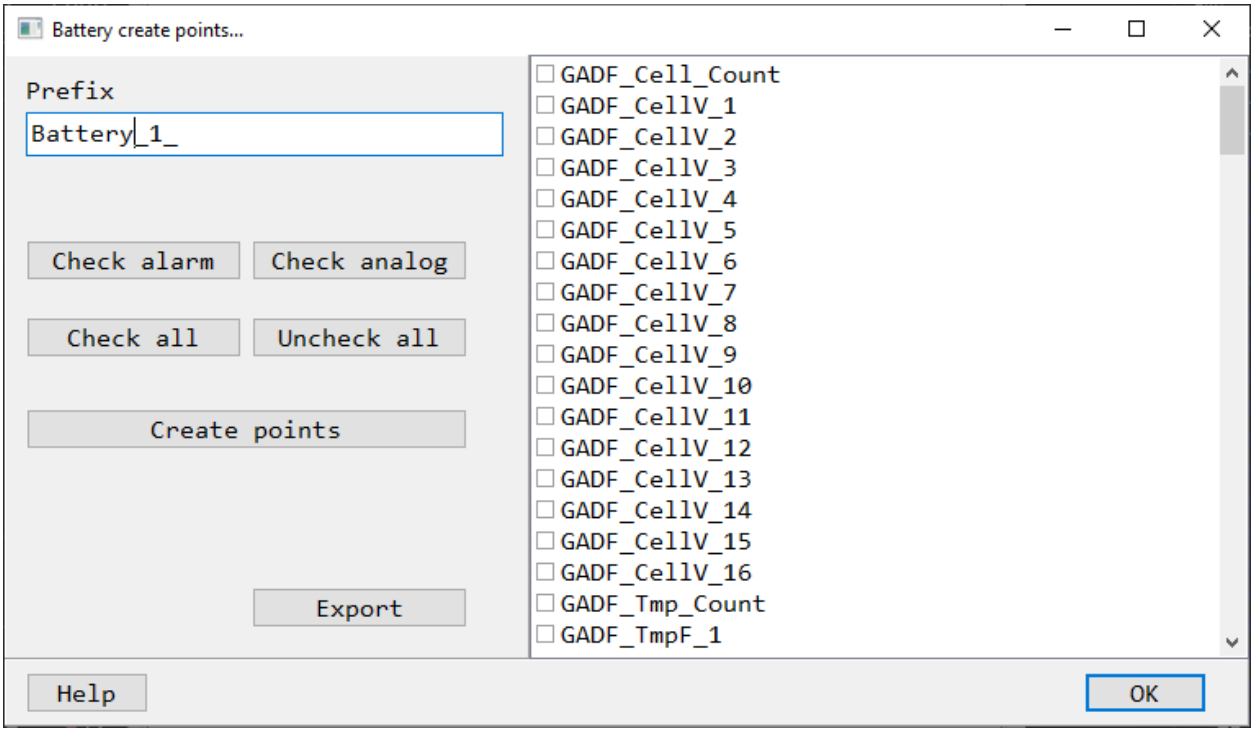
Reduced watchdog logging

When enabled and a watchdog timeout occurs, a single entry will be placed in the event log and the sound will be queued (if configured). When the slave device responds, after a watchdog timeout, a single entry will be placed in the event log. The watchdog counter will continue to count each time the watchdog timer expires.

Version

A two digit number representing the protocol version. Some batteries ignore this value and other batteries will not respond if the version number is not supported. 2.0 and 2.1 was valid for all our test batteries. Note: Remove the “.”, 2.0 = 20 and 2.1 = 21, etc..

Create points



The batteries respond with data structures. The HMI applies "address source" references to the data fields in the structure. All the supported data points are listed on the right for the battery type selected.

Prefix

When using the "Create points" to automatically create the selected point, this value is prefixed before the "address source" is added to the tagname

Example:
prefix is "Battery_1_", source GADF_CellV_1 and GADF_CellV_2 are enabled (checked).
Two points will be created
#1 Tagname: **Battery_1_ GADF_CellV_1**, source address GADF_CellV_1
#2 Tagname: **Battery_1_ GADF_CellV_2**, source address GADF_CellV_2

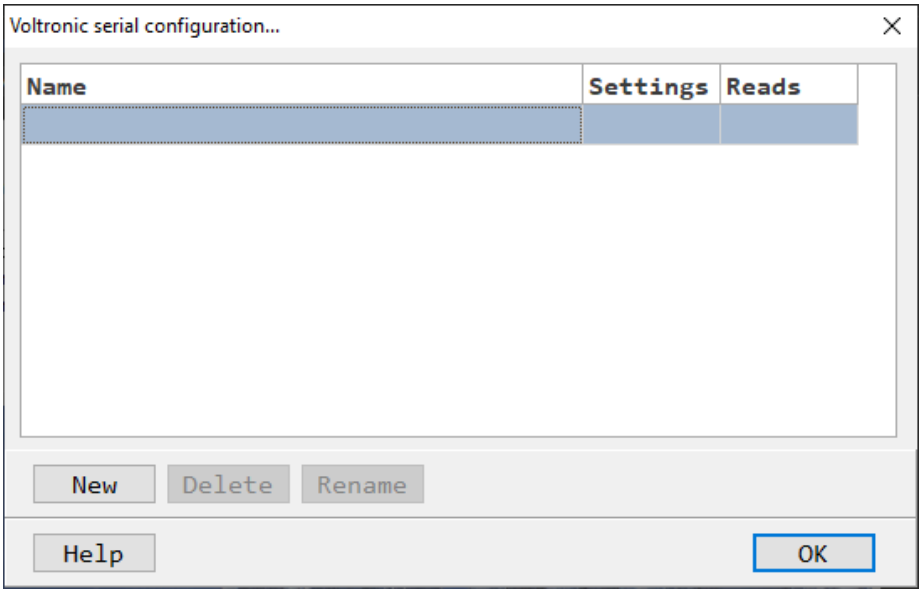
Export

The values listed on the right can be exported. The export includes the "address source", the point type (analog/digital) and the source type if applicable.

Inverter

Voltronic (Serial)

Each Voltronic object controls one serial port and communicates with one inverter.



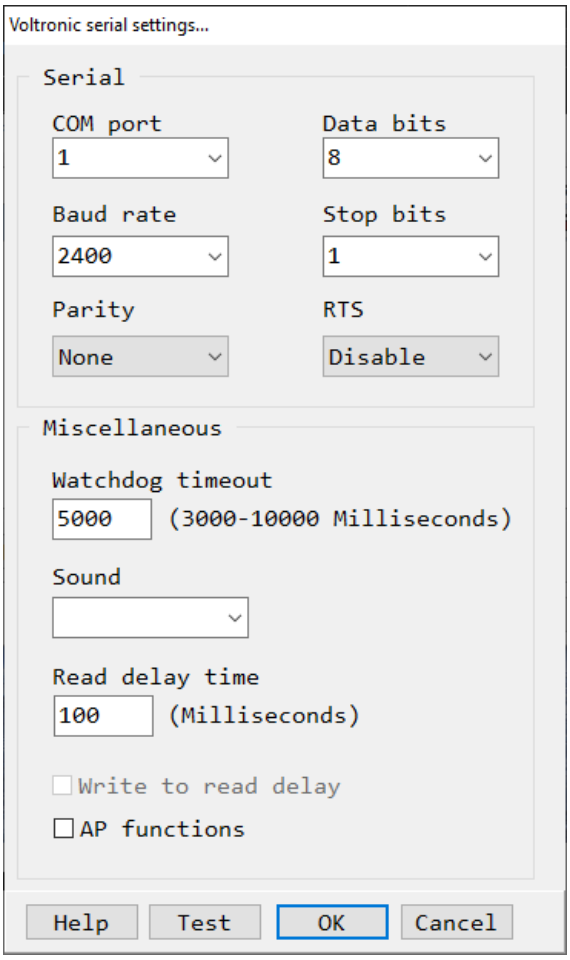
Each Voltronic object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Voltronic object select the "Delete" button.

Settings



Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

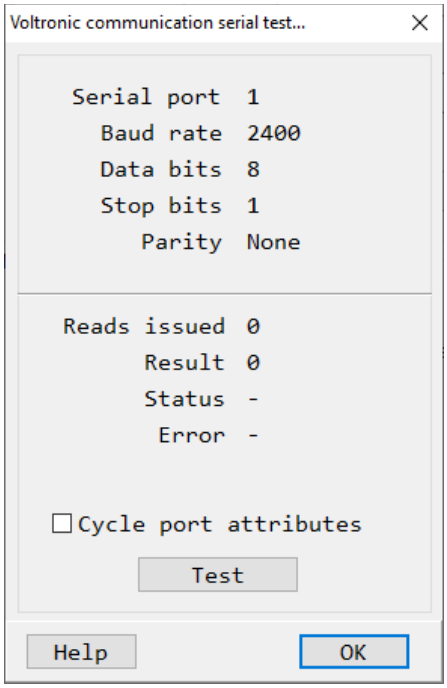
Write to read delay

After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a 'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

AP functions

See analog functions.

Test button



When the test button is selected the program will attempt to read the “^P003PI” data (Query protocol ID) from the selected slave.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Reads

Voltronic serial reads...

#	Query	Enabled	Description
1	^P003PI	☐	Query protocol ID
2	^P003ID	☐	Query series number
3	^P004VFW	☐	Query CPU version
4	^P005VFW2	☐	Query secondary CPU version
5	^P005VFWT	☐	Query DSP and MCU version
6	^P003MD	☐	Query device model
7	^P005PIRI	☐	Query rated information
8	^P003GS	☐	Query general status
9	^P004GS2	☐	Query generator and secondary output information
10	^P003PS	☐	Query power status
11	^P004PS2	☐	Query generator and secondary output information
12	^P004MOD	☐	Query working mode
13	^P003WS	☐	Query warning status
14	^P005FLAG	☐	Query enable/disable flag status
15	^P002T	☐	Query current time
16	^P003ET	☐	Query total energy generated
17	^P004COV	☐	Query AC input voltage acceptable range for feed power

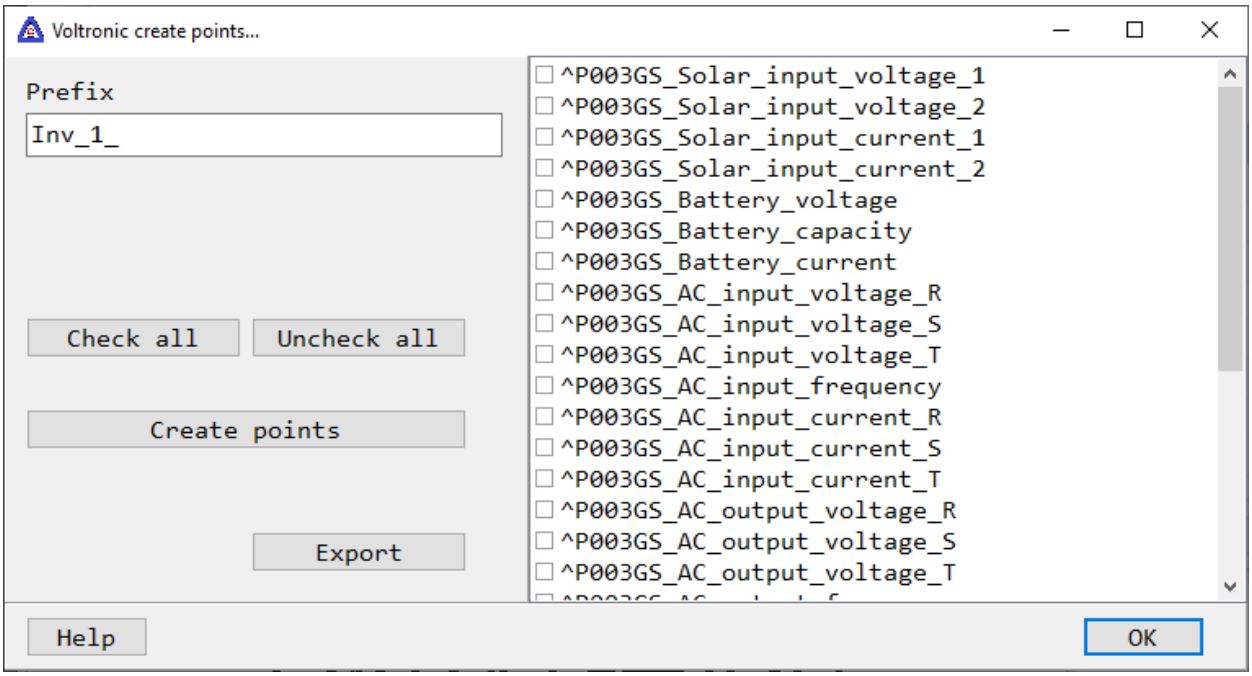
Help | Create points | OK

The protocol returns data in structures. The HMI parses the structures. The "source address" for the structure elements can be seen when the mouse is over a query "description" cell. Example for "^P003GS":

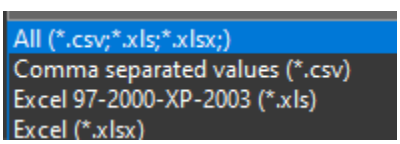
```
^P003GS_Solar_input_voltage_1
^P003GS_Solar_input_voltage_2
^P003GS_Solar_input_current_1
^P003GS_Solar_input_current_2
^P003GS_Battery_voltage
^P003GS_Battery_capacity
^P003GS_Battery_current
^P003GS_AC_input_voltage_R
^P003GS_AC_input_voltage_S
^P003GS_AC_input_voltage_T
^P003GS_AC_input_frequency
^P003GS_AC_input_current_R
^P003GS_AC_input_current_S
^P003GS_AC_input_current_T
^P003GS_AC_output_voltage_R
^P003GS_AC_output_voltage_S
^P003GS_AC_output_voltage_T
^P003GS_AC_output_frequency
^P003GS_Inner_temperature
^P003GS_Component_max_temperature
^P003GS_External_battery_temperature
^P003GS_Setting_change_bit
^P003GS_L1-L2_OP_Angle
```

At runtime a “query” command is sent to the inverter for each enabled query. The first enabled read is issued, the complete list is processed and begins again at the first enabled query. Queries can be enabled/disabled via the “SetPortEnable” script command (Use the number in the # column). At least one query must always be enabled.

When at least one query is enabled the “Create points” button is enabled. Selecting the button opens the create points dialog. This provides a method to create “points” with the correct point port name, point type, source address and data type. Check (enable) the points to create.



The data can be exported via the “Export” button to a file:



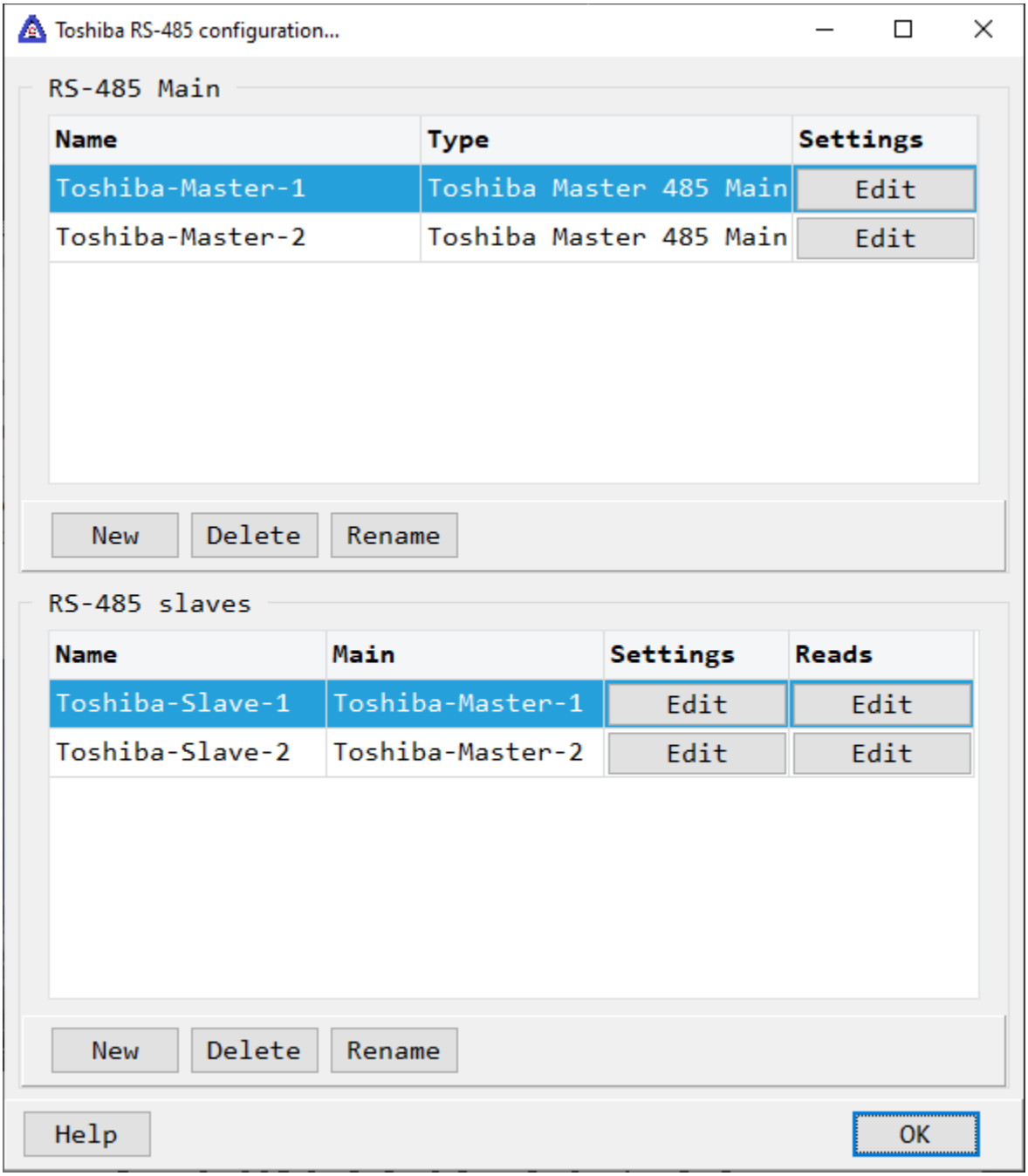
Prefix

When using the “Create points” to automatically create the selected points, this value is prefixed before the “address source” to create a tagname.

Note: These queries return string data:
**^P003ID* **^P004VFW* **^P005VFW2* **^P005VFWT*
Use the script command “StringGet” to access the string.
Examples:
sValue:=StringGet('<port name>', '*^P003ID_Series_number',0,0);
sValue:=StringGet('<port name>', '*^P005VFWT_MCU_version',0,0);

Toshiba Serial 232/485

Each Toshiba master object controls one serial port and can have one or many slave objects. The slaves are configured to address one device.



Note: RS-485 is sensitive to improper wiring and/or terminations. Unpowered units can cause data echoes and other reliability issues. Please follow all RS-485 wiring guidelines.

Toshiba serial main

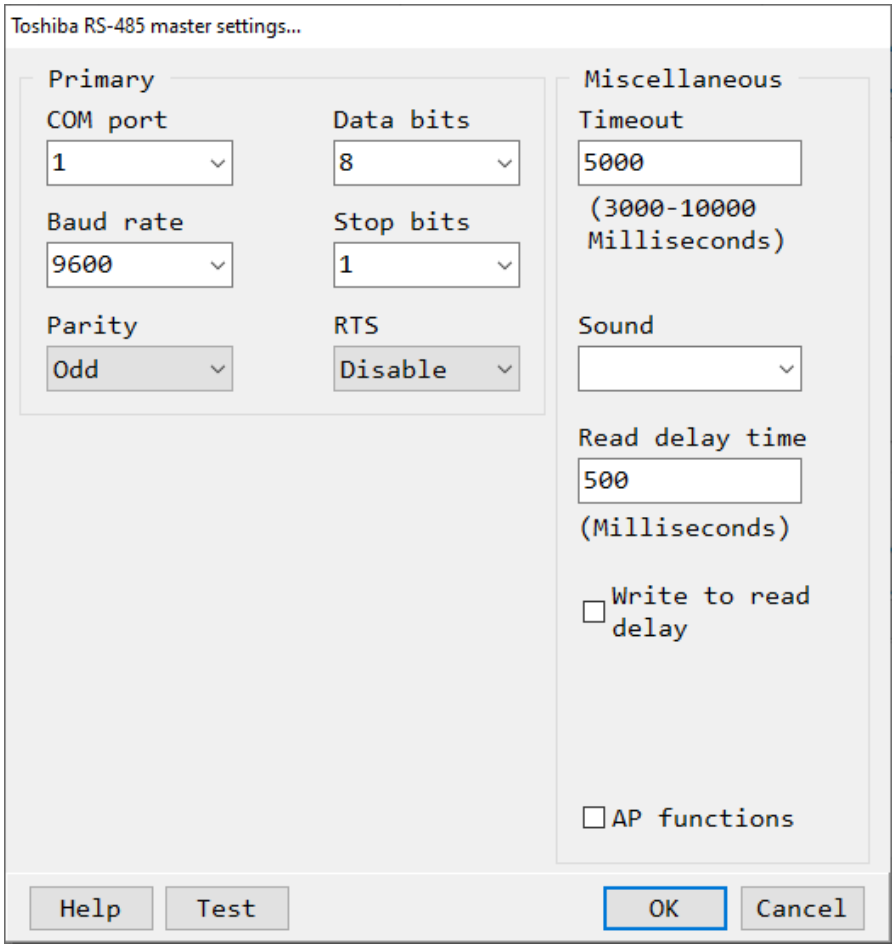
Each Toshiba master object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Toshiba master object select the "Delete" button.

Settings



Select the port attributes as required.

If a non-standard baud rate is required and the com port on the PC supports the baud rate, type in the desired rate. (Do not include any commas. The number must be a whole number.)

Miscellaneous

RTS

See here.

Timeout

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read request.

Operation:

The read request are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.

To prevent flooding due to write request the next read request is always issued after a write request.

If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

Write to read delay

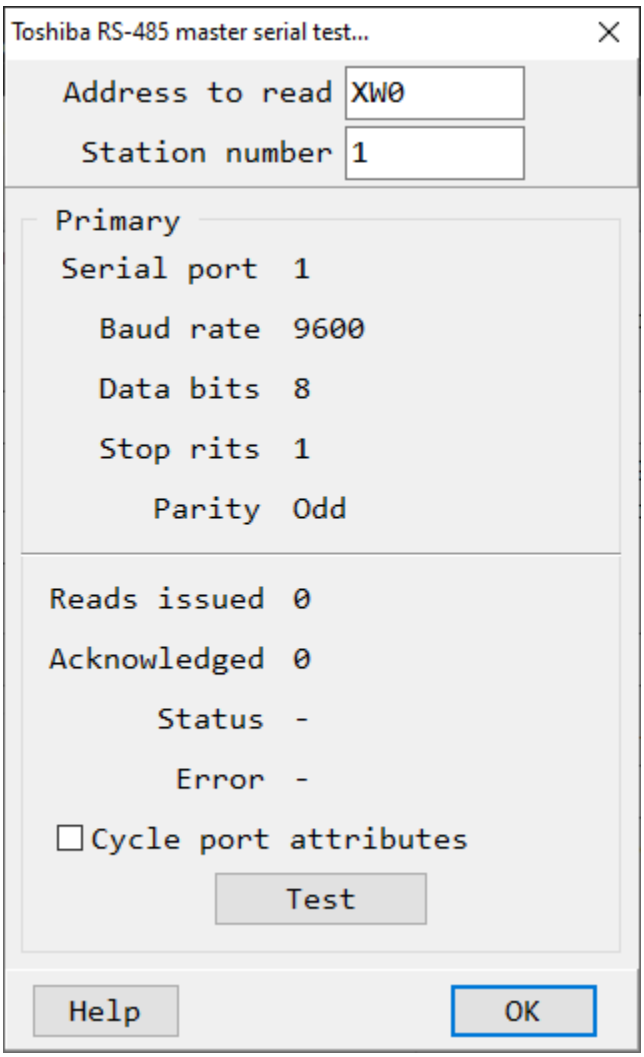
After the write response is received from the PLC the next read is issued. Normally, the next read request can be sent without delay. Some computers, mostly notebooks/laptops and some external devices cannot switch from transmit to receive fast enough causing the read request after the write response to fail, generating a

'watchdog' time out. If this is occurring, enable this attribute and the 'read delay time' will be used to delay the read request.

AP functions

See analog functions.

Test button



When the test button is selected the program will attempt to read one point of data from the device at the address entered.

The program will attempt to use the communication parameters configured.

If the "Cycle port attributes" checkbox is checked the program will cycle through the user configured communication settings and some standard parameters. The test will loop checking for a connection. If a connection is detected a dialog box will appear and the looping will end.

Toshiba serial slaves

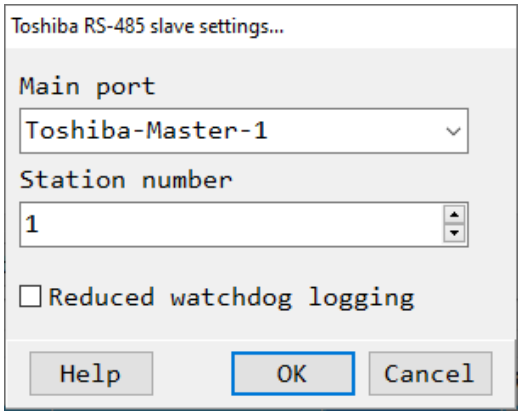
Each Toshiba slave object is listed in the window.

To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Toshiba slave object select the "Delete" button.

Settings



Main port

This is the master RS-485 port the slave is linked to. A port must be linked to a master for runtime operations.

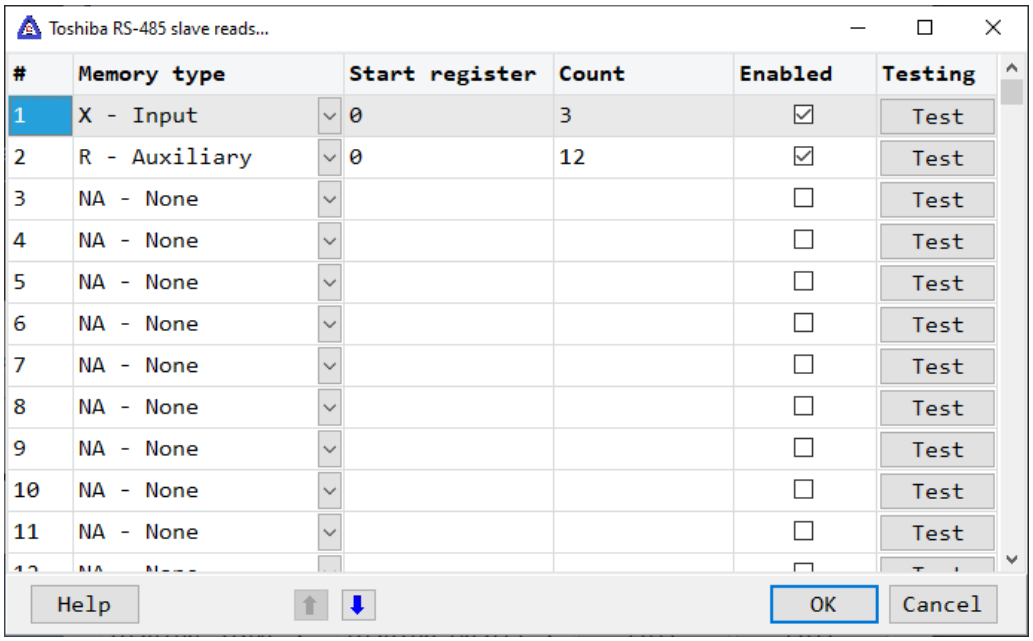
Station number (Future)

This is the station number of the slave device. (1 - 255)

Reduced watchdog logging

When enabled and a watchdog timeout occurs, a single entry will be placed in the event log and the sound will be queued (if configured). When the slave device responds, after a watchdog timeout, a single entry will be placed in the event log. The watchdog counter will continue to count each time the watchdog timer expires.

Reads



#	Memory type	Start register	Count	Enabled	Testing
1	X - Input	0	3	☑	Test
2	R - Auxiliary	0	12	☑	Test
3	NA - None			☐	Test
4	NA - None			☐	Test
5	NA - None			☐	Test
6	NA - None			☐	Test
7	NA - None			☐	Test
8	NA - None			☐	Test
9	NA - None			☐	Test
10	NA - None			☐	Test
11	NA - None			☐	Test

The address ranges shown may or may not be present in the slave device.

Registers

Address examples

X01	Word 0 bit 1
X283	Word 28 bit 3
X12E	Word 12 bit E
Y37	Word 3 bit 7
Y21F	Word 21 bit F
T3	Timer 3
C17	Counter 17
T.64	Timer 64 complete flag
D44	Data register 44

Note: For X, Y, R and S registers the Toshiba specification states the last digit must be in hexadecimal and the leading digits up to 3 places must be decimal. 1F is legal, F1 is not. In hexadecimal when 1 is added to 9F the result is A0. A0 would not be legal per the specification. After 9F in the specification is 100. Word 10 bit 0. Verify the external device has the address desired.

Point configuration source field for analog points.

Source Type	Range	Register count (words)
None	0-65535	1
Float IEEE-754	standard format	2
Integer	-2,147,483,648 .. 2,147,483,647	2
BCD2	Not applicable	
Small integer	-32,768 .. 32,767	1
Unsigned integer	0 ... 4,294,967,295	2

Notes:
1) Points using two registers must start on an even register; Valid D0, D2, D4, etc. Invalid D1, D3, D5, etc.
2) Using the 'StringSet' script command is limited to 32 registers. Using format 0, 1, or 2 allows for a maximum of 64 characters. Using format 3 or 4 allows for a maximum of 32 characters.
3) Per the Toshiba specification, T. and C. address are read only.
4) When writing the accumulator of a timer and the timer is not controlled via ladder logic, the timer complete flag will be cleared.

Memory type

Register Type	Prefix	Data Type	Format
Input discrete	X, XW	Boolean/Word	DDH (Dec, Dec, Hex), DD(Dec, Dec)
Output relay	Y, YW	Boolean/Word	DDH (Dec, Dec, Hex), DD(Dec, Dec)
Auxiliary relay	R, RW	Boolean/Word	DDDH (Dec, Dec, Dec, Hex), DDD(Dec, Dec, Dec)
Special devices	S, SW	Boolean/Word	DDH (Dec, Dec, Hex), DD(Dec, Dec)
Timer	T	Word Dec	
Counter	C	Word Dec	
Data Register	D	Word Dec	
Link register	L, LW	Boolean/Word	DDH (Dec, Dec, Hex), DD(Dec, Dec)

Start register

The starting register to read. This is the word address of the memory area.

Count

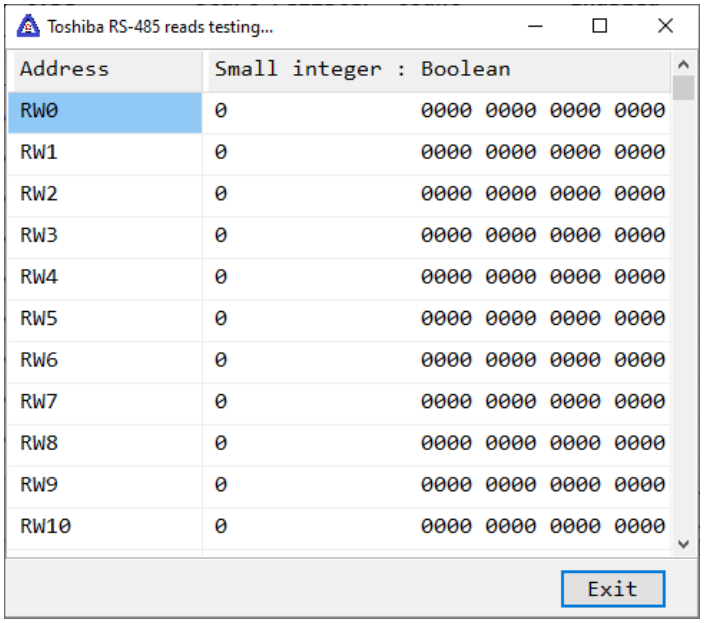
This is the number of words/items to return. The per read limit is 32.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



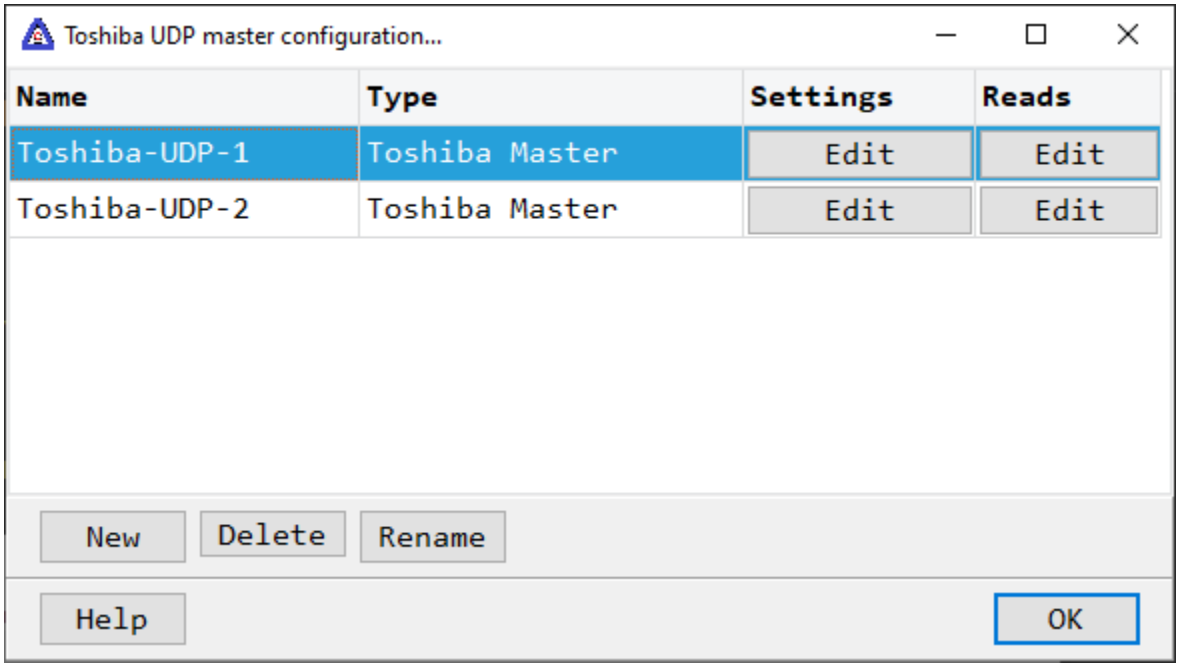
When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

Toshiba UDP

Each Toshiba UDP master object is listed in the window.

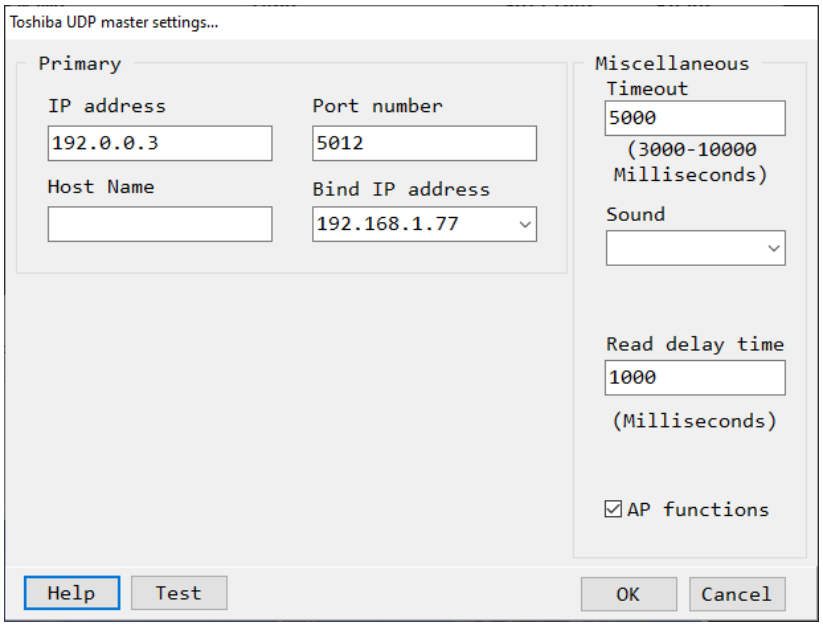


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a Toshiba UDP master object select the "Delete" button.

Settings



If a "host name" is entered, the IP address is ignored.

Bind IP address See here.

Miscellaneous

Watchdog timer

The timer begins timing when the request for data is transmitted to the device. The watchdog timer is reset when data is received via the configured media or the timer completes.

When the timer completes an entry is made in the event log and a sound is played if configured. To not play a sound delete, the value in the sound name field.

Read delay time

To allow the throttling of data request to the slave device the value entered is the delay time between read requests.

Operation:

The read requests are issued, if the read request is enabled, from top to bottom as configured.

A read request is issued. When a response to the last issued command is received and a write request is pending it is issued.

When a response to the last issued command is received and the time value is greater than 0 the timer starts. When the timer expires the next read request is issued. If the time value is not greater than 0 the next read request is issued.
To prevent flooding due to write request the next read request is always issued after a write request.

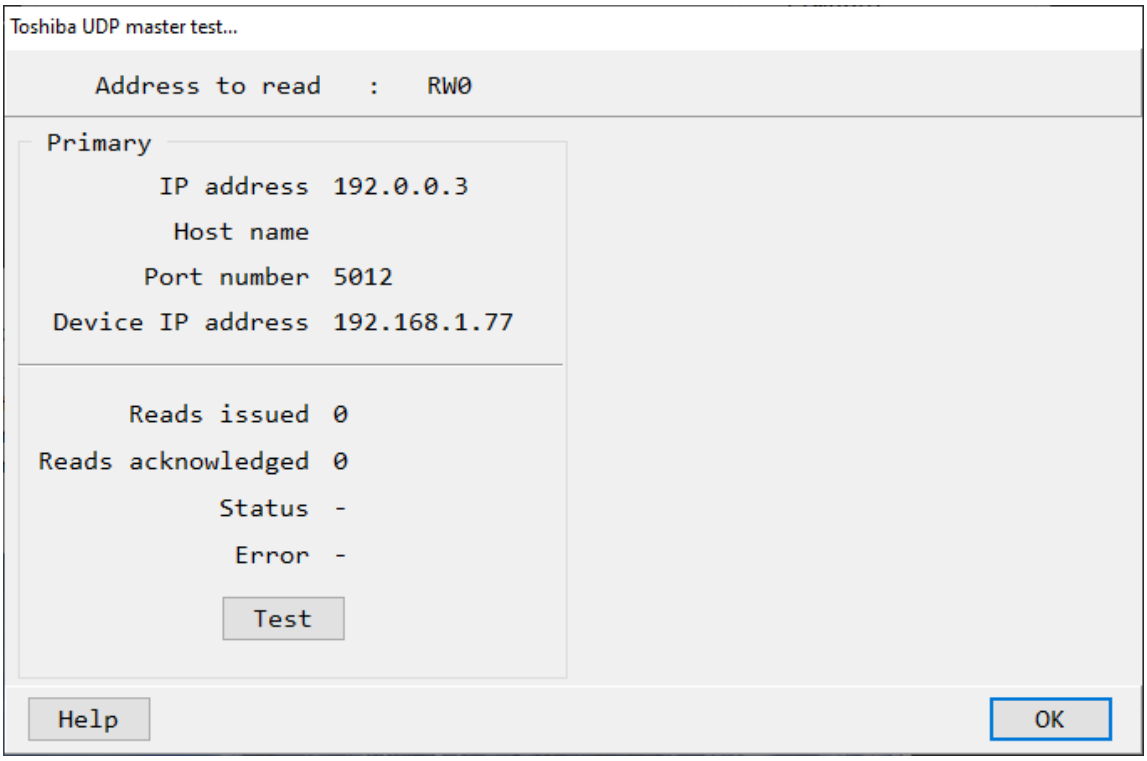
If the watchdog timer times out it fires the read/write processing logic.

If a response is not received for a write command the write is attempted two more times. If it fails all three times an entry is made in the event log.

AP functions

See analog functions.

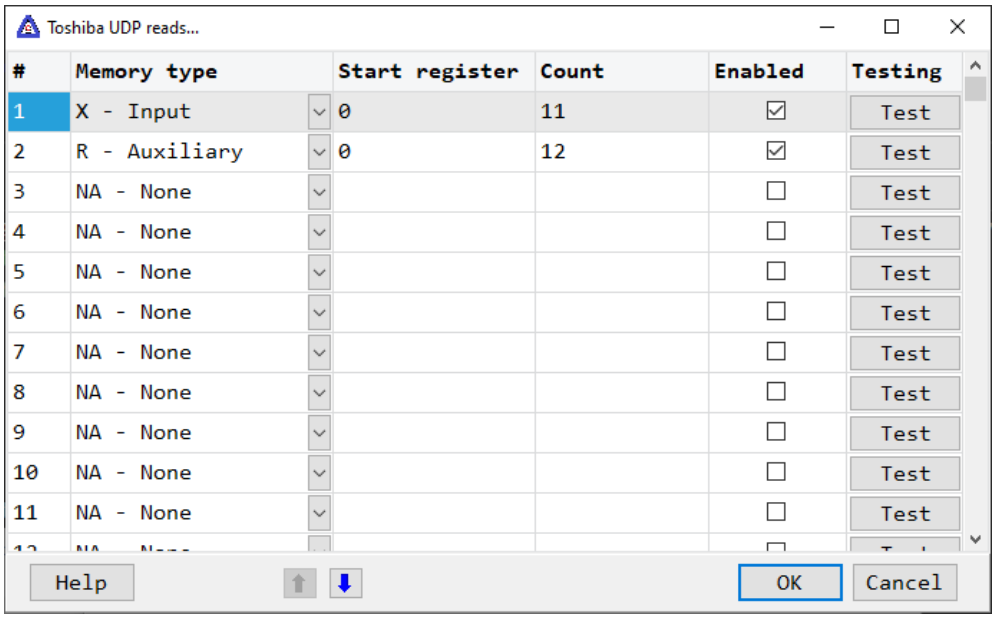
Test button



When the test button is selected the program will send a read request to read one word starting at RW0.

The program will attempt to use the communication parameters configured.

Reads



#	Memory type	Start register	Count	Enabled	Testing
1	X - Input	0	11	☑	Test
2	R - Auxiliary	0	12	☑	Test
3	NA - None			☐	Test
4	NA - None			☐	Test
5	NA - None			☐	Test
6	NA - None			☐	Test
7	NA - None			☐	Test
8	NA - None			☐	Test
9	NA - None			☐	Test
10	NA - None			☐	Test
11	NA - None			☐	Test

The address ranges shown may or may not be present in the slave device.

Registers

Address Examples

X01	Word 0 bit 1
X283	Word 28 bit 3
X12E	Word 12 bit E
Y37	Word 3 bit 7
Y21F	Word 21 bit F
T3	Timer 3
C17	Counter 17
T.64	Timer 64 complete flag
D44	Data register 44

Note: For X, Y, R and S registers the Toshiba specification states the last digit must be in hexadecimal and the leading digits up to 3 places must be decimal. 1F is legal, F1 is not. In hexadecimal when 1 is added to 9F the result is A0. A0 would not be legal per the specification. After 9F in the specification is 100. Word 10 bit 0. Verify the external device has the address desired.

Point configuration source field for analog points.

Source Type	Range	Register count (words)
None	0-65535	1

Float IEEE-754	standard format	2
Integer	-2,147,483,648 .. 2,147,483,647	2
BCD2	Not applicable	
Small integer	-32,768 .. 32,767	1
Unsigned integer	0 ... 4,294,967,295	2

Notes:

1) Points using two registers must start on an even register; Valid D0, D2, D4, etc. Invalid D1, D3, D5, etc.
2) Using the 'StringSet' script command is limited to 32 registers. Using format 0, 1, or 2 allows for a maximum of 64 characters. Using format 3 or 4 allows for a maximum of 32 characters.
3) Per the Toshiba specification, T. and C. address are read only.
4) When writing the accumulator of a timer and the timer is not controlled via ladder logic, the timer complete flag will be cleared.

Memory type

Register Type	Prefix	Data Type	Format
Input discrete	X, XW	Boolean/Word	DDH (Dec, Dec, Hex), DD(Dec, Dec)
Output relay	Y, YW	Boolean/Word	DDH (Dec, Dec, Hex), DD(Dec, Dec)
Auxiliary relay	R, RW	Boolean/Word	DDDH (Dec, Dec, Dec, Hex), DDD(Dec, Dec, Dec)
Special devices	S, SW	Boolean/Word	DDH (Dec, Dec, Hex), DD(Dec, Dec)
Timer	T	Word Dec	
Counter	C	Word Dec	
Data Register	D	Word Dec	
Link register	L, LW	Boolean/Word	DDH (Dec, Dec, Hex), DD(Dec, Dec)

Start register

The starting register to read. This is the word address of the memory area.

Count

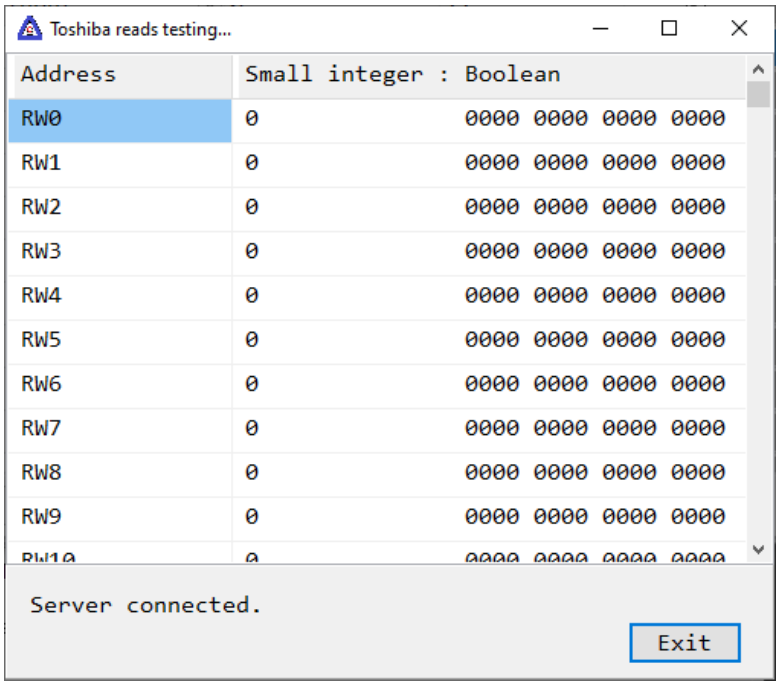
This is the number of words/items to return. The per read limit is 32.

Enabled

The read request are processed from 1 to n. If the read is enabled it will be processed. If the read is disabled the read will be skipped. The enable attribute is accessible from scripts.

For example: One or more screens that displays some data. This data does no alarming or trending and is only viewed from the user while viewing a screen. A read can be created that request the data. When the user opens the screen for viewing the read can be enabled and when the user closes the screen the read can be disabled.

Test



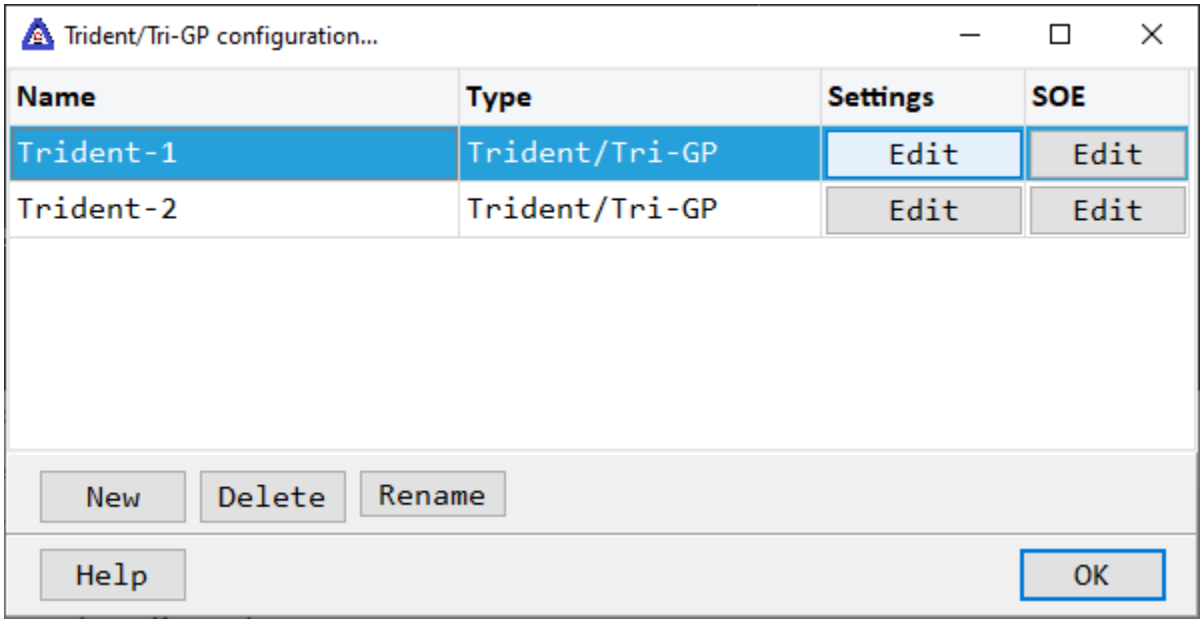
When the button is selected the program will attempt to access the data in the device using the communication parameters configured. By default the primary configuration is used for testing. To use the secondary configuration hold down the "Ctrl" key while pressing the test button.

Error messages

No memory type selected	A memory type must be selected.
Start out of range	Start register must be: 0 - 65535
Count exceeds limit	The number of registers to read is too high for the memory type.
Start register + count exceeds limit	Register start + count greater than memory end.
Count < 1	Must read at least one register.

TRIDENT/TRI-GP

Each Trident/Trip-GP object is listed in the window.



To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

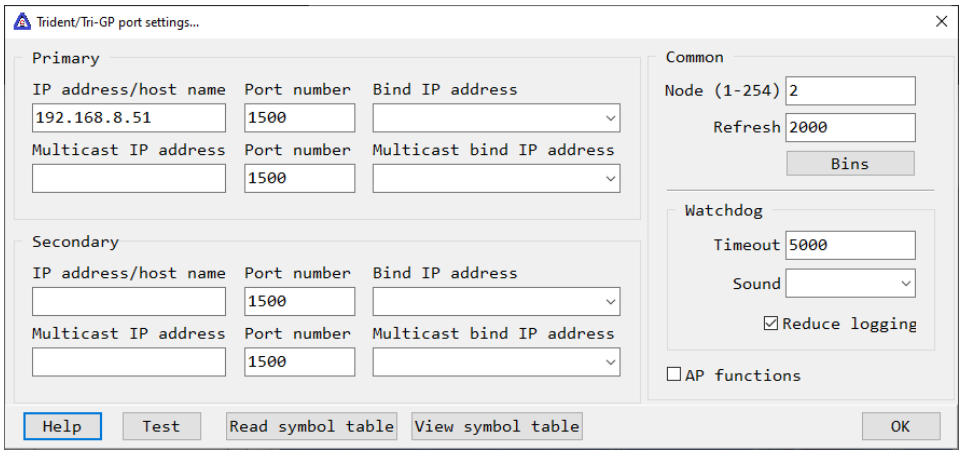
To rename an object select the "Rename" button and enter a new name.

To delete a Trident/Trip-GP object select the "Delete" button.

Notes:

1) Trident/Trip-GP uses UDP for communications. While “binding” to a NIC is not required it is suggested.
2) The primary channel IP address and port number must be present. All other communication channels are optional.
3) The secondary channels operate in “hot standby”. If the primary communication channel has a watchdog timeout, the secondary channel(s) will begin data parsing.
4) Trident/Tri-GP uses version 1 of the TSAA protocol. The HMI does not support the TriStation emulator. The emulator emits data that is not restricted to version 1 of the TSAA protocol.
5) Our testing used TriStation version 5.0.0 Build 539.
6) If collecting SOE data and multicast is enabled, duplicate entries may be placed in the log.
7) At runtime monitoring start the symbol table information is collected from the controller.
8) To force HMI runtime to fetch symbol table information, use the script command “SetPortReadEnable” with index 1. Example: value:=SetPortReadEnable('Tri-Gp-Pump',1,true);

Settings



Primary

IP Address This value can be a IPv4 address or a host name.
Port number The port number (default 1500 for TSAA).
Bind IP Address The NIC to bind communication flow. Optional but suggested.

Multicast (optional)
IP Address This value can be a IPv4 address or a host name.
Port number The port number (default 1500 for TSAA).
Bind IP Address The NIC to bind communication flow. Optional but suggested.

Secondary (optional)

IP Address This value can be a IPv4 address or a host name.
Port number The port number (default 1500 for TSAA).
Bind IP Address The NIC to bind communication flow. Optional but suggested.

Multicast (optional)
IP Address This value can be a IPv4 address or a host name.
Port number The port number (default 1500 for TSAA).
Bind IP Address The NIC to bind communication flow. Optional but suggested.

Common

Node	The processor chassis address.
Refresh	The, time in milliseconds, between broadcast of the requested bins.

Bins

Specifies the bin data to request from the controller.

Watchdog

The watchdog timer for Trident/Tri-GP is a free running timer that begins when runtime monitoring starts and is reset/restarted when data is received. The primary and secondary each have a watchdog timer. UDP is a connectionless protocol, the watchdog timer value should be at least 2.5 times the “refresh” time. If multicast is enabled, the timer value should be at least 2.5 time the update rate configured in the TriStation.

When the primary watchdog timer timeouts out, a secondary port is configured and active, the secondary port begins processing data and continues until the primary communications is restored.

At runtime start and when communications is restored, after a watchdog timeout, the primary will fetch the symbol configuration data. The secondary only fetches the symbol configuration data at runtime start if the primary does not establish communications.

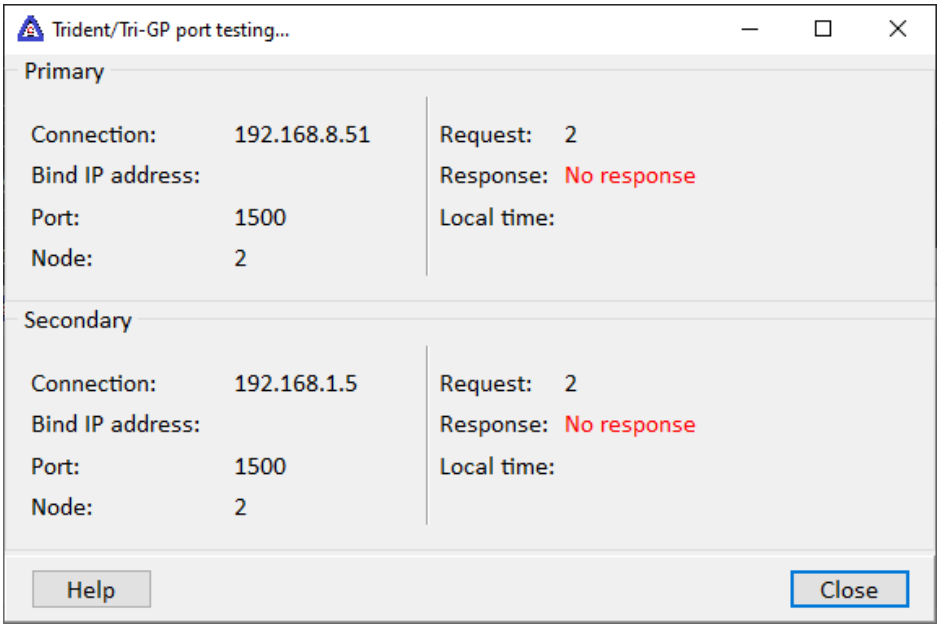
Sound	If configured, the sound to play when the timer completes.
Reduced logging	If enabled and a watchdog timeout occurs, only one entry will be placed in the event log. The watchdog timeout condition must “reset/clear” and timeout again before another entry is added to the event log.

AP functions

See analog functions.

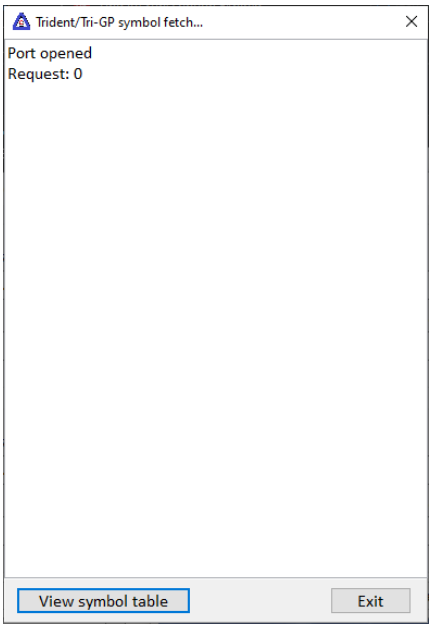
Test

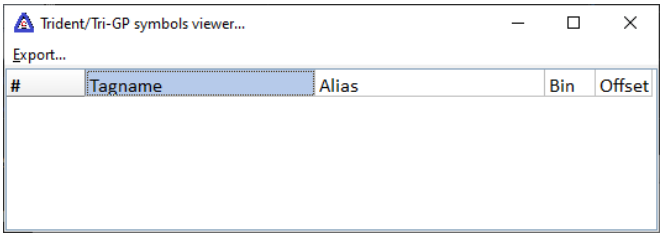
The program will attempt to connect to the PLC and read the system time.



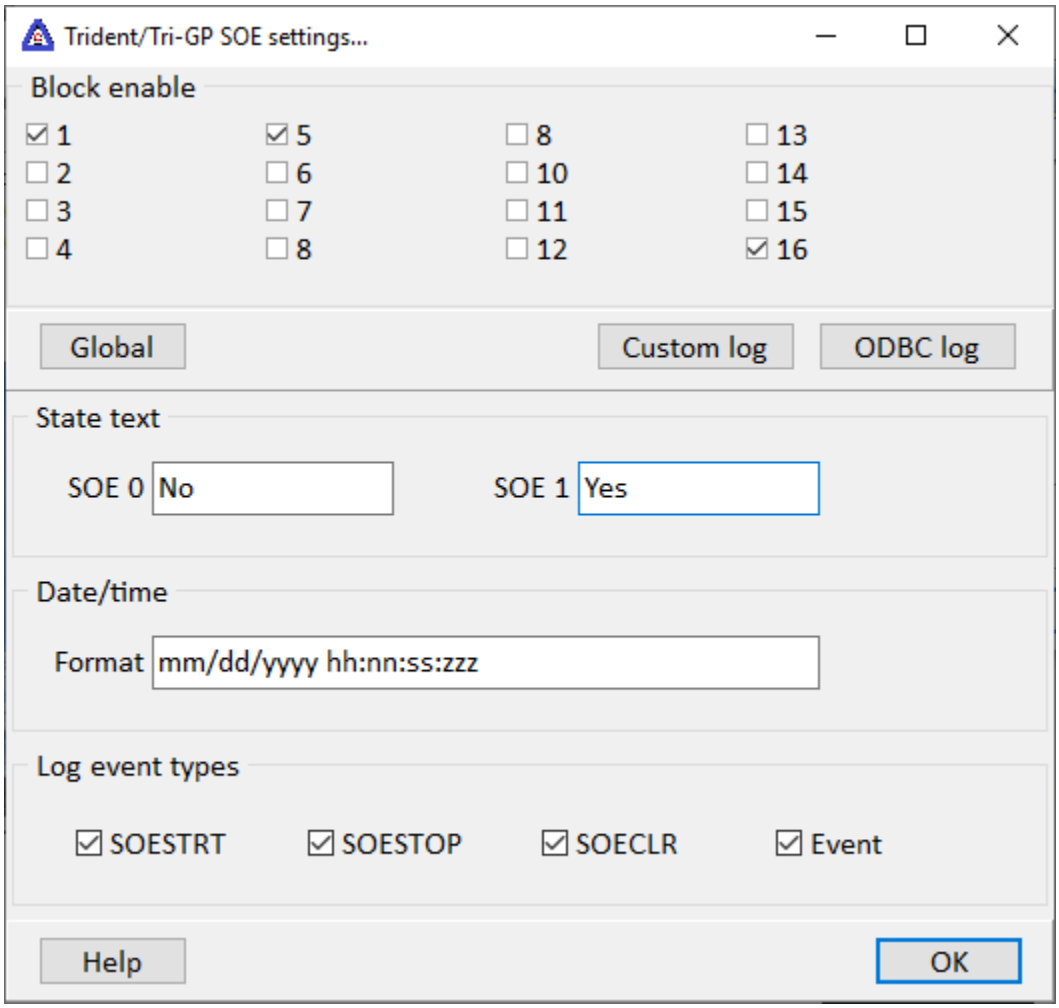
Read/View symbol table

Provided to read and view the symbol table information in the controller. The HMI reads the symbol table information at runtime start, after a primary watchdog timeout or when commanded via a “SetPortReadEnable” script command. See note #8 above.





SOE (Sequence of events)



The HMI can collect the SOE block states and the entries in the blocks.

Notes:

1) When HMI runtime monitoring begins. all the block states and entries are collected. If logging is enabled the logs may contain duplicate entries.
2) When multicast is enabled and logging is enabled, the logs may contain duplicate entries.
3) To force HMI runtime to fetch all SOE information, use the script command "SetPortReadEnable" with index 2. Example: value:=SetPortReadEnable('Tri-Gp-Pump',2,true); This will clear all internal SOE data and a fetch of SOE data when the controller sends the "SOE Data Available" message.

Block enables

Select the blocks for SOE entry collection. The HMI receives block state data for all SOE blocks and will only request/collect entries for enabled blocks.

State text

SOE 0/SOE 1 The text to place in the log for an event entry value of 0 and 1.

Date/time

Format This defines the format for the date/time log entries when Date/Time Is enabled. See "Mask" for example formats. A blank format uses the "c" mask.

Log event types

SOE entries can be a timestamp or data event type. The timestamp has four types. Enable the types required for logging.

Example log:

Block #	Entry type	Reason	Date/Time	Offset	Value
1	Timestamp	SOESTRT	10/28/2019 09:48:37:119		
1	Timestamp	Event	10/28/2019 10:02:41:744		
1	Data entry			0	True

Custom log

Log name The name of the custom log.

Column/Data Select the SOE entry fields to log and in what order. If not all columns are configured the first "Undefined" field defines the end of columns.

To use the custom logging, configure the custom log as needed. The HMI will log the values to the log as required.

ODBC log

Logger name The name of the ODBC logger.

Column/Data Select the SOE entry fields to log and in what order. If not all columns are configured the first "Undefined" field defines the end of columns.

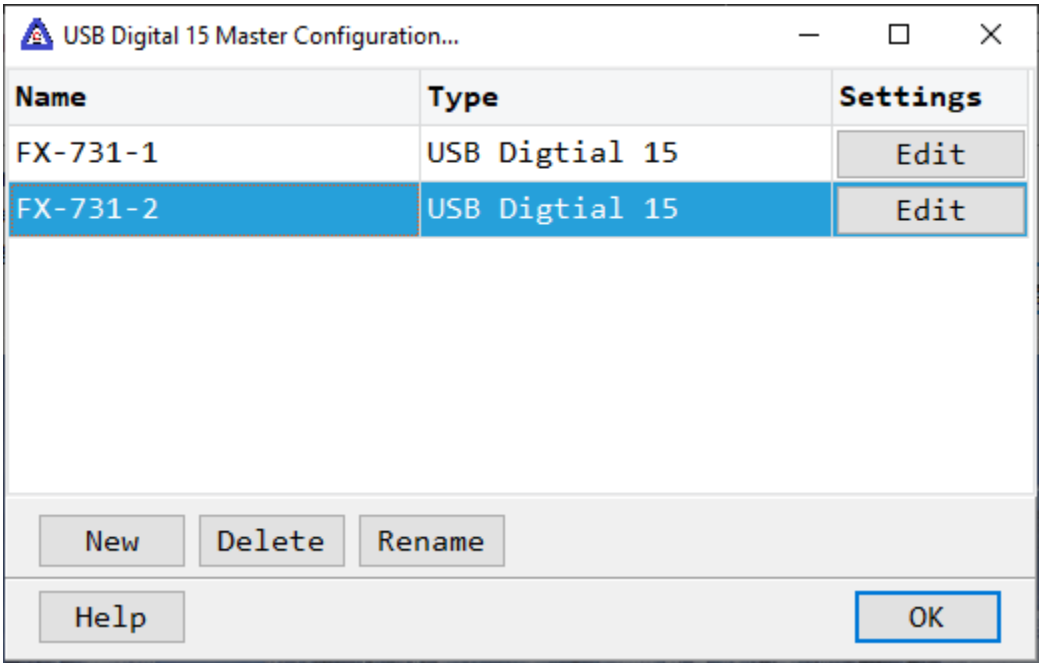
To enable ODBC logging:

1) Create an ODBC logger port.
2) Configure the connection parameters.
3) Add **ODBCLoggingControlExternal=1** to the connection parameters.
4) Check the "Enabled" checkbox.
5) Set the "Refresh time" to 0 (zero).
6) Set the "Table name" as needed.

7) Define the table field names. The count of fields names must match the “Field/Data” count. The field names must be the values in the database table. Leave the “Source data” values empty. The SOE ODBC logger will log the values defined, in the order configured, using the field names defined.
8) Select the created ODBC Logger port in the “Logger name” drop list.
9) Select the SOE entry fields to log and in what order.

USB Digital I/O 15 (FX-731C)

Each USB Digital I/0 15 master object is listed in the window.

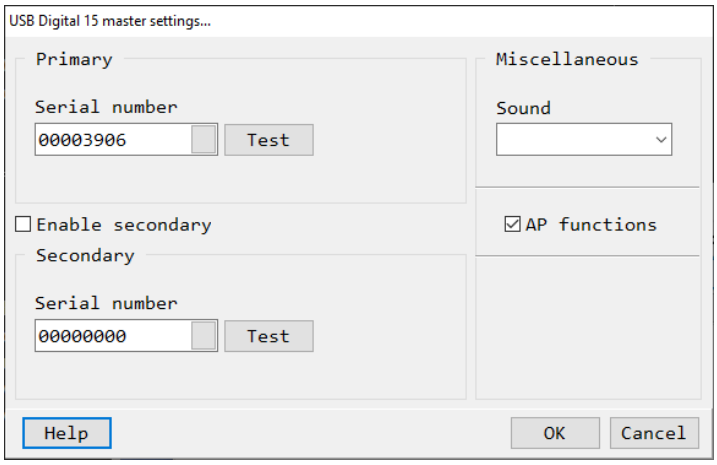


To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.

To rename an object select the "Rename" button and supply a new name.

To delete a USB Digital I/0 15 master object, select the "Delete" button.

Settings



The port has a primary configuration and if enabled a secondary configuration. The I/O device reports any change of state in the digital inputs to the program.

Serial number

Each digital I/O 15 device has a unique serial number. The number is eight digits long and is in hexadecimal. Any leading zeroes (0) are needed. Selecting the button in the serial number field will display all connected devices. The serial number in the field must exactly match the device serial number.

Selecting the test button will display the I/O for the serial number selected.
CAUTION! Changing the state of outputs may cause harm to personal or machinery.

Miscellaneous

Sound

The device reports changes in state of the inputs to the program when they occur. The program accesses the device approximately every 5 seconds to verify the device is connected. If the device does not answer the watchdog flag is set, an entry is made in the event log and a sound is played if configured. To not play a sound, delete the value in the sound name field.

AP functions

See analog functions.

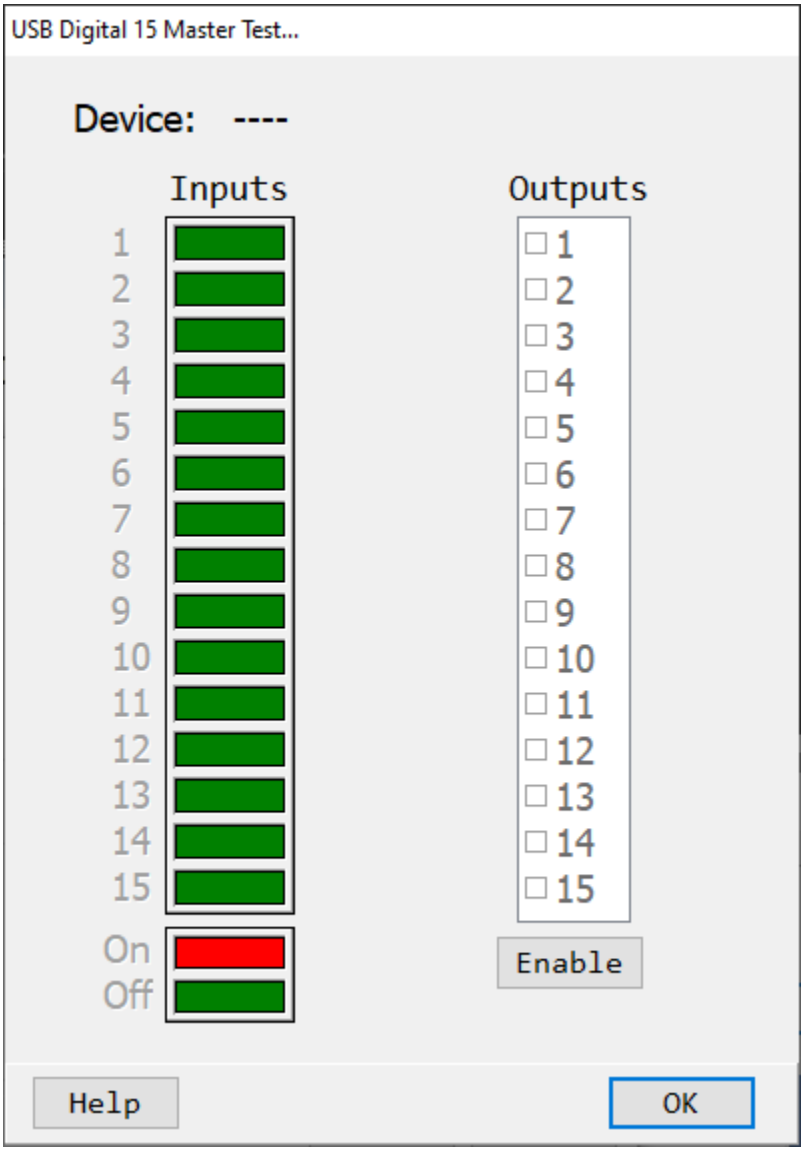
Addressing

I for inputs and Q for outputs.

The device has 15 inputs and 15 outputs.

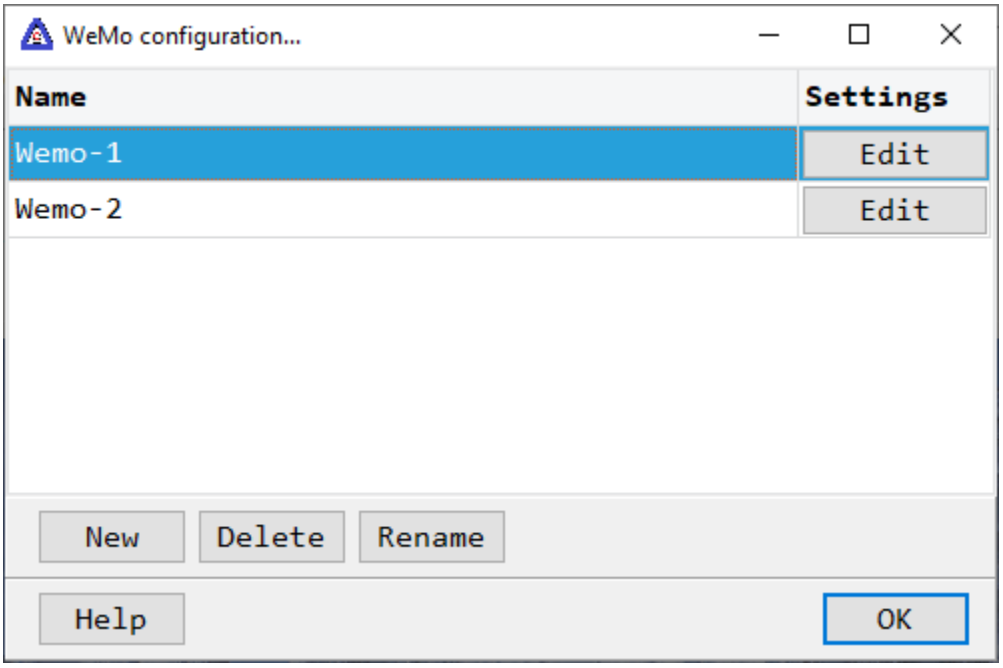
I1 - I15
Q1 - Q15

Testing



WeMo

Each WeMo master object is listed in the window.



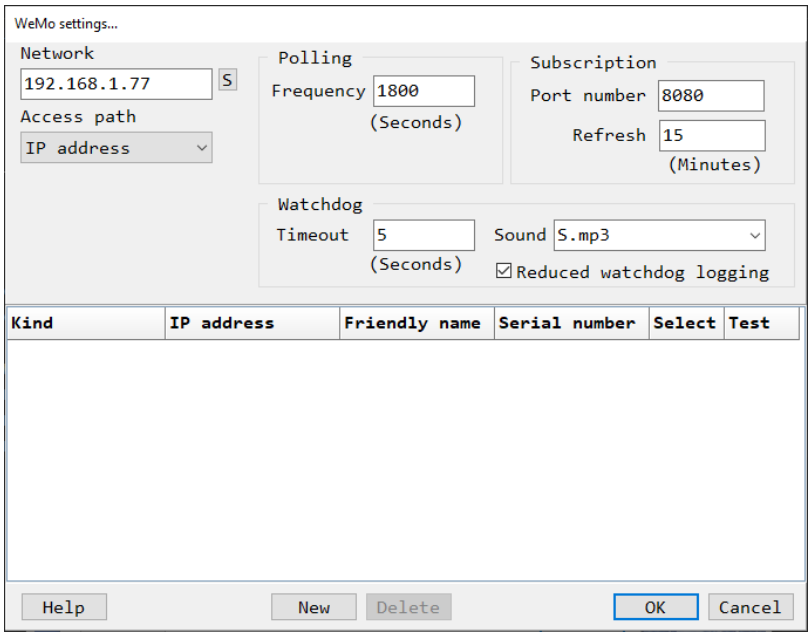
To create a new object select the "New" button and enter a name. Each name must be unique across all tag names.
To rename an object select the "Rename" button and supply a new name.

To delete an object, select the "Delete" button.

Notes:

1) Sometimes WeMo devices will not respond to a “Search” command. This seems to happen when a device has not been accessed in X amount of time. There is no action(s) to perform to make a device respond. To this end, during runtime, when a device does not respond (watchdog timeout or has never responded), the HMI will being issuing a “Search” command every 30 seconds until all configured devices have responded.
2) A WeMo device does not always use the same port number. The “Search” command is the only method to determine the correct port number. When runtime monitoring starts, a search is executed and can delay the “quality” status of a device from becoming “good”, until the device responds.
3) While the HMI provides three access paths only the IP address is used to read/write/verify the device.
4) Call SetPortReadEnable with an index of zero (0) and the port will issue a search command.
5) Call SetPortReadEnable with an index of one (1) and the port will poll all the devices, if the polling process is not in progress.

Settings



Each WeMo port is a collection of WeMo devices residing on the same network.

Network

Modern computers can contain multiple network interfaces. This IP address specifics the network interface device containing the WeMo devices for this port. The selected address can only be used by one WeMo port.

Access path

This property selects which attribute, of the WeMo device, to use as a point source address for point configuration. It is also used to verify connection to the correct device. For example:

1) If the device IP address is dynamic, the address could change on a device reset or loss of power.
2) If the device is replaced, the serial number will be different.
3) If the device is reset, the “friendly name” will be lost.

When runtime monitoring begins, a network search for WeMo devices is performed. The “access path” selection is used, by the program logic, to determine if the correct device was located.

Polling

When the timer expires all configured devices are queried. When the last configured device responds or times out, the timer is reset and when the timer completes the polling logic again executes and the process is repeated. Polling is another method to check for device access at runtime.

Subscription

The WeMo device can send a message to the HMI when the I/O state changes. The WeMo system is designed to report via notifications, not via polling.

Port number:
: Select the TCP port number to use for WeMo subscription messages. (8080 is the default.)

Refresh:
: This property determines the search/subscribe logic execution. (15 minutes is the default.)

Watchdog timeout

This is the amount of time to wait for the device to respond to a query or command before repeating or moving to the next device in the queue.

Sound

If a device does not respond the “watchdog” flag is set and if configured, the sound will play. To not play a sound, delete the value in the sound name field.

Reduced watchdog logging

When enabled and a watchdog timeout occurs, a single entry will be placed in the event log and the sound will be queued (if configured). When the device responds, after a watchdog timeout, a single entry will be placed in the event log.

New button

Select the “New” button to add an un-configured WeMo device to the device list. Manual entry of configuration data is via the gird cells.

Delete button

Select the “Delete” button to delete the selected WeMo device.

Select button

When selected, the search dialog will appear. A “discover” command will be transmitted and the WeMo devices that respond will be queried and listed in the grid. Select a device and then select the “OK” button (or double click a device in the grid) and the device information will be copied to the device configuration list.

Addressing

X for digital inputs, Y for digital outputs and AX for analog inputs.

Device	Inputs (digital)	Outputs (digital)	Inputs (analog)
Switch		Y1	
Insight switch		Y1	AX1-AX10
Light switch		Y1	
Maker	X1 (Sensor)	Y1 (Relay)	

Examples:

Access path is “IP address”
192.168.1.5.Y1, 10.0.0.0.AX1

Access path is “Friendly name”
Front light.Y1, SpaceHeater.AX1

Access path is “Serial number”
224229K0101887.Y1, 421229K0101887.AX7

Insight switch
Maker

Insight switch

Belkin does not publish what the analog data represents for the Insight switch. This is a best guess and could be completely wrong. **Use at your own risk.**

Address	Description (Not verified)	Data type
AX1	Last time output changed state (Unix time stamp)	Signed integer 64
AX2	Last time output was on (in seconds)	Default
AX3	Amount of time output on today (in seconds)	Default
AX4	Amount of time output on in last two weeks (in seconds)	Default
AX5	Timespan (in seconds)	Default
AX6	Average power (watts???)	Default
AX7	Current power usage (milliwatts)	Default
AX8	Energy used today (watts???)	Default
AX9	Energy used total (watts???)	Float
AX10	Unknown; seems to always be 8000--Note 2	Default

Notes:

1) Default data type is 16 bit integer. Select none in the point configuration source type field.
2) AX10 is not returned via the "Subscribe" service.

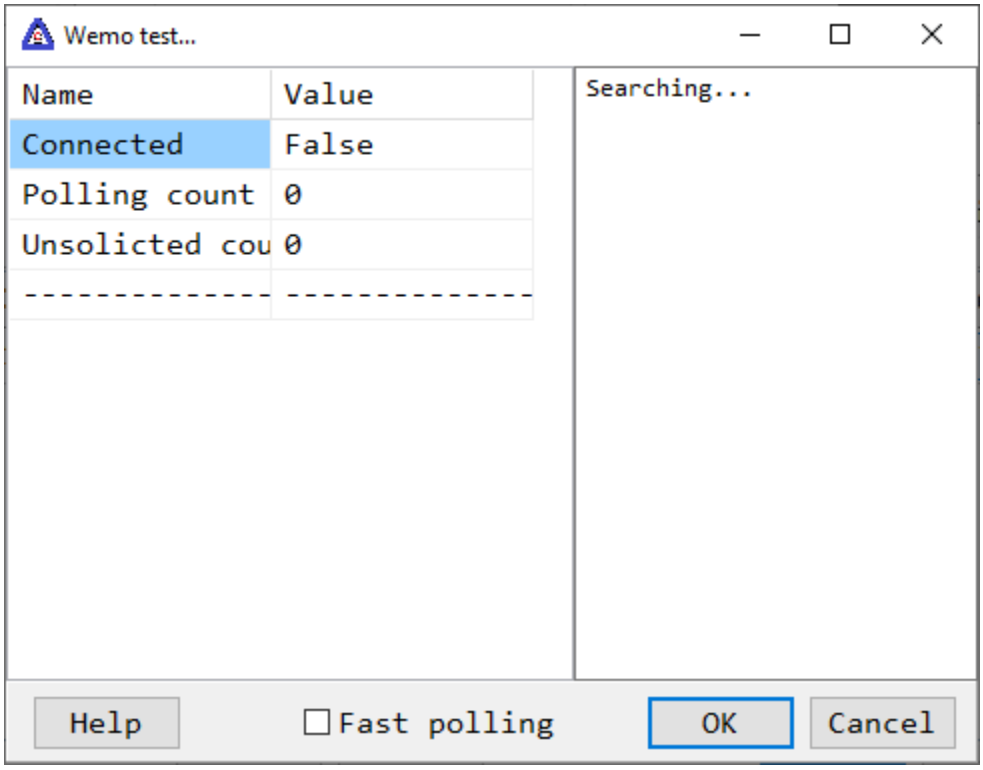
Return to device list

Maker

Note:

1) The sensor input (X1) is active low. On the WeMo application is it named "Sensor triggered" when the sensor input is low (0). The HMI interprets the low or zero (0) as a false. Use the digital point "Invert" property as needed.

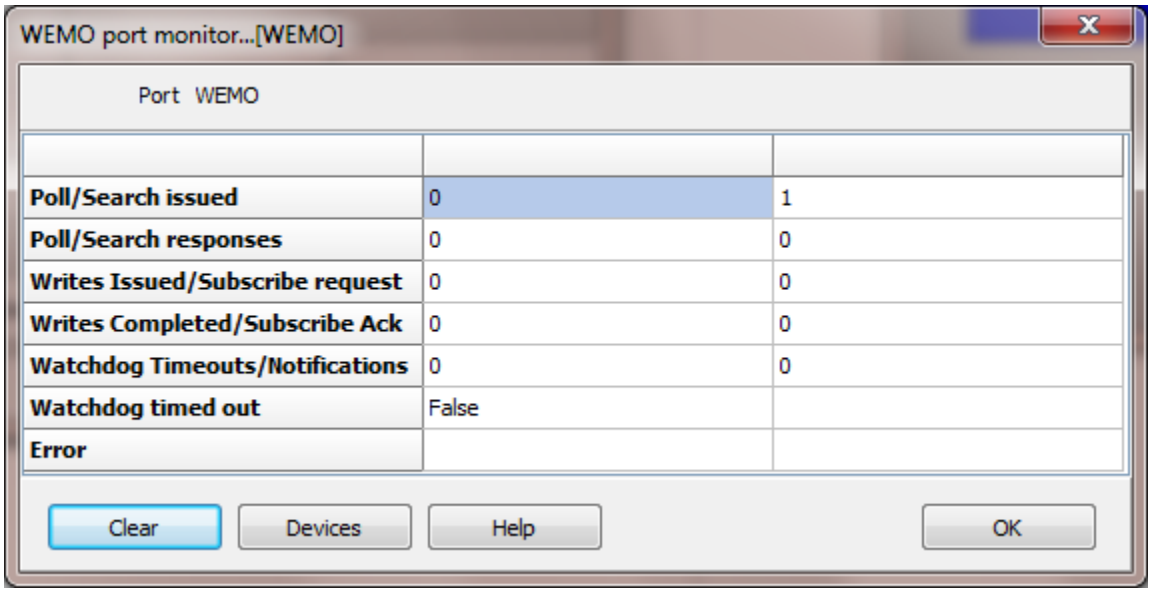
Testing



Fast polling

Normally, subscriptions are enabled and polling is used to periodically verify the device is still responding to queries/events. The testing poll time is set as configured. If fast polling is enabled the testing poll timer is set to one (1) second.

Port monitor



Poll/Search issued

The first column is the “poll” commands issued counter.

The second column is the “search” command issued counter.

Poll/Search responses

The first column is the “poll” responses counter. If all device are responding, this value will be the same as “poll issued”. (Maybe off by one because of timing.)

The second column is the “search” command responses. For each search command all devices should respond. The search command is issued at runtime start (until all devices respond) and if any devices fails to respond to a poll command.

Write/Subscribe request

The first column is the “write” command counter. This only applies to devices that provide an output and a command is initiated by the HMI.

The second column is the “subscribe” command counter. After the device responds to the search command, a command is sent to the device to instruct the device to send “notify” message when a change in the device is detected.

Writes completed/Subscribe Ack

The first column is the “write” completed counter. When a write command is issued to a device, the device responds with an acknowledgement.

The second column is the “Acknowledgement” counter. After the subscribe request is sent to the device, the device will send an “Acknowledgement” response.

Watchdog timeouts/Notify

This counter is the watchdog timeouts for all devices, combined.

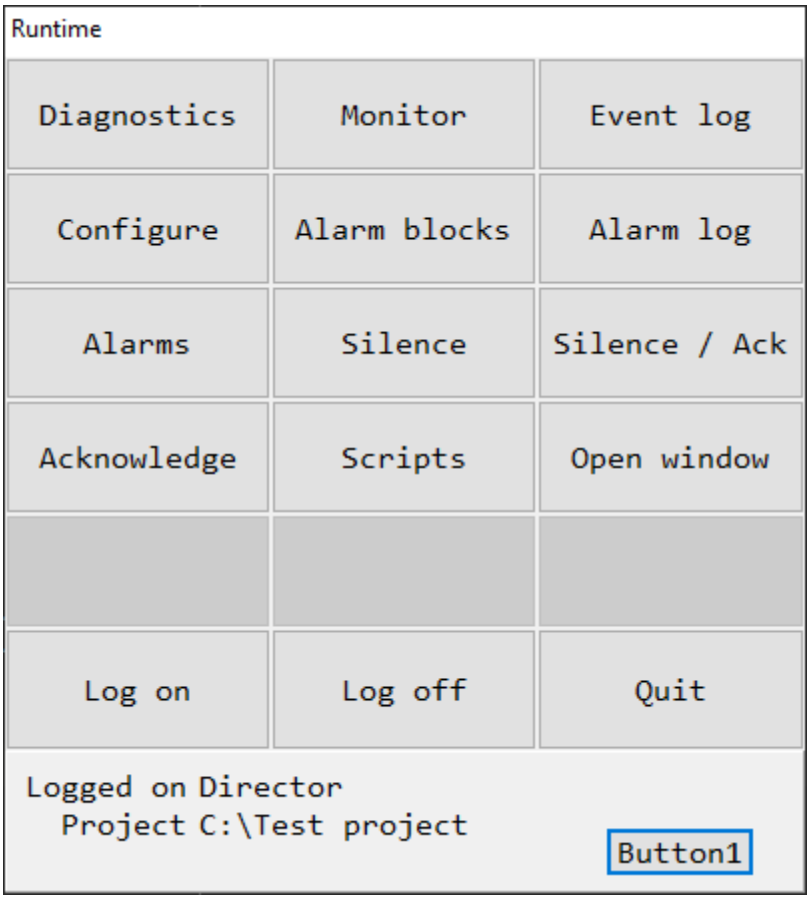
The second column is the “notify” counter, from all device, combined. After the subscribe request is sent to the device, the device will send notify messages for changes in the device.

Watchdog timed out

This is a false/true condition that will indicate if a device did not respond to a poll command. The state is momentary because after the watchdog timeout is detected by the program logic, the logic begins testing the next device.

RUNTIME

PANEL

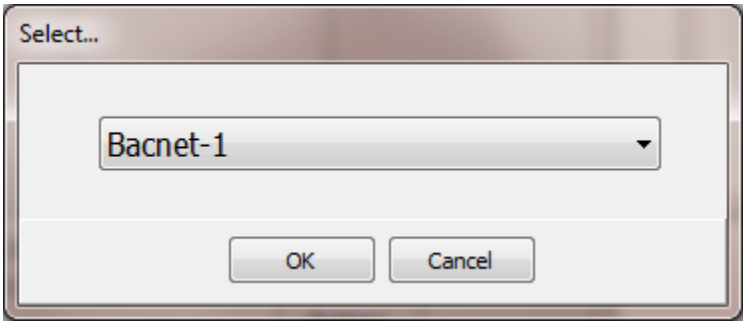


The runtime panel is the main window for the HMI monitoring program. The buttons will be enabled/disabled based on the logged on user and user level.

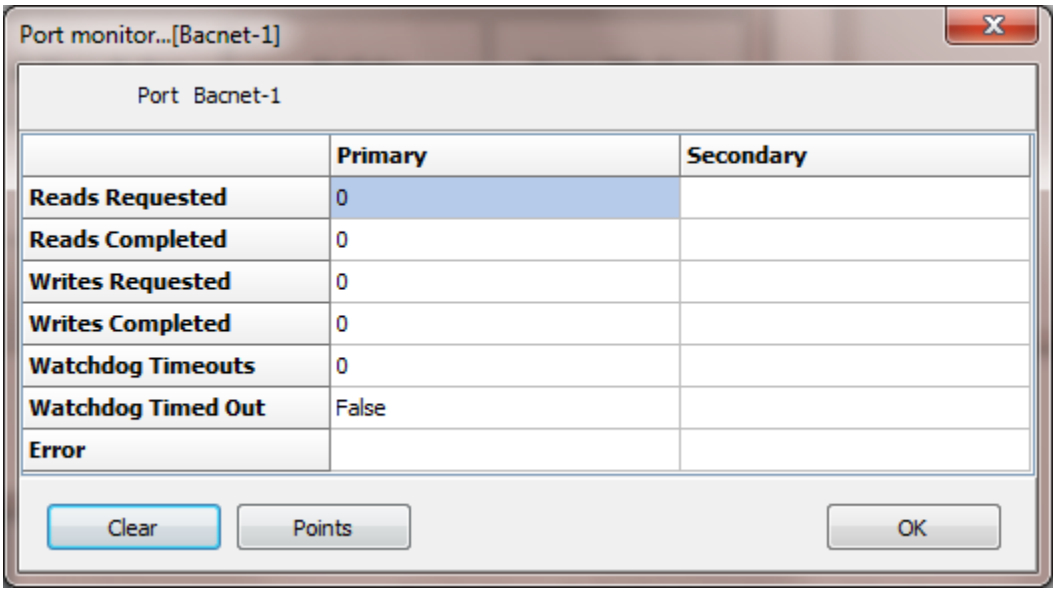
Closing the window will end runtime monitoring and launch the configuration program.

Diagnostics

This button will display a window to allow for the selection of a port.



After the port is selected the port diagnostics window will appear.

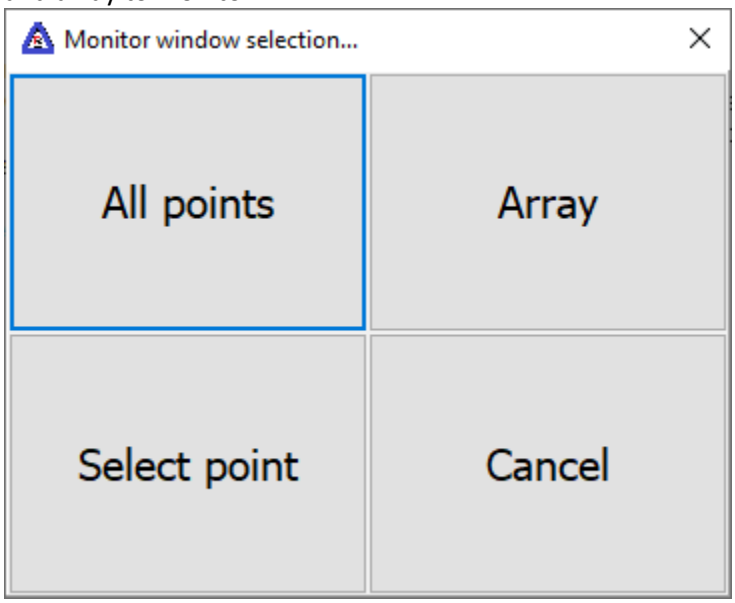


Most of the port diagnostic windows appear like the one above and display the same information for the selected port. Some port types display a different window because the diagnostic data for the port is customized for the port type. Multiple port diagnostic windows can be open concurrently.

Most of the port diagnostic windows are also able to show the raw data collected from the external device.

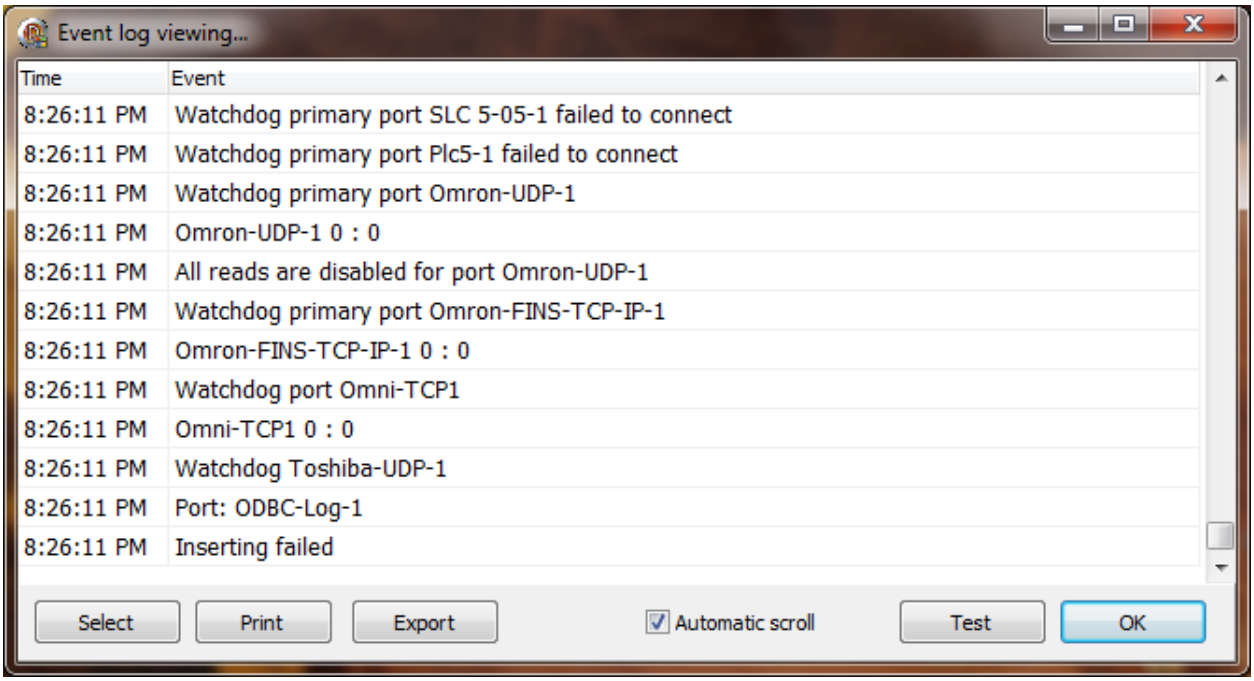
Monitor

This button will display a window for selecting to monitor all points, a point or select and array to monitor.



Event log

This button will display the event log. The event log is a friend. ☺ If something is not working as expected, this is the first place to look for the cause. The HMI logs thousands of possible errors; graphics, scripting, points, connections, free disk space, bad address, invalid format, etc..

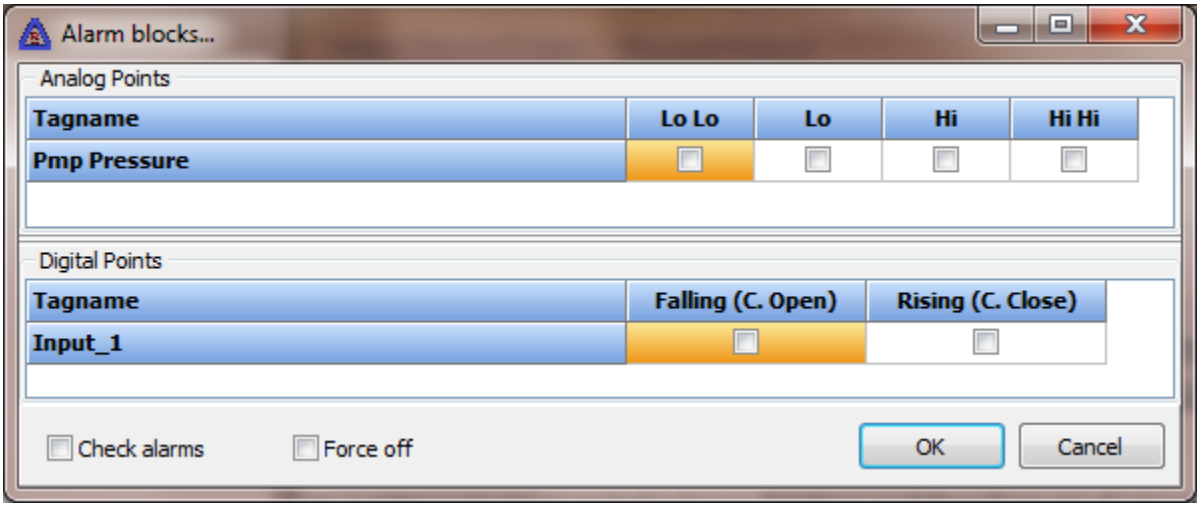


Configure

This button will launch the configuration program. The most common reason to launch the configuration program from runtime is to edit a graphic screen. Changes made to graphic screens are applied when the window is opened at runtime. If the window is open, close it and re-open it to see any changes made and saved in the configuration program.

Note: All most all other changes that can be made in configuration mode will not apply to runtime mode until runtime is stopped and started.

Alarm Blocks



This button provides for changing the alarm blocks. Alarm blocks can also be changed via scripting. The required user level is set on the "Miscellaneous" panel.

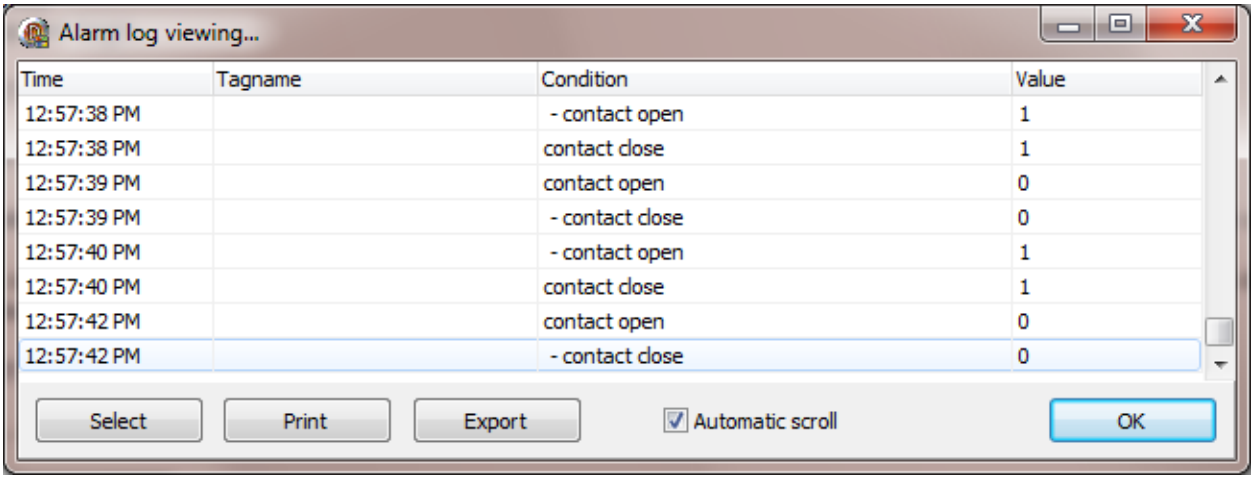
Check alarms

When the OK button is selected the selected block state is set for all points. If the "Check alarms" checkbox is enabled, the point will be processed, checking for a new alarm. This is most useful when removing a block from a point.

Force off

When the OK button is selected the selected block state is set for all points. If the "Force off" checkbox is enabled, the point will be processed. If blocking for the point alarms is enabled and the point alarm is active, the alarm will be cleared.

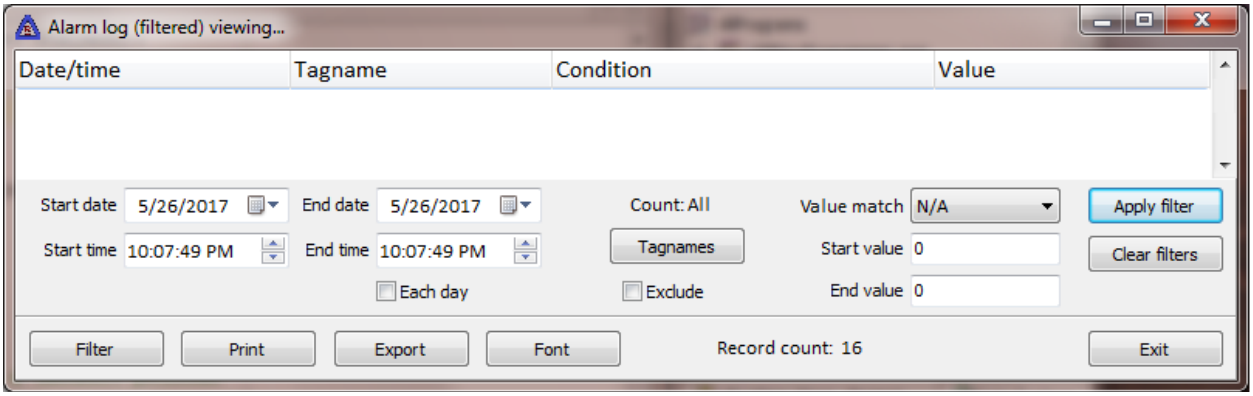
Alarm Log



This button displays the alarm log. These are entries in the disk based alarm log for points that have been configured to log the alarm condition to disk. An entry is made when the alarm condition becomes true and then when the alarm condition returns to false. The “-“ is the first character in the “Return to Normal” indication.

Alarm log (filtered)

This Alarm log (filtered) can be used to view the alarm log for a single day or multiple days and filter the log entries. Also see "Search Logs" dialog.



To display the window use the "OpenAlarmLogFilterWindow" script command. If no filtering is applied, the current day's alarm log is loaded and displayed. Changes to the alarm log while the window is open, are not displayed in the window. The filter settings are saved when the window is closed.

Start date/end date

These properties define the alarm log date range to be displayed/searched.

Start time/end time

These properties define the alarm log time range to be displayed/searched and how the times are applied depends on the value of the next property "Each day". If the start and end time are the same value, the time filter is not applied.

Each day

This property defines how the time values are applied to the filter command.
The examples assume only the time filter is applied.

Example 1
The start and end date are the same date.
The start time is 10:00:00 and the end time is 10:15:00.
When the filter is applied only the alarms logged between the configured times are displayed.

Example 2
The start date is May 1, 2017 and end date is May 4, 2017.
The start time is 10:00:00 and the end time is 10:15:00.
The "Each day" property is false.
When the filter is applied only the alarms logged between
May 1, 2017 10:00:00 through May 4, 2017 10:15:00 are displayed.

Example 3
The start date is May 1, 2017 and end date is May 4, 2017.
The start time is 10:00:00 and the end time is 10:15:00.
The "Each day" property is true.
When the filter is applied only the alarms logged:
May 1, 2017 10:00:00 through 10:15:00 and
May 2, 2017 10:00:00 through 10:15:00 and
May 3, 2017 10:00:00 through 10:15:00 and
May 4, 2017 10:00:00 through 10:15:00 are displayed.

Tagnames

This property defines if the a tagname filter should be applied to the alarm logs.

Exclude

This property defines how the tagname filter will be applied.
When this property is false, only the tagnames selected will pass the filter and be displayed.
When this property is true, the selected tagnames will be "excluded" (filtered out) and not displayed.

Value match

This property defines a filter to be applied to the "value" field.

Value	Description	Start value	End value
N/A	No filter applied		
Less than	Value less than start value	Required	
Equal to	Value equal to start value	Required	
Greater than	Value greater than start value	Required	
Not equal to	Value not equal to start value	Required	
Between	Value between start value and end value	Required	Required

Start/end value

These properties defines the end points for the "value match" filter.

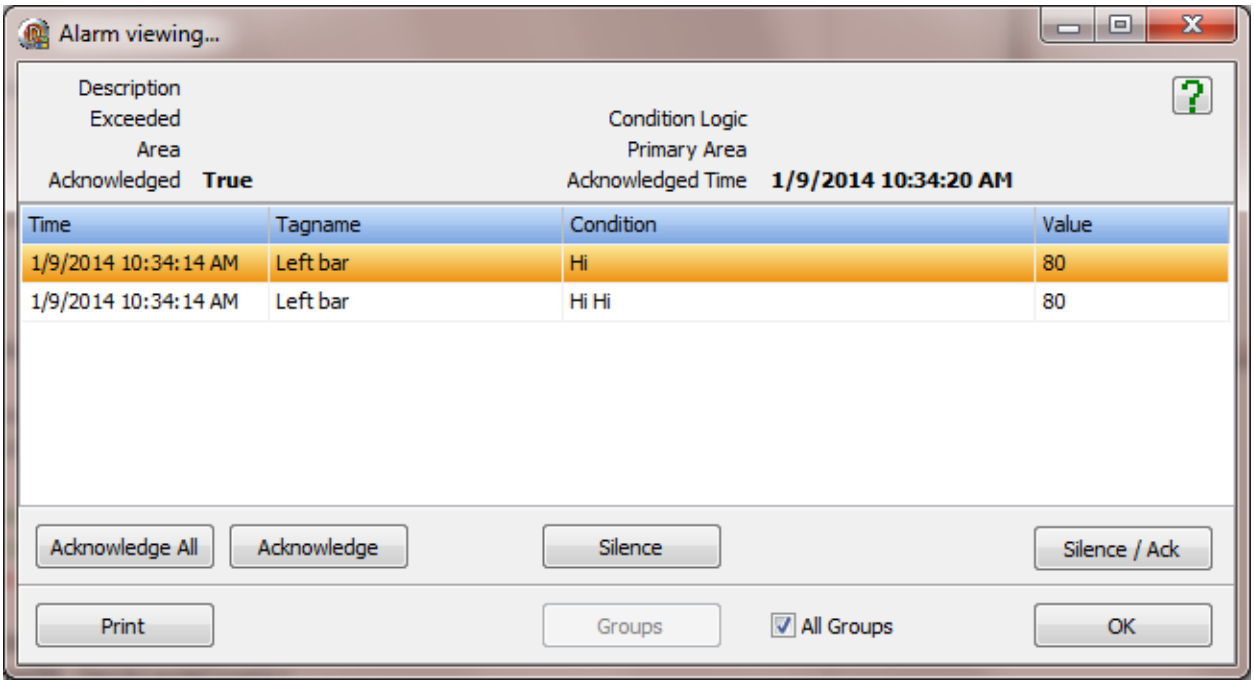
Apply filter

After setting the filter properties, select the “Apply filter” button and the configured alarm logs will be load, filtered and displayed in the grid.

Clear filters

This button sets the date/time properties to the current date and clears all other filters.

Alarms

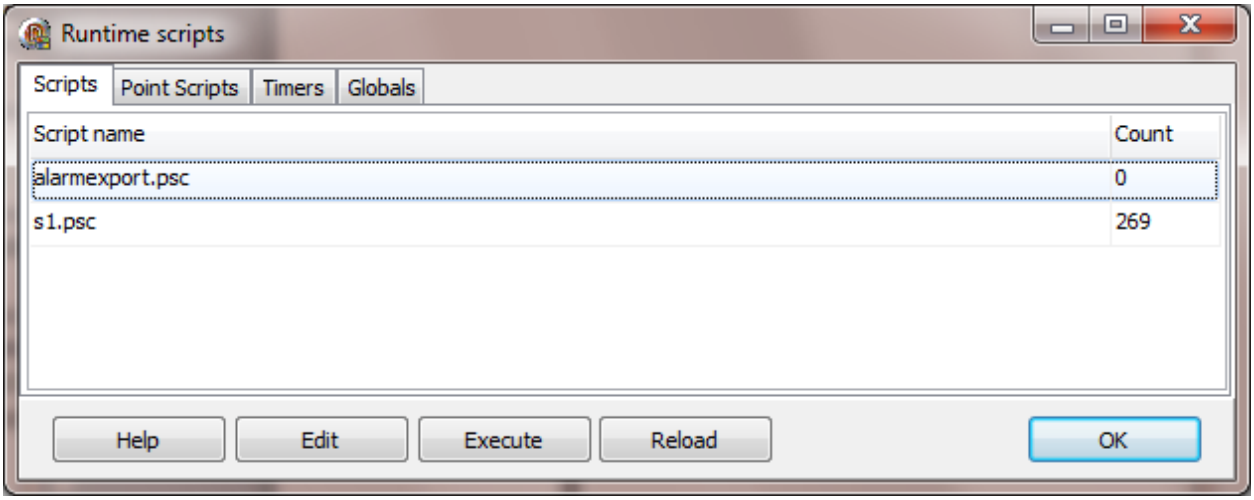


This button displays the active alarms window. These are alarms that are currently active.

Silence
Silence/Ack
Acknowledge (selected alarm)
Acknowledge All (all alarms)

These buttons execute script commands. (Follow the hyperlink)

Scripts



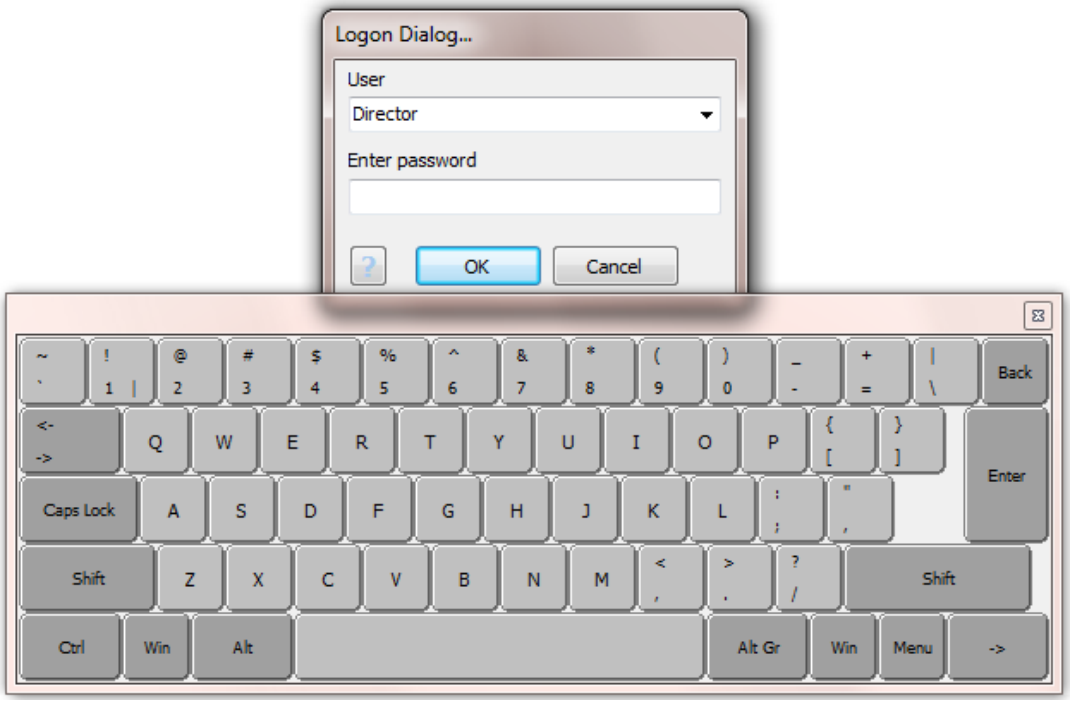
This button displays the runtime scripts window.

Open Window

This button executes the script "OpenWindowUserSelect" command.

Log On

This button displays the “Log On” window with or without the virtual keyboard.



A user can log on by selecting a name in the drop down list, enter the correct password and select "OK". If the password is not correctly entered for the user name selected, the beep sound will play.

New installation

A project is a collection of files in a directory. The project name is the path to the file collection.

When the program is first run a directory selection dialog is displayed and the user is asked to select the project directory. A button to create a new directory is located in the bottom left corner. If the user creates a new directory (project path) and selects it a user named "Director" is created and the password is blank (no password).

The user "Director" is always present and has the highest level of configuration. The password can be changed via Users.

Log Off

This button logs off the current user.

Quit

This button ends runtime monitoring.

There are many other windows that can appear at runtime. For example, the windows that appear to allow for user input when a mouse or script command is called, the different “read” monitor windows, unique windows for port monitoring, etc. As always, if questions or need help, contact support.

OVERRIDE FILE

The HMI runtime program uses the project configuration to define the monitoring operations. At times a project may be used in more than one application.

Example 1
A project to monitor a pumping station has a TCP/IP MODBUS master. The IP address is 192.168.1.1. Another pumping station has the same configuration except for the IP address. The IP address is 192.168.1.2.

Example 2
One application has three boilers. Each boiler is accessed via serial MODBUS master. Another location only has two boilers. Disabling one MODBUS master is needed.

Example 3
A project to monitor a wellhead is using a serial MODBUS master. The serial port on the installed computer uses com 2. A notebook computer sometimes used to connect only has a com 1.

A method to override some of the configuration attributes is provided in the "runtime override" feature.

When launching the Configure.exe program a file can be passed in a launch parameter to be used when runtime monitoring starts.

To accomplish, a text file needs to be created. It uses the "ini" file format described below.

Create a shortcut to the Configure.exe. Right click on the icon and select "properties".

In the target field will be the path to the program. "C:\Program Files\ <program folder>\Configure.exe". The path might be different.

Add the path to the file created to the target. Example: "C:\Program Files\ <program folder>\Configure.exe" "C:\HMIOverride.ini"

Include the quotes. Use a double quote. Do not use two single quotes.

To use a different override file change the path.

Using the override feature one project configuration is used many times with changes made in the override file.

The file is made up of sections. The sections can be in any order. Only the sections needed are required. If a section is not needed it does not need to be present. Not all fields of a section need to be used. Only those fields needed must be present. To enter comments into the file place a semi-colon (;) as the first character on the line.

Format

Each section starts with [<section name>] followed by the section fields. All of the lines after the section declaration until the next section declaration or the end of file belong to the section declared above.

The fields of a section are comprised of a field name, an equal sign and the field value.

Example:

Field name	**Sign**	**Value**
primarySlaveAddress	=	1

If a value is not assigned to a field name then the field is not needed and can be removed.

Note: The filed names and section names are case sensitive. The assigned section names must be used if the section is needed. The variable field names must exactly match the names used in the project configuration.

Sections

[Port Section]

This section is used to override port configurations. Each port to have a configuration attribute overridden must have a separate section. Not all possible field names apply to all port types.

The section name is the port tagname. Example [Turbine_A] or [PUMP101].

All of the fields have a configuration item with a similar name. Refer to the help for the port type for descriptions.

These are the possible field names for a port section.

disable= 1 to disable 0 is no action

primarySlaveAddress=
secondarySlaveAddress=
primarySourceAddress=
secondarySourceAddress=
readDelayTime=
watchdogTime=
watchdogSound=
primaryHostname=
primaryIPAddress=

primaryPortNumber=	
secondaryHostname=	
secondaryIPAddress=	
secondaryPortNumber=	
primaryComPort=	
primaryBaudRate=	
primaryParity=	
primaryDataBits=	
primaryStopBits=	
secondaryComPort=	
secondaryBaudRate=	
secondaryParity=	
secondaryDataBits=	
secondaryStopBits=	
readDisabled1= 1 to disable 0 is no action	
...	
readDisabled32=	
primarySourceAddress=	DF1
primaryDestinationAddress=	DF1
primaryChecksum=	DF1 CRC or BCC
secondarySourceAddress=	DF1
secondaryDestinationAddress=	DF1
secondaryChecksum=	DF1 CRC or BCC
masterID=	
channelTimeout=	
primaryIsBinary=	0 or 1
primaryBindIPAddress=	
primaryNode=	
secondaryIsBinary=	0 or 1
secondaryBindIPAddress=	
secondaryNode=	

[WINDOW_DISABLE]

The field names of the section are the names of the windows to be disabled.

Example:

Pump3StartStop=1	1 to disable 0 is no action
Level7=1	

[PROJECT_SETTINGS]

loggedOnName=
soundDelayAmount=
homeScreenName=
onStartRTScript=

onDuringRTScript=

[POINT_INITIAL_VALUE]

This section is used to set the default item for host tags. For analog 5000, for digital 5007. For host pointer tags the item number is 5009. If an external tag is referenced it ONLY sets the default item to the value.

tagname1=56.7
tagname2=1
tagname3=48

[POINT_PORT]

This section is used to set the port for a point.

tagname1=Ethernet-1
tagname2=Serial-1
tagname3=SomePortName

DDE SERVER

The DDE server provides functions to read/write points and script global items.

DDE configuration is via a file in the project directory. The filename is “DDESettings.ini” and if the file is found the DDE server is enabled.

The file format is based on sections and name=value pairs, the “INI” (initialization) file format. Each section has name=value pairs.

Note: DDE is an older technology and we only tested with Excel. If assistance is needed, contact support.

[Common]

update=2000

DDE items are linked to points and script global and must be updated. Normally the point will update any linked object but, not DDE items.

The value is the number of milliseconds between update checks. An update check compares the last update value to the current value and if the values are different the DDE item is updated.

This property is optional and if not present the update rate will be 2000.

Use caution when altering this value. Too low a value can result in HMI instability.

logPokes=1

The DDE server can accept DDE “Pokes”. If a point access rights is configured for read/write or write, or a poke to a script global is initiated and this value is “1”, the poke command will be logged to the event log.

If this value is “0” pokes will not be logged. This property is optional and if not present poke logging will not occur.

[TagItems]

<dde item name>=<point tagname.item number>

This section defines the points to be monitored by DDE items. The DDE topic name for this section is “Points”.

Each name=value pair defines the DDE item name “<DDE item name>” and the point.item number “<point tagname.item number>”.

Examples:

Pmp1Pressure=Pmp1Press.5000
Pmp1Running=Pmp1Press.5007

Each DDE item name must be unique across all item names.

Note: For Excel we had difficulties with “Name” errors if the DDE item name ended with a digit.

[ScriptGlobals]

<dde item name>=<script global section.item>

This section defines the script globals to be monitored by DDE items. The DDE topic name for this section is “SG”.

Each name=value pair defines the DDE item name “<DDE item name>” and the script global section.item “<script global section.item >”.

Examples:

LoggedOnUser=User.name
IPAddress=Comp.address

Each DDE item name must be unique across all item names.

Note: For Excel we had difficulties with “Name” errors if the DDE item name ended with a digit.

DDE Client

The DDE client provides functions to read/write host points.

DDE configuration is via a file in the project directory. The filename is "DDEClientSettings.ini" and if the file is found, the DDE client will be enabled.

The file format is based on sections and name=value pairs, the "INI" (initialization) file format. Each section has name=value pairs.

Note: DDE is an older technology and we only tested with Excel. If assistance is needed, contact support.

[Common]

update=2000

DDE items are linked to points and must be updated. Normally the point will update any linked object but, not DDE items.

The value is the number of milliseconds between update checks. An update check compares the last update value to the current value and if the values are different the DDE item value is sent to the server .

This property is optional and if not present the update rate will be 2000.

Use caution when altering this value. Too low a value can result in HMI instability.

logPokes=1

The DDE client can "poke" a server item. If a point value changes and a "poke" is executed and the "logPokes" value is "1", the poke command will be logged to the event log.

If this value is "0" pokes will not be logged. This property is optional and if not present poke logging will not occur.

[<service name>~<topic name>]

The name of the service and topic. Normally the program (exe) name without the ".exe" extension and the topic name. e.g [Excel~Sheet1]

Multiple server connections are supported and each sever connection must have a unique <section name>~<topic name>.

Each item in a section is the <DDE item name>=<point tagname.item number>

Examples:

Pmp1Pressure=Pmp1Press.5000
Pmp1Running=Pmp1Running.5007

Each DDE item name must be unique across all item names for the <service>~<topic name>.

Notes:

1) For Excel, we had difficulties with "Name" errors if the DDE item name ended with a digit.
2) For Excel, the item name is the cell coordinates as rowXcolY or the item name can be the name of the cell. Example R1C1 (Row 1, column 1) or SomeCellName

OPC Server

The OPC server, in the runtime program, is active if any point has the "OPC Published" property enabled.

The OPC server is not registered with the OS when the HMI installer is executed. In the <installation directory>/Tools, are two files.

Right click the mouse on the icon for **RegisterOPCServer.bat** and select "Run as administrator". Select "y" and press the "Enter" key. The HMI server will be **registered** with the OS.

Right click the mouse on the icon for **UnRegisterOPCServer.bat** and select "Run as administrator". Select "y" and press the "Enter" key. The HMI server will be **unregistered** with the OS.