FactoryCast HMI 1.7

Setup Manual
Premium and Quantum HMI Modules

12/2011

35007632.05

www.schneider-electric.com

Schneider Electric

Table of Contents

Safety Information

Important Information

NOTICE

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

⚡ The addition of this symbol to a Danger safety label indicates that an electrical hazard exists, which will result in personal injury if the instructions are not followed.

⚠ This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

⚠ DANGER
DANGER indicates an imminently hazardous situation which, if not avoided, **will result in** death or serious injury.

⚠ WARNING
WARNING indicates a potentially hazardous situation which, if not avoided, **can result in** death or serious injury.

⚠ CAUTION

CAUTION indicates a potentially hazardous situation which, if not avoided, **can result in** minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

PLEASE NOTE

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

About the Book

At a Glance

Document Scope

This document describes the implementation procedure for FactoryCast HMI, the configuration of the Premium TSX WMY 100 module under PL7 and Unity Pro, and the configuration of the Quantum 140 NWM 100 00 module under Concept and Unity Pro.

Validity Note

The characteristics presented in this manual should be the same as those that appear online. In line with our policy of constant improvement we may revise content over time to improve clarity and accuracy. In the event that you see a difference between the manual and online information, use the online information as your reference.

Related Documents

Title of Documentation	**Reference Number**
Communication Applications Setup Manual, Volume 3	TLX DS COM PL7 •• E
Web Designer for FactoryCast HMI User Manual	35016143

You can download these technical publications and other technical information from our website at www.schneider-electric.com.

Product Related Information

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

No part of this document may be reproduced in any form or by any means, electronic or mechanical, including photocopying, without express written permission of Schneider Electric.

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Keep strict access to the embedded server by configuring passwords.
- Carefully select the symbols and direct addresses you authorize to be modified online.
- Do not authorize online modification of variables of critical nature concerning human and material safety.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

⚠ WARNING

AVOIDANCE OF HAZARDS

This product must only be used for information exchange. It is not a safety product.

- Do not rely on this product to alert personnel to situations that could involve imminent hazards to personnel, equipment or material. Independent safety alarms must be provided.
- Do not rely on this product or the information conveyed to initiate corrective action. Procedures must be in place to control the plant and processes in accordance with industry practices and relevant regulatory directives.
- Do not allow users to change critical or safety related process parameters or variables.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

User Comments

We welcome your comments about this document. You can reach us by e-mail at techcomm@schneider-electric.com.

Configuration of FactoryCast HMI

I

Subject of this Part

This part describes how to configure FactoryCast HMI modules.

What's in this Part?

This part contains the following chapters:

Introduction to FactoryCast HMI

1

Introduction to FactoryCast HMI

Presentation

FactoryCast HMI gives you the best in Web technology combined with the well-known reliability of PLCs.

The FactoryCast HMI package contains:

- a FactoryCast HMI module that runs the HMI functions and services:
  - **TSX WMY 100** for Premium PLCs,
  - **140 NWM 100 00** for Quantum PLCs.
- Web Designer: a Windows-based software that can be used to create, test and manage FactoryCast HMI applications.

FactoryCast HMI Built-In Functions

The following FactoryCast HMI functions are built into a module:

- Embedded Web server with predefined setup, diagnostic and monitoring pages and hosting of user defined Web pages and documents (pdf, doc, etc.).
- SOAP/XML communication server *(see page 501)* offering interoperability and vertical integration in your information management systems.

FactoryCast HMI Built-In Services

The following FactoryCast HMI services are built into a module:

- **Device** *(see page 23)*: used to declare and define PLC parameters, and to create and configure FactoryCast HMI variables.
- **Database connection** *(see page 42)*: used to archive FCHMI variables in an external database (SQL Server, Oracle, etc.).
- **Email** *(see page 28)*: used to send an email when the status of a variable changes, a threshold is exceeded, an alarm is triggered, etc.
- **Calculation** *(see page 36)*: used to perform calculations using FCHMI variables without employing CPU resources on the PLC.
- **Active pages** *(see page 58)*: used to create HTML pages using FCHMI variables without employing CPU resources on the PLC.
- **Datalogging** *(see page 50)*: save information such as configuration, variable’s values, log files into the internal memory of the module.
- **Recipe** *(see page 64)*: used to change the values of a set of variables when a predefined event occurs.
- **Recipe Database** *(see page 71)*: used to update values of a set of variables with values contained in a database table when a predefined event occurs.

Configuration of FactoryCast HMI

2

Subject of this Chapter

This chapter describes the procedure to be followed to configure FactoryCast HMI services.

It describes all the built-in services:

- Device
- Email
- Calculation
- Active Pages
- Datalogging
- Recipe
- Recipe Database

What’s in this Chapter?

This chapter contains the following sections:

2.1 Description of Services

Subject of this Section

This section provides you with a description of how to configure the different services available with FactoryCast HMI.

NOTE:

- The services are configured using Web Designer for FactoryCast HMI. Before creating a new service, you must create a Web Designer project (refer to the Web Designer for FactoryCast HMI User Manual).
- The maximum number of services for:
  - a 140 NWM 100 00 module is 6.
  - a TSX WMY 100 module is 7.

What’s in this Section?

This section contains the following topics:

Device Service Configuration

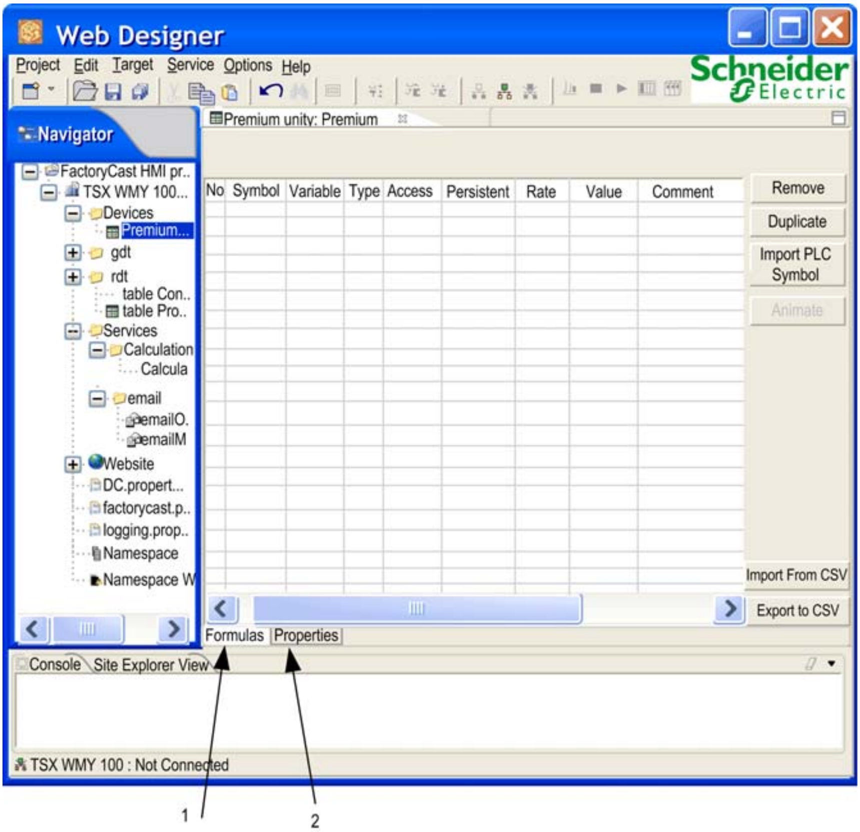
Presentation

The Device service enables you to create symbols associated with PLC variables. These variables are associated with symbols that will be used by other services (email, calculation, database,...).

The variables come from the internal RAM of either:

- the CPU module located on the same rack as the NWM and WMY modules. or
- CPU modules (4 other modules possible) using Xway addressing (WMY only).

The following figures shows you the properties configuration screen for the device service:



1 Use this button to configure the service variables.

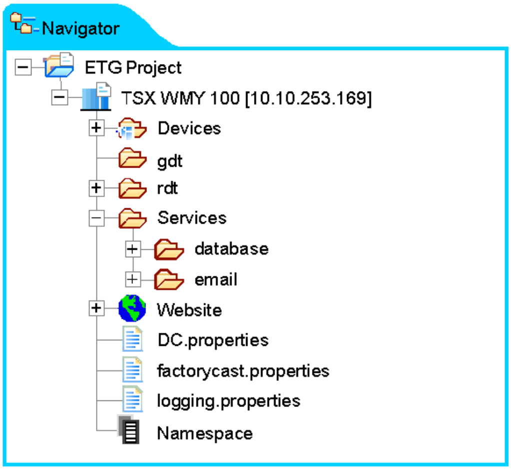
2 Use this button to configure the service properties.

Creating a Device Service

To create a new device service:

- double click the name of your device in the browser, or
- right click the name of your device in the browser, then click **Edit**.

The following figure shows you the Web Designer browser:

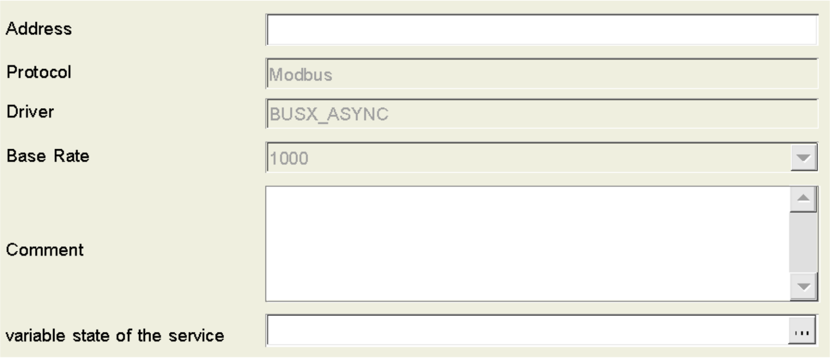


Navigator

- ETG Project
  - TSX WMY 100 [10.10.253.169]
    - Devices
    - gdt
    - rdt
    - Services
      - database
      - email
    - Website
    - DC.properties
    - factorycast.properties
    - logging.properties
    - Namespace

Configuring the Properties

The following figure shows you the properties configuration screen for the device service:



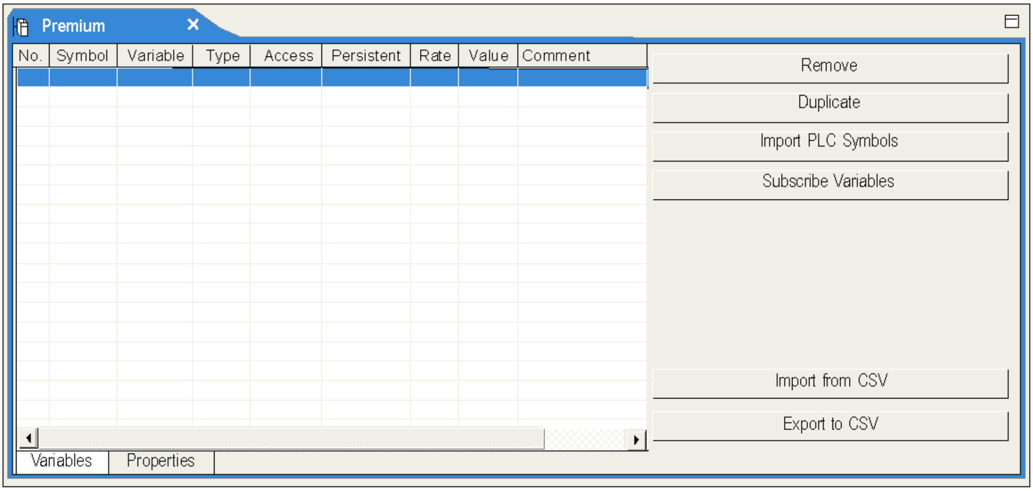
Address	
Protocol	Modbus
Driver	BUSX_ASYNC
Base Rate	1000
Comment	
variable state of the service	

This table describes the fields that make up the properties configuration screen:

Fields	Function
Address	If the module is located on the same rack as the CPU, the address is 'localhost'. If you use a Premium platform, you can specify the Xway address of the device that you want to reach.
Protocol	The protocol is determined by the project type (shaded), i.e. modbus (for Quantum PLCs), or UNI-TE (for Premium PLCs), or UMAS (for Premium/Quantum Unity).
Driver	The driver is determined by the project type (shaded), i.e. either BUSX_ASYNC for Premium modules, or BUSX for Quantum modules.
Base rate	Used to define the default frequency at which variables are read.
Comment	Used to add a comment to the device service properties. This comment is only visible in this screen.
Service status variable	Used to determine the status of the device service.

Configuring Variables

This table enables you to import variables coming from a PLC:



This table describes the various fields that make up the variables configuration screen:

Fields	Function
Input fields	
No	ID of the variable, this number is generated automatically.
Symbol	Name of the variable stored in the module.
Variable	Physical address of the variable.
Type	Used to define the type of variable. The following types may be used: ● BOOL ● EBOOL ● SHORT ● USHORT ● INT ● UINT ● DINT ● REAL ● TIME ● DATE ● TOD ● STR
Access	Access right: Read (R) or Read/Write (R/W).
Persistent	Variable polled in the local image of the module.
Rate	Variable scan frequency (in ms).
Value	Variable value.
Comment	Used to add a comment to the device service properties. This comment is only visible in this screen.
Buttons	
Remove	Delete the selected variable from the table.
Duplicate	Duplicate the selected variable.
Import PLC symbols	Use this button to import PLC symbols from a FEF (PL7), STU (Unity Pro), XVM (Unity Pro) or PRJ (Concept) file. **NOTE**: Concept projects (*.prj*) can only be imported if the ExportVariables parameter in *concept.ini* is set to 1. You must modify this parameter before saving the project under Concept (re-start Concept after modifying the parameter setting).
Subscribe Variables	Use this button to animate variables when the project is in online mode, with connections either to the simulator or to the module.
Import from CSV	Use this button to import symbols from a *.csv* file. This file may be created using Excel or a text editor (in this case, change the *.txt* file extension to *.csv*). The following syntax must be used: *symbol;variable;type;period;comment*
Export to CSV	Use this button to export the variable table for the current service to a *.csv* file.

NOTE: For more information about variable management, refer to the Variable Management chapter of the *Web Designer for FactoryCast HMI User Manual*.

Limitation

The number of variables stored in the module is limited to 1,000 for all device services.

The maximum number of Device service is:
- 1 with a NWM module,
- 5 with a WMY module.

Email Service

Presentation

Use the Email service to send an email if:

- an alarm is triggered,
- there is a change in the status of a variable,
- a threshold is overrun.

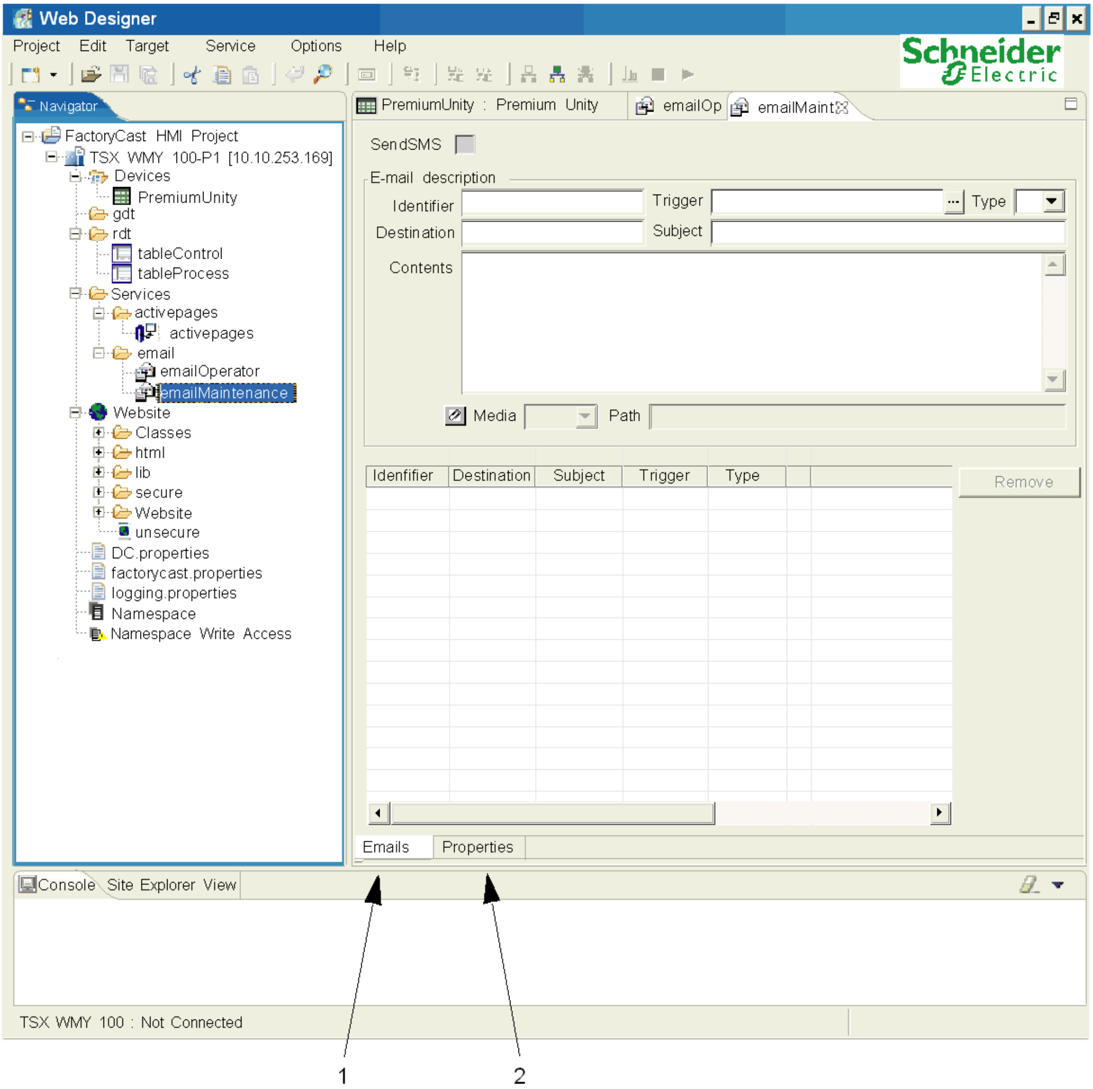
You can associate Email service with variables coming from your Namespace or the calculation service.

The email service uses methods and technologies that are not real-time, and which in certain cases, are not able to guarantee proper reception of information by the recipient.

NOTE: Schneider Electric does not guarantee:

- integrity of data sent by the module with the email service,
- the exact time at which the message has been sent.

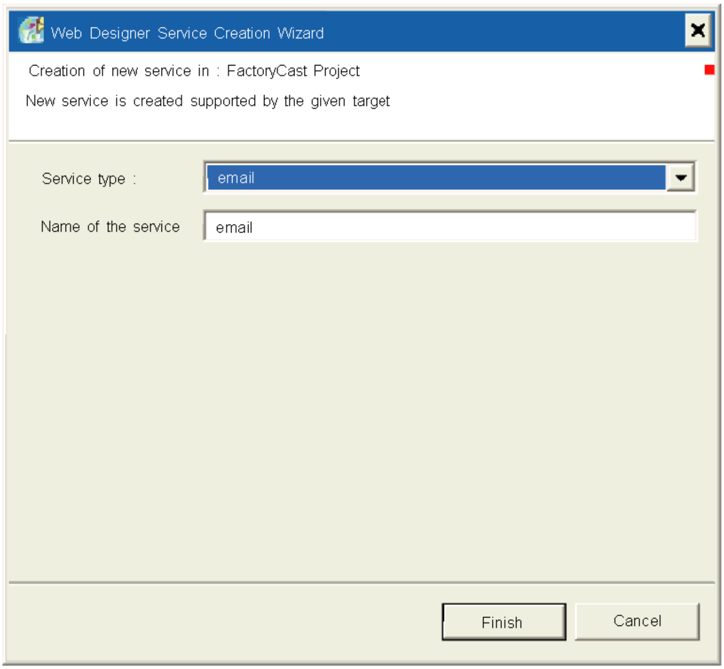
The following figure shows you the configuration properties screen for Email service:



1 Use this button to configure Emails sent by the email service.
2 Use this button to configure Email properties.

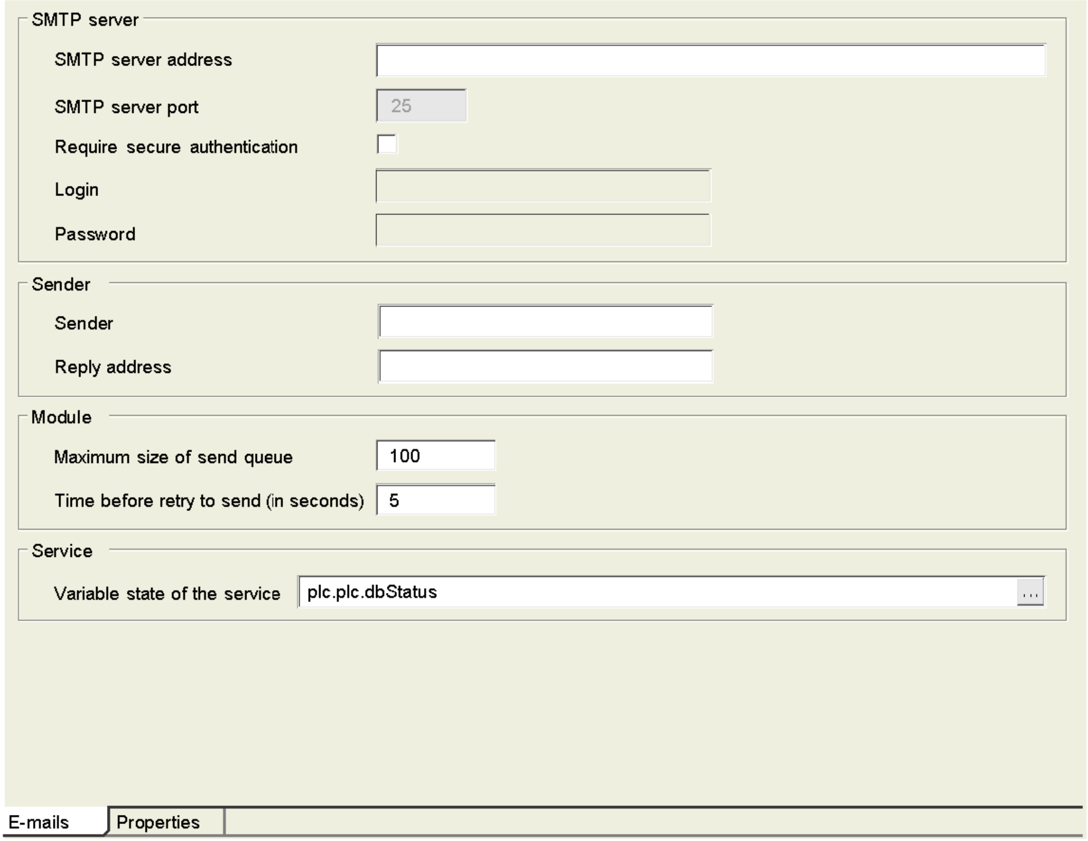
Creating an Email Service

The table below describes how to create an Email service:

Step	Action
1	Click the name of your project in the browser.
2	To add the new service you can: ● Click **Project** → **New** → **Service** or, ● Right-click **Services** in the browser, then click **New Service**. **Result**: the following window appears.  Web Designer Service Creation Wizard Creation of new service in : FactoryCast Project New service is created supported by the given target Service type : email Name of the service email
3	Select **Email** in the **Service Type** list box.
4	Enter a service name or click on **Finish** to keep the name by default.
5	The Email service has been created.

Configuring the Server

The properties configuration screen for the Email service is shown below:



SMTP server

SMTP server address

SMTP server port: 25

Require secure authentication □

Login

Password

Sender

Sender

Reply address

Module

Maximum size of send queue: 100

Time before retry to send (in seconds): 5

Service

Variable state of the service: plc.plc.dbStatus

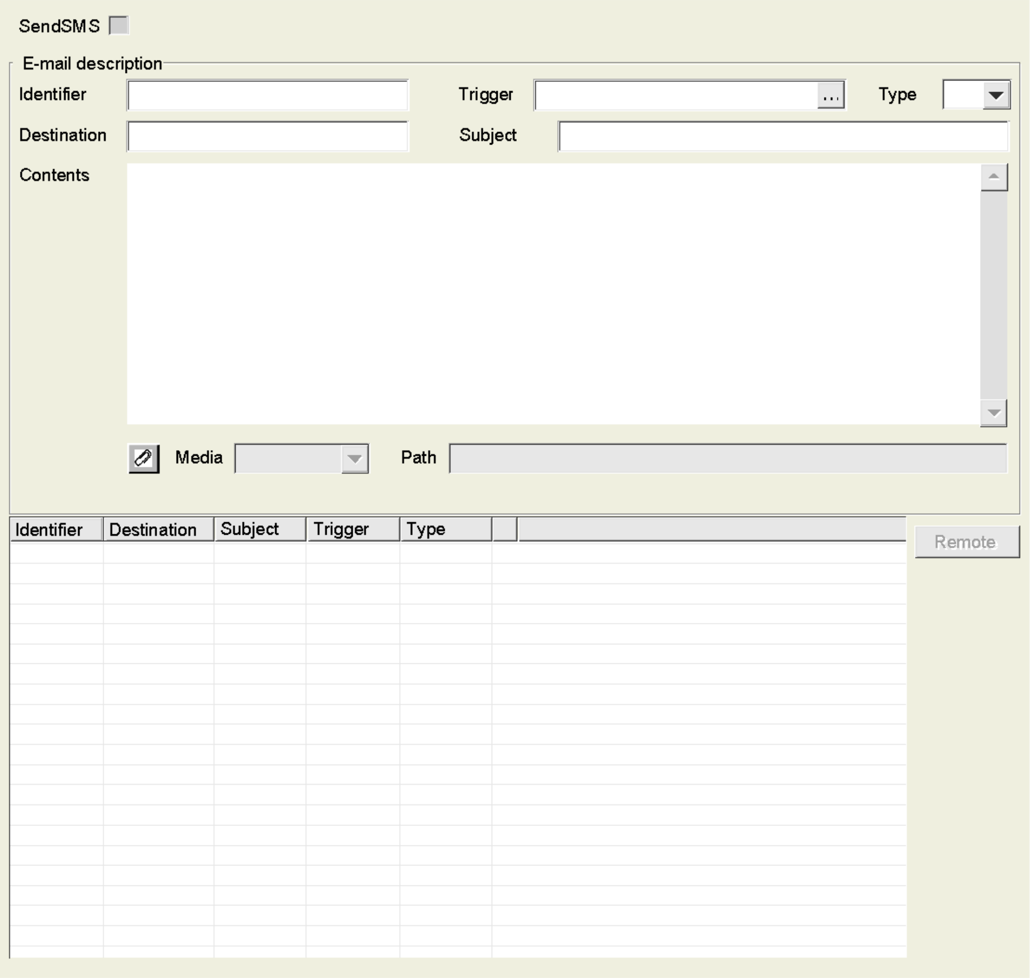
The fields in the properties configuration screen are:

Fields	**Function**
SMTP server	
SMTP server address	SMTP server address.
SMTP server port	TCP port used by the SMTP server (generally port 25).
Require Secure Authentication	Select this box if authentication is needed to access the SMTP server.
Login	Login to access the SMTP server.
Password	Password to access the SMTP server.
Sender	
Sender	Email address of the sender of the message.
Reply address	Email address to which a reply will be sent if the user clicks **Reply**.
Module	
Maximum size of the send queue	Maximum number of emails that can be stored in the buffer memory before being sent: ● Default value = 100, ● Minimum value = 30, ● Maximum value= 200.
Time before retry is sent (in seconds)	Delay before emails stored in the buffer memory are re-sent after communication interruption: ● Default value = 15 s, ● Minimum value = 5 s, ● Maximum value= 3,600 s.
Service	
Service status variable	Used to determine the status of the Email service.

NOTE: When the maximum number of emails is reached (100), no further messages can be stored.

Configuring the Email Service

The Email configuration screen is shown below:



The fields in the Email screen are:

Fields	Function	
Send SMS	This function is not available for FactoryCastHMI products. Select this box to send a SMS instead of an email.	
Identifier	Email address or phone number of the sender (in case of SMS) of the message.	
Trigger	Events that will trigger the Email.	
Type	**NY**: Notify	Triggered by bit status change or word value change.
	RE: Rising Edge	Triggered by a bit rising edge or by an increasing word value.
	FE: Falling Edge	Triggered by a bit falling edge or by a decreasing word value.
	BQ: Bad Quality	Triggered if the trigger status is Bad quality.
Destination	Email address(es) of the receiver(s) of the message.	
Subject	A brief summary of the contents of the message.	
Contents	Type the content of the message in this area.	
📎	Click this button to attach a file to the email.	
Media	Select the media from which the attached file comes from.	
Path	Specify the path of the file in the media selected in Media target: ● to send a datalogging file, type */USERDATA/file.csv*, ● to send another file, type */wwwroot/path/file.xxx*. Where *file* is the name of the file and *xxx* is the extension of the file.	

NOTE: The following fields are mandatory to record and save an Email: **Identifier**, **Trigger** and **Destination**.

Including Dynamic Data

The email service enables you to include dynamic data in the body of the email.

To include dynamic data:

- place brackets before and after the variable name, or
- double click the **Contents** text box to display the **Variable Section** window. Select the variables you want to monitor in the list.

Example: *{plc.plc1.value1}* gives you the value of the value1 variable created in the device service.

Limitation

The number of messages you can configure in the project is restricted to 100.

The maximum number of Email services is 2.

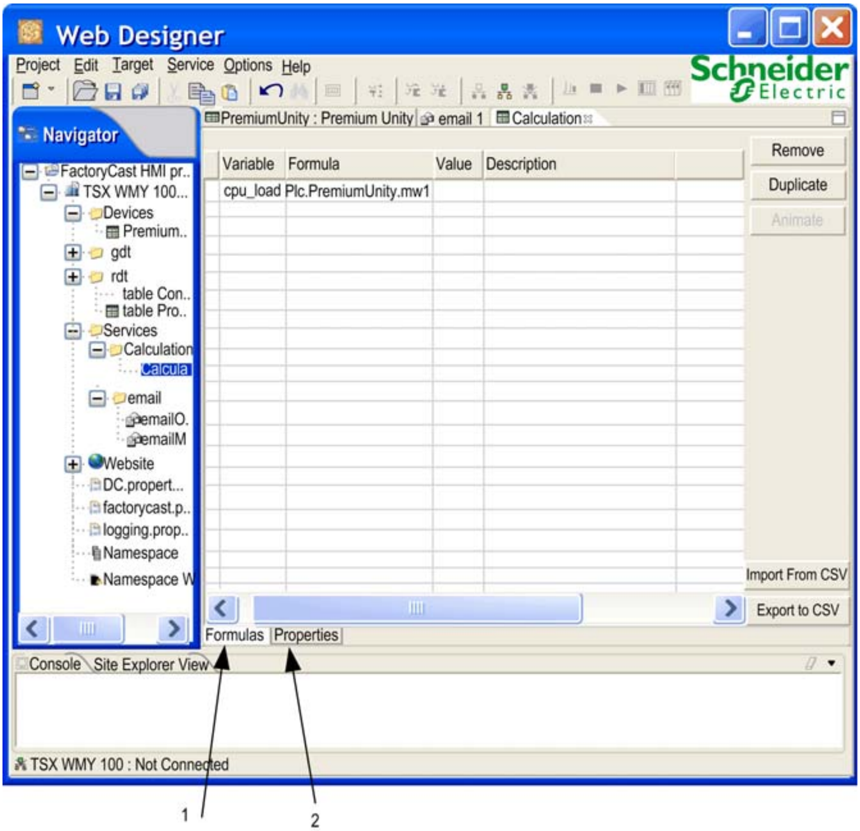
NOTE: The SMTP server must allow PLAIN or LOGIN authentication. Other authentication protocols are not supported.

Calculation Service

Presentation

Use the Calculation service to perform operations on or to combine variables. You can also use it to view the result of operations in real time to facilitate application debugging.

The configuration properties screen for the Calculation service is shown below:



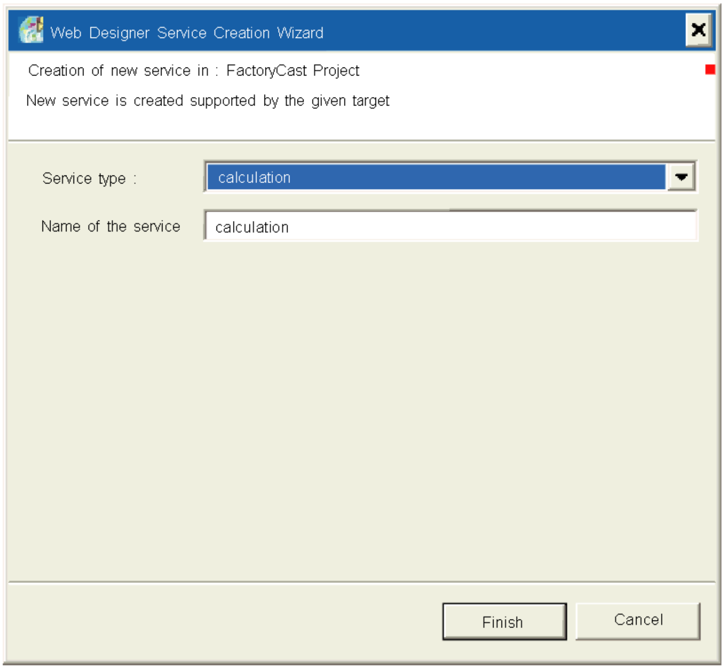
Variable	Formula	Value	Description
cpu_load	Plc.PremiumUnity.mw1		

TSX WMY 100 : Not Connected

1 Use this button to configure a formula.
2 Use this button to configure properties common to all formulas.

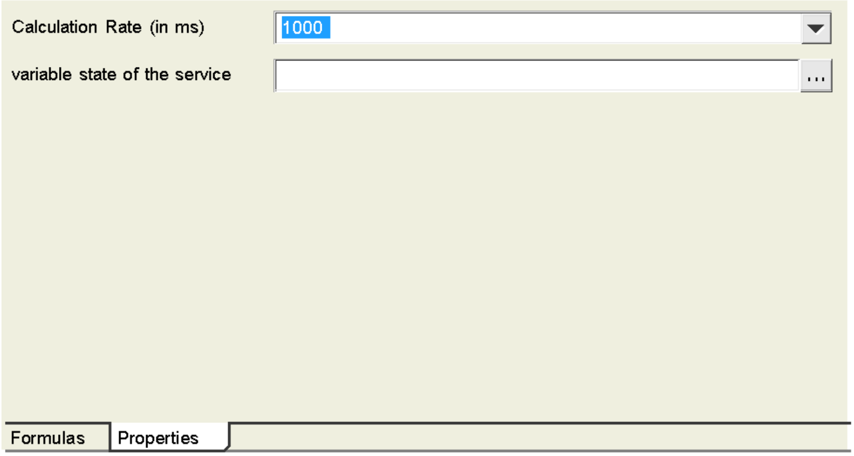
How to Create a Calculation Service

The table below describes how to create a Calculation service:

Step	Action
1	Click the name of your project in the browser.
2	To add the new service you can: ● Click **Project** → **New** → **Service** or, ● Right-click **Services** in the browser, then click **New Service**. **Result**: the following window appears:  Web Designer Service Creation Wizard Creation of new service in : FactoryCast Project New service is created supported by the given target Service type : calculation Name of the service calculation
3	Select **Calculation** in the **Service Type** list box.
4	Enter the service name or click **Finish** to keep the name by default.
5	The Calculation service has been created.

Configuring the Properties

The properties configuration screen for the Calculation service is shown below:



Calculation Rate (in ms) 1000

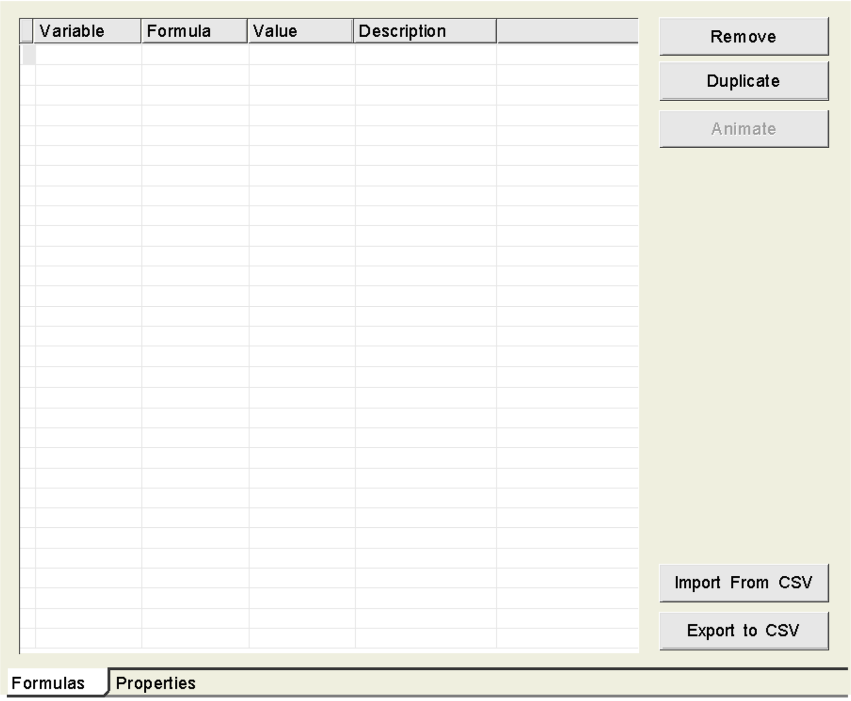
variable state of the service

This table describes the fields in the Calculation configuration screen:

Fields	Function
Calculation rate (in ms)	Frequency of formula execution using variables.
Variable state of the service	Used to determine the status of the calculation service.

Configuring Formulas

The formula configuration screen for the Calculation service is shown below:



This table describes the fields in the formula configuration screen:

Fields	**Function**
Variable	Name of the variable. **Note**: the variable can be a PLC variable that you set to the formula value.
Formula	Formula associated with the variable. The operators are limited.
Value	Gives the result of the formula (value of the variable) in real time. You must be connected and in animation mode.
Description	Comment on the variable.
Buttons	
Remove	Delete the selected variable from the table.
Duplicate	Duplicate the selected variable
Animate	Use this button to switch to animation mode. You must be connected to the module or in simulation mode.
Import from CSV	This button enables you to import symbols from a *.csv* file. You can create this file using Excel or a text editor (in this case, change the *.txt* file extension to *.csv*). You must use the following syntax: *variable;formula;description*.
Export to CSV	This button enables to export the variable table for the current service to a *.csv* file.

List of Available Operators

The following list provides details of numeric binary, boolean and unary operators. The result of the operation is displayed in the Result type column:

Type	Operator	Description	Example
Binary numeric	+	addition	1+2 displays the result "3"
	-	subtraction	5-2 displays the result "3"
	*	multiplication	5*2 displays the result "10"
	/	division	15/3 displays the result "5"
	<	less than	1<2 displays the result "true"
	>	greater than	1>2 displays the result "false"
	<=	less than or equal to	1<=2 displays the result "true"
	>=	greater than or equal to	2>=2 displays the result "true"
	==	equal to	3==2 displays the result "false"
	!=	not equal to	1!=2 displays the result "true"
Boolean numeric	&	and	1+1 displays the result "1"
	|	or	1 | 0 displays the result "1"
Unary	!	inverse	!1 displays the result "0"
	-	sign inversion	-1 displays the result -1
	.<. ? . : .		a<b ? c:d returns c if a<b otherwise the result is d

Formula Execution

The calculation service periodically executes a formula, according to the frequency configured in the **Properties** screen. The formula cells are interpreted then executed one by one from top to bottom. The calculation service updates the associated variable configured in the variable field with the new value.

Limitation

The number of calculations you can configure in the project is restricted to 100.

The maximum number of calculation services is 2.

Database Service

Presentation

This connection enables all internal or process data to be archived so that it can be logged and traced. The data can be archived (written) periodically or for a specific event. These variables can be from PLCs (I/O bits, internal bits, internal words and registers) or local to the module.

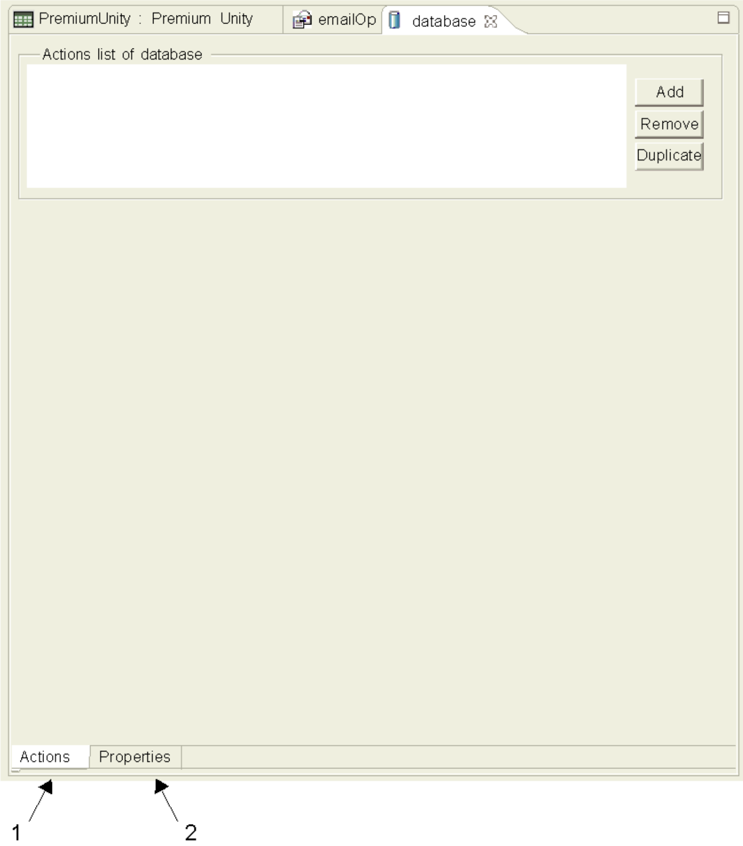
Types of database supported:

- SQL server 6.5, 7, 2000, 2005
- MySQL 4.1 or above
- Oracle 8i, 9i, 10i

NOTE: Schneider Electric does not guarantee:

- integrity of data sent by the module with the database service,
- the exactness of the time at which data are logged.

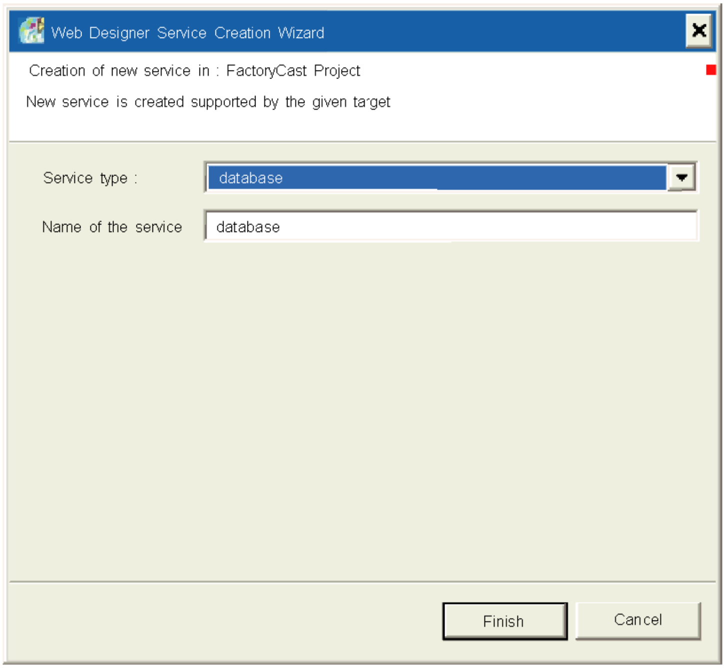
The configuration screen for the Database service is shown below:



1 Use this button to configure the actions.
2 Use this button to configure the general properties of the database.

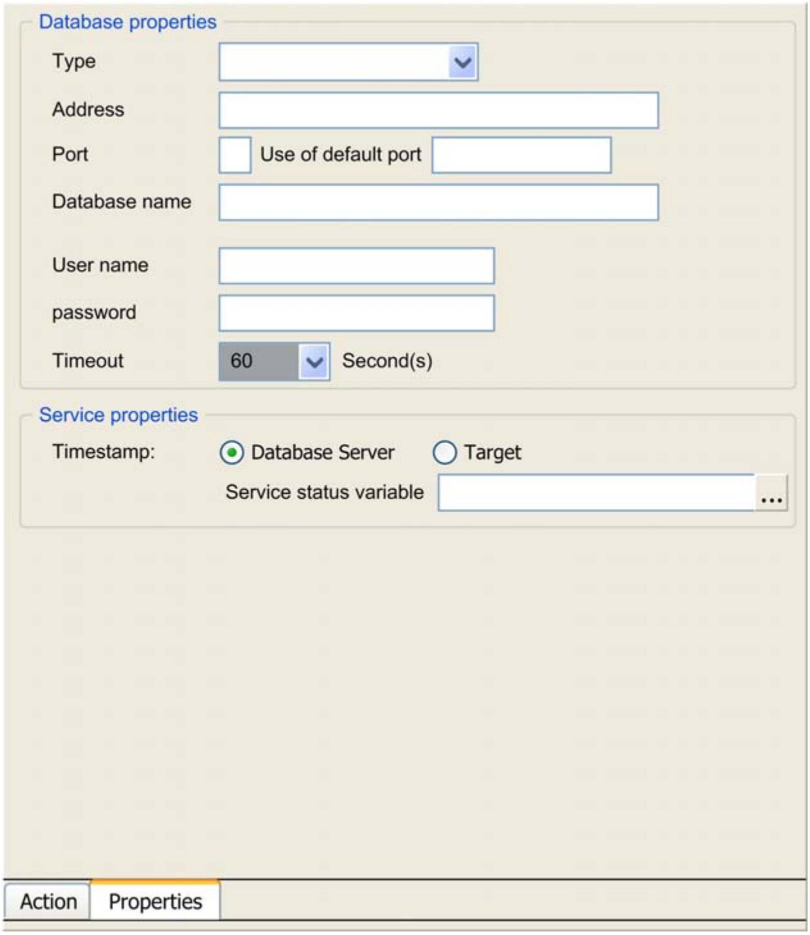
How to Create a Database Service

The following table describes how to create a Database service:

Step	Action
1	Click on the name of your project in the browser.
2	To add the new service you can: ● Click **Project** → **New** → **Service** or, ● Right-click **Services** in the browser, then click **New Service**. **Result**: the following window appears.  Web Designer Service Creation Wizard Creation of new service in : FactoryCast Project New service is created supported by the given ta'get Service type : database Name of the service database
3	Select **database** in the **Service Type** list box.
4	Enter the service name or click **Finish** to keep the name by default.
5	The Database service has been created.

Configuring the Properties

The following table introduces the properties configuration window of the Database service:



Timeout 60 Second(s)

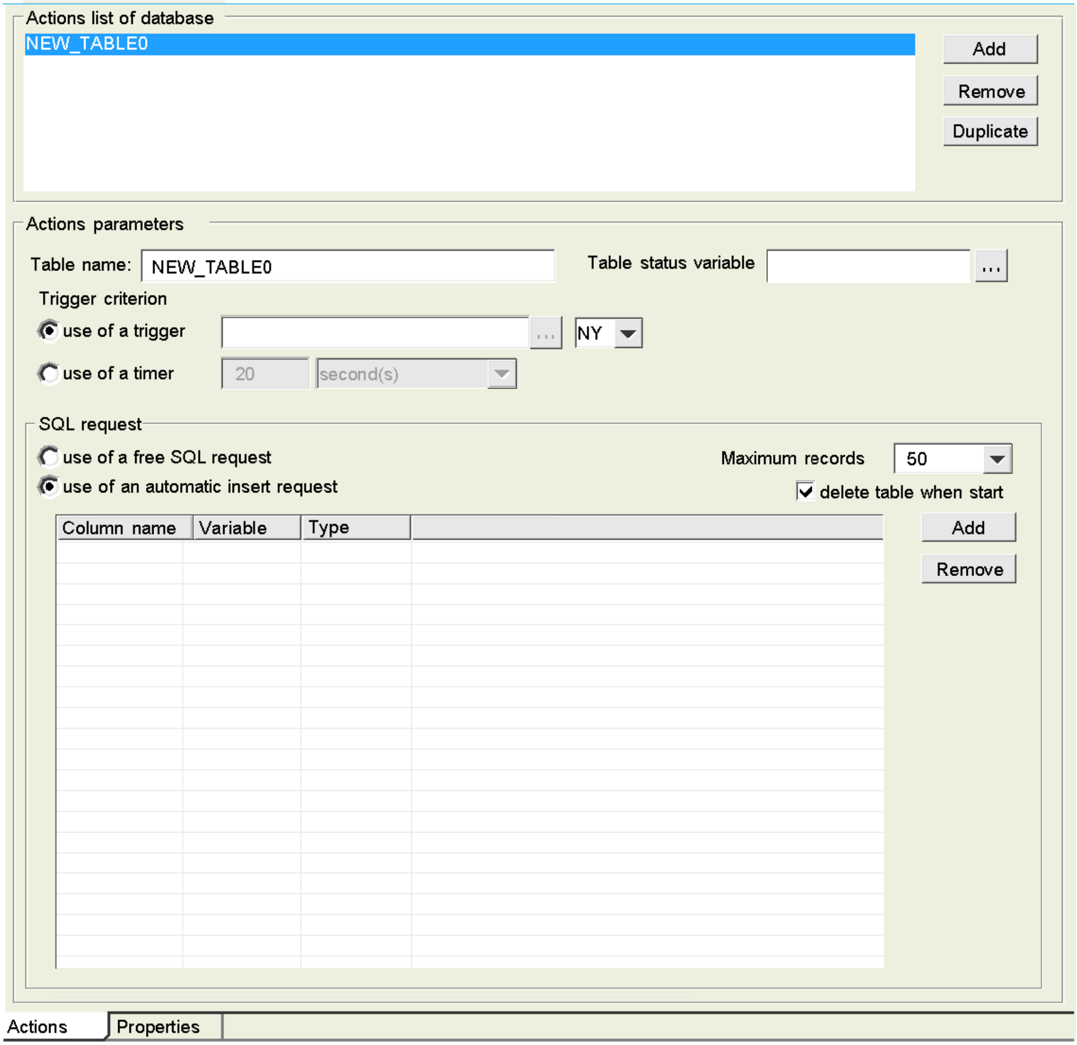
Timestamp: ● Database Server ○ Target

The following table describes the parameters of the database service:

Fields	Function
Type	● Oracle ● MySql ● SQL Server
Address	IP address of the database server.
Port	Used to validate a default address.
Use Default Port	Port used by the database to access the server.
Database Name	Name of the database that will contain all data configured by the user.
User Name & Password	Username and password for database access (database connection string).
Timeout	Period of time that will be allowed to elapse before the system kills the current process when it improperly operates.
Service status variable	Used to determine the status of the database service.
Timestamp server	The date is taken directly from the server for the dating values it receives.
Timestamp Target	The date is taken directly from the module for the dating values it receives.

Configuring the Log Files

The configuration screen for the Database service is shown below:



Actions list of database

NEW_TABLE0

Actions parameters

Table name: NEW_TABLE0

Table status variable

Trigger criterion

● use of a trigger NY

○ use of a timer 20 second(s)

SQL request

○ use of a free SQL request

● use of an automatic insert request

Maximum records 50

☑ delete table when start

Column name	Variable	Type

The following table describes the fields in the log file configuration screen:

Fields	**Function**
Actions list of database	List of all tables configured on the server. Use the **Add**, **Remove** and **Duplicate** buttons to create, delete or copy a table. When you duplicate an existing database, the new name will have the suffix _copy added to its original name.
Actions parameters	Name of the new archiving table. If the table does not exist in the database, the database service will create it on the first connection. To trigger variable logging, you must specify an event. This will either be a periodic event (use periodic log), in which case you need to set a time base, or it will be an event associated with a variable (use trigger), in which case you need to provide the name of the variable (for example, calculation.calculation1.Pressure1).
SQL Request	Two options are available: ● create SQL requests in the **Free Text SQL Request** window, or ● use the table provided to specify the name of the column and the variable type and name. In this case, the column name syntax must comply with the database syntax. As a general rule, avoid using punctuation marks, upper case letters and hyphens, (refer to your database documentation).

Database Properties

The table below shows the equivalents of Xopen types for other databases:

Xopen type	**MySql**	**Oracle**	**SQL Server**
BOOLEAN	SMALLINT	NUMBER(1)	SMALLINT
TINYINT	SMALLINT	SMALLINT	SMALLINT
SMALLINT	SMALLINT	NUMBER	SMALLINT
INTEGER	INTEGER	NUMBER	INTEGER
BIGINT	DECIMAL(200)	NUMBER	NUMERIC(38,0)
DOUBLE	DECIMAL(50,80)	NUMBER	FLOAT(32)
REAL	REAL	REAL	REAL
TIMESTAMP	TIMESTAMP	TIMESTAMP	TIMESTAMP
DATE	DATETIME	DATE	DATETIME
VARCHAR	VARCHAR(255)	VARCHAR(255)	VARCHAR(255)
All other types	VARCHAR(255)	VARCHAR(255)	VARCHAR(255)

Limitation

The maximum number of columns is 50 per table.

The maximum number of database connections that can be configured is 10 per service.

The maximum number of database services is 3.

NOTE: The service database contains a buffer that allows it to store a limited number of events in case of communication failure or network disruption. The number of events must not exceed 5000.

Datalogging Service

Presentation

The Datalogging service enables you to backup information such as configuration parameters, variable’s values, log files coming from the devices connected to the FactoryCast module into the internal memory of the module.

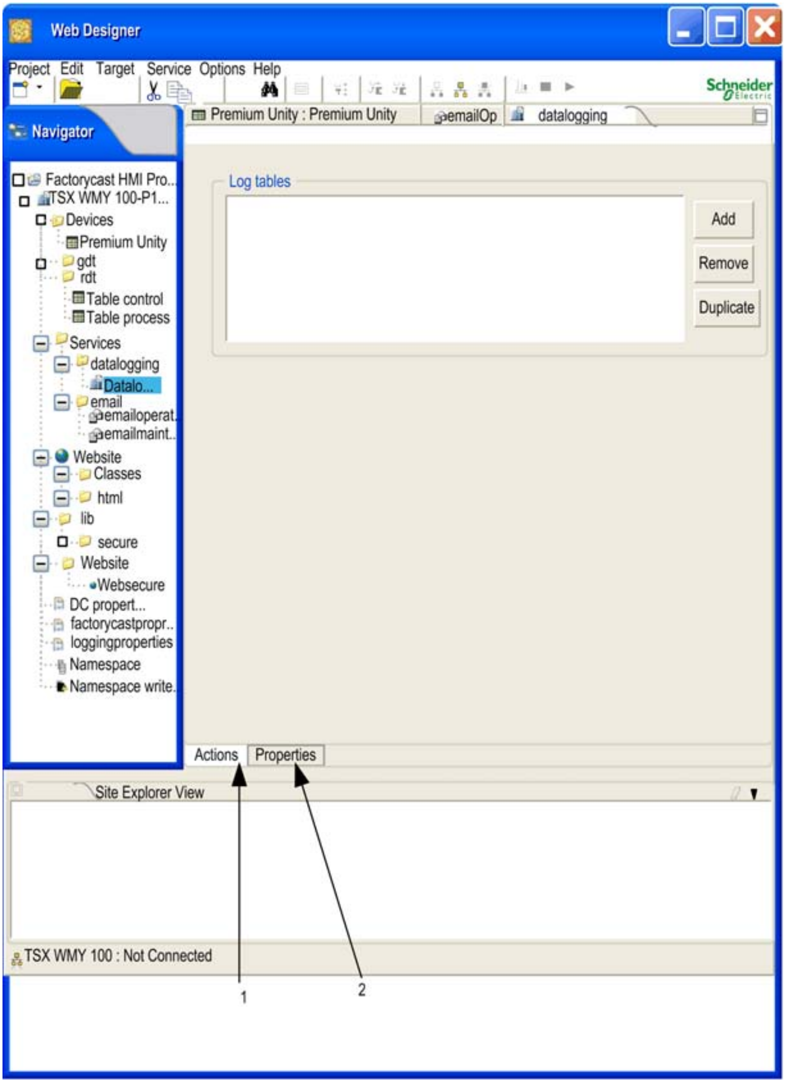
This can be achieved periodically, or when a specific event occurs (configured by the user).

The file system of the FactoryCast module is accessible from any FTP client. You can specify a URL for automatic sending of the information to a remote FTP server.

You can use the log file to animate a Datalogging history object *(see page 202)* in the Graphic Editor.

NOTE: We recommend you to don’t log file at a frequency lower than 30 minutes in the order to save internal flash life.

The properties configuration screen for the datalogging service is shown below:

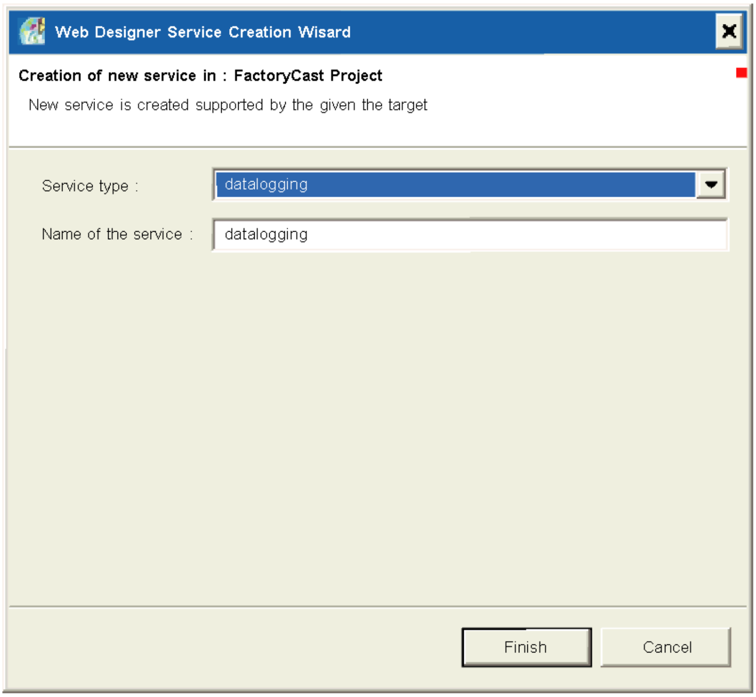


1 Use this button to configure the parameters to be archived.

2 Use this button to configure the general properties of the datalogging service.

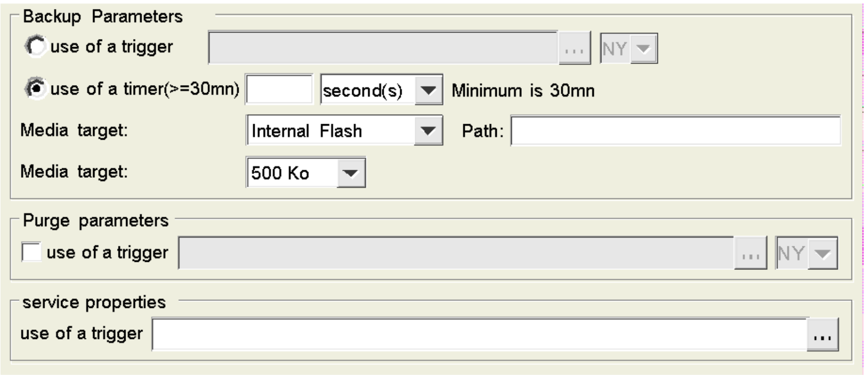
How to Create a Datalogging Service

The following table describes how to create a datalogging service:

Step	Action
1	Click the name of your project in the browser.
2	To add the new service: ● click **Project** → **New** → **Service** or, ● right-click **Services** in the browser, then click **New Service**. **Result**: the following window appears.  Web Designer Service Creation Wisard Creation of new service in : FactoryCast Project New service is created supported by the given the target Service type : datalogging Name of the service : datalogging
3	Select datalogging in the **Service Type** listbox.
4	Enter the service name or click **Finish** to keep the name by default.
5	The datalogging service has been created.

Configuration of Properties

The properties configuration screen for the datalogging service is shown below:



Backup Parameters

○ use of a trigger ... NY

● use of a timer(>=30mn) second(s) Minimum is 30mn

Media target: Internal Flash Path:

Media target: 500 Ko

Purge parameters

□ use of a trigger ... NY

service properties

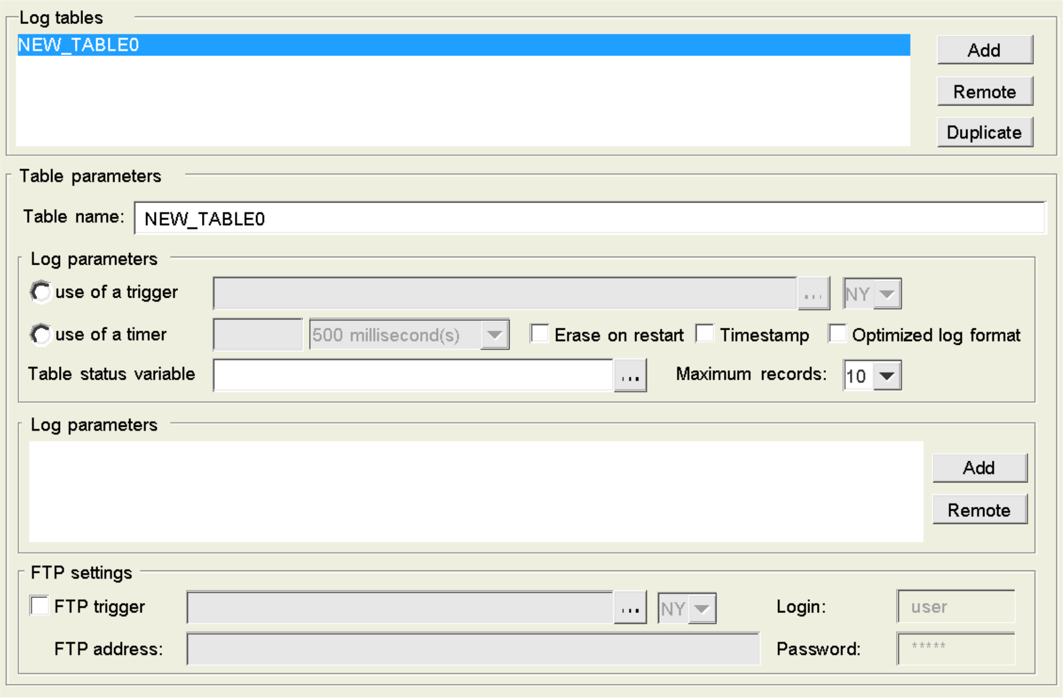
use of a trigger ...

The following table describes the fields of the datalogging configuration screen:

Fields	Function
Backup parameters	
Action parameters	To trigger variable logging, an event must be configured. This will either be a periodic event (use periodic log), in which case you need to set a time base, or it will be an event associated with a variable (use trigger), in which case you need to provide the name of the variable (for example, calculation.calculation1.Pressure1).
Media target	Select the media on which the data will be backup: ● Select **Internal Flash** to store the information on the internal memory of the module. ● Select **Internal Saved RAM** to store the information on the internal RAM of the module (not available for FactoryCast HMI modules). ● Select **CF card** to store the information on the memory card of the module (not available for FactoryCast HMI modules). ● Select **USB Storage** to store the information on a USB memory key connected to the module (not available for FactoryCast HMI modules).
Path	Specify the path of the media selected in **Media target**. The default path is: *NAND\FLASH1\USERDATA*. You can create subdirectories in the *USERDATA* directory. If you want to attached a subdirectory (i.e. *NAND\FLASH1\USERDATA\SUBD1\SUBD2*) to the email, specify its path without typing the default path (i.e. SUBD1\SUBD2).
Log maximum size	Specify the maximum size of the log file.
Purge parameters	
Use of a trigger	Define the event that will purge the current log file(s) and start a new one at service restart.
Service properties	
Variable state of the service	Used to determine the status of the datalogging service.

Log File Configuration

The configuration screen of the datalogging service is shown below:



Log tables

NEW_TABLE0

Table parameters

Table name: NEW_TABLE0

Log parameters

○ use of a trigger NY

○ use of a timer 500 millisecond(s) □ Erase on restart □ Timestamp □ Optimized log format

Table status variable Maximum records: 10

Log parameters

FTP settings

□ FTP trigger NY Login: user

FTP address: Password: *****

The following table describes the fields of the datalogging configuration screen:

Fields	Function
Log Tables	List the current log files stored on the module. It is possible to: ● Create a new log file using the **Add** button, ● Remove a log file using the **Remove** button, ● Duplicate a log file using the **Duplicate** button.
Table name	Name of the log file that will contain all data configured by the user.
Log parameters	
Trigger criteria	To trigger logging of variables, an event must be configured. This will either be a periodic event (**use of a timer**), in which case you need to set a timebase, or it will be an event associated with a FactoryCast variable (**use of a trigger**), in which case you need to provide the name of this variable (for example, calculation.calculation1.Pressure1). It is possible to use both criteria at the same time.
Erase on restart	If the box is checked, the log file will be deleted on restart of the module.
Timestamp	Check this box if you want to have the hour and the date of each event included in the log file.
Optimized log format	Check this box for optimizing the log file format *(see page 57)* of the *.csv* file. In this case, the variable name does not appear in each record.
Table status variable	Used to determine the status of the selected table.
Maximum Record Number	Configure the maximum of records that can be stored in a log file. If that number is reached, a new log file overwrites the old file.
Log variables	
Log variables	This section is used to configure the log files content. If the **Timestamp** checkbox has been checked, the hour and the date of each event is included in the log file. The **variable name** part lists the variables (PLC or Calculation variables) that will be stored in the log file.
FTP settings	
FTP trigger	Specify the event that will trigger logging of variables via FTP.
FTP address	Type the address of the FTP server.
Login and Password	Protection of FTP access by username and password (default: **userdata/userdata**).

File Format

The file format is fixed and cannot be modified. The file is encoded in pure ASCII text with a *.csv* extension. Microsoft Excel is able to open these files.

Example of log file:

```
2003-10-01 02:44:55;plc.plc1.height;150;plc.plc1.length;200;plc.plc1.width;50;
2003-10-01 03:48:08; plc.plc1.height;140;plc.plc1.length;150;plc.plc1.width;30;
```

```
2003-10-01 04:55:10;
plc.plc1.height;220;plc.plc1.length;280;plc.plc1.width;80;2003-10-01 06:01:05;
plc.plc1.height;170;plc.plc1.length;220;plc.plc1.width;60;
```

Example of optimized log file:

```
Date;plc.plc1.height;plc.plc1.length;plc.plc1.width;
```

```
2003-10-01 02:44:55;150;200;50;2003-10-01 03:48:08;140;150;30;2003-10-01
04:55:10;220;280;80;2003-10-01 06:01:05;170;220;60;
```

Limitation

The memory space allocated to the datalogging service is limited to 500 ko on the Flash memory of the module.

You must pay attention to the file size of your log file in order to do not reach the limit of 500 ko. The estimation of the log file size table *(see page 541)* shows you an estimation of the log file size depending on number of variables and the number of logs.

Active Pages Service

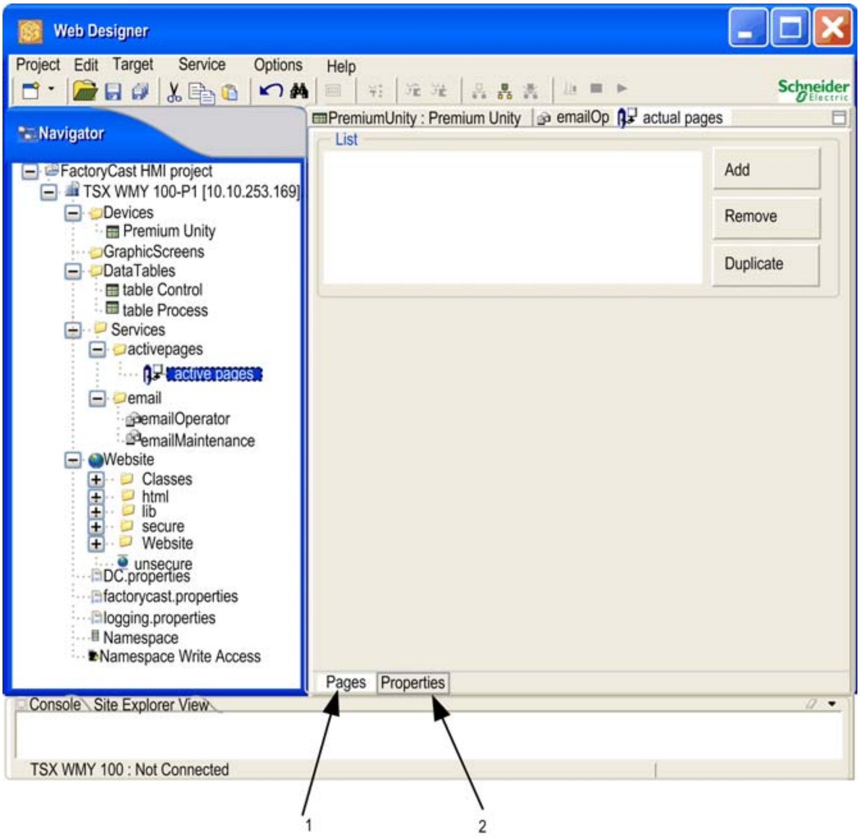
Presentation

The Active Pages service enables you to create your own HTML pages to view variables in real time.

This technology has two benefits:

- The Web page displayed only includes data useful for viewing, meaning data size is kept to a minimum (a few KBs).
- The only protocol used for communication with the browser is HTTP, so this technology can bridge all firewalls used in an internet infrastructure.

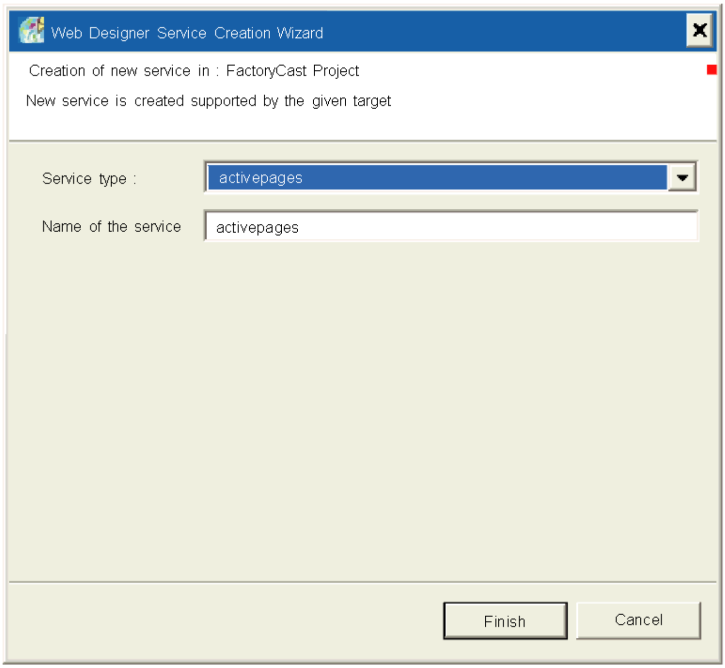
The configuration properties screen of the Active Pages service is shown below:



1 Use this button to configure the Active Pages service.
2 Use this button to configure properties of the pages.

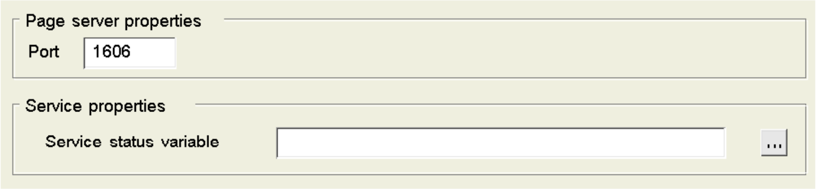
How to Create an Active Pages Service

The following table describes how to create an Active Pages service:

Step	Action
1	Click the name of your project in the browser.
2	To add the new service you can: ● Click **Project** → **New** → **Service** or, ● Right-click **Services** in the browser, then click **New Service**. **Result**: the following window appears.  Web Designer Service Creation Wizard Creation of new service in : FactoryCast Project New service is created supported by the given target Service type : activepages Name of the service activepages
3	Select **active pages** in the **Service Type** list box.
4	Enter the service name or click **Finish** to keep the name by default.
5	The Active Pages service has been created.

Configuring the Properties

The properties configuration screen of the Active Pages service is shown below:



Page server properties

Port: 1606

Service properties

Service status variable:

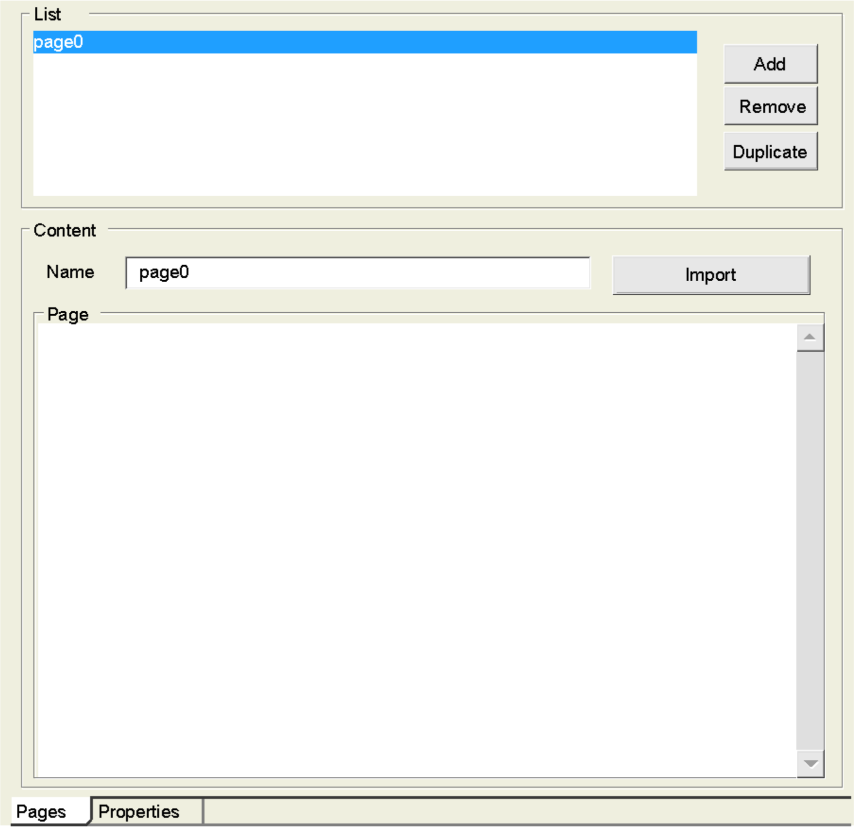
The following table describes the fields of the properties configuration screen:

Fields	Function
Port	Active pages port number for the server IP address. The port numbers are between 1024 and 65535. By default, the first port declared will be assigned number 1606.
Service status variable	Used to determine the status of the Active Pages service.

NOTE: It is possible to create two Actives Pages services. In this case, you must configure the two services with different port numbers.

Configuring Pages

The configuration screen of the Active Pages service is shown below:



List: page0

Name: page0

The following table describes the fields of the configuration screen:

Fields	**Function**
List	List of all the active HTML pages created with the calculation service on the server. Use the **Add**, **Remove** and **Duplicate** buttons to create, delete or copy a page. When you duplicate an existing page, the new name will have the suffix _copy added to its original name.
Import	Use this button to import complete HTML code from an *.htm* or *.txt* file. You can create the file with Frontpage or a text editor. It overwrites the code already written.
Name	Name of the page currently being created. You can rename the page using this field.
Page	Use this field to write HTML code of the current page or to view code which has been imported. Double-clicking in the field enables you to display the list of variables. If you select one of these variables, it is directly incorporated into the code. You can incorporate calls to applets, bitmaps or HTML tags into the page’s HTML code.

Accessing the Pages

To access the pages, use the following syntax:

- http://IP address of module: Active Pages port or
- http://domain name of module: Active Pages port

Example:

- *http://139.160.234.166:1606*
- *http://hmi4:1606*

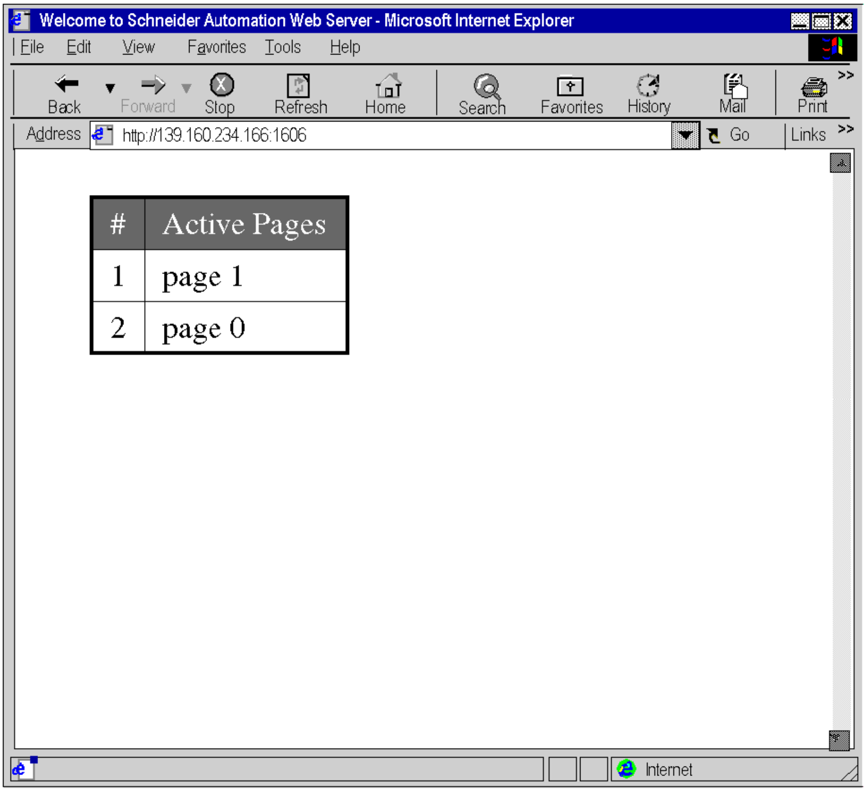
To access a specific page, you must use the following syntax:

- http://IP address of module: Active Pages port / name of page
- http://domain name of module: Active Pages port / name of page

Example:

- *http://139.160.234.166:1606/page0*
- *http://hmi4:1606/page0*

The following diagram illustrates Active Pages:



#	Active Pages
1	page 1
2	page 0

NOTE: Automatic refresh of the page called up is not available by default but you can include a refresh command in the HTML code of the page.

Example of HTML code you can use in a page to be refreshed:

```
<meta http-equiv="refresh" content="1";url=page1>
```

```
where content= "refresh time in seconds";url=name of page to
be reloaded
```

Limitation

The maximum number of active pages is 2 per project.

The active pages service may include up to a maximum of 64 pages in the project.

1,000 variables can be declared for the entire set of project pages.

Recipe Service

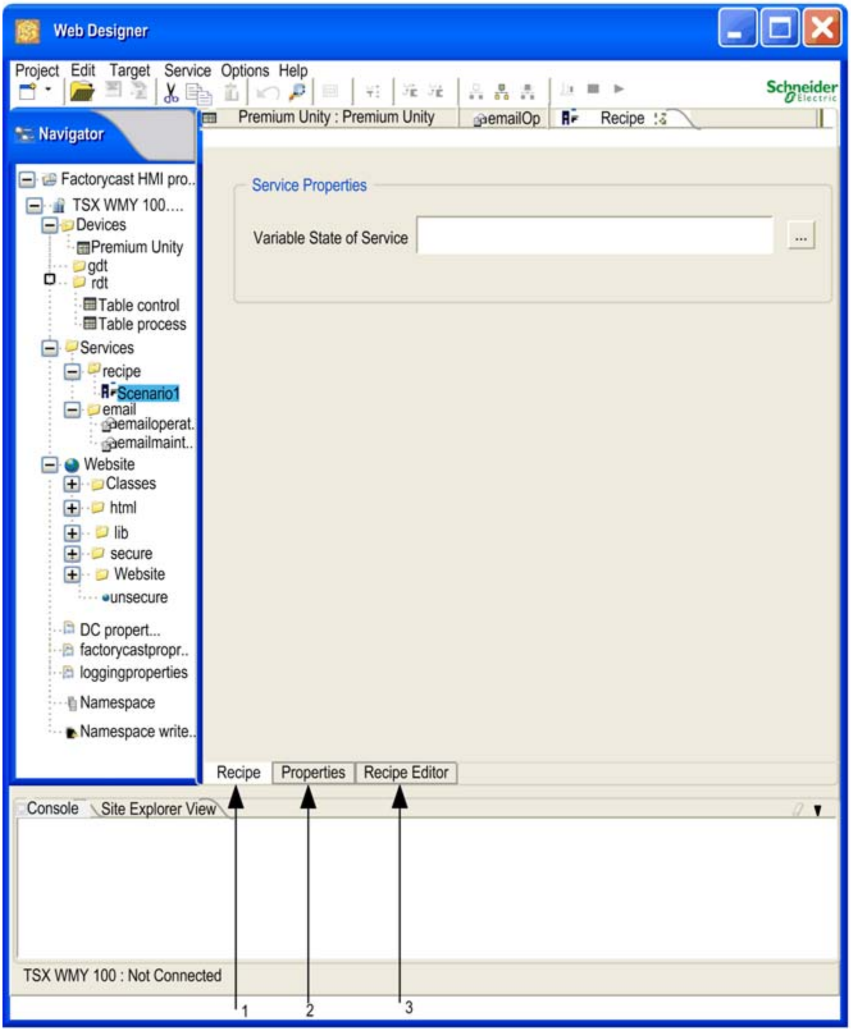
Presentation

The Recipe service is used following an event to initialize a group of variables set at predetermined values. The predetermined values are stored in an XML file.

For this particular event, the recipe service accesses the recipe file (in XML format) through an HTTP server, reads the values that correspond to the variables, then updates the PLC variables.

This XML file can be created either by using the Recipe Editor, or by using an external tool.

The Recipe service configuration screen is shown below:

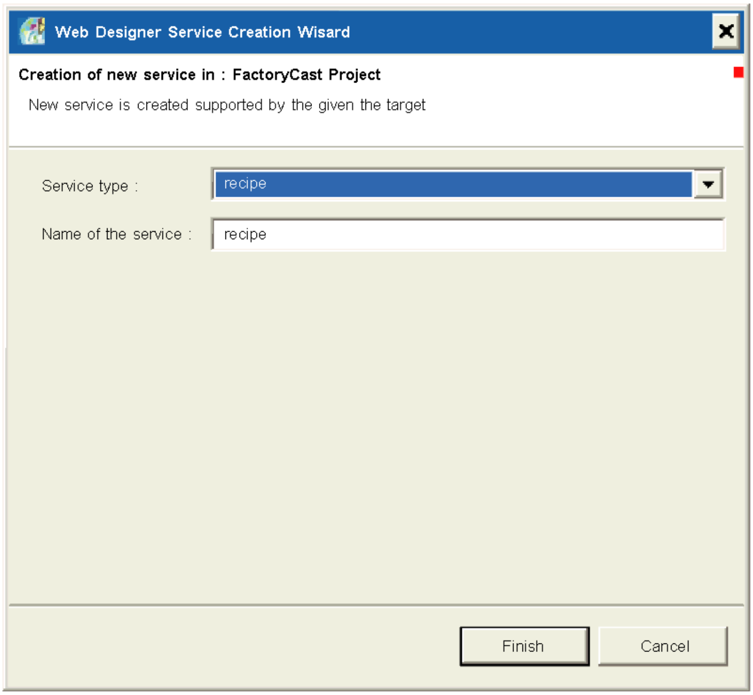


The numbered references are as follows:

Reference	Function
1	View the recipe service’s status with this tab.
2	Configure recipes’ properties with this tab.
3	Configure a recipe’s values with this tab.

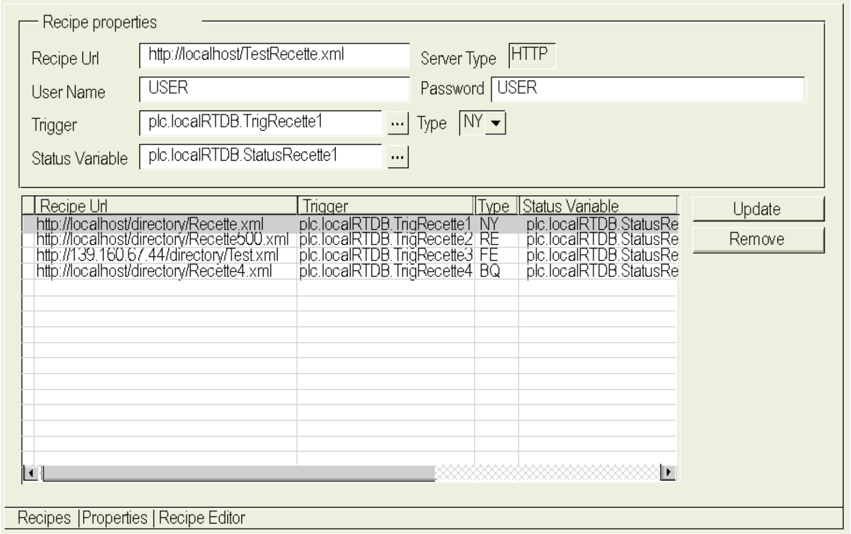
How to Create a Recipe Service

The following table describes how to create a Database connection service:

Step	Action
1	Click on the name of your project in the browser.
2	To add the new service you can: ● click **Project** → **New** → **Service** or, ● right-click **Services** in the browser, then click **New Service**. **Result**: the following window appears.  Web Designer Service Creation Wisard Creation of new service in : FactoryCast Project New service is created supported by the given the target Service type : recipe Name of the service : recipe
3	Select **recipe** in the **Service Type** list box.
4	Enter the service name or click **Finish** to keep the name by default.
5	The recipe service has been created.

Configuring the Properties

The configuration screen of the Recipe service is shown below:



Recipe properties

Recipe Url: http://localhost/TestRecette.xml
Server Type: HTTP
User Name: USER
Password: USER
Trigger: plc.localRTDB.TrigRecette1
Type: NY
Status Variable: plc.localRTDB.StatusRecette1

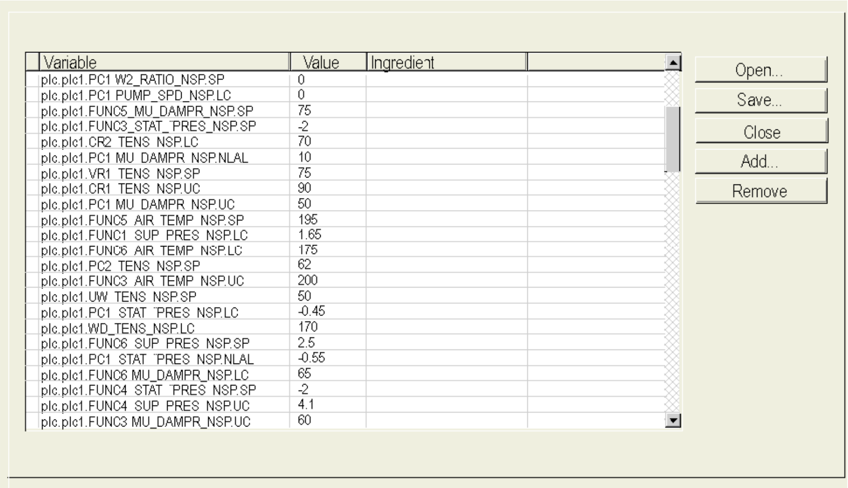
Recipe Url	Trigger	Type	Status Variable
http://localhost/directory/Recette.xml	plc.localRTDB.TrigRecette1	NY	plc.localRTDB.StatusRe
http://localhost/directory/Recette500.xml	plc.localRTDB.TrigRecette2	RE	plc.localRTDB.StatusRe
http://139.160.67.44/directory/Test.xml	plc.localRTDB.TrigRecette3	FE	plc.localRTDB.StatusRe
http://localhost/directory/Recette4.xml	plc.localRTDB.TrigRecette4	BQ	plc.localRTDB.StatusRe

This table describes the various fields that make up the configuration screen:

Field	Function	
Recipe URL	Location of the recipe file. If the file is stored in the module, the URL must be in the format *http://localhost/directory/filename.xml*, or *http://IP address of HTTP server/directory/filename.xml*.	
Username	Username for connecting to the HTTP server.	
Password	Password for connecting to the HTTP server.	
Status Variable	Variable that describes the recipe’s status.	
Trigger	Variable from which the event is triggered.	
Type	NY: Notify	Triggered on a bit or word value state change.
	RE: Rising Edge	Triggered on a rising edge for a bit, or a value increase for a word.
	FE: Falling Edge	Triggered on a falling edge for a bit, or a value decrease for a word.
	BQ: Bad Quality	Triggered if the trigger status is "Bad quality".
Update	To update the selected recipe’s parameters.	
Remove	To remove a recipe from the list.	

Recipe Editor

The Recipe Editor screen is used to input a recipe’s values in an XML-format recipe file:



Variable	Value	Ingredient
plc.plc1.PC1 W2_RATIO_NSP.SP	0	
plc.plc1.PC1 PUMP_SPD_NSP.LC	0	
plc.plc1.FUNC5_MU_DAMPR_NSP.SP	75	
plc.plc1.FUNC3_STAT_¨PRES_NSP.SP	-2	
plc.plc1.CR2 TENS NSP.LC	70	
plc.plc1.PC1 MU DAMPR NSP.NLAL	10	
plc.plc1.VR1 TENS NSP.SP	75	
plc.plc1.CR1 TENS NSP.UC	90	
plc.plc1.PC1 MU DAMPR NSP.UC	50	
plc.plc1.FUNC5 AIR TEMP NSP.SP	195	
plc.plc1.FUNC1 SUP PRES NSP.LC	1.65	
plc.plc1.FUNC6 AIR TEMP NSP.LC	175	
plc.plc1.PC2 TENS NSP.SP	62	
plc.plc1.FUNC3 AIR TEMP NSP.UC	200	
plc.plc1.UW TENS NSP.SP	50	
plc.plc1.PC1 STAT ¨PRES NSP.LC	-0.45	
plc.plc1.WD_TENS_NSP.LC	170	
plc.plc1.FUNC6 SUP PRES NSP.SP	2.5	
plc.plc1.PC1 STAT ¨PRES NSP.NLAL	-0.55	
plc.plc1.FUNC6 MU_DAMPR_NSP.LC	65	
plc.plc1.FUNC4 STAT ¨PRES NSP.SP	-2	
plc.plc1.FUNC4 SUP PRES NSP.UC	4.1	
plc.plc1.FUNC3 MU_DAMPR_NSP.UC	60	

This table describes the "Recipe Editor":

Field	Function
Variable	Name of the variable to which a value is to be given.
Value	Value to attribute.
Ingredient	Optional comment to describe the variable’s role.
Open	To open an XML-format recipe file.
Save	To save a recipe as an XML file.
Close	To close the Recipe Editor.
Add...	To add a variable.
Remove	To remove a variable.

Adding Variables

Step	Action
1	Click **Add**. **Result:** the variable selection box is displayed.
2	Select the variables to insert into the recipe file. Multiple variables may be selected by holding down the Ctrl or Shift keys while left-clicking.
3	Close the window with the Confirm button. **Result:** the selected variables are added to the recipe file.

Saving a Recipe File

Step	Action
1	Click **Save**. **Result:** the **Save as** dialog box is displayed.
2	The file must be saved on disk in order to be transferred to an HTTP server later. If you would like the file to remain local to the FactoryCast HMI module, you must save it in a directory located in the website root of the project currently being configured. For example: *C:\Program Files\Schneider Electric\FactoryCast HMI\workspace\project_name\wwwroot\recipes\my recipe.xml*
3	Close the window with the **Save** button. **Result:** the recipe file is saved.

Transferring a recipe file to a module

Step	Action
1	In the navigator, select the recipe file to transfer in the directory located under the website root directory (*wwwroot*).
2	Right-click **Partial transfer -> :PC -> Module**

NOTE: For a recipe file that is not local to the module, you must first transfer it to the target HTTP server without using the FactoryCast HMI configuration application.

Example of an XML File

```
<?xml version="1.0" encoding="utf-8" ?> -
<table>-
 <data>
  <name>plc.localRTDB.VarRecette403</name>
   <value>4</value>
   <ingredient>douwn part black</ingredient>
   </data>- <data>
  <name>plc.localRTDB.VarRecette402</name>
   <value>3</value>
   <ingredient>up part blue</ingredient>
   </data>- <data>
  <name>plc.localRTDB.VarRecette401</name>
   <value>2</value>
   <ingredient>left part red</ingredient>
   </data>- <data>
  <name>plc.localRTDB.VarRecette400</name>
   <value>1</value>
  </data>
</table>
```

Limitations

- Recipe files must be accessible by an HTTP server.
- A FactoryCast HMI application can only manage a single recipe service.

Recipe Database Service

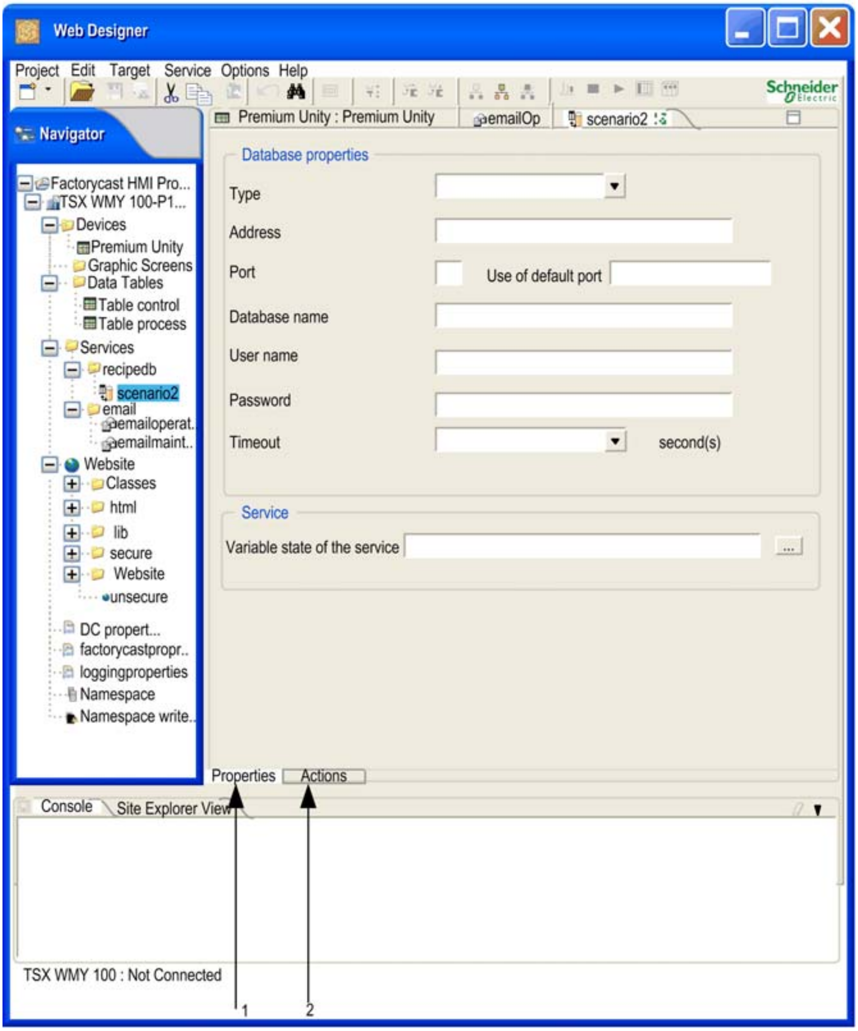
Presentation

The Recipe database service enables you to set the value of variables to predetermined values stored in database tables. You can create a scenario or initialize system variables of your system with values stored in a database (SQLServer, MySql and Oracle).

You can also save the state of your system by storing the value of your system variables in database tables.

This can be achieved periodically, or when a specific event occurs (configured by the user).

The Recipe database service configuration screen is shown below:

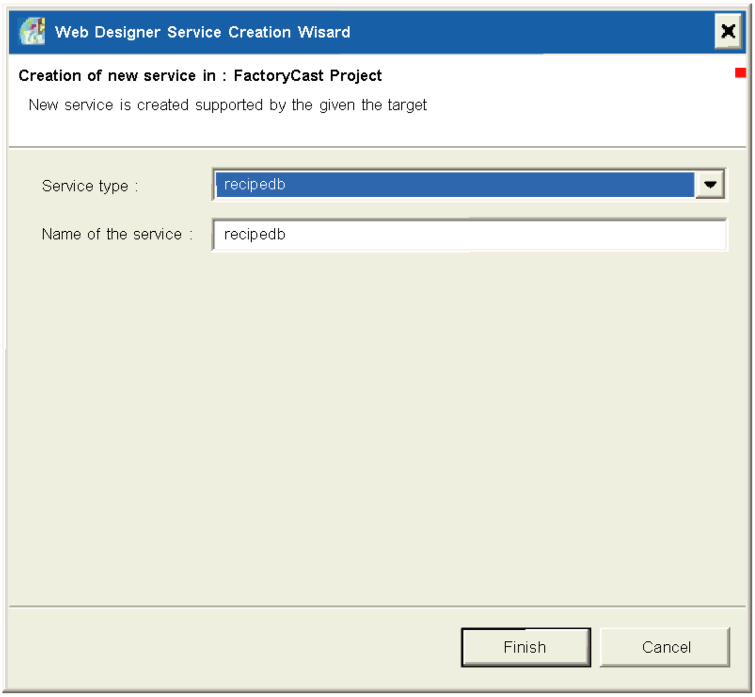


1 Configure the properties of the recipe database system with this tab.

2 Configure the actions to be undertaken with this tab.

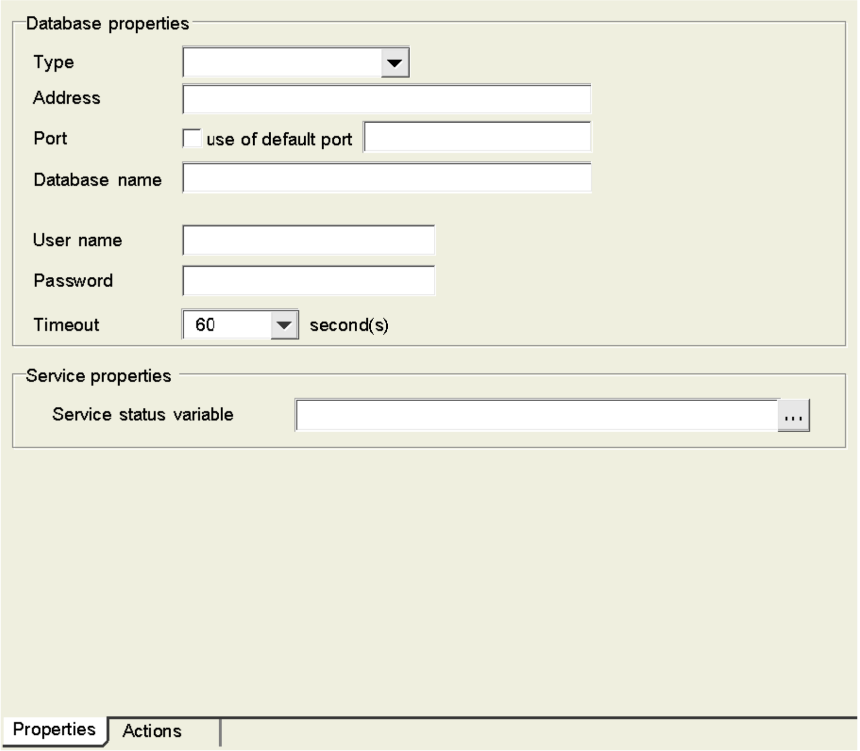
How to Create a Recipe Database Service

The following table describes how to create a recipe database service:

Step	Action
1	Click the name of your project in the browser.
2	To add the new service you can: ● click **Project** → **New** → **Service** or, ● right-click **Services** in the browser, then click **New Service**. **Result**: the following window appears.  Web Designer Service Creation Wisard Creation of new service in : FactoryCast Project New service is created supported by the given the target Service type : recipedb Name of the service : recipedb
3	Select **recipedb** in the **Service Type** listbox.
4	Enter the service name or click **Finish** to keep the name by default.
5	The recipe database service has been created.

Configuration of Properties

The properties configuration window of the Database connection service is shown below:



Database properties

Type

Address

Port □ use of default port

Database name

User name

Password

Timeout 60 second(s)

Service properties

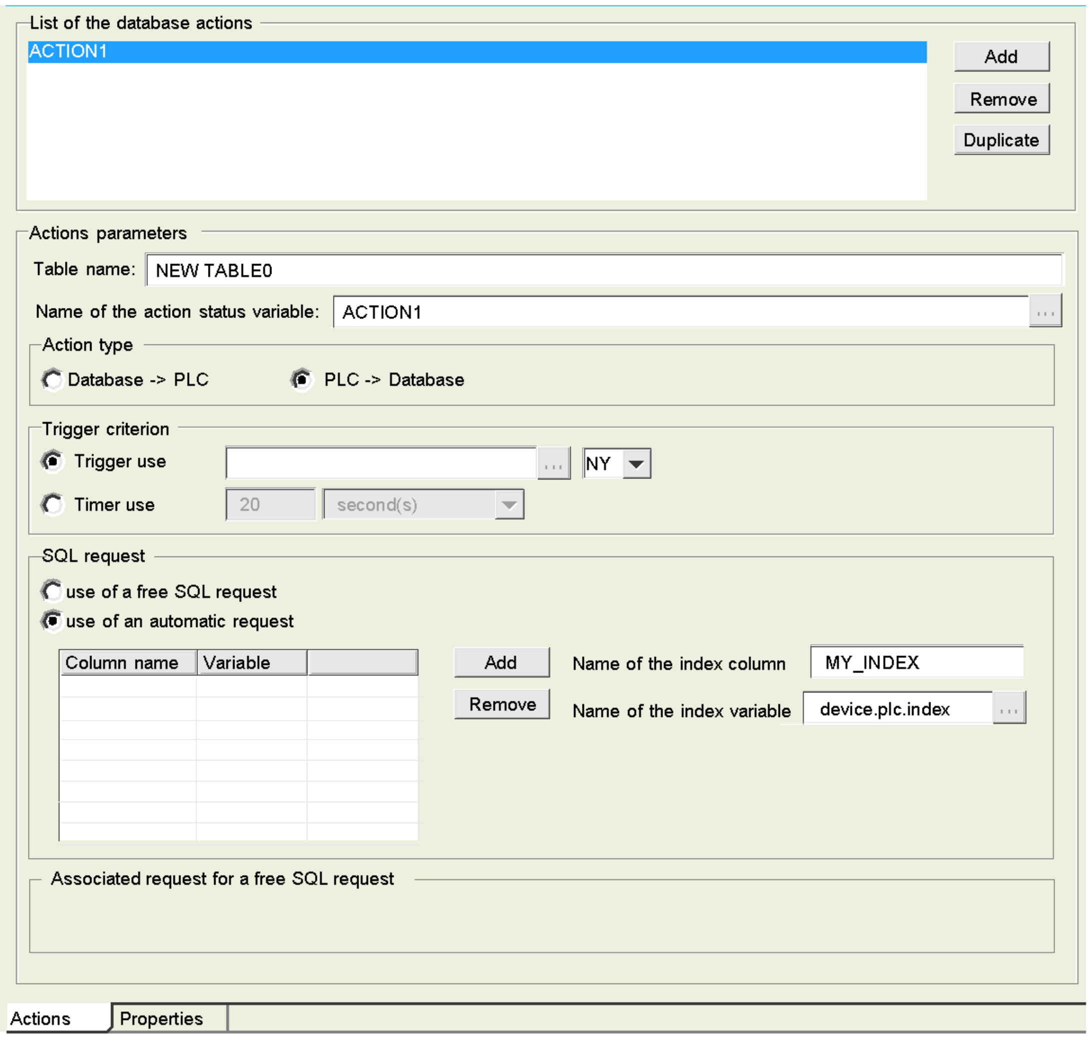
Service status variable ...

The following table describes the fields of the database server configuration screen:

Fields	**Function**
Type	● Oracle 8i, 9i, 10i ● MySql 4.1 or above ● SQL Server 6.5, 7, 2000, 2005
Address	IP address of the database server.
Port	Used to validate a default address.
Use Of Default Port	Port used by the database to access the server.
Database Name	Name of the database that will contain all data configured by the user.
User Name & Password	Username and password (database connection string) to access the database server.
Timeout	Period of time that will be allowed to elapse before the system kills the current process when it improperly operates.
Service status variable	Used to determine the status of the recipe database service.

Configuring the Actions

The action screen of the recipe database service is shown below:



List of the database actions

ACTION1

Actions parameters

Table name: NEW TABLE0

Name of the action status variable: ACTION1

Action type

○ Database -> PLC ● PLC -> Database

Trigger criterion

● Trigger use NY

○ Timer use 20 second(s)

SQL request

○ use of a free SQL request

● use of an automatic request

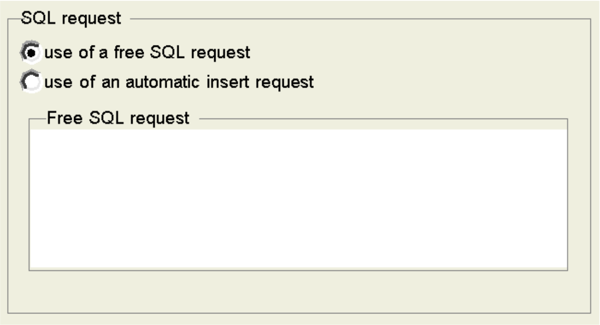
Column name	Variable	

Name of the index column: MY_INDEX

Name of the index variable: device.plc.index

Associated request for a free SQL request

This table describes the fields that make up the action screen:

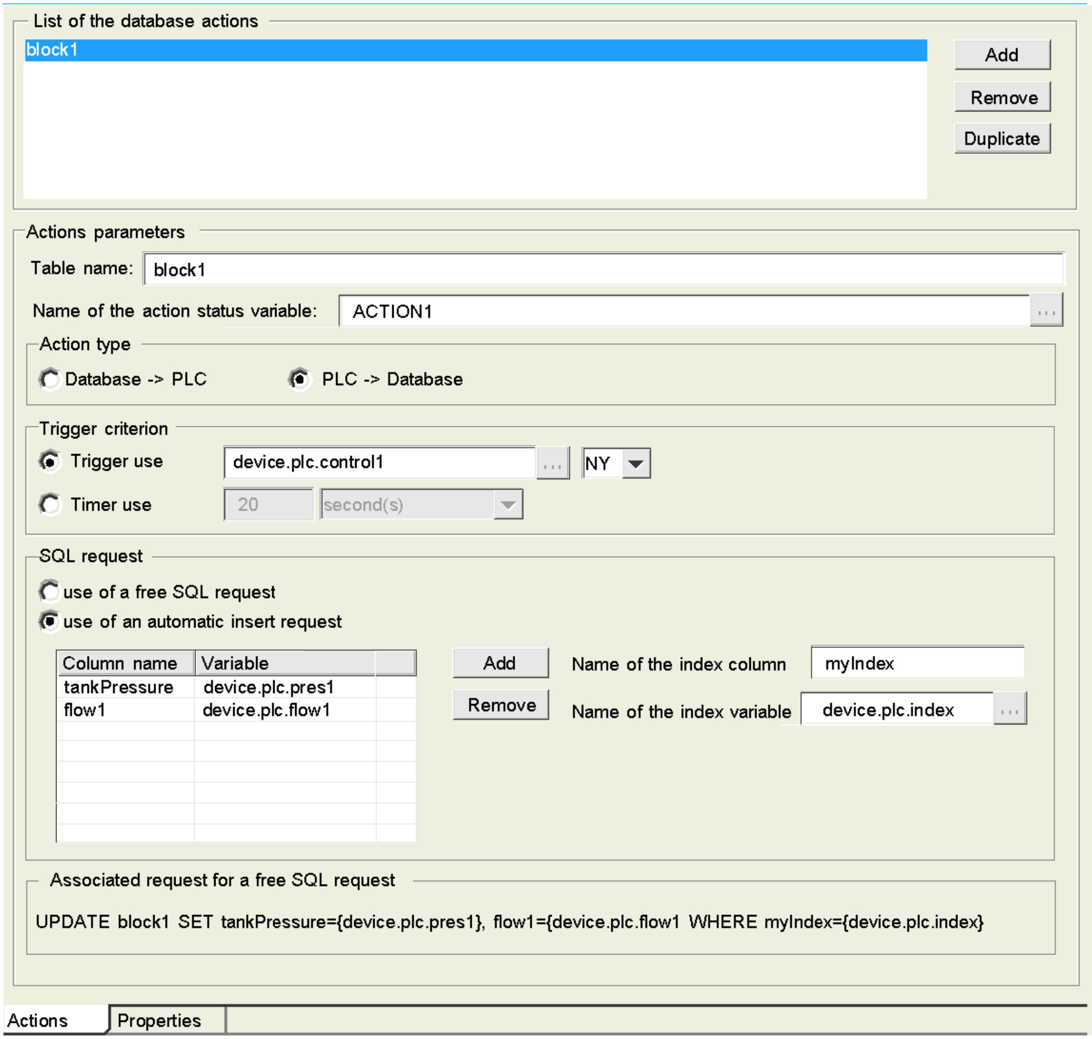
Field	Function	
Name of the status action variable	List the actions already configured.	
Action parameters		
Table name	Type a name to identify the action.	
Action status variable	Type the variable name that will represent the action status.	
Action type		
Database -> PLC	Initialization of your system variable with the value of the database.	
PLC -> Database	Save the current state of the system in the database.	
Trigger criterion		
Trigger use	Specify the variable that will trigger the action.	
	NY: Notify	Triggered on a bit or word value state change.
	RE: Rising Edge	Triggered on a rising edge for a bit, or a value increase for a word.
	FE: Falling Edge	Triggered on a falling edge for a bit, or a value decrease for a word.
	BQ: Bad Quality	Triggered if the trigger status is "Bad quality".
Timer use	Specify the period at which the action will be triggered.	
SQL request		
Use a free SQL request	Type your SQL request using the SQL syntax. In this case the screen looks like this:  SQL request ● use of a free SQL request ○ use of an automatic insert request Free SQL request	

Field	**Function**
Use of an automatic request	This function enables you to update tables without using a SQL syntax. Refer to the following paragraph for more information. Use the table provided for this purpose specifying the name of the column and the type of the variable. In this case, the column name syntax must comply with the database syntax. As a general rule, avoid using punctuation marks, upper case letters and hyphens (refer to your database documentation). **Note:** The name of the action and the name of the table are identical if you use Automatic Request therefore you cannot read and write to the same table in that case. Use the free SQL request to precise the name of the table.
Associated request for a free SQL request	This part displays the SQL request syntax corresponding to the automatic request you have entered.

Automatic Request Example

In this example, you will update the block1 table in the database with PLC values when the state of the device.plc.control1 variable changes.

The following figure shows you the window that corresponds to the example:



List of the database actions

block1

Actions parameters

Table name: block1

Name of the action status variable: ACTION1

Action type

○ Database -> PLC ● PLC -> Database

Trigger criterion

● Trigger use device.plc.control1 NY

○ Timer use 20 second(s)

SQL request

○ use of a free SQL request

● use of an automatic insert request

Column name	Variable
tankPressure	device.plc.pres1
flow1	device.plc.flow1

Name of the index column myIndex

Name of the index variable device.plc.index

Associated request for a free SQL request

UPDATE block1 SET tankPressure={device.plc.pres1}, flow1={device.plc.flow1 WHERE myIndex={device.plc.index}

Step	Action
1	Create a recipe database service.
2	Click the **Actions** tab.
3	Type the name of the table you want to update (i.e. block1) in the **Table name** field.
4	Specify a variable that will represent the action status (i.e. ACTION1) in the **Name of the action status variable** field.
5	Select Database -> PLC in the **Action type** area.
6	Select use of a trigger in the **Trigger criterion** area and select the variable that will trigger the action by clicking ... . You also have to specify the type of event that will trigger the action (i.e. NY).
7	Select use of an automatic request.
8	Click **Add** and type tankPressure for the column name. The column name syntax must comply with the database syntax.
9	Select the corresponding variable (i.e. device.plc.pres1) by clicking ... on the left side of the **Variable** field.
10	Redo step 8 and 9 to add the values of device.plc.flow1 in the flow1 column.
11	Specify a name for the index of the table and select a variable to store the index value.

Limitations

Web Designer for FactoryCast HMI only manages a single recipe database service.

The maximum number of tables is 20 per service.

The maximum number of columns is 50 per table.

Service Status

Presentation

The main purpose of the service status is to provide feedback on the services.

The status value of the service (email, plc, database, calculation...) is copied to a device variable using the Service properties screen. By default, each service has a FactoryCast HMI module internal status variable named as follows: "service type.service name.sstatus" (e.g. device.plc1.sstatus).

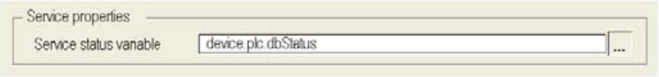
This status variable can be used to send out an email (alarm), to trigger archiving to a database, for management (sending alarm in supervision mode) or for any other action possible in the FactoryCast HMI application.

Example:

To send an email with the database service, you must:

- declare a dbStatus variable in the device and assign it to an address (integer, for example), then import this variable into the device service,
- define the dbStatus variable to trigger an email in the email service.

The service property configuration screens all have a ‘service status variable’ field. An illustration of what it looks like and how it is configured is shown below:



Service properties

Service status variable: device.plc.dbStatus

Simply click on... to display a variable selection window which can be configured, by selecting from a list of INT, UINT, SHORT and USHORT type.

The status values are:

- 0 for a stopped service,
- 1 for a service that is running,
- ≥2 for a service on which an anomaly has occurred,

The codes are assigned in ascending order of severity.

In case of 2 events, the status value corresponds to the most important one.

When a new action is operational (log, creation of a table, etc.), the service is assigned the value 1 to inform you that the service is operating correctly.

Each service has its own codes.

Device Service Codes

Value	Comment
0	The service is stopped.
1	The service is running.
2	The PLC application’s signature is different from the one contained in the FactoryCast HMI application for the device service.

NOTE: Value 2 will only be accessible from the service’s internal status variable (i.e. device.plc.SSTATUS). This value will not be copied to the device variable (except if the variable belongs to another accessible device).

RecipeDB Services Codes

Value	Comment
0	The service is in stop mode.
1	The service is in run mode.
20	Log in table delayed.
21	No access to table.
22	Table no longer exists.
23	Database connection anomaly.
24	Incorrect service configuration.
25	Fatal detected error.

NOTE:

The value 20 should be considered to be a system message, not an alarm. It may occur when:

- the service is started (value not ready),
- the PLC has not had the time to send the data as part of the request,
- the signature of the PLC application is different to that of the FactoryCast HMI application.

The high values take priority. Code 23 has priority over code 21. When code 23 has been corrected, the value 21 can then be sent to the variable declared for the status of the service.

Database Services Codes

Value	Comment
0	The service is in stop mode.
1	The service is in run mode.
20	Log in table delayed.
21	Access to table.
22	Table no longer exists.
23	Database connection anomaly.
50	Buffer is full.
100	Anomaly database configuration.
200	Database undefined.
1000	Fatal detected error.

NOTE: Code 50 : buffer full - The database buffer is full, the new events are not stored anymore. A reconnection of the database or the ethernet cable is required to send the values to the database and store new coming values.

NOTE:

The value 20 should be considered to be a system message, not an alarm. It may occur when:

- the service is started (value not ready),
- the PLC has not had the time to send the data as part of the request,
- the signature of the PLC application is different to that of the FactoryCast HMI application.

The high values take priority. Code 23 has priority over code 21. When code 23 has been corrected, the value 21 can then be sent to the variable declared for the status of the service.

Email Service Codes

Value	Comment
0	The service is in stop mode.
1	The service is in run mode.
10	Email refused by SMTP server.
11	Email not sent, due to a connection interruption.
12	No more connections or queue saturated.

NOTE: The high values take priority. The value 12 has priority but becomes 11 subsequently. Code 11 disappears once the email has been sent correctly.

Datalogging Service Codes

Value	Comment
0	OK.
2	The current file ‘.csv’ is renamed ‘.cs0’.
10	The module cannot reach a variable that should be logged because: ● the variable does not appear in the Web Designer namespace. ● the variable is not write enabled. ● the preceding value has not been updated yet.
11	FTP transfer interruption.
12	The URL specified for FTP cannot be accessed.
13	The internal flash is full
14	The internal RAM is full.
15	The module cannot write on the media specified.
16	The module cannot access the namespace.
17	The maximum number of tables has been reached (10 maximum) in your *.xml* file.
18	The maximum number of variables has been reached in a table.
19	The service is empty, no table defined.

NOTE:

- The value 2 should be considered as a warning, not an alarm.
- The high values take priority. For example: code 17 has priority over code 12. When code 17 has been corrected, the value 12 can then be sent to the variable declared for the status of the datalogging service
- If all tables status values are 0, the service status is set to 1 (for running state).
- The values of the table status variable are identical to the values of the datalogging service status variable.

Recipe Service Codes

Value	Comment
0	The service is in stop mode.
1	The service is in run mode.
60	Detected error in recipe configuration.
61	Detected error when writing the value.
62	Communication error.
63	Error in recipe file.

Codes for Other Services

The other services (calculation, active pages) do not have any specific values. The code only specifies whether the service is stopped (0) or running (1).

Value	Code	Comment
0	Stop	The service is in stop mode.
1	Run	The service is in run mode.

2.2 Device Explorer

Scope of this Section

The Device Explorer feature enables you to display status and diagnostic information of Ethernet devices connected to your FactoryCast product.

What's in this Section?

This section contains the following topics:

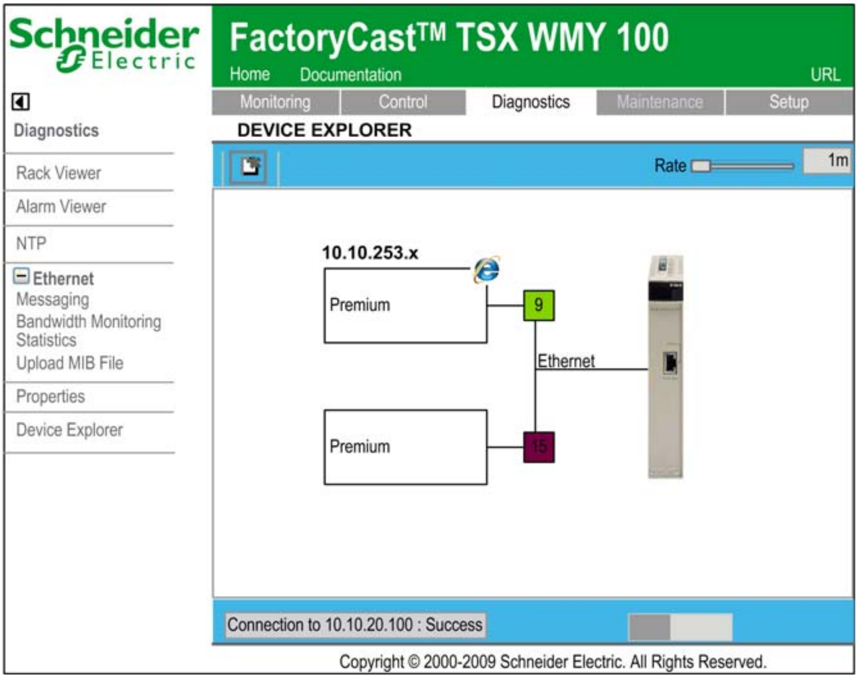
Device Explorer Presentation

Presentation

FactoryCast modules integrate a proxy function. This feature enables you to access status and diagnostic information of devices connected to the module.

Device Explorer Page

The Device Explorer page looks like this in the animation mode:



Connection to 10.10.20.100 : Success

Copyright © 2000-2009 Schneider Electric. All Rights Reserved.

In the animation mode, the FactoryCast module sends an identification request to each devices on the network (cyclic check).

The device appears inoperative in the animation page if:

- it does not answer to the identification request.
- the response to the animation request is negative.

Devices that are in error appear in red otherwise they appear in green.

Accessing the Device

By double clicking the IE icon on the rectangles that symbolize the devices, the website of the device appears in a new tab of the Web browser.

NOTE:

- Your Web browser must manage tabs.
- The device must not be operational.

If the device is connected to the Ethernet network, the module launches a new page with the IP address of the device as URL.

Otherwise, the module acts as a proxy. It launches the website as in the FactoryCast module with its address as a parameter. It is the same website as in the FactoryCast module except that the setup menu is unavailable.

Device Explorer Configuration

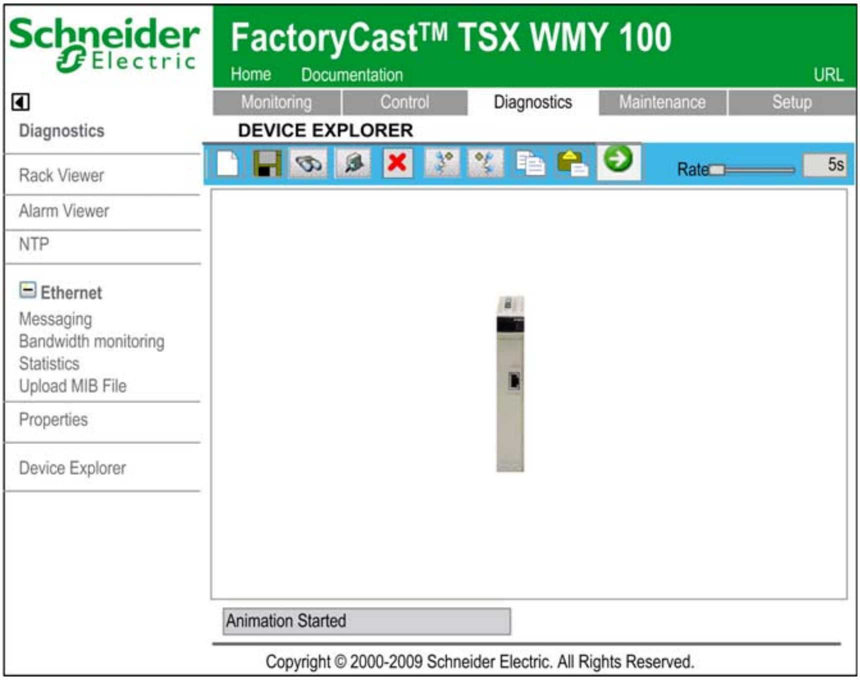
Presentation

To use the Device Explorer feature, you must first define your network configuration.

Accessing the Device Explorer Configuration Page

Step	Action
1	Click the **Diagnostics** link on the Home page.
2	Click the **Device Explorer** link on the diagnostics page.
3	Click to access the configuration.

View of the Device Explorer Configuration Page

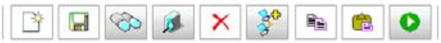


Animation Started

Copyright © 2000-2009 Schneider Electric. All Rights Reserved.

Description of the Toolbar

The toolbar enables you to access the main functions of the configuration page directly by clicking its icons.



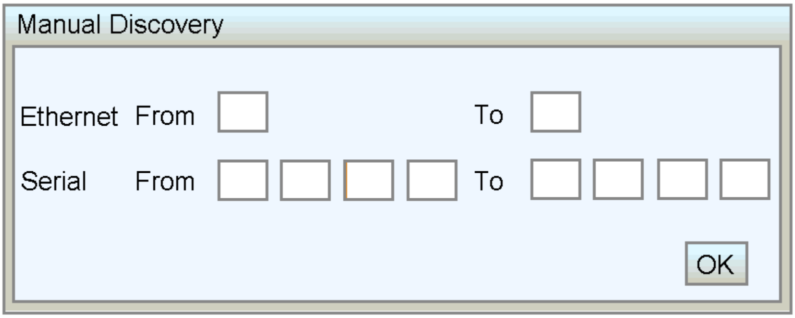
The following table describes the elements of the toolbar:

Button	Function
📄	Access the edition mode.
💾	Save the current network configuration.
🔭	Automatic discovery of the whole network.
🔎	Manual discovery of the network. You must specify the range of addresses that will be scanned.
✕	Delete the selected device.
➕	Add a new device on the Ethernet link.
📑	Copy the selected object to the clipboard.
📋	Paste the clipboard.
▶	Launch the animation mode.

Discovery of the Devices

The Device Explorer applet enables you to define how to scan the network for discovery of the connected devices:

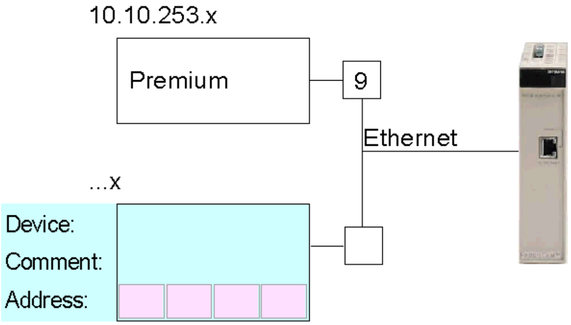
- Click to scan the whole network. The scan starts at IP address = x.x.x.1 and it stops at IP address = x.x.x.254.
- Click to scan a specified range of device addresses. The **Manual Discovery** window appears in which you can define the range of addresses to be scanned.



NOTE:

- The automatic discovery is a process that can take time: around 1 s for an Ethernet device.
- The box in the bottom left corner informs you on the status of the scan. It also displays the address that is currently being scanned.

Manual Edition of the Devices

Step	Action
1	Click to add a device on the Ethernet link. **Result**: An empty box representing the device appears.  10.10.253.x Premium 9 Ethernet ...x Device: Comment: Address:
2	You can add a comment on the device in the **Comment** area.
3	Specify the address of device in the **Address** area (mandatory).
4	Click somewhere else in the window to validate your change.

Modifying and Removing a Device

To modify the properties of a device, simply double click on it.

To remove a device, select it by clicking on it and click ✕ .

Web Site Security

3

Overview

FactoryCast modules must not be used to support critical functions. Before configuring your Web site develop a security plan. Unlike default Web site data, data from a custom Web site is write enabled.

You must pay particular attention to the people with access rights to the site and to the data that can be changed. This chapter describes security events and presents security mechanisms accessible to users of the Web utility.

What's in this Chapter?

This chapter contains the following topics:

Internal Security

Overview

FactoryCast products provide two mechanisms so that only authorized users view and modify your data.

- Password entry
- Access restrictions

⚠ WARNING

UNINTENDED OPERATION

Limit embedded server access to qualified personnel:

- Change passwords monthly
- Do not use simple user names and passwords.
- Disable default passwords before commissioning the module

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Password Protection

Although you may add unprotected Web pages to the site, the default Web pages and any other pages you choose to protect can only be viewed by users who supply the correct user name and password.

Restricting Access to Web Pages

When you create a website and you want to protect it, you must place it in the folder called *secure*. The uploading of the custom website is subject to security conditions linked to a FTP password.

External Security

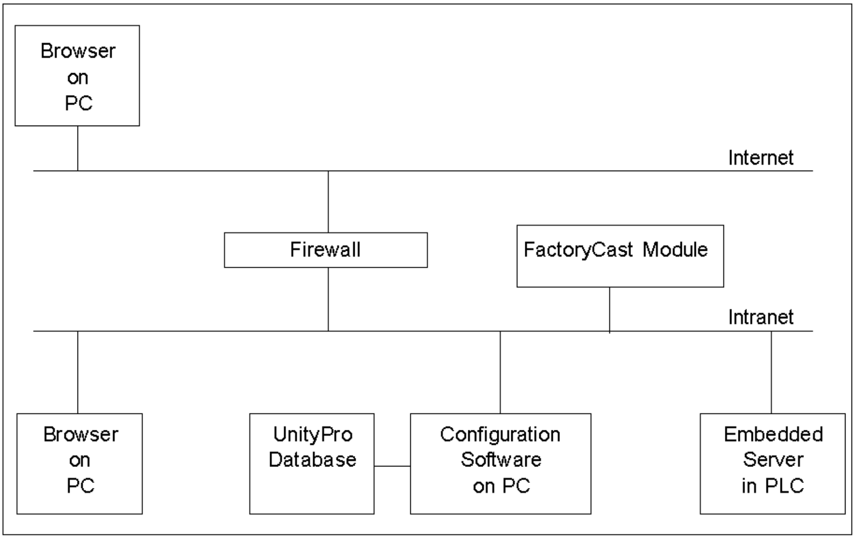
Overview

If your network has been configured to enable users to consult your Internet site, your internet site, your security system is the same as that of an intranet site, only you have an additional security measure: a firewall.

Architecture of a Firewall

A firewall forms a gateway between Internet and your embedded server. You can use a firewall to restrict or forbid access to your website.

A firewall can be configured to authorize network connections to a limited range of ports, or to authorize traffic to or from certain IP addresses.



Types of Firewalls

There are two types of firewalls:

- Network firewalls
- Application firewalls

Network Firewalls

Network firewalls are often installed between the Internet and a single entry point to an intranet or internal protected network.

Application-Level Firewalls

An application firewall works for an application, for example FTP. It intercepts all traffic sent to this application, and decides whether or not to transmit this traffic to the application. Application firewalls are located on individual host computers.

Firewall Configuration

Web Designer uses HTTP, FTP and Schneider Electric Modbus application protocol (MBAP) to access embedded server pages and files. If you want viewers to be able to access your site from the Internet and your embedded server is protected by a firewall, you must configure the firewall to authorize HTTP, FTP and MBAP traffic.

Port	Protocol	Access to...
21	FTP	Protected embedded server files
Higher than 1024		
80	HTTP	Web pages
502	MBAP	Operational data

NOTE:

- The default FTP name and password are USER/USER.
- The FactoryCast client follows the "Firewall Friendly FTP" standard, RFC 1579. It issues an FTP PASV command to the FactoryCast server before any attempt to establish an FTP data connection.
- The online mode of the configuration tool is not operational if the module is protected by a firewall. The ports in this mode are dynamically assigned.

Variable Access Security, Symbol, Direct Address

Presentation

Users who enter the Data Editior write password can modify only variables (symbols) and direct addresses that are write-enabled. When you create a Web-enabled database of variables and direct addresses, you can designate each element as read-only or write-enabled.

Unauthorized or incorrect modifications made to symbols and direct addresses may have undesirable effects or even change the behavior of your application in ways that may be unintended or even hazardous.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Limit access to qualified personnel.
- Carefully select the symbols and direct addresses you authorize to be modified online.
- Do not authorize online modification of variables of critical nature concerning human and material integrity.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Default Web Site for Quantum

4

Overview

When you receive the Embedded Server, it already contains a default Web site with diagnostic, monitoring and setup pages.

You may view these pages and view variables in the editors simply by installing the module and configuring its IP address. To access the site, type the IP address of the module in your browser and enter the default user name and password (USER/USER).

This section describes the pages in the default Web site.

What's in this Chapter?

This chapter contains the following sections:

4.1 Quantum Home Pages

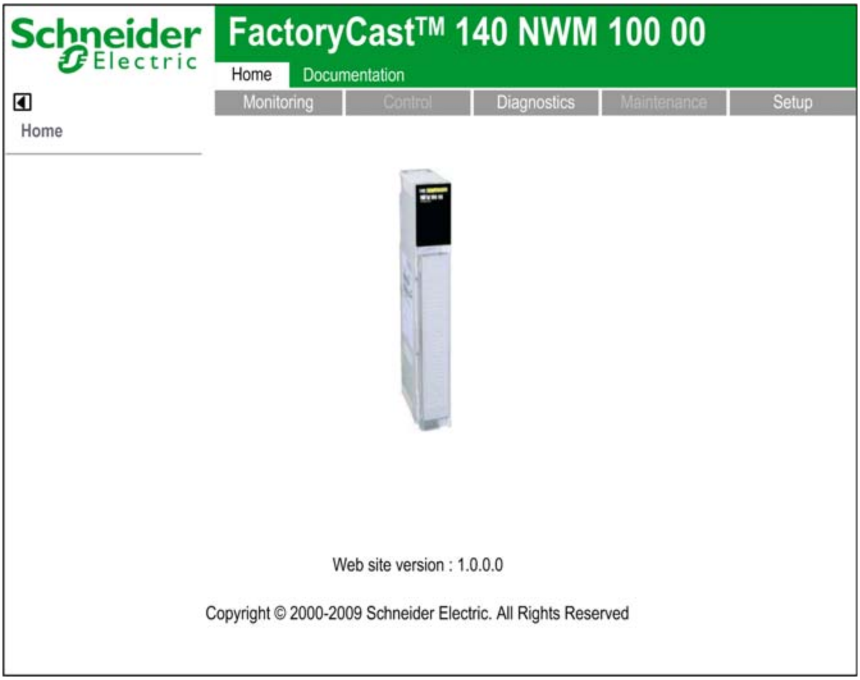
Quantum Home Page

Presentation

You can access the Quantum home page by entering the IP address of the module. No password is required to display this page.

Home Page

The Quantum home page looks like this.



Web site version : 1.0.0.0

Copyright © 2000-2009 Schneider Electric. All Rights Reserved

Links

From the Quantum home page, you can access the following pages:

- Monitoring,
- Diagnostics,
- Setup,
- Documentation for access to the product documentation in HTML format.

4.2 Diagnostics

Subject of this Chapter

The diagnostics link is used to obtain specific diagnostics on the operating status of the module and the CPU.

What's in this Section?

This section contains the following topics:

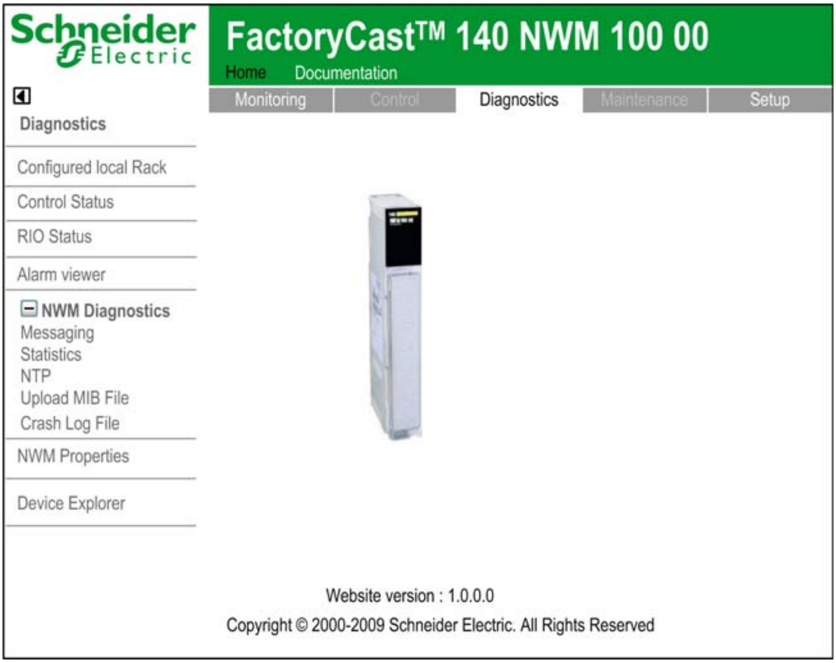
140 NWM 100 00 Module Diagnostics Home Page

Presentation

This page contains a list of links for accessing the different diagnostics pages of the module:

- Configured local Rack for viewing the rack,
- Control Status for viewing the status of the CPU,
- RIO Status for viewing the status of the remote IO,
- Alarm Viewer displays detected application faults *(see page 287)*,
- Messaging for the Messaging service,
- Statistics for an Ethernet Statistics service,
- Upload MIB file for uploading the MIB,
- Crash Log File to diagnose the cause of a crash,
- NWM properties for consulting the module properties.
- Device Explorer *(see page 86)* gives you status and diagnostic information about the devices connected to the module.

View of the Diagnostic Services Home Page



Website version : 1.0.0.0

Copyright © 2000-2009 Schneider Electric. All Rights Reserved

Click on a link to access the desired diagnostics page.

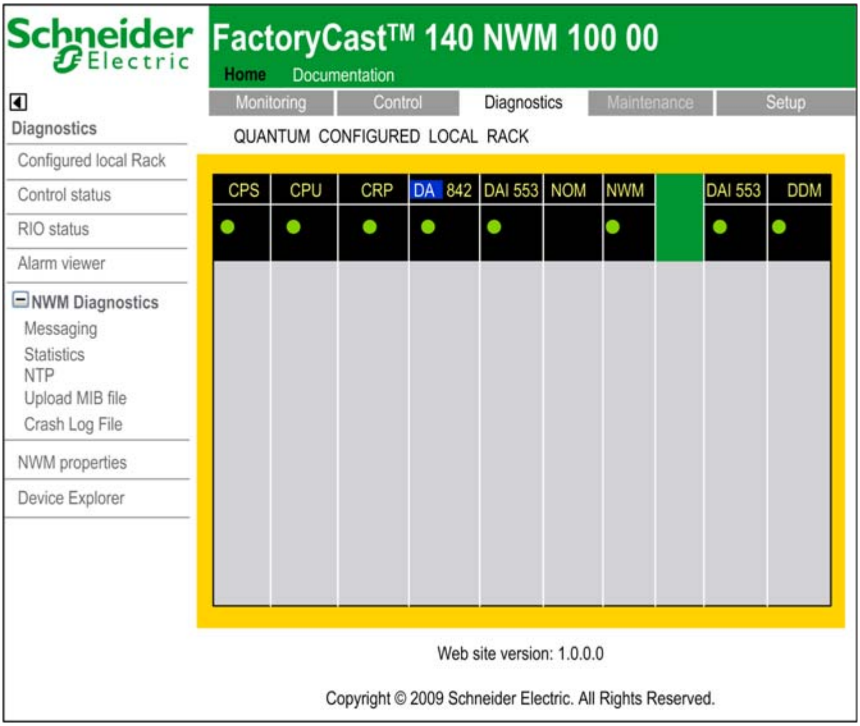
Local Rack Configuration Page

Overview

The Quantum Configured Local Rack page displays the current configuration of the local rack, including the controller, Embedded Server module and any I/O modules. The rack may use up to 16 slots.

Example

The following diagram shows an example of configured local rack page.



QUANTUM CONFIGURED LOCAL RACK

CPS	CPU	CRP	DA 842	DAI 553	NOM	NWM		DAI 553	DDM
●	●	●	●	●		●		●	●

Web site version: 1.0.0.0

Copyright © 2009 Schneider Electric. All Rights Reserved.

Data

Each module is shown in the slot in which it is configured in the rack. The page provides the following information:

- The module type is shown at the top of the page. A question mark means that the module type is unknown or that the slot is empty.
- The module status is indicated by an LED below the module type:
  - A green LED indicates that the module is healthy.
  - A red LED indicates a module malfunction.

Links

When you click on a module, you are directed to another Web page which provides detailed information on the module.

The links located towards the bottom of the page can be used to access the other main diagnostics, data editor and graphic editor pages.

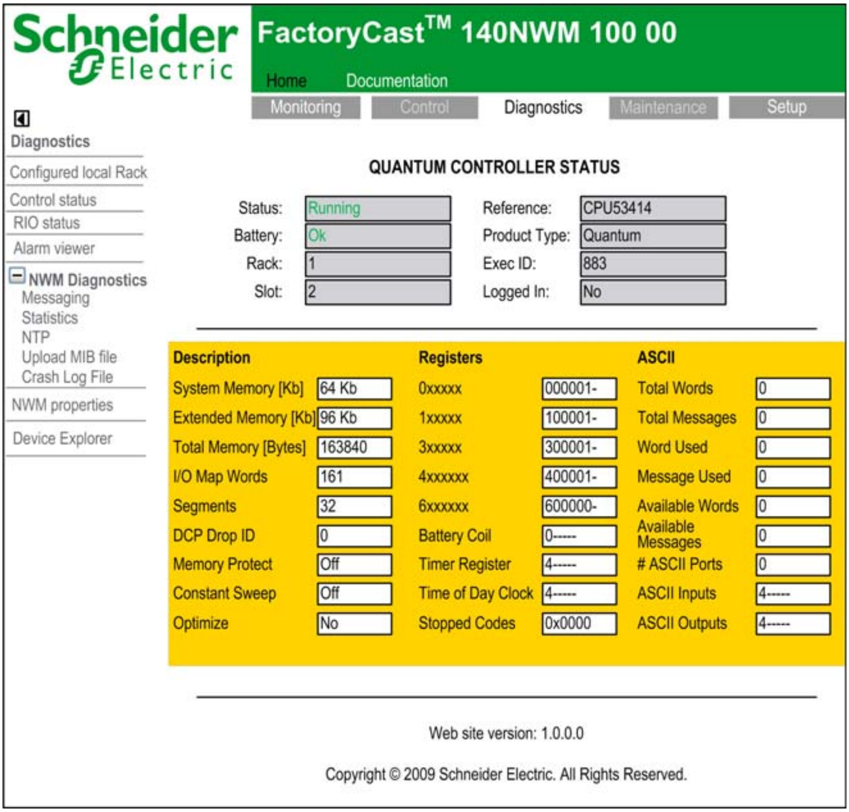
Processor Configuration Page

Overview

The Processor configuration page provides recent information on the controller and its configuration. You can access this page by selecting the model of processor from the configured Local rack page or by using the hypertext link at the bottom of the page.

Example

The following diagram shows an example of the Processor configuration page.



QUANTUM CONTROLLER STATUS

Status:	Running	Reference:	CPU53414
Battery:	Ok	Product Type:	Quantum
Rack:	1	Exec ID:	883
Slot:	2	Logged In:	No

Description		Registers		ASCII	
System Memory [Kb]	64 Kb	0xxxxx	000001-	Total Words	0
Extended Memory [Kb]	96 Kb	1xxxxx	100001-	Total Messages	0
Total Memory [Bytes]	163840	3xxxxx	300001-	Word Used	0
I/O Map Words	161	4xxxxxx	400001-	Message Used	0
Segments	32	6xxxxxx	600000-	Available Words	0
DCP Drop ID	0	Battery Coil	0-----	Available Messages	0
Memory Protect	Off	Timer Register	4-----	# ASCII Ports	0
Constant Sweep	Off	Time of Day Clock	4-----	ASCII Inputs	4-----
Optimize	No	Stopped Codes	0x0000	ASCII Outputs	4-----

Web site version: 1.0.0.0

Copyright © 2009 Schneider Electric. All Rights Reserved.

Dynamic Data

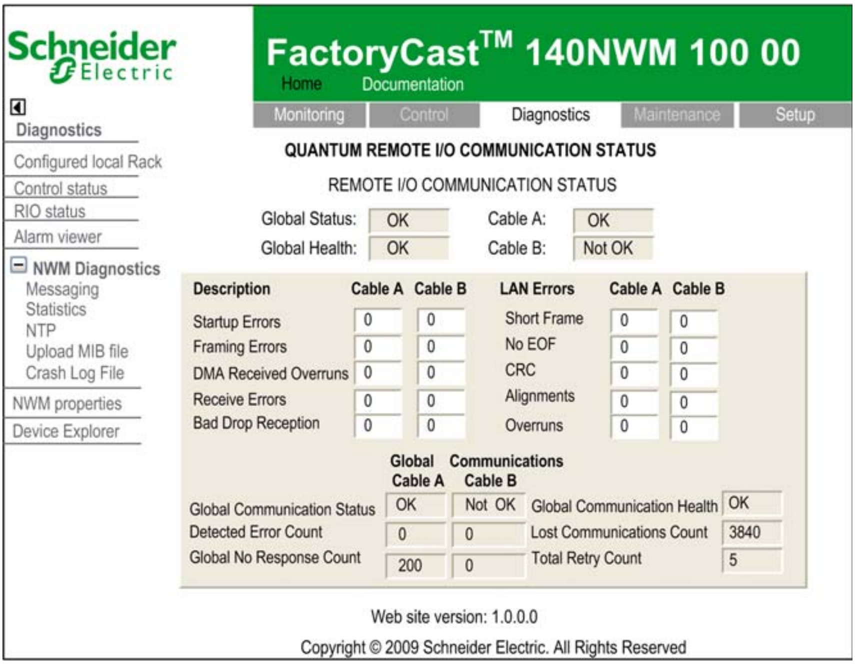
Some data items on this page are dynamic. They are updated periodically at a frequency determined by the performance of the embedded server, the network and the client processor.

Remote I/O Status Page

Presentation

The remote I/O status page provides diagnostics about communication with the remote devices.

View of the Remote I/O Status Page



QUANTUM REMOTE I/O COMMUNICATION STATUS

REMOTE I/O COMMUNICATION STATUS

Global Status:	OK	Cable A:	OK
Global Health:	OK	Cable B:	Not OK

Description	Cable A	Cable B	LAN Errors	Cable A	Cable B
Startup Errors	0	0	Short Frame	0	0
Framing Errors	0	0	No EOF	0	0
DMA Received Overruns	0	0	CRC	0	0
Receive Errors	0	0	Alignments	0	0
Bad Drop Reception	0	0	Overruns	0	0

	Global	Communications		
	Cable A	Cable B		
Global Communication Status	OK	Not OK	Global Communication Health	OK
Detected Error Count	0	0	Lost Communications Count	3840
Global No Response Count	200	0	Total Retry Count	5

Web site version: 1.0.0.0

Copyright © 2009 Schneider Electric. All Rights Reserved

For further information on the status of remote I/Os, refer to the Quantum Remote I/O Manual,.

Ethernet Module Diagnostic Pages

Presentation

This page contains a series of links for accessing the different diagnostics pages for the Ethernet module:

- Messaging (Messaging utility),
- Statistics (Ethernet module statistics),
- NTP (NTP service status),
- Upload MIB File (upload the module MIB source file),
- Crash Log File (display the log of crashes and their causes).

Messaging Page

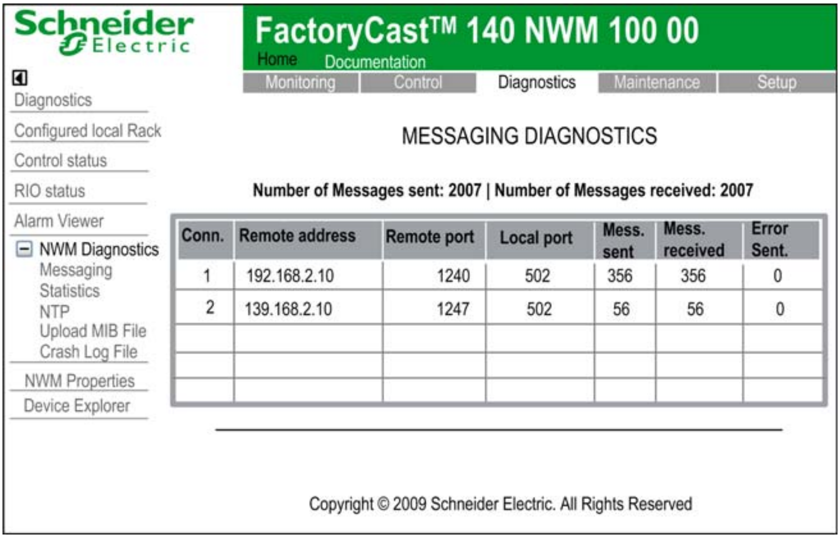
This page provides current information on the open TCP connection on port 502.

The number of messages sent/received on the port can be found at the top of the page.

A table provides, for each connection (numbered from 1 to 64):

- the remote IP address,
- the remote TCP port,
- the local TCP port,
- the number of messages sent from this connection,
- the number of messages received from this connection,
- the number of anomalies on this connection.

The following figures shows you the messaging diagnostics page:



MESSAGING DIAGNOSTICS

Number of Messages sent: 2007 | Number of Messages received: 2007

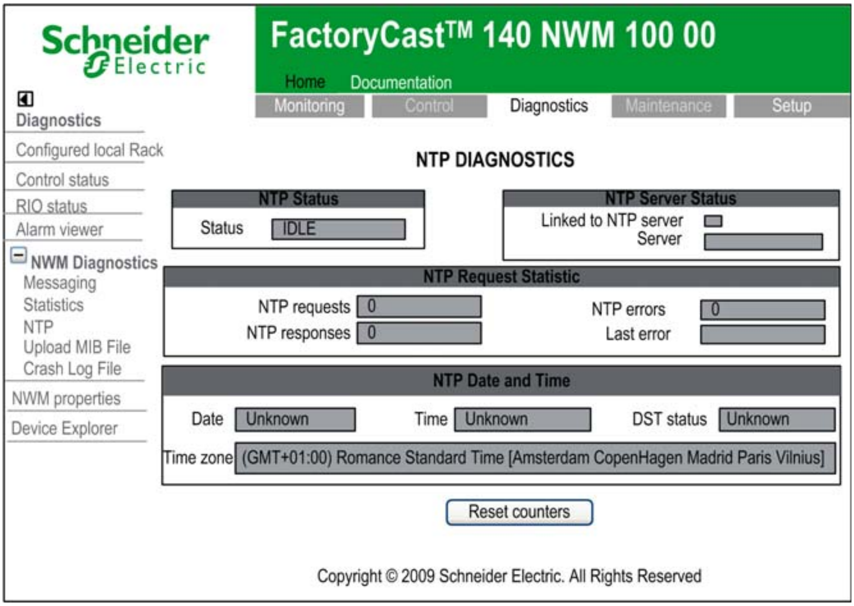
Conn.	Remote address	Remote port	Local port	Mess. sent	Mess. received	Error Sent.
1	192.168.2.10	1240	502	356	356	0
2	139.168.2.10	1247	502	56	56	0

Copyright © 2009 Schneider Electric. All Rights Reserved

Ethernet Module Statistics Page

The Ethernet module statistics page provides information on status, send and receive statistics and system messages for the embedded server module. You can access this page by selecting the NWM module from the local rack or by using the hypertext link at the bottom of the page.

The following figure shows an example of the Ethernet Module Statistics page.



NTP DIAGNOSTICS

NTP Status

Status: IDLE

NTP Server Status

Linked to NTP server: □

Server:

NTP Request Statistic

NTP requests: 0

NTP responses: 0

NTP errors: 0

Last error:

NTP Date and Time

Date: Unknown

Time: Unknown

DST status: Unknown

Time zone: (GMT+01:00) Romance Standard Time [Amsterdam CopenHagen Madrid Paris Vilnius]

Copyright © 2009 Schneider Electric. All Rights Reserved

NOTE: Refer to the following documents for a definition of terms:

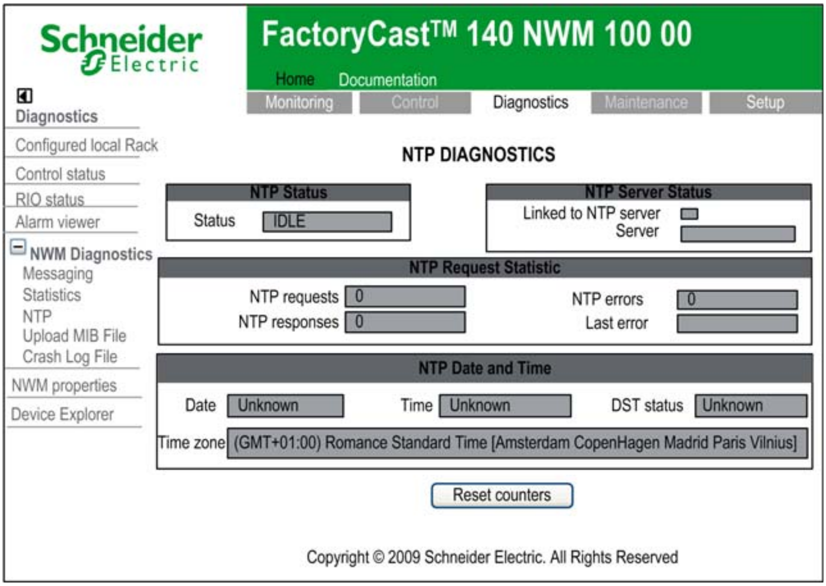
- *Modicon Quantum - Ethernet TCP-IP Module - User Guide* 840 USE 107 00.
- *Modicon Quantum - Ethernet TCP-IP Module - User Guide 840* USE 115 00.

NTP Diagnostics

This page enables you to diagnose the NTP service. It shows you the status of the link with the server as well as the status of the server. It also provides you statistics about the link.

NOTE: Schneider Electric is not responsible of hour veracity provided by the NTP server.

The following figure shows an example of the NTP diagnostics page.



NTP DIAGNOSTICS

NTP Status

Status: IDLE

NTP Server Status

Linked to NTP server:

Server:

NTP Request Statistic

NTP requests: 0

NTP responses: 0

NTP errors: 0

Last error:

NTP Date and Time

Date: Unknown

Time: Unknown

DST status: Unknown

Time zone: (GMT+01:00) Romance Standard Time [Amsterdam CopenHagen Madrid Paris Vilnius]

Copyright © 2009 Schneider Electric. All Rights Reserved

The following table describes the elements of the NTP statistics page:

Text	Description
NTP Status	Indicates the status of the NTP service: ● unconfigured, ● configured.
NTP server status	● Indicates the status of the NTP server: ● OK if the link is available, ● NOK if the link is not available. ● Indicates the NTP servers availability: ● NONE if there is no server, ● PRIMARY give the name of the primary NTP server, ● SECONDARY give the name of the secondary NTP server.
NTP request statistics	● Number of requests to the NTP server, ● Number of anomalies, ● Number of responses from the NTP server, ● Last anomaly.
NTP date and time	● DST status: daylight saving time can be configured automatically or not. ● Date: format Day Month Year. ● Time: format Hour Minute Second.
Time zone	Name of the zone: GMT +/- X.
Reset Counters	Click this button to set all the counters to 0.

MIB File

The Upload MIB File link is used to upload the module MIB file to a directory on a PC connected to the 140 NWM 100 00.

Crash Log Page

The Crash Log File link is used to display entries in the crash log file which enable you to diagnose the cause of a crash. The Crash Log Diagnostics page displays a crash log file in the event of a crash or a status message if no crash has occurred.

Click on Erase crash log file to erase the file.

Diagnostic resources monitoring for Premium

Presentation

This page contains a list of links for accessing the diagnostic resources monitoring:

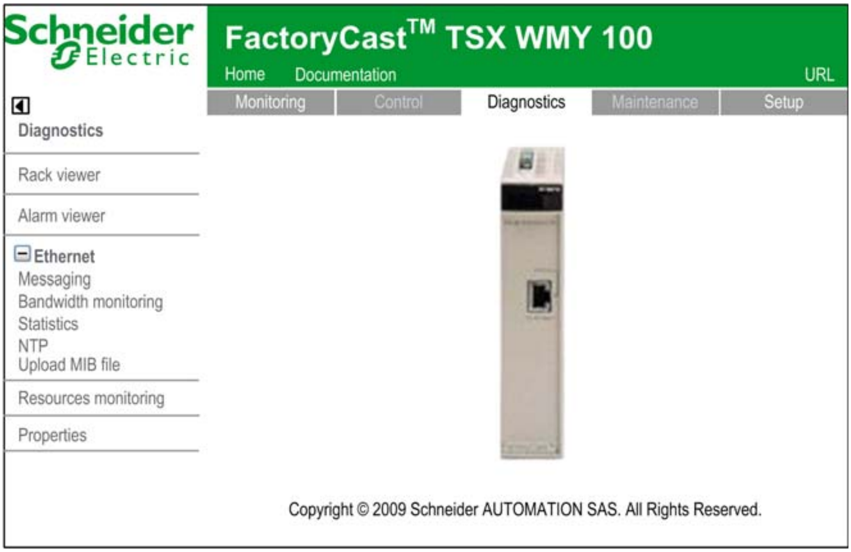
- Rack viewer
- Alarm viewer
- Messaging service
- Bandwidth service
- Ethernet statistics service
- Properties
- NTP
- Upload MIB file

The following table describes the links for diagnostic resources monitoring:

Links	Description
Rack viewer	The Rack viewer page shows the current configuration of the racks, in particular the PLC, the embedded server module and the I/O modules.
Alarm viewer	The Alarm viewer Premium and Quantum PLCs.
Messaging	The Messaging provides current information on the open TCP connection on the port.
Bandwidth Monitoring	The Bandwidth displays the load distribution of the TSX WMY 100 module between the global data services, I/O scanning, messaging and other services.
Ethernet statistics	The Ethernet statistics provides all the lastest information on the status, configuration and functions of the embedded server module.
NTP	The NTP (Network Time Protocol) synchronizes the clocks of computer systems.
Upload MIB file	The Upload MIB file upload the module MIB file to a directory on a PC connected to the TSX WMY 100.
Properties	The properties tab provides information on: ● The software version ● The version of web pages ● The physical properties of the module (Ethernet properties)

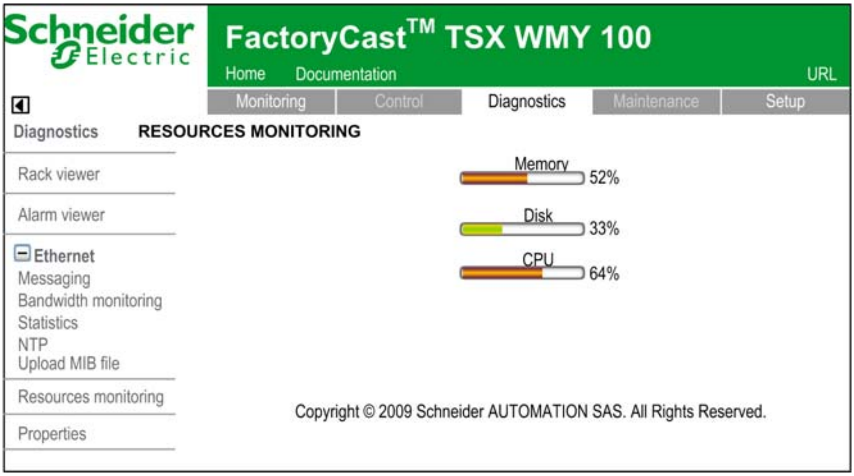
Resources monitoring page

The following figures shows a new resource monitoring diagnostics menu:



Copyright © 2009 Schneider AUTOMATION SAS. All Rights Reserved.

Memory available page:



RESOURCES MONITORING

Memory 52%

Disk 33%

CPU 64%

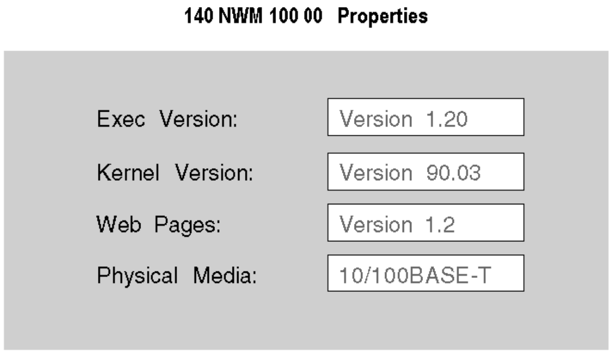
Copyright © 2009 Schneider AUTOMATION SAS. All Rights Reserved.

- **Memory**: Estimation of the RAM load
- **Disk**: Estimation of the Flash space usage
- **CPU**: Estimation of the CPU usage

Properties Page

Presentation

The figure below shows the 140 NWM 100 00 Properties Page. It contains the following information: the version of the executable, the kernel, the Web pages and the physical media used.



140 NWM 100 00 Properties

Exec Version: Version 1.20

Kernel Version: Version 90.03

Web Pages: Version 1.2

Physical Media: 10/100BASE-T

NOTE: The content of the NWM Properties Pages is fixed.

4.3 Setup

Subject of this Chapter

The ‘Setup’ link is used to configure the module.

What’s in this Section?

This section contains the following topics:

Setup Home Page

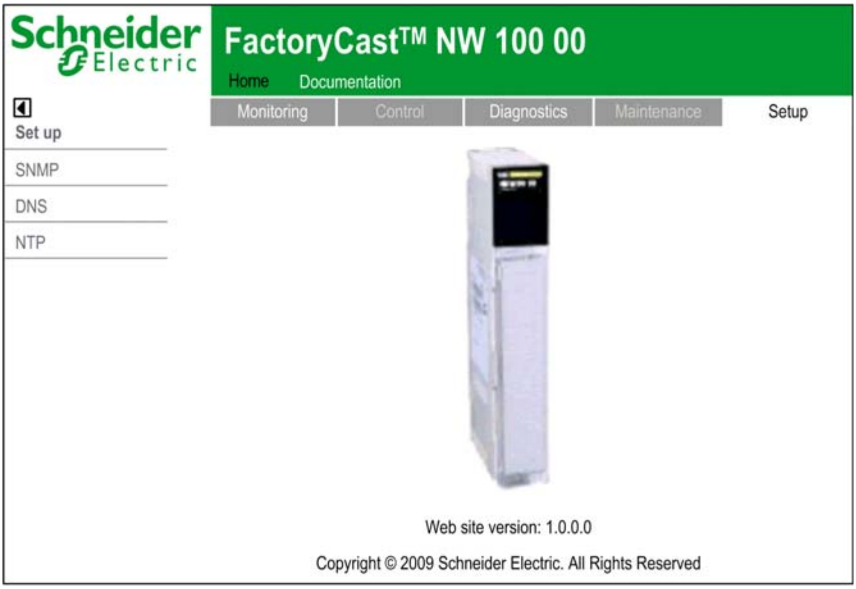
Presentation

This page contains a list of links for accessing the different Setup pages of the module:

- SNMP for configuring the SNMP service,
- DNS for configuring the DNS service *(see page 140)*,
- NTP for vconfiguring the NTP service *(see page 139)*.

Click on a link to access the desired setup page.

View of the Setup Services home page:



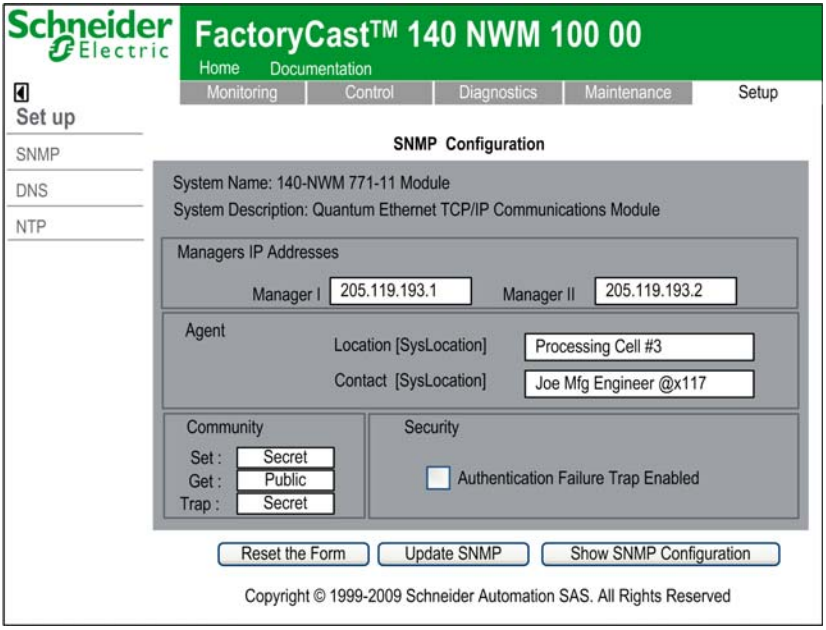
Web site version: 1.0.0.0

Copyright © 2009 Schneider Electric. All Rights Reserved

SNMP Configuration Page

Presentation

The SNMP configuration may already be defined on your 140 NWM 100 00. If it is not, fill out the SNMP Configuration form shown below.



SNMP Configuration

System Name: 140-NWM 771-11 Module

System Description: Quantum Ethernet TCP/IP Communications Module

Managers IP Addresses

Manager I: 205.119.193.1
Manager II: 205.119.193.2

Agent

Location [SysLocation]: Processing Cell #3
Contact [SysLocation]: Joe Mfg Engineer @x117

Community

Set : Secret
Get : Public
Trap : Secret

Security

☐ Authentication Failure Trap Enabled

Copyright © 1999-2009 Schneider Automation SAS. All Rights Reserved

SNMP Configuration Page Fields

The following table describes the specific SNMP fields that can be modified.

Field	**Information to be provided**
Manager I	IP address of first SNMP manager
Manager II	IP address of second SNMP manager
Location [SysLocation] (Position [SysLocation])	Module location
Contact [SysContact]	Name of engineer responsible for system
Set	User level authorized to set the configuration
Get	User level authorized to view the configuration
Trap	User level authorized to trap information
Authentication Failure Trap Enabled	Enables authentication of community name
Show SNMP Configuration	Display current SNMP configuration
Reset the Form	Clear the fields
Update SNMP	Modify SNMP configuration

Completion Message

When you click on Update SNMP, a new page containing the message "The SNMP database has been updated" is displayed. This page contains the same links as those featured on the SNMP Configuration page.

NOTE: The 140 NWM 100 00 module must be reset for these changes to take effect.

4.4 Viewer

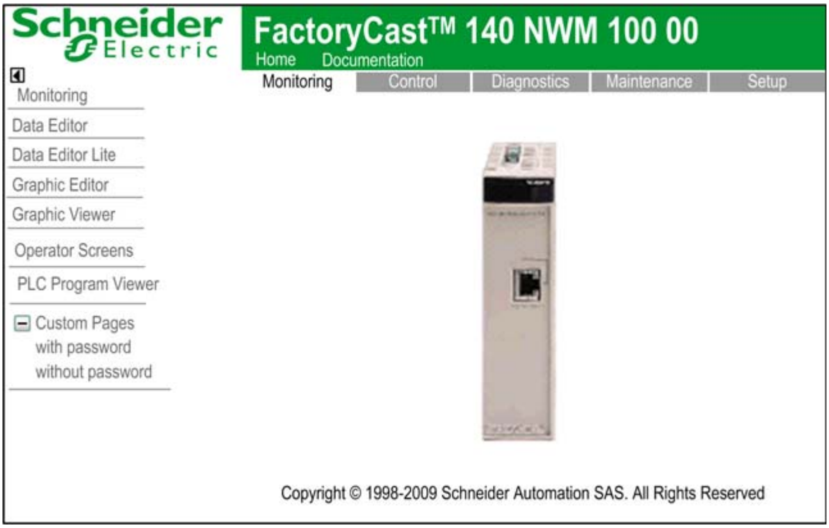
Monitoring Pages

Presentation

This page lists the various viewer services supported by the default web server of the module and provides links for accessing the services you require.

Link	**Description**
Data Editor	Create variable data tables, so as to be able to determine their value when the table is animated. See *(see page 144)*.
Data Editor Lite	Create variable data tables, with some limitations, so as to be able to determine their value when the table is animated. See *(see page 158)*.
Graphic Editor	Create graphics, so as to be able to determine the value of variables when the graphic is animated. See *(see page 169)*
Graphic Viewer	Display graphics, so as to be able to determine the value of variables when the graphic is animated. See *(see page 217)*
Operator Screens	Display graphic pages created with UnityPro. See *(see page 219)*
Custom Pages with Password	Display screen pages created by the user (accessible with password). See *(see page 225)*
Custom Pages without Password	Display screen pages created by the user (accessible without password). See *(see page 225)*

View of the Monitoring Hmoe Page



FactoryCast™ 140 NWM 100 00

Copyright © 1998-2009 Schneider Automation SAS. All Rights Reserved

Default Premium Web site

5

Subject of this Chapter

When you receive the TSX WMY 100 FactoryCast HMI module, it already contains a Web site by default.

To display these pages, simply install the module and configure its IP address. To access the site, enter the IP address of the module in your browser and the default username along with the "USER" password.

This chapter describes the pages of the default Web site.

What's in this Chapter?

This chapter contains the following sections:

5.1 Premium Home Pages

Home Page of TSX WMY 100 Module

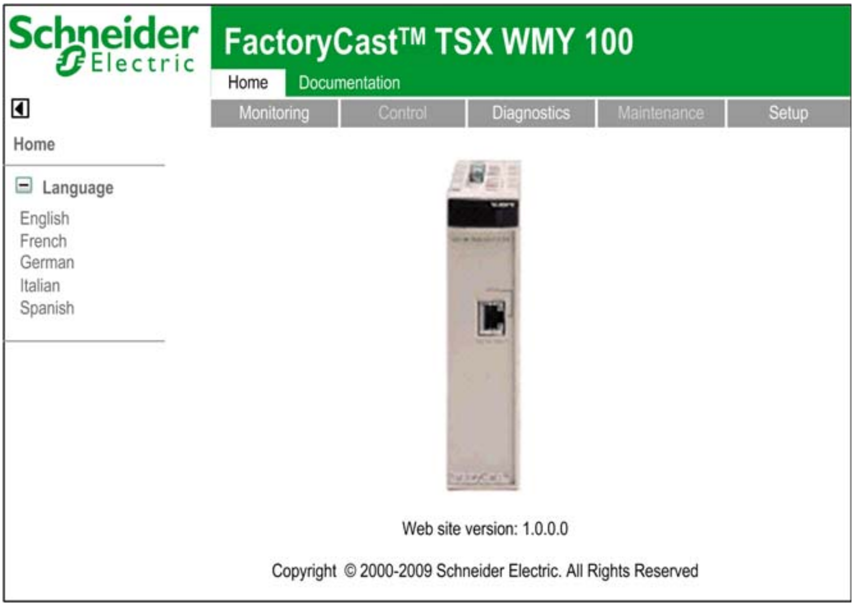
Presentation

The FactoryCast Web server home page of the TSX WMY 100 is the first page that the visitors sees on connection. No password is required to access this page.

The home page provides you links to:

- the monitoring home page *(see page 141)*,
- the diagnostics home page *(see page 123)*,
- the setup home page *(see page 136)*,
- the documentation for accessing the product documentation in HTML format,
- change the language of the website.

View of the Home Page



Web site version: 1.0.0.0

Copyright © 2000-2009 Schneider Electric. All Rights Reserved

5.2 Diagnostics

Subject of this Chapter

The diagnostics link is used to obtain specific diagnostics on the operating status of the module and the CPU.

What’s in this Section?

This section contains the following topics:

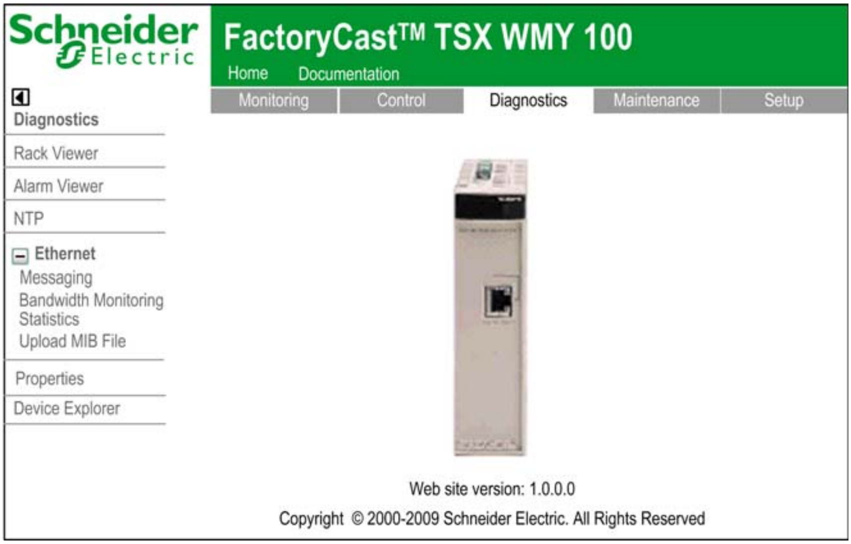
Ethernet Module Diagnostics Home Page

Presentation

This page contains a list of links for accessing the diagnostic pages for the Ethernet module:

- Rack Viewer,
- Alarm Viewer,
- NTP,
- Messaging service,
- Bandwidth service,
- Ethernet Statistics service,
- MIB Upload,
- Module properties.
- Device Explorer *(see page 86)*

View of the Diagnostics Home Page



Web site version: 1.0.0.0

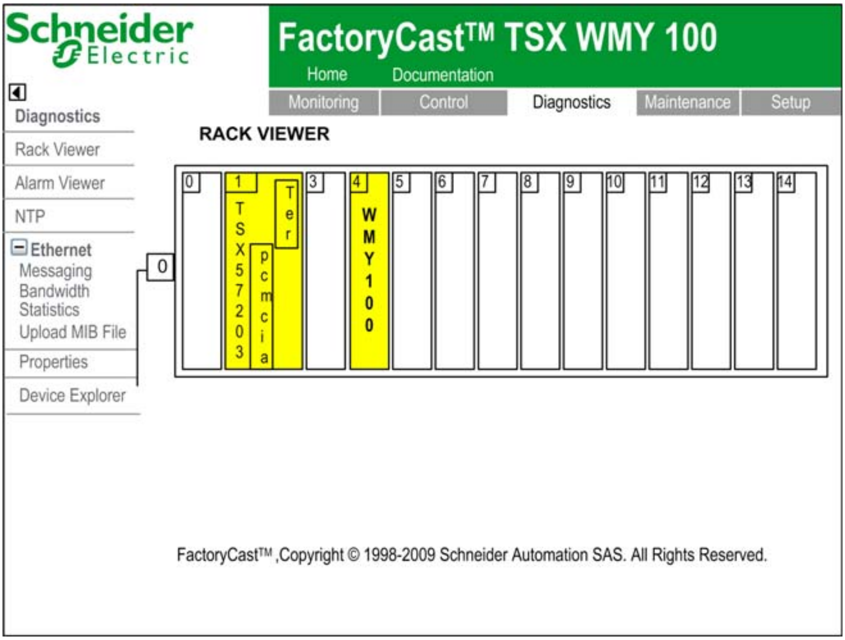
Copyright © 2000-2009 Schneider Electric. All Rights Reserved

Rack Viewer Page

Presentation

The Rack Viewer page shows the current configuration of the (local or remote) racks, in particular the PLC, the embedded server module and the I/O modules.

View of the Rack Viewer Page



FactoryCast™ ,Copyright © 1998-2009 Schneider Automation SAS. All Rights Reserved.

Data

The following information is provided for each module shown in the rack:

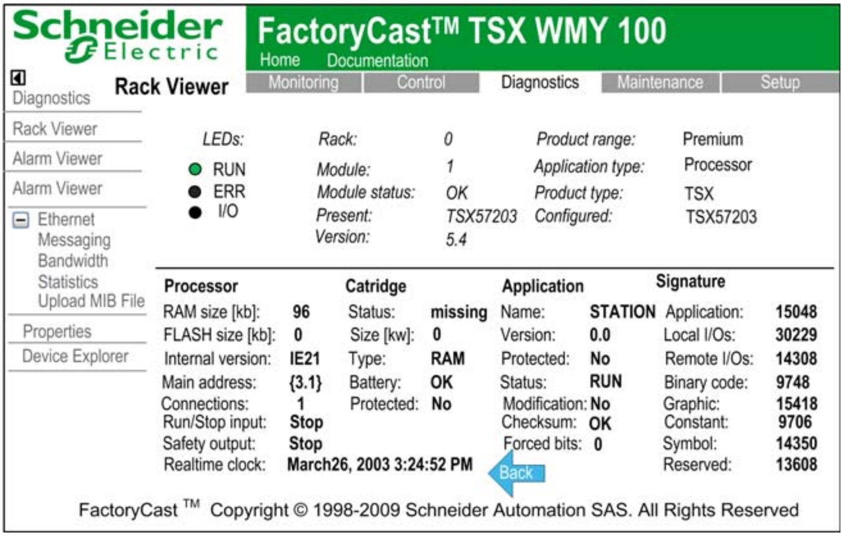
- A vertical label indicating the type of module and the product reference.
- The box in the upper left hand corner of the module indicates the slot number and operating state of the module:
  - A yellow box indicates that the module is healthy.
  - A red box indicates that the module is not operating correctly.

Links

Click on an element in the rack to obtain more detailed information (processor, module, Ter port, etc.).

A line descending from Rack#0 (to the left of the rack) is a link to the next rack. When you roll over this link, it changes into a red arrow. Each rack in the configuration can be displayed. The links towards the top of the page take you to Rack#0.

Here is an example of a PLC properties page for the above rack.



Rack Viewer

LEDs:		Rack:	0	Product range:	Premium
●	RUN	Module:	1	Application type:	Processor
●	ERR	Module status:	OK	Product type:	TSX
●	I/O	Present:	TSX57203	Configured:	TSX57203
		Version:	5.4		

Processor		Catridge		Application		Signature	
RAM size [kb]:	96	Status:	missing	Name:	STATION	Application:	15048
FLASH size [kb]:	0	Size [kw]:	0	Version:	0.0	Local I/Os:	30229
Internal version:	IE21	Type:	RAM	Protected:	No	Remote I/Os:	14308
Main address:	{3.1}	Battery:	OK	Status:	RUN	Binary code:	9748
Connections:	1	Protected:	No	Modification:	No	Graphic:	15418
Run/Stop input:	Stop			Checksum:	OK	Constant:	9706
Safety output:	Stop			Forced bits:	0	Symbol:	14350
Realtime clock:	March26, 2003 3:24:52 PM					Reserved:	13608

FactoryCast ™ Copyright © 1998-2009 Schneider Automation SAS. All Rights Reserved

The Back arrow takes you back to the Rack Viewer page for this PLC.

Dynamic Data

The indicators at the top and to the left of the screen provide dynamic references concerning the state of the PLC:

LEDs	Color when On	Meaning when On	Meaning when flashing	Meaning when Off
RUN	Green	Application running	Stopped	detected PLC error
ERR	Red	PLC error	Not configured	OK
I/O	Red	I/O event	---	OK
COM	Yellow	Communication interruption	---	OK

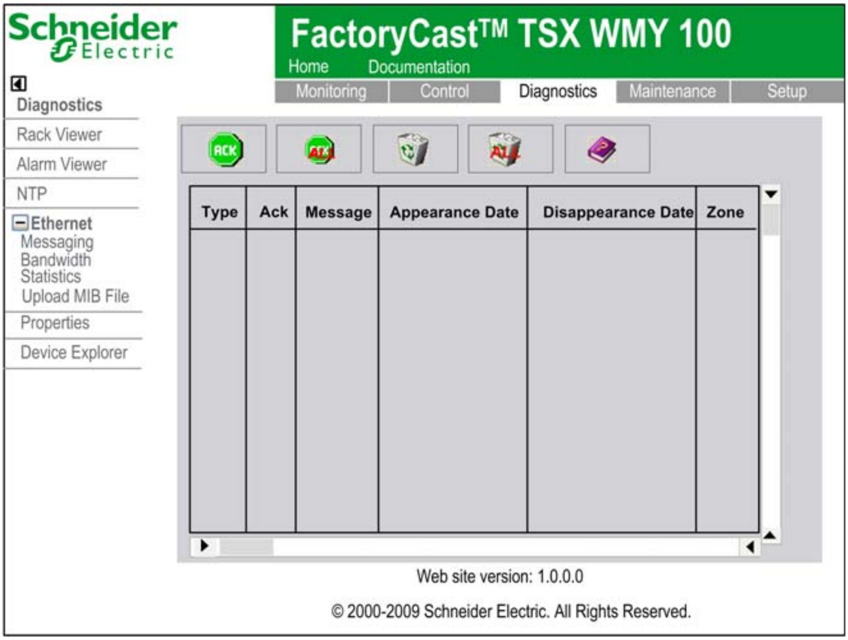
The dynamic reference LEDs concerning the status of the embedded server module are:

LEDs	Color	LED on	LED flashing	LED off
RUN	Green	Operating normally	---	Power off
ERR	Red	Module fault	Not configured	Operating normally
STS	Red	Network address incorrect or station outside limits	---	OK

Alarm Viewer Page

Presentation

This page is used to view and acknowledge PLC system alarms *(see page 287)*.



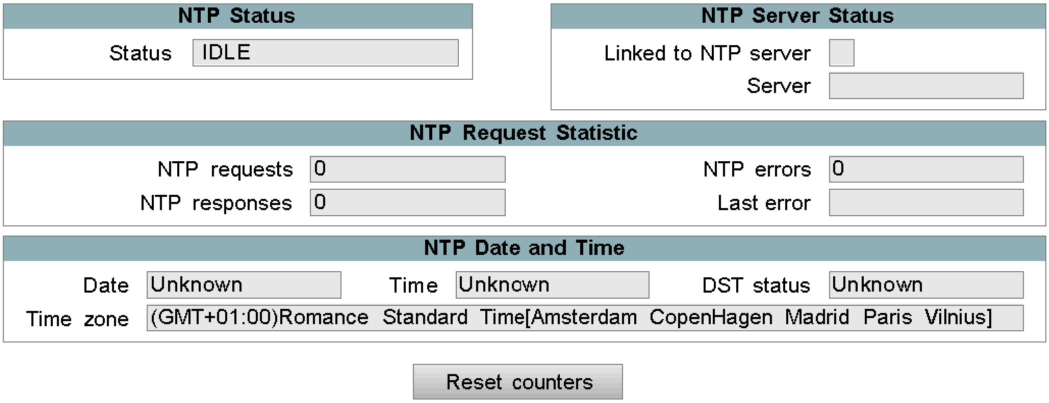
Type	Ack	Message	Appearance Date	Disappearance Date	Zone

Web site version: 1.0.0.0

© 2000-2009 Schneider Electric. All Rights Reserved.

NTP Statistics Page

View of the NTP Statistics Page



NTP Status

Status: IDLE

NTP Server Status

Linked to NTP server: □

Server:

NTP Request Statistic

NTP requests: 0

NTP errors: 0

NTP responses: 0

Last error:

NTP Date and Time

Date: Unknown

Time: Unknown

DST status: Unknown

Time zone: (GMT+01:00)Romance Standard Time[Amsterdam CopenHagen Madrid Paris Vilnius]

Element of the NTP Statistics Page

Text	Description
NTP Status	Indicates the status of the NTP service: ● unconfigured, ● configured.
NTP server status	● Indicates the status of the NTP server: ● OK if the link is available, ● NOK if the link is not available. ● Indicates the NTP servers available: ● NONE if there is no server, ● PRIMARY give the name of the primary NTP server, ● SECONDARY give the name of the secondary NTP server.
NTP request statistics	● Number of requests to the NTP server. ● Number of anomalies. ● Number of responses from the NTP server. ● Last anomaly.
NTP date and time	● DST status: daylight saving time can be configured automatically or not. ● Date: format Day Month Year. ● Time: format Hour Minute Second.
Time zone	Name of the zone: GMT +/- X.
Reset Counters	Click **Reset counters** to set all the counters to 0.

Ethernet Module Diagnostics Pages

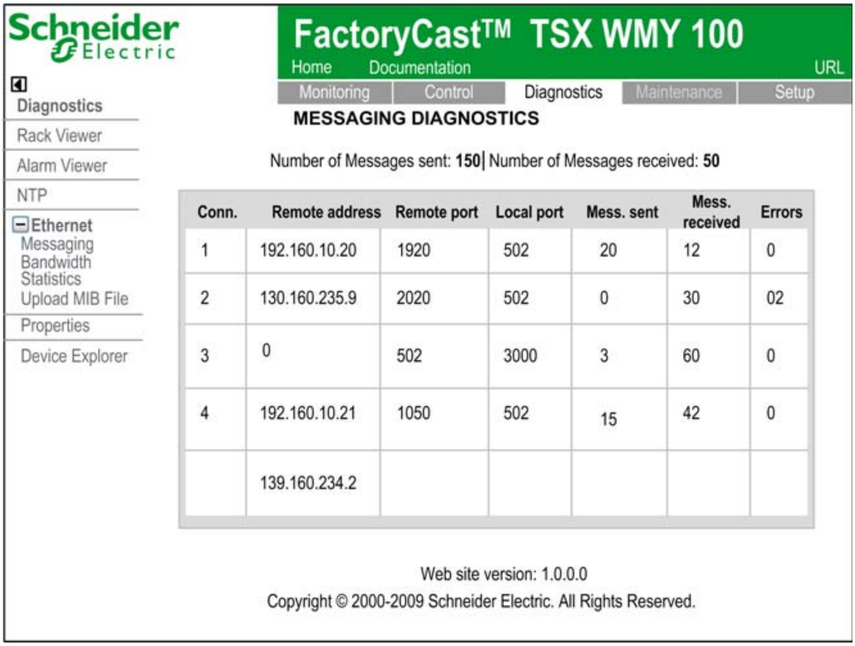
Presentation

The Ethernet Diagnostics home page contains a list of links for accessing the different Ethernet diagnostics pages for the module:

- Messaging service,
- Bandwidth service,
- Ethernet Statistics service,
- MIB Upload.

Messaging Diagnostics Page

This page provides current information on the open TCP connection on port 502:



MESSAGING DIAGNOSTICS

Number of Messages sent: **150**| Number of Messages received: **50**

Conn.	Remote address	Remote port	Local port	Mess. sent	Mess. received	Errors
1	192.160.10.20	1920	502	20	12	0
2	130.160.235.9	2020	502	0	30	02
3	0	502	3000	3	60	0
4	192.160.10.21	1050	502	15	42	0
	139.160.234.2					

Web site version: 1.0.0.0

Copyright © 2000-2009 Schneider Electric. All Rights Reserved.

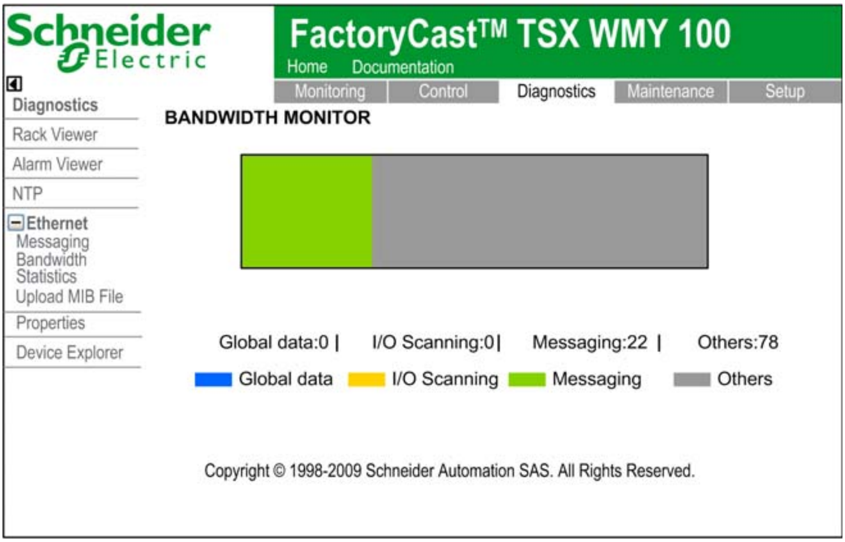
The number of messages sent/received on the port can be found at the top of this page.

A table provides, for each connection (numbered from 1 to 64):

- the remote IP address,
- the remote TCP port,
- the local TCP port,
- the number of messages sent from this connection,
- the number of messages received from this connection,
- the number of anomalies on this connection.

Bandwidth Page

This page displays the load distribution of the TSX WMY 100 module between the Global Data services, I/O Scanning, Messaging and other services.



BANDWIDTH MONITOR

Global data:0 | I/O Scanning:0| Messaging:22 | Others:78

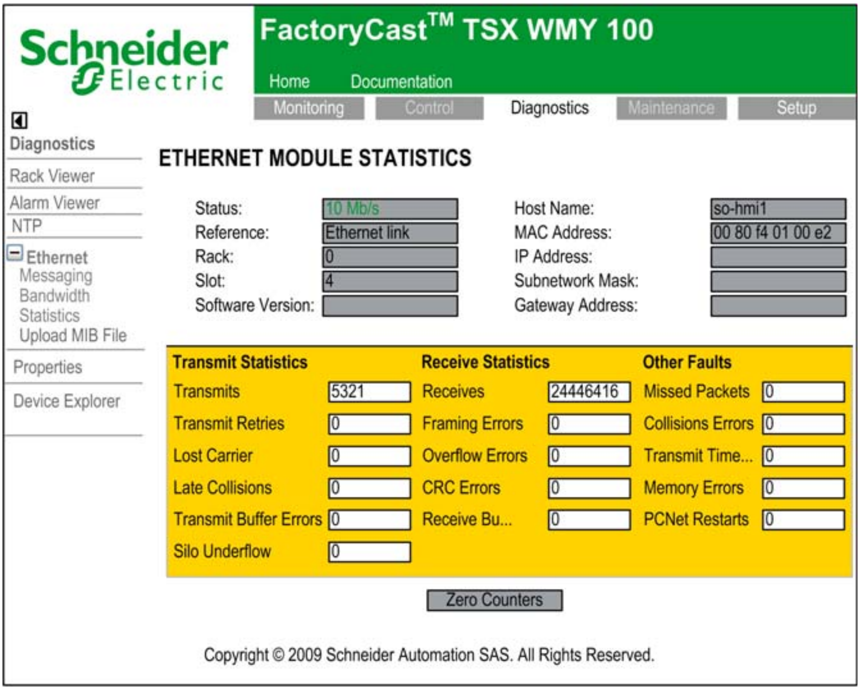
🟦 Global data 🟨 I/O Scanning 🟩 Messaging ⬜ Others

Copyright © 1998-2009 Schneider Automation SAS. All Rights Reserved.

NOTE: As Global data and I/O Scanning services are not supported on this module, they will always be set to 0.

Ethernet Statistics Page

This page provides all the latest information on the status, configuration and functions of the embedded server module.



ETHERNET MODULE STATISTICS

Status:	10 Mb/s	Host Name:	so-hmi1
Reference:	Ethernet link	MAC Address:	00 80 f4 01 00 e2
Rack:	0	IP Address:	
Slot:	4	Subnetwork Mask:	
Software Version:		Gateway Address:	

Transmit Statistics		Receive Statistics		Other Faults	
Transmits	5321	Receives	24446416	Missed Packets	0
Transmit Retries	0	Framing Errors	0	Collisions Errors	0
Lost Carrier	0	Overflow Errors	0	Transmit Time...	0
Late Collisions	0	CRC Errors	0	Memory Errors	0
Transmit Buffer Errors	0	Receive Bu...	0	PCNet Restarts	0
Silo Underflow	0				

Copyright © 2009 Schneider Automation SAS. All Rights Reserved.

Upload MIB File Page

Click this link to transfer the MIB file from the Ethernet module to a PC connected to the module.

Diagnostic resources monitoring for Quantum

Presentation

This page contains a list of links for accessing the diagnostic resources monitoring:

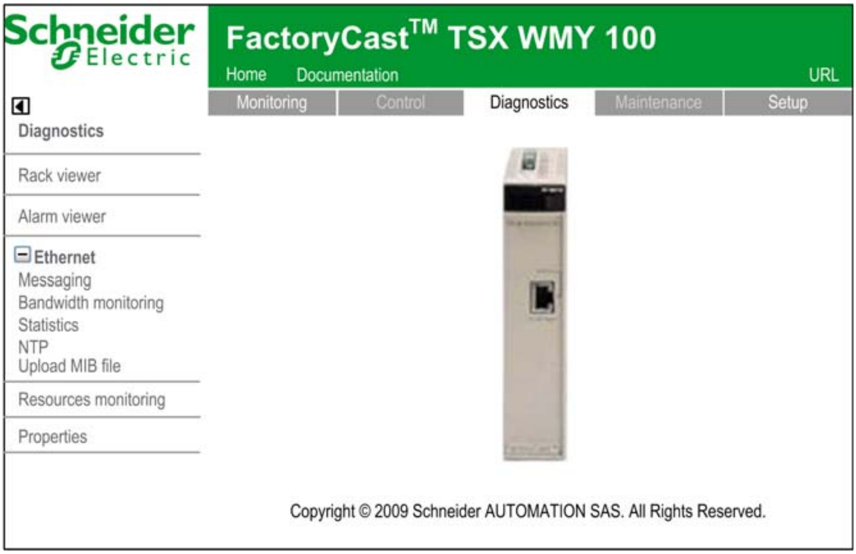
- Rack viewer
- Alarm viewer
- Messaging service
- Bandwidth service
- Ethernet statistics service
- Properties
- NTP
- Upload MIB file

The following table describes the links for diagnostic resources monitoring:

Links	Description
Rack viewer	The Rack viewer page shows the current configuration of the racks, in particular the PLC, the embedded server module and the I/O modules.
Alarm viewer	The Alarm viewer Premium and Quantum PLCs.
Messaging	The Messaging provides current information on the open TCP connection on the port.
Bandwidth Monitoring	The Bandwidth displays the load distribution of the TSX WMY 100 module between the global data services, I/O scanning, messaging and other services.
Ethernet statistics	The Ethernet statistics provides all the lastest information on the status, configuration and functions of the embedded server module.
NTP	The NTP (Network Time Protocol) synchronizes the clocks of computer systems.
Upload MIB file	The Upload MIB file upload the module MIB file to a directory on a PC connected to the TSX WMY 100.
Properties	The properties tab provides information on: ● The software version ● The version of web pages ● The physical properties of the module (Ethernet properties)

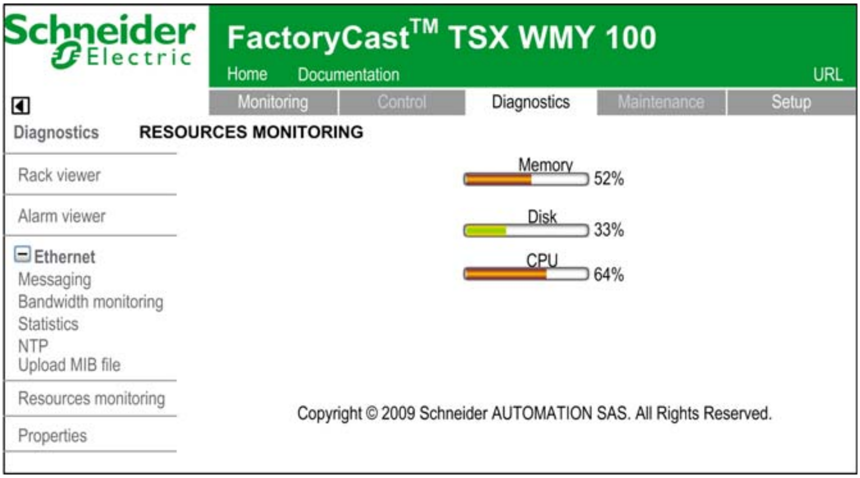
Resources monitoring page

The following figures shows a new resource monitoring diagnostics menu:



Copyright © 2009 Schneider AUTOMATION SAS. All Rights Reserved.

Memory available page:



Copyright © 2009 Schneider AUTOMATION SAS. All Rights Reserved.

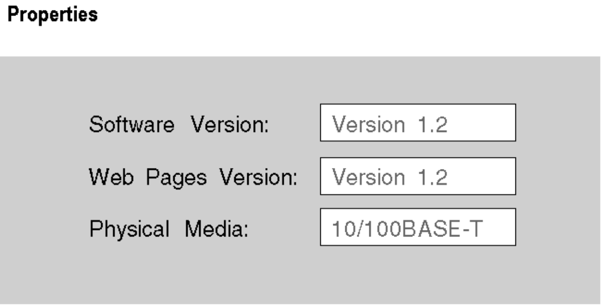
- **Memory**: Estimation the load on the RAM
- **Disk**: Estimation space free on the FLASH1
- **CPU**: Estimation the load on the CPU in use

Properties Page

Presentation

The properties tab provides information on:

- the software version,
- the version of Web pages,
- the physical properties of the module (Ethernet properties).



Properties

Software Version: Version 1.2

Web Pages Version: Version 1.2

Physical Media: 10/100BASE-T

5.3 Setup

Subject of this Chapter

The ‘**Setup**’ link is used to configure the module.

What’s in this Section?

This section contains the following topics:

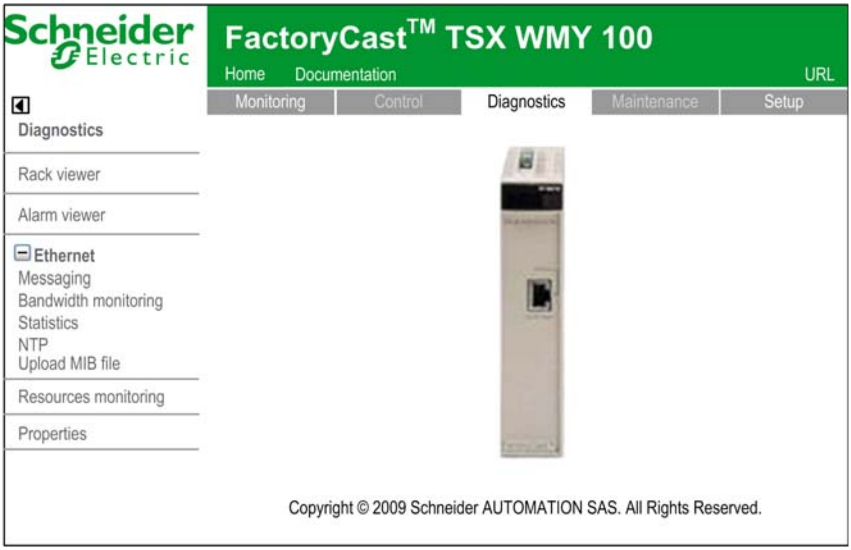
Module Setup Pages

Presentation

This page contains a list of links for accessing the different setup pages for the Ethernet module:

- Security,
- NTP client,
- DNS.

View of the Setup Home Page



Security Page

Presentation

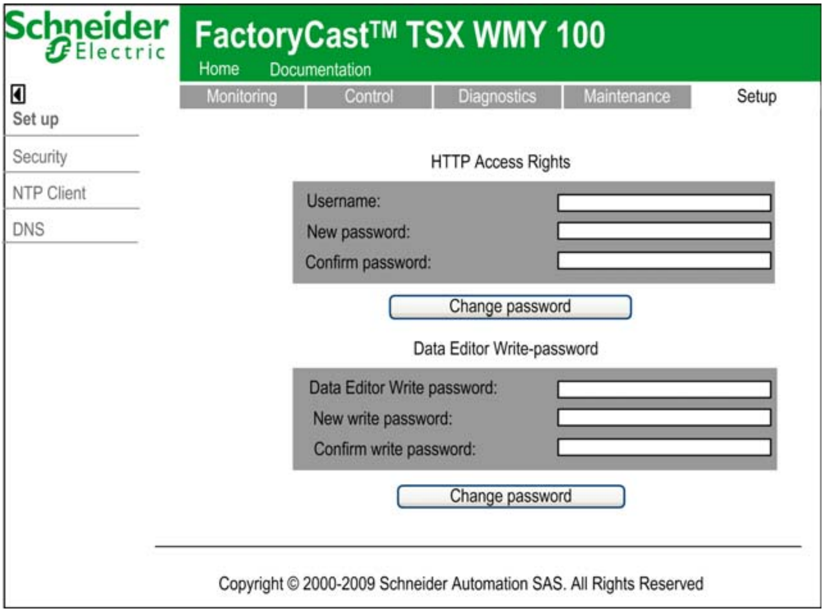
The security page is used to modify:

- the user name and password to access the diagnostics and setup links,
- the user name and password needed to write variables to the data editor.

NOTE:

- The user name and passwords must be no longer than 15 characters long in letters or numbers (a-z, A-Z and 0-9).
- The default values of the Username/Password fields protecting access to the Web pages are **USER**/**USER**.

View of the Security Page



Copyright © 2000-2009 Schneider Automation SAS. All Rights Reserved

NTP Service Configuration

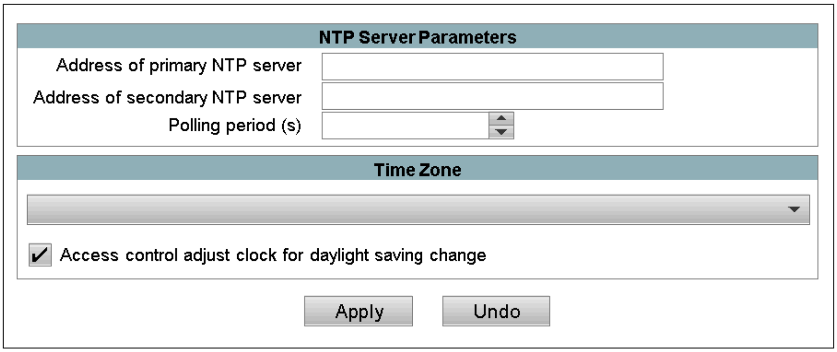
Presentation

The NTP (Network Time Protocol) is a protocol used for synchronizing the clocks of computer systems.

You can access the NTP configuration page via the Setup menu of the website.

View of the NTP Configuration Page

The parameters are configured in the following window:


NTP Server Parameters
Address of primary NTP server
Address of secondary NTP server
Polling period (s)
Time Zone
Apply
Undo

☑ Access control adjust clock for daylight saving change

Parameters

Parameters to be configured:

Parameters	Action
IP address of primary NTP server	Set the IP address of the primary NTP server.
IP address of secondary NTP server	Set the IP address of the secondary NTP server.
Polling period	Specify the polling period for updating time.
Time zone	Specify the local time zone (GMT, GMT+1, ...).
Automatic adjust clock for daylight saving change	Define if daylight saving time is managed.
Edit buttons	
Apply	Confirm the input.
Undo	Cancel the current changes.

DNS Service Configuration

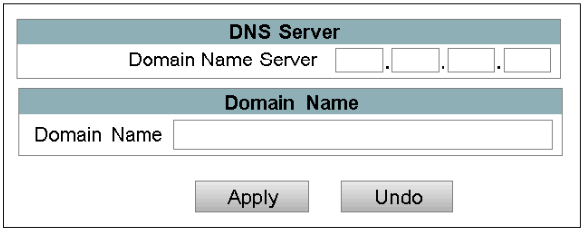
Presentation

The DNS (Domain Name System) is a distributed database that stores and associates many types of information with domain names, but most importantly, it translates domain names (computer hostnames) to IP addresses.

You can access DNS configuration page via the Setup menu of the website.

View of the DNS Configuration Page

DNS protocol parameters are configured in the DNS configuration page:



Parameters

Parameters to be configured:

Parameters	Values
Input fields	
Domain Name Server	Set the IP address of the DNS server.
Domain Name	Specify the domain name of your network for the module.
Edit buttons	
Apply	Confirm the input.
Undo	Cancel the current changes.

5.4 Monitoring

Monitoring Home Page

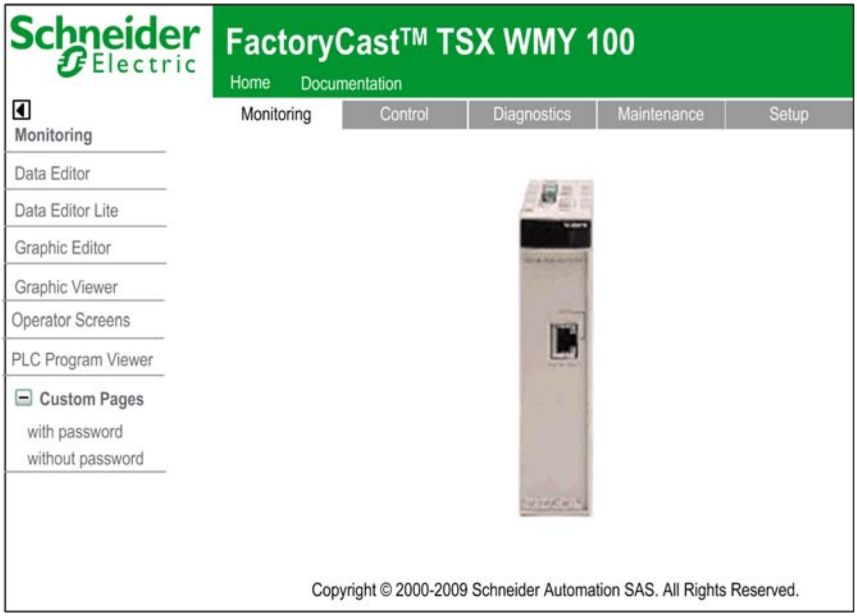
Presentation

The monitoring home page lists the various viewer services supported by the default web server of the module and provides links for accessing the services you require:

Link	Description
Data Editor	Create variable data tables, so as to be able to determine their value when the table is animated. See *(see page 144)*.
Data Editor Lite	Create variable data tables, with some limitations, so as to be able to determine their value when the table is animated. See *(see page 158)*.
Graphic Editor	Create graphics, so as to be able to determine the value of variables when the graphic is animated. See *(see page 169)*
Graphic Viewer	Display graphics, so as to be able to determine the value of variables when the graphic is animated. See *(see page 217)*
Operator Screens	Display graphic pages created with UnityPro. See *(see page 219)*
PLC Program Viewer	Display UnityPro programs in run mode using a Web browser. See *(see page 221)*
Custom Pages with Password	Display screen pages created by the user (accessible with password). See *(see page 225)*
Custom Pages without Password	Display screen pages created by the user (accessible without password). See *(see page 225)*

View of the Monitoring Home Page

The monitoring home page looks like this:



Schneider Electric

FactoryCast™ TSX WMY 100

Home Documentation

Monitoring | Control | Diagnostics | Maintenance | Setup

Monitoring

- Data Editor
- Data Editor Lite
- Graphic Editor
- Graphic Viewer
- Operator Screens
- PLC Program Viewer
- ⊟ Custom Pages
  - with password
  - without password

Copyright © 2000-2009 Schneider Automation SAS. All Rights Reserved.

Monitoring

6

Subject of this Chapter

This chapter describes the editors Java applets that enable you to create either dynamic data tables or dynamic graphic displays, to visualize operator screens and animated PLC programs coming from your UnityPro application. These editors are dynamically updated with run-time data from the PLC.

What's in this Chapter?

This chapter contains the following sections:

6.1 Data Editor

Subject of this Section

The Data Editor allows you to edit/create data monitoring tables or to display Data tables. Data tables provide read/write access to application data and devices registers. Write access is password protected. Allowing write access can change system behavior.

⚠ WARNING

UNINTENDED OPERATION

- Limit embedded server access to qualified personnel.
- Make sure variables that can be written are accessible by trained personnel only (password protect).
- Critical control variables must not be given write access via this mechanism.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

This section describes how to use the Data Editor to display and modify the values of symbol variables and direct addresses.

What's in this Section?

This section contains the following topics:

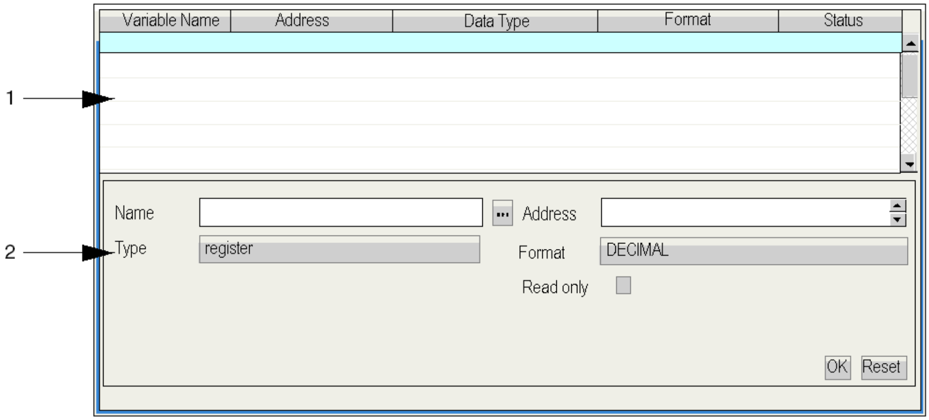
Data Editor

Overview

Data Editor is a Java applet that enables you to create dynamic data tables that can be updated with run-time data from the PLC.

Elements of Data Editor

The following illustration shows you the Data Editor:



Type register

Format DECIMAL

Number	Description
1	List of the variables included in this table.
2	The configuration area makes it possible to: ● select and/or modify a symbol, ● select and/or modify an address, ● select the variable type, ● select the variable's display format, ● check the read-only option.

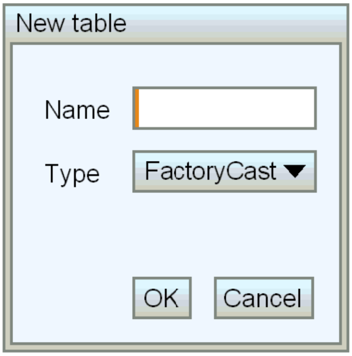
Creating a Data Template

Overview

To display some symbols (variables) or direct addresses, you must create a new data template.

Creating a Data Template

Follow the steps in the table below to create a data template:

Step	Description
1	If you would like a new one, select the button on the toolbar. Result: The **New table** box appears.  New table Name Type FactoryCast
2	Enter the name of the new Data template.
3	Click **OK**.

NOTE: Save *(see page 154)* the current spreadsheet before selecting a new spreadsheet. Selecting a new spreadsheet deletes the current spreadsheet.

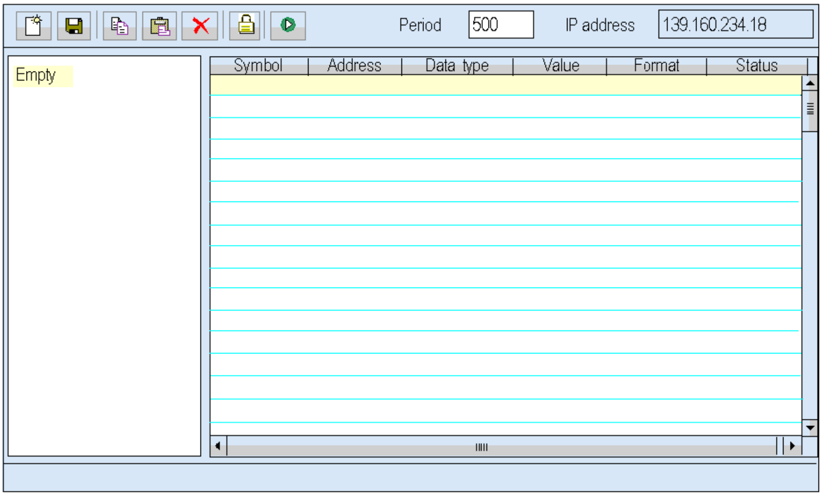
Data Editor Spreadsheet

Pressentation

Data Editor displays data in a spreadsheet with the following fields:

- Symbol
- Address
- Data Type
- Value
- Format
- State

View of the Spreadsheet



Symbol Field

The Symbol column contains the names of Concept, PL7 or Unity Pro symbol variables.

The only variable symbols which may be used in Data Editor are those previously defined by the Web Designer for FactoryCast HMI configuration tool. The symbol variables are grouped in a file called "Namespace". If you try to enter a symbol variable that is not in the namespace, a "variable not found" message appears.

NOTE: The Namespace must have been created using a program identical to the one operating in the PLC. If the namespace was created using a different program, its name is displayed at the bottom of Data Editor.

Address Field

The Address column contains direct addresses and addresses of Concept, PL7 or Unity Pro symbol variables. Any direct address may be viewed by entering its reference in this field. This direct address does not need to be referenced in the namespace.

Valid Direct Addresses for Quantum

- Coils (0x)
- Discrete inputs (1x)
- Input registers (3x)
- Output/holding registers (4x)
- Extended memory registers (6x)

For Unity Quantum PLC, direct addresses also include:

- %Mi (same as for 0X coils)
- %Ii (same as 1x for discrete inputs)
- %IWi (same as 3x for input registers)
- %MWi, %MDi, %MFi (same as 4x for holding registers
  For a Unity Quantum PLC, a single bit of any "word address" (for example, %MWi, %IWi) can be specified by appending ".j" to the address, where "j" is a bit index in the range of 0 (LSB) to 15 (MSB). For example, bit 4 of the value at %MW101 would be specified as %MW101.4.
  Also for a Unity Quantum PLC, a direct address can include an index specification that allows it to be treated as an array variable. Indexed addressing can be used with a %Mi, %MWi, %MDi or %MFi address by appending "[j]" to the address of the beginning of the array, where "j" is an unsigned integer value. For example, the third value of an array of float values starting at %MF201 would be specified as %MF201[2].

Valid Direct Addresses for Premium

Here are the valid direct addresses for Premium, except Unity Premium.

Address	Type	R(ead) or W(rite) Access
%KWi	WORD 16	R
%KDi	WORD 32	R
%MDi	WORD 32	R/W
%SDi	WORD 32	R/W
%Irs.c	BOOLEAN	R
%Qrs.c	BOOLEAN	R/W
%Mi	BOOLEAN	R/W
%Si	BOOLEAN	R/W
%MFi	REAL 32	R/W

Address	Type	R(ead) or W(rite) Access
%IWrs.c.i	WORD 16	R
%MWi	WORD 16	R/W
%SWi	WORD 16	R/W
%QWrs.c.i	WORD 16	R/W
r=rack number, s=slot number, c=channel number, and i=range number		

Here are the valid direct addresses for Unity Premium.

Address	Type	R(ead) or W(rite) Access
%KWi	WORD 16	R
%KDi	WORD 32	R
%MDi	WORD 32	R/W
%SDi	WORD 32	R/W
%Mi	BOOLEAN	R/W
%Si	BOOLEAN	R/W
%MFi	REAL 32	R/W
%MWi	WORD 16	R/W
%SWi	WORD 16	R/W
r=rack number, s=slot number, c=channel number, and i=range number		

Here are the valid FIP I/O addresses for Premium, except Unity Premium.

Address	Type	R(ead) or W(rite) Access
%I\rs.2.d\m.c	BOOLEAN	R
%Q\rs.2.d\m.c	BOOLEAN	R/W
%IW\rs.2.d\m.c.i	WORD 16	R
%QW\rs.2.d\m.c.i	WORD 16	R/W
%MW\rs.2.d\m.c.i	WORD 16	R/W
%KW\rs.2.d\m.c.i	WORD 16	R
r=rack number, s=slot number, d=device number, m=module number, c=channel number and i=range number		

Here are the valid FIP I/O addresses for Unity Premium.

Address	Type	R(ead) or W(rite) Access
%I\bs.cp\0.m.c	BOOLEAN	R
%Q\bs.cp\0.m.c	BOOLEAN	R/W
%IW\bs.cp\0.m.c.i	WORD 16	R
%QW\bs.cp\0.m.c.i	WORD 16	R/W
bs=bus number, cp=connection point, m=module number, c=channel number and i=range number		

For Unity Premium, a single bit of any "word address" (for example, %MWi, %SWi, %KWi) can be specified by appending ".j" to the address, where "j" is a bit index in the range of 0 (least significant bit) to 15 (most significant bit). For example, bit 4 of the value at %MW101 would be specified as %MW101.4.

Also for Unity Premium PLC, a direct address can include an index specification that allows it to be treated as an array variable. Indexed addressing can be used with a %Mi, %MWi, %MDi, %MFi, %KWi or %KD address by appending "[j]" to the address of the beginning of the array, where "j" is an unsigned integer value. For example, the third value of an array of float values starting at %MF201 would be specified as %MF201[2].

NOTE: Addressing of I/O modules concerns TSX DEY, TSX DSY, TSX DMY, TSX AEY, TSX ASY as well as the TBX and Momentum families.

Data Type Field

The Data Type field contains the data type of the symbol variable or direct address. Symbol variable data types appear automatically when the symbol variable is located. Direct address data types must be set by the user from a drop-down list.

The following data types are valid.

Abbreviation	Data type
INT	16-bit signed integer
UINT	16-bit unsigned integer
DINT	32-bit signed integer
UDINT	16-bit unsigned integer
REAL	32-bit IEEE floating point
TIME	32-bit unsigned integer (in ms)
DATE (1)	Date (32-bit BCD)
TOD (1)	Time of Day (32-bit BCD)
BOOL	1 internal bit (Boolean)
Legend (1) Not for Quantum	

Value Field

The Value column will be filled with the value of the symbol variable or direct address. This field is updated continuously in animation mode.

Format Field

The Format field contains the format type for displaying the value of the symbol variable or direct address. The following formats are accepted.

Abbreviation	Format Type
bool	Boolean
dec	Decimal
hex	Hexadecimal
binary	Binary
ASCII	Bytes displayed as ASCII characters
time	Day_hr_min_sec_ms
date (1)	YYYY-MM-DD or HH:MM:SS
Legend (1) Not for Quantum	

Status Field

The Status column contains messages about the status of communications with the symbol variable or direct address. If communications are normal, the status message is "OK".

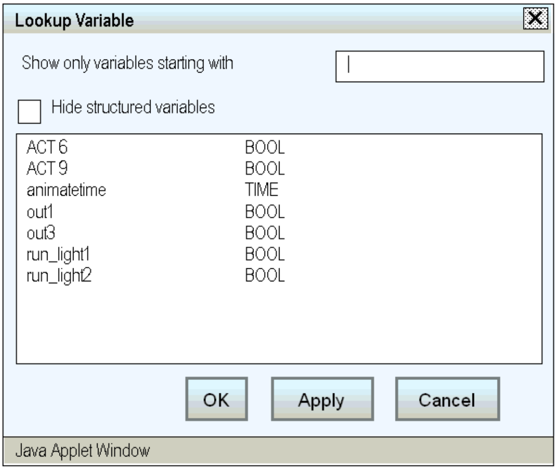
If there is a communication interruption with the symbol variable or direct address, the Status column contains a system message describing the event.

Inserting a Symbol (Variable) in a Data Template

Presentation

If you want to view or modify the value of a symbol (variable) in the namespace, you must insert that symbol (variable) in a data template.

Inserting a Symbol (Variable)

Step	Action
1	Double-click on an empty row in the spreadsheet. **Result**: The data editor’s configuration area appears.
2	In the configuration area, click on the ... button. **Result:** The **Lookup Variable** dialog box appears.  Lookup Variable Show only variables starting with □ Hide structured variables ACT 6 BOOL ACT 9 BOOL animatetime TIME out1 BOOL out3 BOOL run_light1 BOOL run_light2 BOOL
3	Select the symbol (variable) you want to insert in the data template.
4	Click **OK**. **Result:** The symbol (variable) you selected is displayed in the **Symbol** field.
5	In the configuration area, click on **Apply**. **Result:** A new row corresponding to the symbol (variable) you selected is displayed in the spreadsheet.

Inserting a Direct Address in a Data Template

Presentation

If you want to view or modify the value of a direct address, you must insert that direct address in a data template.

Allowing write access can change system behavior.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Limit embedded server access to qualified personnel.
- Password-protect access to the embedded server.
- Carefully select the symbols and direct addresses you authorize to be modified online.
- Do not authorize online modifications of critical process variables.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Inserting a Direct Address

Step	Action
1	Double-click on an empty row in the spreadsheet. **Result**: The data editor’s configuration area appears.
2	In the **Address** field of the configuration area, enter the variable’s Direct address.
3	In the configuration area, click on **Apply**. **Result:** A new row that corresponds to the variable address is displayed in the spreadsheet.

Saving a Data Template

Overview

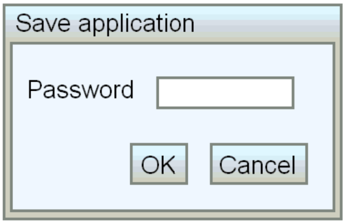
If you save a data template, you can use it again to view or modify the same symbols (variables) or direct addresses.

NOTE:

- Be careful when you modify and save a data table, the last modifications will be saved and will overwrite the existing table even if someone else has created it.
- If someone is viewing a data table which is overwritten, the modifications will be seen only on the next access to the data editor

Saving a Data Template

Follow the steps in the table below to save a data template.

Step	Action
1	Click the 💾 button on the tool bar. Result: The **Save application** box appears. 
2	Enter the password (USER by default).
3	Click **OK**.

Using an Existing Data Template

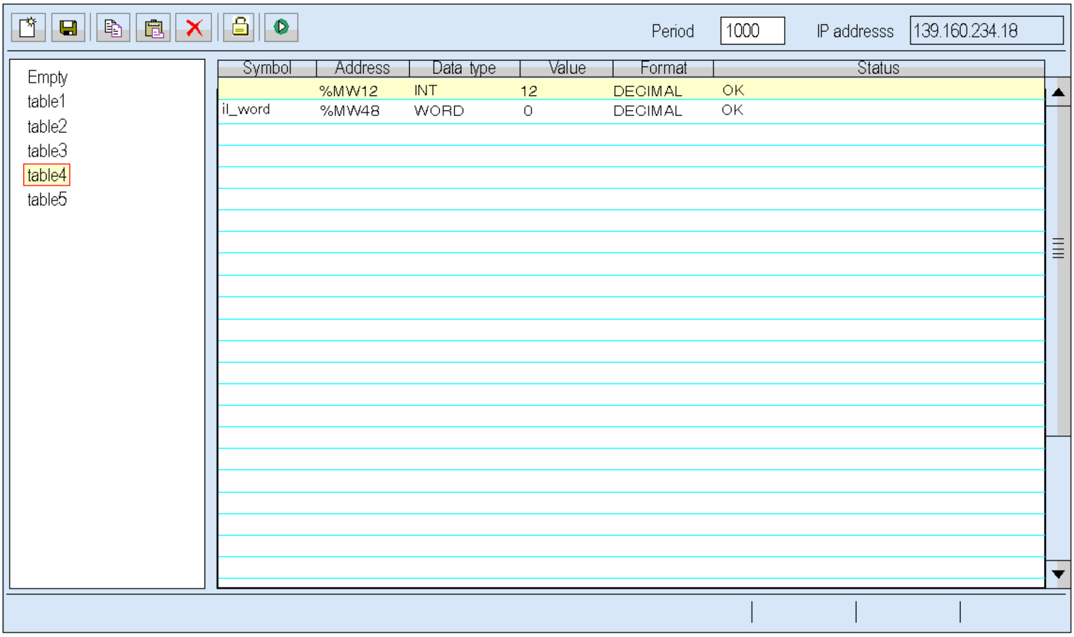
Overview

After you save a data template, you can use it to view or modify the values of the same symbols (variables) and direct addresses.

Finding a Data Template

A treeview menu on the Data Editor lists all the saved data templates.

DATA EDITOR



Period 1000 IP addresss 139.160.234.18

Empty
table1
table2
table3
table4
table5

Symbol	Address	Data type	Value	Format	Status
	%MW12	INT	12	DECIMAL	OK
il_word	%MW48	WORD	0	DECIMAL	OK

Retrieving a Data Template

Select the data template you want from the treeview menu. It will appear on a spreadsheet.

Modifying Data Values in a Data Template

Overview

You can use the Data Editor to modify the value of a symbol (variable) and of a direct address, and send the new value(s) to the controller.

Allowing write access can change system behavior.

> **⚠ WARNING**
>
> **UNINTENDED EQUIPMENT OPERATION**
>
> - Keep strict access to the embedded server by configuring passwords.
> - Carefully select the symbols and direct addresses you authorize to be modified online.
> - Do not authorize online modifications of critical process variables.
> - Do not use default or obvious password.
> - Limit access to trained personnel.
>
> **Failure to follow these instructions can result in death, serious injury, or equipment damage.**

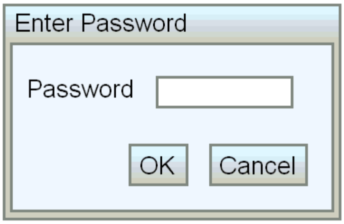
For example, say you have programmed a pushbutton object to jog a motor when the button is depressed and to stop jogging when the button is released. If communications are lost while the button is depressed, the motor will continue to jog even when the button is released. Graphic objects should not be used to control situations such as this unless other interlock methods are installed in the system.

Restrictions on Modifying Data

You can only modify the value of a symbol (variable) or of a direct address that is write-enabled in the namespace.

Modifying Data

Follow the steps in the table below to modify data:

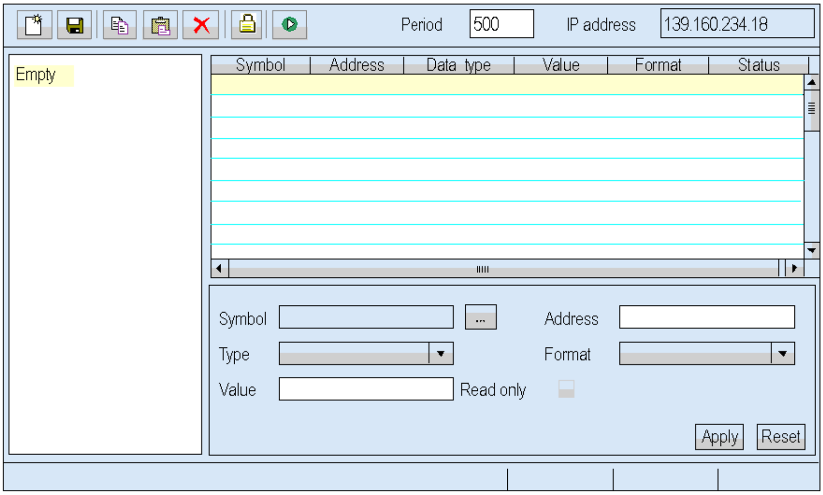
Step	Action
1	Double-click on a spreadsheet row that corresponds to the symbol (variable). **Result**: The data editor’s configuration area appears.
2	In the **Value** field of the configuration area, change the value of the symbol (variable) or direct address.
3	Click **Apply**.
4	**Result**: The **Password** dialog appears.  Enter Password Password
5	Enter the password (USER by default).
6	Click **OK**. **Result**: The new value is sent to the controller.

6.2 Data Editor Lite

Data Editor Lite

Overview

Data Editor Lite is a compact version of Data Editor that is faster to download, especially for use over a dialup connection. It provides the same interface than the standard Data Editor but with some restrictions on accessible data type.



Variables

Data Editor Lite accepts the following IEC and Concept variables:

Address	Type	Display
%MW IEC internal word	INT	DECIMAL
%MD IEC double word	DINT	DECIMAL
%M IEC internal bits	BOOL	BOOLEAN
400000 Concept variable	INT or DINT	DECIMAL
000000 Concept variable	BOOL	BOOLEAN

Tables

Data Editor Lite can reuse the tables created with Data Editor and Pocket Data Editor. However, Data Editor tables use a wider range of variable types than in Data Editor Lite. When Data Editor Lite encounters a variable it cannot manage, it displays `Not Supported`. In this case, the variable cannot be edited.

Operating Mode

Refer to the Operating Mode section for Data Editor *(see page 144)*.

NOTE: You cannot insert symbols (variables) in a Data Template with Data Editor Lite, you can only insert direct addresses.

6.3 Pocket Data Editor

Overview

This section describes the Pocket Data Editor, which runs on Pocket PC.

What’s in this Section?

This section contains the following topics:

Pocket Data Editor

Overview

Pocket Data Editor is a version of the Remote Data Editor that operates in a Pocket PC environment. A Pocket PC with a WiFi connection can be used as a mobile client station to access data on remote equipment. Pocket Data Editor supports remote data monitoring, remote control, and remote diagnostics.

Pocket PC Requirements

The Pocket PC requirements are:

- only Pocket PCs with ARM processors
- Pocket PC Windows 2003 mobile edition

Pocket PC Windows 2002 mobile edition is not supported.

Capabilities

Pocket Data Editor can open, edit, and save the same data tables as the PC version in a format appropriate for Pocket PCs.

NOTE: If you have developed custom pages for your Web server, Internet Explorer for Pocket PC can access the same custom HTML pages on an equipment Web site. A Pocket PC screen is smaller than one on a PC, so the Web site must be adapted to return the appropriate page based on the client's browser type.

Availability

Pocket Data Editor is available on the FCHMI Web server. It can be used with the following modules:

- a TSX NWY for Quantum,
- a TSX WMY for Premium.

NOTE: Pocket Data Editor is disabled when the Pocket PC is connected to and synchronized with a PC via USB using the ActivSync tool.

NOTE: Some communication interruptions can occur between a Pocket PC and the FactoryCast module when the WiFi (802.11) power save mode is activated on a Pocket PC. See your Pocket PC user manual for the procedure to disable this feature.

Variables

Pocket Data Editor accepts the following variables:

Address	**Type**	**Display**
%MW IEC internal word	INT	DECIMAL
%MD IEC double word	DINT	DECIMAL
%M IEC internal bits	BOOL	BOOLEAN
400000 Concept variable	INT or DINT	DECIMAL
000000 Concept variable	BOOL	BOOLEAN

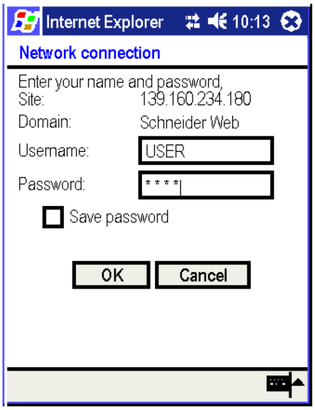
Using Pocket Data Editor

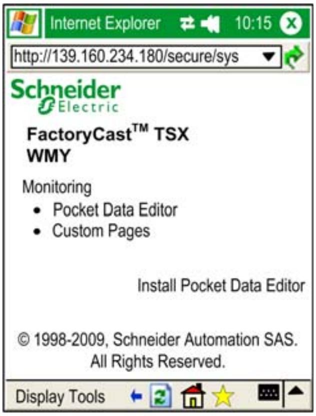
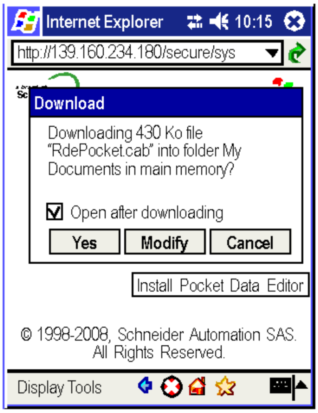
Overview

When you first connect a Pocket PC to the FactoryCast Web server, you must download and install the Pocket Data Editor software on your Pocket PC. After this, you can launch Pocket Data Editor by clicking on the **Pocket Data Editor** link on the equipment Web site home page.

Installing Pocket Data Editor

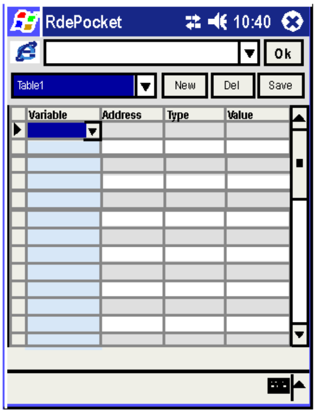
The following table describes the installation of Pocket Data Editor:

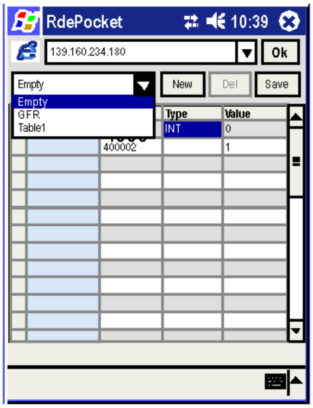
Step	Action
1	Turn on the Pocket PC.
2	Launch Internet Explorer.
3	Enter the IP address of the remote equipment. **Result**: The following window appears:  Network connection Enter your name and password, Site: 139.160.234.180 Domain: Schneider Web Username: USER Password: **** □ Save password

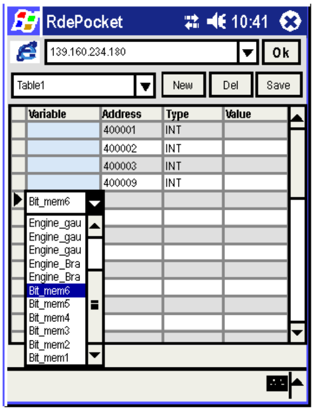
Step	Action
4	Enter the username and password. (The defaults are USER and USER.) **Result**: the FactoryCast Server home page appears:  Internet Explorer 10:15 http://139.160.234.180/secure/sys Display Tools Schneider Electric FactoryCast^{TM} TSX WMY Monitoring • Pocket Data Editor • Custom Pages Install Pocket Data Editor © 1998-2009, Schneider Automation SAS. All Rights Reserved.
5	Click on **Install Pocket Data Editor**. **Result**: The following window appears:  Internet Explorer 10:15 http://139.160.234.180/secure/sys Yes Modify Cancel Display Tools Download Downloading 430 Ko file "RdePocket.cab" into folder My Documents in main memory? ☑ Open after downloading Install Pocket Data Editor © 1998-2008, Schneider Automation SAS. All Rights Reserved.
6	Click on **YES**.

Using Pocket Data Editor

The following table describes the use of Pocket Data Editor

Step	Action
1	Click on the **Pocket Data Editor** link on the equipment Web site home page. **Result**: The following window appears:  RdePocket 10:40 Ok Table1 New Del Save Variable Address Type Value Push the **New** button to create a new table after entering its name. Push the **Del** button to delete the current table. Push the **Save** button to save the new version of the table after entering the HTTP password.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Select the desired table via the drop-down menu:

RdePocket
10:39
139.160.234.180
Ok
Empty
New
Del
Save
Empty
GFR
Table1

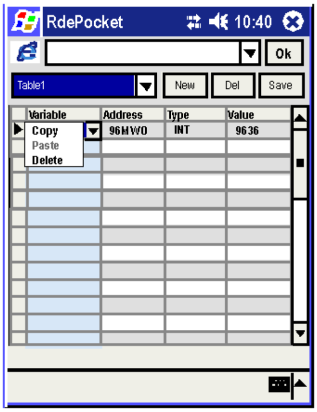
<table>
<tr><th></th><th>Type</th><th>Value</th></tr>
<tr><td></td><td>INT</td><td>0</td></tr>
<tr><td>400002</td><td></td><td>1</td></tr>
</table>
The Pocket PC version can use the same tables defined in Data Editor for PC and in Data Editor Lite. Not all variables are accessible. (Restrictions are detailed in the Variables section <i>(see page 161)</i>. A selected variable may be reported as <b>not supported</b> in the status bar. For an unsupported variable type, the string ??? appears in the value cell.</td></tr>
<tr><td>3</td><td>If you want to view or modify the value of a PLC variable, insert the variable symbol or address in the Pocket Data table.

RdePocket
10:41
139.160.234.180
Ok
Table1
New
Del
Save
Bit_mem6
Engine_gau
Engine_gau
Engine_gau
Engine_Bra
Engine_Bra
Bit_mem6
Bit_mem5
Bit_mem4
Bit_mem3
Bit_mem2
Bit_mem1

<table>
<tr><th>Variable</th><th>Address</th><th>Type</th><th>Value</th></tr>
<tr><td></td><td>400001</td><td>INT</td><td></td></tr>
<tr><td></td><td>400002</td><td>INT</td><td></td></tr>
<tr><td></td><td>400003</td><td>INT</td><td></td></tr>
<tr><td></td><td>400009</td><td>INT</td><td></td></tr>
</table>
You can add new addresses, following the restrictions detailed in the Variables section <i>(see page 161)</i>. To modify a PLC variable, select the corresponding <b>value</b> cell and enter the new value.</td></tr>
</table>

Copy, Paste, or Delete Table Rows

To copy, paste, or delete, select one or more rows by clicking in the first cell (to the left of the **variable** cell). The selected rows are then highlighted. Hold down the mouse button in this cell until a popup menu appears, then select the desired action.

- To copy, select one or more rows that have values (that are not empty).
- To paste the copied rows, select a row, empty or not. If several rows are copied, the other rows are pasted one after the other.
- To delete one or more rows, select rows that have values (that are not empty) and choose delete.



Variable	Address	Type	Value
	96MW0	INT	9636

Using customized pages

If you want to display a custom page, click on the Custom pages link on the equipment Web site home page.

Launch Pocket Data Editor from its icon

Pocket Data Editor can also be launched directly from its program icon.

6.4 Graphic Editor

Subject of this Section

This section describes the functions and characteristics of the Graphic Editor. The Graphic Editor is a Web page that enables the user to create dynamic graphic displays with a browser and using a predefined set of graphic objects. The Graphic Editor is both a graphic editor that can be used to create and modify displays, and a Runtime environment that allows the user to view animated displays using data from the PLC.

What's in this Section?

This section contains the following topics:

Overview of the Graphic Editor

Top Window

The Graphic Editor is made up of three windows. The top window features an area for presenting the user commands and dialog boxes for creating, saving, reading and editing a graphic display.

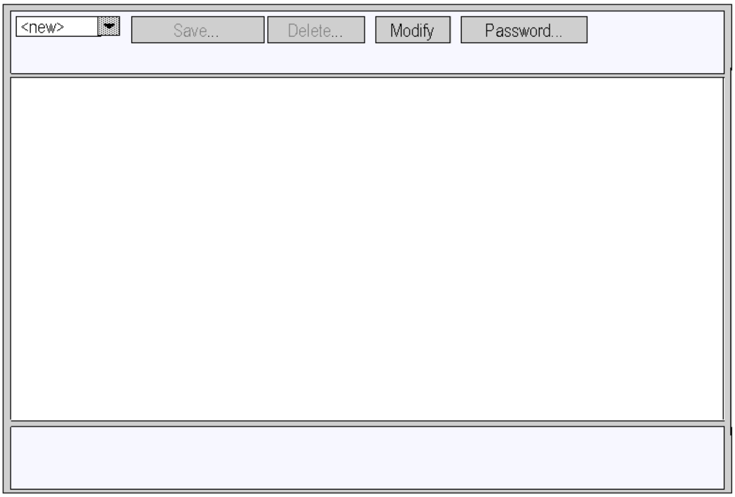
Display Window

The display window proposes a presentation area of the current graphic display. When you create a new graphic display, this window turns into a blank space into which you can add the graphic objects that will make up the required graphic display.

Message Window

This window proposes a scrolling message window used to display the messages generated by the Graphic Editor. The figure below shows the Graphic Editor with its initial top window and display window.

Graphic Editor



Graphic Objects

All the graphic objects supplied with the Graphic Editor are able to communicate with the PLC. They are also designed as standalone objects, which means no connection is needed between the objects, and that each object is capable of operating on its own.

Allowing write access can change system behavior.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Password-protect access to the embedded server.
- Carefully select the symbols and direct addresses you authorize to be modified online.
- Do not authorize online modifications of critical process variables.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

NOTE: Write access is controlled by a password (Security) whose default value is USER.

Creating and Modifying Graphic Displays

If you wish to create and modify a graphic page, click **Modify...** which will give you access to the specific editing functions. Then, simply select the objects from the palette, place them in a space, move them and adjust their size, and define their properties. You can immediately test the graphic display updated with the PLC data by quitting edit mode (click **Done**). If you want to keep the graphic display you have created, you can save it in the PLC for future use by clicking **Save...**, on condition you have entered the correct password.

NOTE: Be careful when you modify and save a graphic page, the last modifications will be saved and will overwrite the existing page even if someone else has created it.

Toolbar

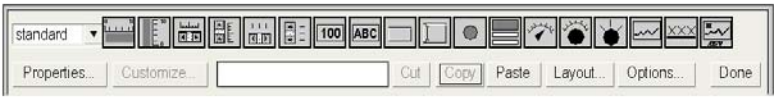
Overview

The Graphic Editor applet’s top window consists of several "dialog panels," only one of which is shown at a time. Switching from one dialog to another is done by clicking buttons on the current dialog. This section describes the dialog panels that comprise the top window.

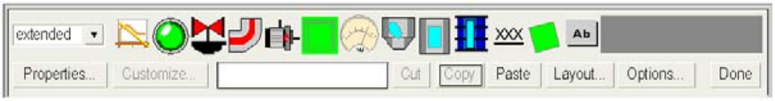
Top Dialog

The **Edit Dialog** allows you to select a graphic object for placement in the display window, and provides access to all graphic editing functions. The available graphic objects are presented in a set of palettes, with one palette visible at a time. There are two palettes.

The standard palette:



The extended palette:

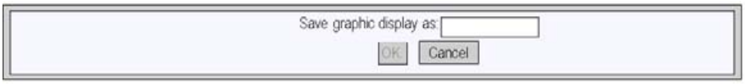


The controls in the **top dialog box** provide the following functions.

- **Drop-down List**. The drop-down list box shows all the graphic pages that have been saved to the FactoyCast HMI module, and are available for retrieving. When you select a graphic page from this list, the graphic display currently visible in the window is replaced with the one selected. If the current graphic page has been modified since it was last saved, you will be asked for confirmation that the changes are to be discarded. If the special entry <new> is chosen from the list, then the display window is cleared, and a new graphic page can be created.
- **Save**. The **Save** button causes the **Save dialog box** to become visible. This button is disabled until you have entered a correct write-enable password.
- **Delete**. The **Delete...** button causes the **Delete dialog box** to become visible. This button is disabled until you have entered a correct password, or if the current graphic display has not yet been saved.
- **Modify**. The **Modify...** button causes the **Edit dialog box** to become visible.
- **Password**. The **Password...** button causes the **Password dialog box** to become visible.
- **Information display area**. The information display area shows the name and version of the Concept, PL7 or Unity Pro program that is running in the connected PLC.

Save Dialog

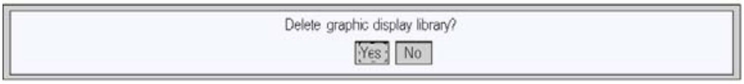
The **Save dialog box** allows you to save the current graphic display.



When the **Save dialog box** is presented, the name of the current graphic page is shown in the dialog's text field. If the current graphic page has never been saved (i.e., a "new" graphic display), then the text field is blank. Once you have either accepted the current name (a "save" operation) or provided a new name (a "save as" operation), then you can click the **OK** button to save the contents of the current graphic display to the Web server module. The **Cancel** button will cause the **Top dialog box** to be shown again, with no action being taken.

Delete Dialog

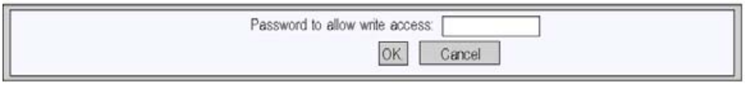
The **Delete dialog box** allows you to delete the current graphic page.



If you click the **Yes** button, the existing graphic display window is cleared and the graphics file on the Web server module is deleted. Clicking the **No** button will cause the **Top dialog box** to be shown again, with no action being taken.

Password Dialog

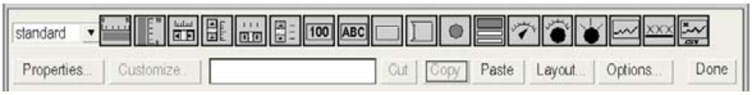
The **Password dialog box** allows you to enter the password that enables those user functions that modify graphic display files or PLC run-time data values.



If you enter the correct password and click the **OK** button, then you will be allowed to save and delete the current graphic display. Correct password entry also permits you to write new values to the PLC (via those graphic objects that support writing values to a PLC, if any). If you click the **OK** button when the text field is empty, then the current password permissions, if any, are cleared. The **Cancel** button will cause the **Top dialog box** to be shown again, with no changes made to current password permissions.

Edit Dialog

The **Edit dialog box** allows you to create or modify a graphic page, by selecting a graphic object for placement in the display window, and accessing all the graphic editing functions. The graphic objects available are presented in a single object palette.

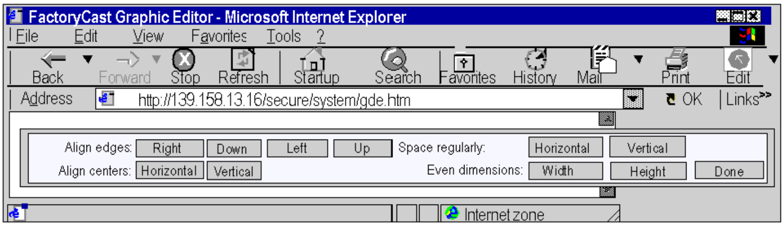


The controls of the **Edit dialog box** provide the following functions:

- The **Drop-down List Box** shows the set of palettes that are available. When you select the name of a palette from the list, the palette area of the dialog displays the selected one’s graphic objects.
- The **Palette** shows the graphic objects that are in the current palette with an icon that depicts each graphic object’s type (meter, button, etc.). When you click any of the icons in the palette, a graphic object of the corresponding type becomes selected for insertion. While the Graphic Editor is in "insert mode," if you click in an open area of the display window, an instance of the selected graphic object is inserted into the graphic display.
- The **Information Area** shows the name and size of the graphic object that is currently selected.
- The **Cut** button causes the currently selected graphic object(s) to be removed from the graphic display and saved to a buffer (i.e., an internal clipboard), replacing any existing contents of the buffer.
- The **Copy** button causes the currently selected graphic object(s) to be copied to the buffer, replacing any existing contents.
- The **Paste** button causes the content of the clipboard to be inserted into the upper left corner of the graphic display. The pasted graphic objects can then be moved to the desired location in the display.
- The **Properties** button causes the Properties Sheet for the currently selected graphic object to be shown.
- The **Customize** button causes the Customizer for the currently selected object to be shown, if the graphic object has been provided with one.
- The **Layout** button shows the **Layout dialog box**.
- The **Options** button shows the **Options dialog box**.
- The **Done** button causes the **Top dialog box** to be shown again.

Layout Dialog

The **Layout dialog box** allows you to change the position and size of a group of graphic objects.


FactoryCast Graphic Editor - Microsoft Internet Explorer
File Edit View Favorites Tools ?
Back Forward Stop Refresh Startup Search Favorites History Mail Print Edit
Address http://139.158.13.16/secure/system/gde.htm OK Links
Align edges: Right Down Left Up
Space regularly: Horizontal Vertical
Align centers: Horizontal Vertical
Even dimensions: Width Height Done
Internet zone

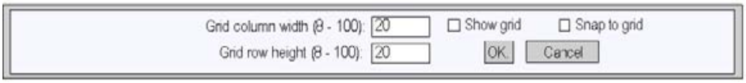
The controls of the **Layout dialog box** provide the following functions.

- The **Right**, **Bottom**, **Left**, and **Top** buttons can be used to align the edges of the selected graphic objects so that their specified sides are at the same position. At least two graphic objects must be selected for these buttons to be enabled.
- The **Horizontal**, and **Vertical** buttons are used to align the centers of the graphic objects. At least two graphic objects must be selected for these buttons to be enabled.
- The **Horizontal** and **Vertical** buttons are used to space the selected graphic objects regularly, in order that the horizontal or vertical spacing between the objects is the same. At least three graphic objects must be selected for these buttons to be enabled.
- The **Width** and **Height** buttons are used to achieve parity in dimensions of the graphic objects, so the selected width or height corresponds. At least two graphic objects must be selected for these buttons to be enabled.
- The **Done** button causes the **Edit dialog box** to be shown again.

NOTE: For all layout operations (except **Space evenly**) one of the selected objects is considered the "reference object" to which all other selected objects refer in order to know their new position or dimension. For example, when the "Width" button is pressed, all of the selected objects will have their width changed to match the width of the reference object. The reference object is differentiated from the other selected objects by making its selection box a different color than the others.

Options Dialog

The **Options dialog box** is used to change the settings related to a grid drawn in the display window. The grid is solely for assistance in editing or creating a graphic display and is shown only when the Graphic Editor is in "edit mode."



Grid column width (8 - 100): 20 □ Show grid □ Snap to grid

Grid row height (8 - 100): 20

The controls of the **Options dialog box** provide the following function.

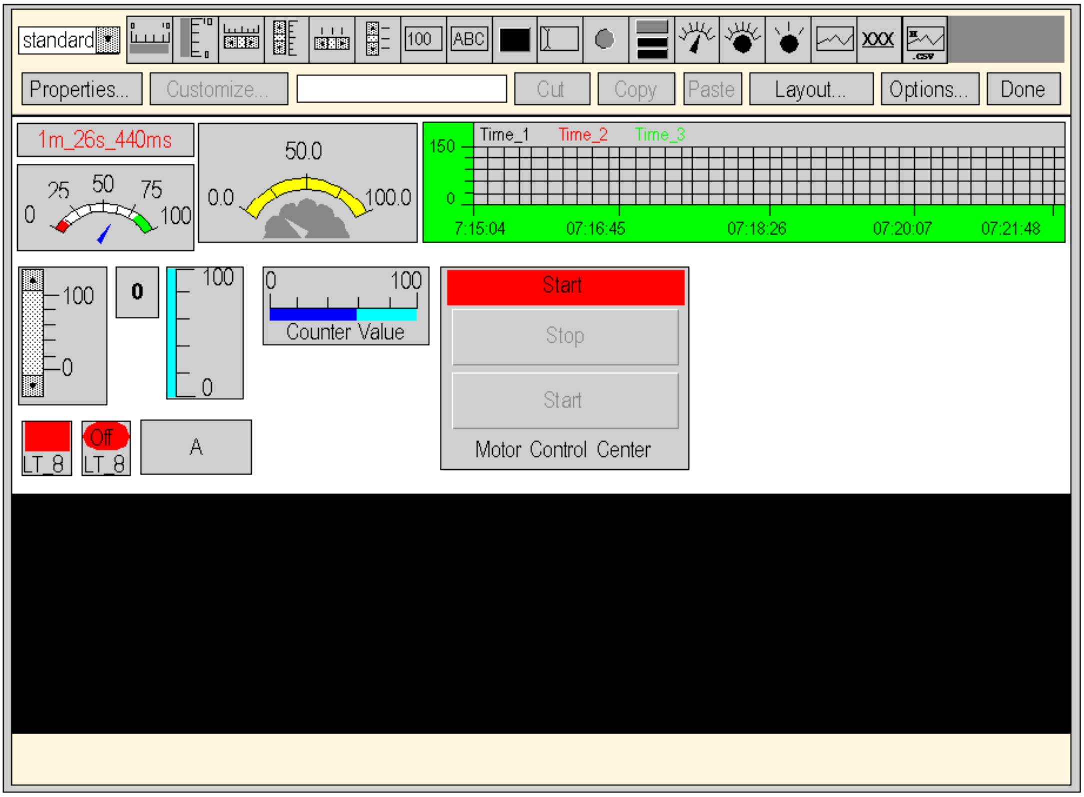
- The cell size of the grid can be changed by the entering the grid’s column width and row height into the dialog’s text fields.
- If the **Show grid** check-box is checked, the grid will be drawn; otherwise, no grid will be shown.
- If the **Snap to grid** check-box is checked, then, when you change the size or position of a graphic object, the changed coordinate(s) or dimension(s) is automatically adjusted so that it coincides with a grid point.
- The **OK** button causes the current option settings to become active, and the **Edit dialog box** to be shown again.
- The **Cancel** button causes the **Edit dialog box** to be shown again, with no option settings being changed.

User Functions of the Display Window

Overview

The user functions available in the display window of the **Graphic Editor** are used to select and move objects, and redefine their size. All moving or size redefinition operations require the graphic object(s) to be selected before being modified. The object selected appears framed on the screen.

The figure below shows the **Graphic Editor** screen.



Selecting Graphic Objects

The selection status of a graphic object (selected/deselected) can be defined by the following user actions:

- You can select a single graphic object by simply clicking on it. If other objects are already selected, they are automatically deselected.
- You can select several graphic objects by creating a rectangle over them in the display window. Left-click with the mouse in an open area of the display window (not on a graphic object) and, holding down the button, drag the curser to draw a dotted frame. One of the corners of the rectangle will be where you first clicked, and the corner diagonally opposite will follow the path of the curser. When you release the button, all the objects within the box will be selected. All objects outside this box will be deselected.
- You can select or deselect a graphic object by holding down the CTRL key and clicking on an object. Here, the selection status of all other objects will remain unchanged. With this action, each graphic object can be added to the current group of selected or deleted objects.
- You can select a graphic object by holding down the SHIFT key and clicking on an object. Here, the selection status of all other objects will remain unchanged. With this method, when one object is selected it becomes the *reference object* (see the layout dialog box in section *Toolbar, page 171*) for the group of objects selected. The main aim of this action is to change the reference object in a group of selected objects. The **layout** operations come later.
- You can deselect all the graphic objects by clicking in an open area of the display window (and not on a graphic object).

Defining the Size of the Graphic Objects

You can modify the size of a graphic object by first selecting it then using the mouse to change the size of the object’s selection box. As you move the curser on an object’s selection box, this changes to reflect the type of operation to perform to redefine the size. If you press the left mouse button with the curser on an object’s selection box, then hold down the button and move the object, a dotted frame appears. When you release the button, the size of the object is modified to correspond to the frame you have drawn. There are eight possible ways of redefining the size, according to the part of the object’s selection box from which you drag the mouse. The corners of the frame will only move the adjacent sides, and each side of the frame can be moved on its own.

Moving Graphic Objects

The position of a graphic object in the display window can be modified using the mouse. If you press the left mouse button with the curser on an object, then hold down the button and move the object, a dotted frame appears. When you release the button, the position of the object is modified to correspond to the frame you have drawn.

You can move several graphic objects by first selecting the objects to move, then moving the whole group as you would a single object. When you move a group of objects, a box delimits each object of the group.

Defining the Properties of the Graphic Objects

You can define the properties of a graphic object using the **Property Sheet** (see property sheet in section *Property Sheet, page 179*). If the property sheet is displayed, the characteristics of the graphic object selected can be edited. You can display the property sheet by clicking **Properties...** or double-clicking anywhere on the selected object in the display window.

Customizing Complex Graphic Objects

Certain complex graphic objects have a large number of properties. The configuration process for graphic objects like these using the property sheet can be long. To simplify the configuration of complex graphic objects, you can use a customization module. This is a dialog window designed specially to configure associated graphic objects. . This button presents the customization module of the graphic object. When you double-click on a graphic object with a customization module, this is what is presented, not its properties sheet. If a graphic object has a customization module, the only characteristic to appear in its properties sheet is its name.

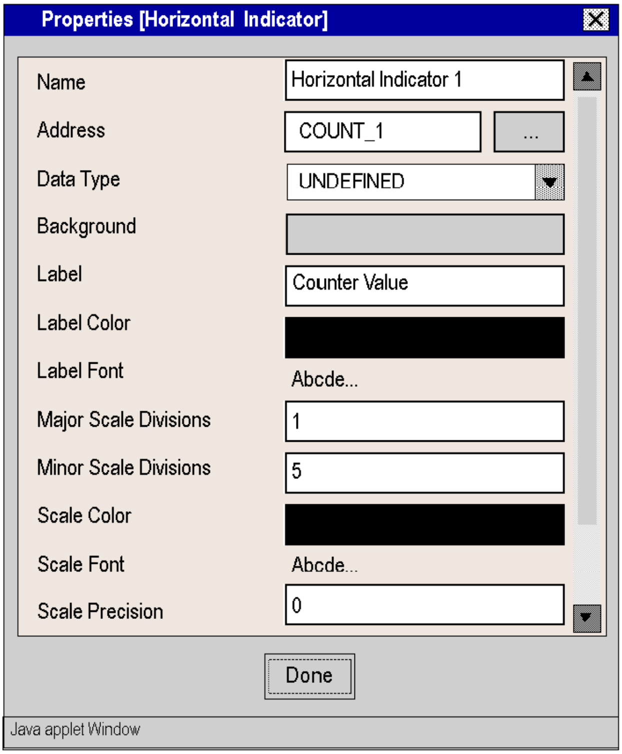
Displaying a Background Image

The Graphic Editor display has a **Background Image** characteristic which can be used to specify an image to appear in the background of the display. This image can be a GIF file or a JPEG file. All files are placed in the /wwwroot directory of the embedded server. For example, if the image "cool.gif" was placed in the /wwwroot/images directory of the embedded server, the background image property must be set to /wwwroot/images.

Property Sheet

Overview

The Property Sheet is a "floating" (non-modal) dialog that presents all the settable properties of the currently selected graphic object.



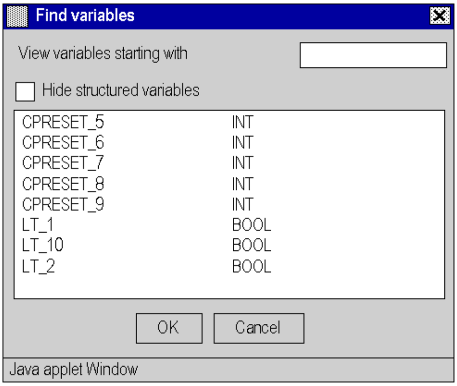
Name	Horizontal Indicator 1
Address	COUNT_1 ...
Data Type	UNDEFINED
Background	▨
Label	Counter Value
Label Color	■
Label Font	Abcde...
Major Scale Divisions	1
Minor Scale Divisions	5
Scale Color	■
Scale Font	Abcde...
Scale Precision	0

The properties of a graphic object are specific to an object's type. The properties are presented in a scrollable list, with the name and the value of each property listed. See *Graphic Objects, page 184* for a description of the graphic objects provided with the Graphic Editor.

Find variables dialog box

For each of the graphic objects provided with the Graphic Editor, a property editor is provided for its **Address** property. This property editor not only allows you to directly enter the address of a Quantum/Premium/Micro register (or Concept/PL7/Unity Pro variable name), but also provides access to the **Find variables dialog box**. The Lookup Dialog allows you to pick a Concept/PL7/Unity Pro symbol (variable) name from a list of symbol (variables) that have been "Web enabled" by Web Designer.

This is the **Find variables dialog box**.



CPRESET_5	INT
CPRESET_6	INT
CPRESET_7	INT
CPRESET_8	INT
CPRESET_9	INT
LT_1	BOOL
LT_10	BOOL
LT_2	BOOL

Security

Security

Your API data is protected by three security measures.

- The HTML page containing the Graphic Editor applet has been placed in a "secure" directory on the Web module; the Web browser user is therefore asked to enter a password enabling him/her to download the HTML page.
- You must enter the correct password in the **Password** dialog box to be able to save/delete the files or send the data values to the connected PLC. For the transfer of data values to the PLC, the Graphic Editor reinforces the "read only" mode by deactivating the user commands of all the graphic objects.
- The FactoryCast configuration program lets you indicate that an element is in read only. The **Graphic Editor** reinforces the "read only" attribute of a symbol variable or address by rejecting all requests in order to define a new value for the data, and warning the user through the message window **Graphic Editor.**

Parameters of the Graphic Editor Applet

Overview

The **Graphic Editor** supports three applet parameters for customizing its behavior. The applet parameters are indicated by the <PARAM> labels in the <APPLET> label of the Graphic Editor HTML page. The parameters recognized by the **Graphic Editor** applet are:

- **LOAD**—This parameter asks the **Graphic Editor** to automatically load a graphic file when it is launched. If the specified file does not exist, a message is delivered. If the parameter does not appear in the <APPLET> label, no graphic file is automatically loaded on startup. In this case you must select the initial table file from the list provided by the **Graphic Editor**.
- **MODE**—This parameter tells the **Graphic Editor** whether it has to start normally in edit mode or in view mode. If it is launched in view mode, it will **only** display the display window. When this parameter is used with the LOAD parameter, a website can be designed using HTML pages that are reserved for specific graphic display. The user is not required to explicitly select any graphic files, which is typical HMI behavior. The possible values for this parameter are
  - **EDIT**—The **Graphic Editor** starts normally in edit mode (default value).
  - **VIEW_RO**—The **Graphic Editor** starts normally in read only. The Web browser user will not be authorized to send values to the PLC.
  - **VIEW_RW**—The **Graphic Editor** starts normally in read/write. The Web browser user will be authorized to send values to the PLC, having first entered the write access password.
- **AUTO_LOGIN**—This parameter tells the **Graphic Editor** to automatically enter the password required to write access the PLC. If the **MODE** parameter is set to **VIEW_RW** or **EDITI** and also set to **TRUE**, the **Graphic Editor** grants write access to the PLC without asking the user to enter the password. The possible values for this parameter are **FALSE** (by default) and **TRUE**.

Example

The following example shows an applet invocation marker that starts the **Graphic Editor** in view mode and automatically changes the graphic file called **UNIT_1**. The Web browser user will be authorized to send values to the PLC though graphic objects that take into account the sending of values if the password for write access in entered.

```
<APPLET codebase="/classes"
archive="SAComm.jar,GDE.jar,Widgets.jar"
code="com.schneiderautomation.gde.GdeApplet"
width="700" height="514">
<PARAM name="LOAD" value="UNIT_1">
<PARAM name="MODE" value="VIEW_RW">
<PARAM name="AUTO_LOGIN" value="FALSE">
</APPLET>
```

Graphic Objects

Presentation

The set of graphic objects provided in the **Graphic Editor** is intended to support building graphic displays that mimic conventional instrument panels. All of the data monitoring and control objects have built-in communication capabilities and are designed as stand-alone graphic objects.

Additionally, to support customers that want to put several simple applets on a single HTML page, each object in the **Graphic Editor** set is provided in an applet version. When used in conjunction with the `LiveBeanApplet`, the **Graphic Editor** graphic objects can be used in the same way as the `LiveLabelApplet`.

This section provides a description of the standard graphic objects and their properties.

Horizontal Indicator

A Horizontal Indicator provides an analog representation of the value of a symbol (variable) or direct address in a PLC by drawing a horizontal bar whose length is proportional to the value as a percentage of its range in engineering units. Optionally, a digital indication of the value can be shown in the center of the bar area.

The properties for the Horizontal Indicator are listed below:

Property	Description	Limits
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Major Scale Divisions	The number of major (labeled) scale divisions	0 to 100
Minor Scale Divisions	The number of minor (unlabeled) scale divisions	0 to 100
Scale Color	The color for the scale and its labels	
Scale Font	The font for scale labels	

Property	Description	Limits
Scale Precision	The number of fractional digits to be shown for scale labels (Set to -1 to use a general exponential format.)	-1 to 6
Maximum EU Value	The maximum value, in engineering units, of the direct address or symbol (variable)	
Minimum EU Value	The minimum value, in engineering units, of the direct address or symbol (variable)	
Maximum PLC Value	The maximum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Minimum PLC Value	The minimum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Value Visible	Indicates whether a digital display of the scaled value is to be shown	
Value Font	The font for the digital display of the value, if shown	
Bar Background	The background color for the bar indicator area	
Bar Color	The color for the indicator bar (when scaled value within High/Low limits)	
High High Limit Value	The value in engineering units for the ’High High’ limit	
High High Limit Color	The color for the indicator bar when scaled value is greater than the ‘High High’ limit	
High Limit Value	The value in engineering units for the ‘High’ limit	
High Limit Color	The color for the indicator bar when scaled value is greater than the ‘High’ limit	
Low Limit Value	The value in engineering units for the ‘Low’ limit	
Low Limit Color	The color for the indicator bar when scaled value is less than the ‘Low’ limit	
Low Low Limit Value	The value in engineering units for the ‘Low Low’ limit	
Low Low Limit Color	The color for the indicator bar when scaled value is less than the ‘Low Low’ limit	
Limit Deadband	The deadband (as percentage of EU range) to apply to High/Low limit checking	0 to 10
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	
PLC Value	A simulated, raw (unscaled) input value for testing the graphic object	See Note 3,*Notes, page 204*

Vertical Indicator

A Vertical Indicator provides an analog representation of the value of a symbol (variable) or direct address in a PLC by drawing a vertical bar whose length is proportional to the value as a percentage of its range in engineering units.

The properties for the Vertical Indicator are listed below:

Property	**Description**	**Limits**
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Major Scale Divisions	The number of major (labeled) scale divisions	0 to 100
Minor Scale Divisions	The number of minor (unlabeled) scale divisions	0 to 100
Scale Color	The color for the scale and its labels	
Scale Font	The font for scale labels	
Scale Precision	The number of fractional digits to be shown for scale labels (Set to -1 to use a general exponential format.)	-1 to 6
Maximum EU Value	The maximum value, in engineering units, of the direct address or symbol (variable)	
Minimum EU Value	The minimum value, in engineering units, of the direct address or symbol (variable)	
Maximum PLC Value	The maximum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Minimum PLC Value	The minimum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Bar Background	The background color for the bar indicator area	
Bar Color	The color for the indicator bar (when scaled value within High/Low limits)	
High High Limit Value	The value in engineering units for the ’High High’ limit	

Property	Description	Limits
High High Limit Color	The color for the indicator bar when scaled value is greater than the ‘High High’ limit	
High Limit Value	The value in engineering units for the ‘High’ limit	
High Limit Color	The color for the indicator bar when scaled value is greater than the ‘High’ limit	
Low Limit Value	The value in engineering units for the ‘Low’ limit	
Low Limit Color	The color for the indicator bar when scaled value is less than the ‘Low’ limit	
Low Low Limit Value	The value in engineering units for the ‘Low Low’ limit	
Low Low Limit Color	The color for the indicator bar when scaled value is less than the ‘Low Low’ limit	
Limit Deadband	The deadband (as percentage of EU range) to apply to High/Low limit checking	0 to 10
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	
PLC Value	A simulated, raw (unscaled) input value for testing the graphic object	See Note 3,*Notes, page 204*

Horizontal or Vertical Slider

A Horizontal or Vertical Slider provides an analog representation of the value of a symbol (variable) or direct address in a PLC by drawing a scroll bar whose "thumb" position is proportional to the value as a percentage of its range in engineering units. With a mouse, a user can change the value of the scroll bar and cause a new value to be sent to the PLC.

The properties for the Horizontal or Vertical Slider are listed below:

Property	Description	Limits
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Major Scale Divisions	The number of major (labeled) scale divisions	0 to 100

Property	Description	Limits
Minor Scale Divisions	The number of minor (unlabeled) scale divisions	0 to 100
Scale Color	The color for the scale and its labels	
Scale Font	The font for scale labels	
Scale Precision	The number of fractional digits to be shown for scale labels (Set to -1 to use a general exponential format.)	-1 to 6
Maximum EU Value	The maximum value, in engineering units, of the direct address or symbol (variable)	
Minimum EU Value	The minimum value, in engineering units, of the direct address or symbol (variable)	
Maximum PLC Value	The maximum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Minimum PLC Value	The minimum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Block Increment	The amount that the scaled value should change when the scroll bar’s scroll area is clicked	
Unit Increment	The amount that the scaled value should change when the scrollbar’s arrow buttons are clicked	
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	

Horizontal or Vertical Selector

A Horizontal or Vertical Selector allows a user to make a selection from a set of choices. When a selection is made, the value corresponding to the choice is sent to the PLC. The choices are shown as labels of a "scale," with the current selection indicated by the position of the "thumb" of a scroll bar.

The properties for the Horizontal or Vertical Selector are listed below:

Property	Description	Limits
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Choices	The choices for the selector. Each choice is given as a ‘label=value’ entry (when a user selects ‘label,’ ‘value’ is sent to PLC)	Minimum of two choices required
Label	The label to be displayed as part of the graphic object	

Property	Description	Limits
Label Color	The color for the label	
Label Font	The font for the label	
Scale Visible	Indicates whether a "scale," labeled with the choices, is to be shown	
Scale Color	The color for the scale and its labels	
Scale Font	The font for scale labels	
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	

Digital Indicator

A Digital Indicator provides a numeric representation of the value of a symbol (variable) or direct address in a PLC. The value can be shown in various formats, and can be made to change color when a preset high or low limit is exceeded.

The properties for the Digital Indicator are listed below:

Property	Description	Limits
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Value Format	The format (decimal, hex, etc.) to use in displaying the scaled value	
Value Precision	The number of fractional digits to be shown for the scaled value (Set to -1 to use a general exponential format.)	-1 to 6
Value Background	The background color for the value display area	
Value Color	The text color for the digital display of the value	
Value Font	The font for the digital display of the value	
Units	The label for the engineering units of the value (appended to the numeric display of the value)	
Maximum EU Value	The maximum value, in engineering units, of the direct address or symbol (variable)	

Property	Description	Limits
Minimum EU Value	The minimum value, in engineering units, of the direct address or symbol (variable)	
Maximum PLC Value	The maximum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Minimum PLC Value	The minimum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
High High Limit Value	The value in engineering units for the ’High High’ limit	
High High Limit Color	The color for the indicator bar when scaled value is greater than the ‘High High’ limit	
High Limit Value	The value in engineering units for the ‘High’ limit	
High Limit Color	The color for the indicator bar when scaled value is greater than the ‘High’ limit	
Low Limit Value	The value in engineering units for the ‘Low’ limit	
Low Limit Color	The color for the indicator bar when scaled value is less than the ‘Low’ limit	
Low Low Limit Value	The value in engineering units for the ‘Low Low’ limit	
Low Low Limit Color	The color for the indicator bar when scaled value is less than the ‘Low Low’ limit	
Limit Deadband	The deadband (as percentage of EU range) to apply to High/Low limit checking	0 to 10
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	
PLC Value	A simulated, raw (unscaled) input value for testing the graphic object	See Note 3,*Notes, page 204*

Message Display

A Message Display shows a text message based on the value of a symbol (variable) or direct address in a PLC. For each specified message, a value is also specified that will trigger its display.

The properties for the Message Display are listed below:

Property	**Description**	**Limits**
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Messages	The set of messages to display. Each message is given as a ‘value=text’ entry (when the PLC value equals ‘value’, ‘text’ is displayed as the message)	Minimum of one message required
Message Background	The background color for the message display area	
Message Color	The color for the message text	
Message Font	The font for the message text	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	
PLC Value	A simulated input value for testing the graphic object	See Note 3,*Notes, page 204*

Push Button

A Push Button allows a user to send preset value(s) to a PLC when clicked with the mouse.

The properties for the Push Button are listed below:

Property	Description	Limits
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Values	The value(s) to send to the PLC	See Note 4,*Notes, page 204*
Reset Values	The value(s) to send to the PLC after the reset delay time has expired. If no reset values are provided, no reset action will occur.	
Reset Delay	The delay time (milliseconds) that the Push Button should wait after sending the value(s) to the PLC before sending the reset value(s).	0-2000
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Button Label	The text label for the button	
Button Background	The color for the knob	0 to 100
Button Label Color	The color for the button label	
Button Label Font	The font for the button label	
Border Width	The width (in pixels) for the graphic object's border	0 to 32
Border Color	The color for the graphic object's border	

Direct Output Station

The Direct Output Station allows a user to enter a value into a text input field directly with their keyboard. When the entered text represents a numeric value that is within preset high and low limits, a **Set** button is enabled. While the **Set** button is enabled, the entered value will be sent to the PLC whenever the user presses either the **Set** button or the ENTER key (if the input field has keyboard input focus).

The properties for the Direct Output Station are listed below:

Property	**Description**	**Limits**
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Maximum EU Value	The maximum value, in engineering units, of the direct address or symbol (variable)	
Minimum EU Value	The minimum value, in engineering units, of the direct address or symbol (variable)	
Maximum PLC Value	The maximum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Minimum PLC Value	The minimum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Maximum Input	The maximum value, in engineering units, that is valid for the entered input value	
Minimum Input	The minimum value, in engineering units, that is valid for the entered input value	
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	

LED

The Indicator Light provides a dual-state indication of the value of a symbol (variable) or direct address in a PLC. Unless the **Input Inverted** property is set to **TRUE**, an input value of zero is deemed **OFF** and a non-zero value is deemed **ON**. If the **Flash Interval** property is set to greater than zero, the light will flash while the input value is on.

The properties for the Indicator Light are listed below:

Property	Description	Limits
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Off Word	The text to show when the input value is off	
Off Word Background	The background color of the light when the **Off Word** is shown	
Off Word Color	The color for the **Off Word** text	
Off Word Font	The font for the **Off Word** text	
On Word	The text to show when the input value is on	
On Word Background	The background color of the light when the **On Word** is shown	
On Word Color	The color for the **On Word** font	
On Word Font	The font for the **On Word** text	
Flash Interval	The flashing time period (in milliseconds) of the light when the input value is on. Set to zero for no flashing.	200 to 2000
Shape	The shape (circle, rectangle, etc.) of the light	
Input Inverted	If **TRUE**, inverts the input value. (Light will show the **Off Word** when input value is on.)	
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	
PLC Value	A simulated input value for testing the graphic object	See Note 3,*Notes, page 204*

Motor Control Station

The Motor Control Station is designed to mimic the typical start/stop push button station that is often used to control motors. This graphic object is essentially a composite of two Push Buttons and an Indicator Light. In order to make it easier to set this object's many properties, a Customizer is provided. All of the properties (except Name) are set with its Customizer, not with the **Graphic Editor's** Property Sheet.

The properties for the Motor Control Station are listed below:

Property	Description	Limits
Name	The name for the graphic object	
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Border Width	The width (in pixels) for the graphic object's border	0 to 32
Border Color	The color for the graphic object's border	
LED	Same properties as the Indicator Light graphic object, excluding the common properties listed above	
Top Push Button	Same properties as the Push Button graphic object, excluding the common properties listed above	
Bottom Push Button	Same properties as the Push Button graphic object, excluding the common properties listed above	

Analog Meter

An Analog Meter provides an analog representation of the value of a symbol (variable) or direct address in a PLC by drawing a pointer on a circular dial whose position is proportional to the value as a percentage of its range in engineering units. The size of the meter's circular dial (degrees sweep of a circle), the colors for the dial, and the style of the pointer can all be set.

The properties for the Analog Meter are listed below:

Property	**Description**	**Limits**
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Major Scale Divisions	The number of major (labeled) scale divisions	0 to 100
Minor Scale Divisions	The number of minor (unlabeled) scale divisions	0 to 100
Scale Color	The color for the scale and its labels	
Scale Font	The font for scale labels	
Scale Precision	The number of fractional digits to be shown for scale labels (Set to -1 to use a general exponential format.)	-1 to 6
Maximum EU Value	The maximum value, in engineering units, of the direct address or symbol (variable)	
Minimum EU Value	The minimum value, in engineering units, of the direct address or symbol (variable)	
Maximum PLC Value	The maximum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Minimum PLC Value	The minimum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Dial Degrees Sweep	The amount of a circular arc to use for drawing the dial	60 to 300
Pointer Type	The type (needle, arrow head, etc.) of pointer to use	
Pointer Color	The color for the pointer	
Dial Color	The color for the dial (that part that is within the High/Low limits)	
High High Limit Value	The value in engineering units for the ‘High’ limit	
High High Limit Color	The color for the indicator bar when scaled value is greater than the ‘High High’ limit	
High Limit Value	The value in engineering units for the ‘High’ limit	

Property	Description	Limits
High Limit Color	The color for the indicator bar when scaled value is greater than the ‘High’ limit	
Low Limit Value	The value in engineering units for the ‘Low’ limit	
Low Limit Color	The color for the indicator bar when scaled value is less than the ‘Low’ limit	
Low Low Limit Value	The value in engineering units for the ‘Low Low’ limit	
Low Low Limit Color	The color for the indicator bar when scaled value is less than the ‘Low Low’ limit	
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	
PLC Value	A simulated, raw (unscaled) input value for testing the graphic object	See Note 3,*Notes, page 204*

Rotary Slider

A Rotary Slider provides an analog representation of the value of a symbol (variable) or direct address in a PLC by drawing a knob on a circular dial whose position is proportional to the value as a percentage of its range in engineering units. The size of the circular dial (degrees sweep of a circle) and knob color can be set. With a mouse, a user can change the position of the knob and cause a new value to be sent to the PLC.

The properties for the Rotary Slider are listed below:

Property	Description	Limits
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Major Scale Divisions	The number of major (labeled) scale divisions	0 to 100
Minor Scale Divisions	The number of minor (unlabeled) scale divisions	0 to 100
Scale Color	The color for the scale and its labels	
Scale Font	The font for scale labels	
Scale Precision	The number of fractional digits to be shown for scale labels (Set to -1 to use a general exponential format.)	-1 to 6
Dial Degrees Sweep	The amount of a circular arc to use for drawing the dial	60 to 300
Dial Color	The color for the dial	
Knob Color	The color for the knob	
Maximum EU Value	The maximum value, in engineering units, of the direct address or symbol (variable)	
Minimum EU Value	The minimum value, in engineering units, of the direct address or symbol (variable)	
Maximum PLC Value	The maximum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Minimum PLC Value	The minimum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3,*Notes, page 204*
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	

Rotary Selector

A Rotary Selector allows a user to make a selection from a set of choices. When a selection is made, the value corresponding to the choice is sent to the PLC. The choices are shown as labels of a "scale," with the current selection indicated by the position of the knob. The size of the circular dial (degrees sweep of a circle) and knob color can be set.

The properties for the Rotary Selector are listed below:

Property	Description	Limits
Name	The name for the graphic object	
Address	The direct address (or the name of a symbol (variable)) to monitor	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable)	See Note 2,*Notes, page 204*
Background	The background color for the graphic object	
Choices	The choices for the selector. Each choice is given as a ‘label=value’ entry (when a user selects ‘label,’ ‘value’ is sent to PLC)	Minimum of two choices required
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Scale Visible	Indicates whether a "scale," labeled with the choices, is to be shown	
Scale Color	The color for the scale and its labels	
Scale Font	The font for scale labels	
Dial Degrees Sweep	The amount of a circular arc to use for drawing the dial	60 to 300
Knob Color	The color for the knob	
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	

Trend Recorder

A Trend Recorder provides a continuous, time-based charting of the value of up to six symbol (variable)s or direct addresses in a PLC. A Trend Recorder emulates a strip-chart recorder, with the pens on the right, and the "paper" moving from right to left. A vertical scale can be shown on the left side of the chart for showing the range of the values being recorded, and a horizontal scale can be shown below the chart for showing the time span of the chart. The rate at which the chart is updated, and the appearance of the chart can be set.

In order to make it easier to set this object’s many properties, a Customizer is provided. All properties (except Name) are set with its Customizer, not with the **Graphic Editor’s** Property Sheet.

The following table describes properties for the Trend Recorder. Properties available for each pen are described in the next table:

Property	Description	Limits
Name	The name for the graphic object	
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Major Scale Divisions	The number of major (labeled) scale divisions	0 to 100
Minor Scale Divisions	The number of minor (unlabeled) scale divisions	0 to 100
Scale Color	The color for the scale and its labels	
Scale Font	The font for scale labels	
Scale Precision	The number of fractional digits to be shown for scale labels (Set to -1 to use a general exponential format.)	-1 to 6
Maximum EU Value	The maximum value, in engineering units, of the direct address or symbol (variable)	
Minimum EU Value	The minimum value, in engineering units, of the direct address or symbol (variable)	
Update Period	The update interval (in seconds) for the chart	0.5 to 120
Time Scale Divisions	The number of horizontal scale divisions	0 to 6
Chart Background	The color for the chart area	
Grid Color	The color of the grid drawn in the chart area	
Vertical Grid Divisions	The number of vertical divisions for the grid	0 to 100
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	

These Trend Recorder properties are available for each pen:

Property	Description	Limits
Address	The direct address (or the name of a symbol (variable)) to monitor.	See Note 1,*Notes, page 204*
Data Type	The data type of the direct address or symbol (variable).	See Note 2,*Notes, page 204*
Maximum PLC Value	The maximum raw (unscaled) value of the direct address or symbol (variable) in the PLC.	See Note 3,*Notes, page 204*
Minimum PLC Value	The minimum raw (unscaled) value of the direct address or symbol (variable) in the PLC.	See Note 3,*Notes, page 204*
Pen Color	The color of the "pen" used to record the scaled value.	
Pen Label	The label used to identify the pen.	

Display Link

A Display Link is a special graphic object that allows the user to switch to another graphic display by clicking on it with a mouse. To indicate to the user that the object is a link to another display, the text label for the link is underlined and the mouse cursor changes to a hand icon when the mouse is moved over it. This object is especially useful when the **Graphic Editor** is used in its **view mode**, where no drop-down list of graphic displays is available for selecting a display.

A Display Link can also be used as a hyperlink to an HTML file. If a URL is entered as the **Link Display Name**, the URL can be opened in a new browser window if the user presses the SHIFT key while they click the link; otherwise, the existing browser window is replaced with the URL when the link is clicked.

If the **Link Display Name** is blank, then the **Label** will be shown as not underlined, and the displayed object becomes a simple text label.

The properties for the Display Link are listed below:

Property	Description	Limits
Label	The label for the link	
Link Display Name	The name of the graphic display to be loaded when the link is clicked, or a URL of a Web page	
Label Color	The color for the label	
Label Font	The font for the label	

Datalogging History

A Datalogging History provides a continuous, time-based charting of the value of up to six symbols (variables) coming from the log file of the Datalogging service. A Datalogging History emulates a strip-chart recorder, with the pens on the right, and the "paper" moving from right to left. A vertical scale can be shown on the left side of the chart for showing the range of the values being recorded, and a horizontal scale can be shown below the chart for showing the time span of the chart.

NOTE: In order to plot the Datalogging History, you must select the Timestamp option in the Datalogging Service configuration window.

In order to make it easier to set this object’s many properties, a Customizer is provided. All properties (except Name) are set with its Customizer, not with the **Graphic Editor’s** Property Sheet.

3 buttons are available in edition and animation mode:

- Reload: the Datalogging History object is a static widget. This button enables to refresh the value used to build the chart.
- +: zoom on the trend. It decreases the time scale in order to have a better vision of a part of the trend.
- -: zoom out on the trend. It increases the time scale in order to have a larger vision of the trend.

If you place the mouse cursor on a point of the trend, a tooltip appears displaying the exact value at that point. Stay pressed and rollover several points to display the tooltips of all of those points.

Releasing the mouse button and rolling over any point will cleanup existing tooltips and display a new one.

Right click on it to make it disappear.

The following table describes properties for the Datalogging History. Properties available for each pen are described in the next table:

Property	Description	Limits
Name	The name for the graphic object	
Background	The background color for the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color for the label	
Label Font	The font for the label	
Major Scale Divisions	The number of major (labeled) scale divisions	0 to 100
Minor Scale Divisions	The number of minor (unlabeled) scale divisions	0 to 100
Scale Color	The color for the scale and its labels	
Scale Font	The font for scale labels	

Property	Description	Limits
Scale Precision	The number of fractional digits to be shown for scale labels (Set to -1 to use a general exponential format.)	-1 to 6
Maximum EU Value	The maximum value, in engineering units, of the symbol (variable)	
Minimum EU Value	The minimum value, in engineering units, of the symbol (variable)	
Time Scale Divisions	The number of horizontal scale divisions	0 to 6
Chart Background	The color for the chart area	
Grid Color	The color of the grid drawn in the chart area	
Vertical Grid Divisions	The number of vertical divisions for the grid	0 to 100
Border Width	The width (in pixels) for the graphic object’s border	0 to 32
Border Color	The color for the graphic object’s border	
Zoom Coefficient	Set a coefficient for the zoom. In example, if the zoom coefficient is set to 2, the timescale will be divided by 2 by clicking **+** in animation mode.	

These Datalogging History properties are available for each pen:

Property	Description	Limits
Name of the CSV file	The name of the CSV file used to build the trend. **Note:** the log file must include timestamps.	
Address	The name of a symbol (variable) to monitor.	
Data Type	The data type of the symbol (variable). **Note:** the data type must be numerical.	
Maximum PLC Value	The maximum raw (unscaled) value of the symbol (variable) in the PLC.	See Note 3,*Notes, page 204*
Minimum PLC Value	The minimum raw (unscaled) value of the symbol (variable) in the PLC.	See Note 3,*Notes, page 204*
Pen Color	The color of the "pen" used to record the scaled value.	
Pen Label	The label used to identify the pen.	

Notes

These are the notes for the chapter.

<table>
<tr><td><b>1.</b></td><td colspan="2">If the Address property of a graphic object is a direct address, the Data Type property is set to UNDEFINED, a default Data Type (BOOL, INT,DINT or REAL based on the implied size of the data value) is used. If the Address property is a symbol (variable) name, the Data Type property does not have to be specified and can be set to UNDEFINED. However, if the Data Type property is specified for a symbol (variable), it must exactly match the symbol (variable)’s actual data type.
If the Address property is a direct address for a discrete PLC reference (Quantum 0x/1x reference), the Data Type property must be set to BOOL. The Data Type property may be set to BOOL only for a discrete PLC reference.</td></tr>
<tr><td rowspan="15"><b>2.</b></td><td colspan="2">The meaning of the possible values of the Data Type property are:</td></tr>
<tr><td><b>Data Type</b></td><td><b>Meaning</b></td></tr>
<tr><td>UNDEFINED</td><td>no data type specified</td></tr>
<tr><td>BOOL</td><td>1-bit discrete (Boolean)</td></tr>
<tr><td>SHORT</td><td>8-bit signed integer</td></tr>
<tr><td>USHORT</td><td>8-bit unsigned integer</td></tr>
<tr><td>INT</td><td>16-bit signed integer</td></tr>
<tr><td>UINT</td><td>16-bit unsigned integer</td></tr>
<tr><td>DINT</td><td>32-bit signed integer</td></tr>
<tr><td>UDINT</td><td>32-bit unsigned integer</td></tr>
<tr><td>REAL</td><td>32-bit IEEE floating point</td></tr>
<tr><td>TIME</td><td>32-bit unsigned integer (in milliseconds)</td></tr>
<tr><td>DATE</td><td>Date (32-bit BCD)</td></tr>
<tr><td>TOD</td><td>Time of Day (32-bit BCD)</td></tr>
<tr><td>DT</td><td>Date and Time (64-bit BCD)</td></tr>
<tr><td><b>3.</b></td><td colspan="2">The limits for the Maximum PLC Value and Minimum PLC Value properties are the natural limits of the Data Type property that is set. A Data Type setting of UNDEFINED is treated as a REAL with respect to its limit values.</td></tr>
<tr><td><b>4.</b></td><td colspan="2">For a Push Button, a minimum of one value must be provided. If the Address property is a symbol (variable) name, then only one value will ever be sent to the PLC, and any additional values are ignored. If the Address property is a direct address, then all of the values provided will be sent to the PLC as an array of values starting at the specified direct address.</td></tr>
</table>

Extended Graphic Objects

Overview

The set of extended graphic objects provided in Graphic Editor is intended to support building graphic displays that mimic advanced graphic panels. All of the data monitoring and control objects have built-in communication capabilities and are designed as standalone graphic objects.

Additionally, to support customers who want to put several simple applets on a single HTML page, each object in the Graphic Editor set is provided in an applet version. When used in conjunction with the `LiveBeanApplet`, Graphic Editor graphic objects can be used in the same way as the `LiveLabelApplet`.

Be aware, however, that if communication to the device linked to the graphic object is lost, the object becomes inoperative without the end device’s knowledge.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not use graphic objects in situations where loss of communication to the module can affect human or material integrity.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

For example, say you have programmed a pushbutton object to jog a motor when the button is depressed and to stop jogging when the button is released. If communications are lost while the button is depressed, the motor will continue to jog even when the button is released. Graphic objects should not be used to control situations such as this unless other srike interlock methods are installed in the system.

Extended Analog Meter

An extended analog meter provides an analog representation of the value of a symbol (variable) or direct address in a PLC. It draws a pointer on a circular dial whose position is proportional to the value as a percentage of its range in engineering units. The size of the meter’s circular dial (measured in degrees of a circle), the dial colors, and the pointer style can all be set.

The properties of the extended analog meter are listed below:

Property	**Description**	**Limits**
Name	The name of the graphic object	
Address	The direct address or the name of a symbol (variable) to monitor	See Note 1, *Notes, page 216*
Data Type	The data type of the direct address or symbol (variable)	See Note 2, *Notes, page 216*
Background	The background color of the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The label color	
Label Font	The label font	
Maximum EU Value	The maximum value, in engineering units, of the direct address or symbol (variable)	
Minimum EU Value	The minimum value, in engineering units, of the direct address or symbol (variable)	
Maximum PLC Value	The maximum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3, *Notes, page 216*
Minimum PLC Value	The minimum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3, *Notes, page 216*
Bitmap Choices	The bitmap to display	
Dial Degrees Sweep	The range in which degrees vary	
Pointer Type	The graphic display of the pointer	
Pointer Color	The color of the pointer	
Border Width	The width (in pixels) of the graphic object’s border	0 to 32
Border Color	The color of the graphic object’s border	
PLC Value	The default value of the PLC	

ASCII Text Writer

The ASCII Text Writer is based on the message display widget. It allows you to input new text.

The properties of the ASCII Text Writer are listed below:

Property	Description	Limits
Name	The name of the graphic object	
Address	The direct address or the name of a symbol (variable) to monitor	See Note 1, *Notes, page 216*
Max. Text Length	The maximum length of the text	
Text Color	The color of the text	
Text Font	The font used for the text	
Swap Bytes	False if target order of bytes is same as PC	
Value	The text itself	

Bar Graph

A bar graph provides an analog representation of the value of a symbol (variable) or direct address in a PLC. It draws a vertical bar whose length is proportional to the value as a percentage of its range in engineering units.

The properties of the bar graph are listed below:

Property	Description	Limits
Name	The name of the graphic object	
Address	The direct address or the name of a symbol (variable) to monitor	See Note 1, *Notes, page 216*
Data Type	The data type of the direct address or symbol (variable)	See Note 2, *Notes, page 216*
Background	The background color of the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color of the label	
Label Font	The font used on the label	
Maximum EU Value	The maximum value, in engineering units, of the direct address or symbol (variable)	
Minimum EU Value	The minimum value, in engineering units, of the direct address or symbol (variable)	
Maximum PLC Value	The maximum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3, *Notes, page 216*
Minimum PLC Value	The minimum raw (unscaled) value of the direct address or symbol (variable) in the PLC	See Note 3, *Notes, page 216*

Property	Description	Limits
Bar Background	The background color of the bar indicator area	
Bar Color	The color of the indicator bar (when scaled value is within High/Low limits)	
High High Limit Value	The value of the High High limit in engineering units	
High High Limit Color	The color of the indicator bar when the scaled value is greater than the High High limit	
High Limit Value	The value of the High limit in engineering units	
High Limit Color	The color of the indicator bar when scaled value is greater than the High limit	
Low Limit Value	The value of the Low limit in engineering units	
Low Limit Color	The color of the indicator bar when the scaled value is less than the Low limit	
Low Low Limit Value	The value of the Low Low limit in engineering units	
Low Low Limit Color	The color of the indicator bar when the scaled value is less than the Low Low limit	
Limit Deadband	The deadband (as percentage of EU range) to apply to High/Low limit checking	0 to 10
Border Width	The width (in pixels) of the graphic object’s border	0 to 32
Border Color	The color of the graphic object’s border	
PLC Value	A simulated, raw (unscaled) input value for testing the graphic object	See Note 3, *Notes, page 216*

Bitmap

The bitmap widget displays a static bitmap on the screen.

The properties of the bitmap widget are listed below:

Property	Description	Limits
Name	The name of the graphic object	
Background	The background color of the graphic object	See Note 1, *Notes, page 216*
Label	The label to be displayed as part of the graphic object	
Label Color	The color of the label	
Label Font	The font used for the label	
Border Width	The width (in pixels) of the graphic object's border	
Border Color	The color of the graphic object's border	
Bitmap Choices	File names of custom bitmaps to display. Refer to the next paragrah for information on adding images into the module.	

Adding Images

You can add images into the module in one of the following way:

- Add your images into the images.zip file (path: *FLASH1/wwwroot*).
- Create a directory into the module (i.e. *FLASH1/wwwroot/bitmaps*). Copy your images into this directory. In this case, you need to specify the path of the images you want to use (i.e. *FLASH1/wwwroot/bitmaps/key.gif*).

Step	Action
1	Create an *images* folder on your PC.
2	Copy the images you want to use in this folder.
3	Import the user.jar file from the module to the PC (path: *NAND/FLASH1/wwwroot/classes*) using a FTP client.
4	Open the *user.jar* file using a file archiver.
5	Drag and drop the images folder in the *user.jar* file. Make sure the relative path of the image files is 'images/'.
6	Transfer the *user.jar* file back to the module using a FTP client.

Generic Bitmap

The generic bitmap widget lets you display one static bitmap for each distinct value of a PLC variable. It can be used to display dynamic animations, for instance the changing level in a tank.

The properties of the Generic Bitmap widget are listed below:

Property	Description	Limits
Name	The name of the graphic object	
Address	The direct address or the name of a symbol (variable) to monitor	See Note 1, *Notes, page 216*
Data Type	The data type of the direct address or symbol (variable)	See Note 2, *Notes, page 216*
Background	The background color of the graphic object	See Note 1, *Notes, page 216*
Label	The label to be displayed as part of the graphic object	
Label Color	The color of the label	
Label Font	The font used for the label	
Bitmap Choices	File names of custom bitmaps to display. Refer to the next paragrah for information on adding images into the module.	
Border Width	The width (in pixels) of the graphic object’s border	
Border Color	The color of the graphic object’s border	
PLC Value	A simulated input value for testing the graphic object behavior	

Graphic Link

A graphic link is a special graphic object that lets you switch to another graphic display by clicking on it. Graphic links can also be recognized by their underlined labels, and the mouse cursor changes to a hand icon when the mouse moves over them. This object is especially useful when Graphic Editor is used in view mode, where no drop-down list of graphic displays is available.

A graphic link can also be used as a hyperlink to an HTML file. If a URL is entered as the **Link Display Name**, the URL can be opened in a new browser window by simultaneously pressing the SHIFT key and clicking the link. Otherwise, the URL opens in the existing browser window when the link is clicked.

If the **Link Display Name** is blank, then the label is not underlined, and the displayed object becomes a simple text label.

The properties of the display link are listed below:

Property	Description	Limits
Label	The link label	
Link Display Name	The name of the graphic display to be loaded when the link is clicked, or the URL of a Web page	
Label Color	The color of the label	
Label Font	The font used for the label	
Bitmap Choices	The filename of the bitmap on which to click	

Indicator Light

The indicator light displays the value of a symbol (variable) or direct address in a PLC with two possible states. An input value of 0 is considered off and a non-zero value is considered on. If the **Flash Interval** property is set to a value greater than 0, the light flashes while the input value is on. There is a bitmap for the on state and a different one for the off state.

The properties of the indicator light are listed below:

Property	Description	Limits
Name	The name of the graphic object	
Address	The direct address or the name of a symbol (variable) to monitor	See Note 1, *Notes, page 216*
Data Type	The data type of the direct address or symbol (variable)	See Note 2, *Notes, page 216*
Background	The background color of the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color of the label	
Label Font	The font used for the label	
OFF Word	The text to display when the input value is off	
OFF Bitmap Choice	The light bitmap when the OFF word is displayed	
OFF Word Color	The color of the OFF word font	
OFF Word Font	The font used for the OFF word text	
ON Word	The text to display when the input value is on	
ON Bitmap Choice	The light bitmap when the ON word is displayed	
ON Word Color	The color of the ON word font	
ON Word Font	The font used for the ON word text	

Property	Description	Limits
Flash Interval	The flashing time period (in ms) of the light when the input value is on. Set to 0 for no flashing.	200 to 2000
Input Inverted	If true, inverts the input value. (Light will show the OFF word when input value is on.)	
Border Width	The width (in pixels) of the graphic object's border	0 to 32
Border Color	The color of the graphic object's border	
PLC Value	A simulated input value for testing the graphic object	See Note 3, *Notes, page 216*

Motor

The motor widget displays the value of a symbol (variable) or direct address in a PLC with three possible states. An input value of 0 is considered off, a value of 1 is considered on, and other values are considered default. The three states are represented by different bitmaps.

The properties of the motor widget are listed below:

Property	Description	Limits
Name	The name of the graphic object	
Address	The direct address or the name of a symbol (variable) to monitor	See Note 1, *Notes, page 216*
Data Type	The data type of the direct address or symbol (variable)	See Note 2, *Notes, page 216*
Background	The background color of the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color of the label	
Label Font	The font used for the label	
OFF Word	The text to display when the input value is off	
OFF Bitmap Choice	The motor bitmap when the OFF word is displayed	
OFF Word Color	The color of the OFF word font	
OFF Word Font	The font used for the OFF word text	
ON Word	The text to display when the input value is ON	
ON Bitmap Choice	The motor bitmap when the ON word is displayed	
ON Word Color	The color of the ON word font	
ON Word Font	The font used for the ON word text	
DEFAULT Word	The text to display when the input value is ON	

Property	Description	Limits
DEFAULT Bitmap Choice	The motor bitmap when the DEFAULT word is displayed	
DEFAULT Word Color	The color of the DEFAULT word font	
DEFAULT Word Font	The font used for the DEFAULT word text	
Border Width	The width (in pixels) of the graphic object’s border	0 to 32
Border Color	The color of the graphic object’s border	
PLC Value	A simulated input value for testing the graphic object	See Note 3, *Notes, page 216*

Pipe

The pipe displays the value of a symbol (variable) or direct address in a PLC with two possible states. An input value of 0 is considered off and a non-zero value is considered on. There is a bitmap for the on-state and a different one for the off-state.

The properties of the pipe are listed below:

Property	Description	Limits
Name	The name of the graphic object	
Address	The direct address or the name of a symbol (variable) to monitor	See Note 1, *Notes, page 216*
Data Type	The data type of the direct address or symbol (variable)	See Note 2, *Notes, page 216*
Background	The background color of the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color of the label	
Label Font	The font used for the label	
OFF Word	The text to display when the input value is off	
OFF Bitmap Choice	The pipe bitmap when the OFF word is displayed	
OFF Word Color	The color of the OFF word font	
OFF Word Font	The font used for the OFF word text	
ON Word	The text to display when the input value is on	
ON Bitmap Choice	The pipe bitmap when the ON word is displayed	
ON Word Color	The color of the ON word font	
ON Word Font	The font used for the ON word text	

Property	Description	Limits
Border Width	The width (in pixels) of the graphic object’s border	0 to 32
Border Color	The color of the graphic object’s border	
PLC Value	A simulated input value for testing the graphic object	See Note 3, *Notes, page 216*

Push Button

A push button sends preset value(s) to a PLC when the user clicks it with the mouse.

The properties of the Push Button are listed below:

Property	Description	Limits
Name	The name of the graphic object	
Address	The direct address or the name of a symbol (variable) to monitor	See Note 1, *Notes, page 216*
Data Type	The data type of the direct address or symbol (variable)	See Note 2, *Notes, page 216*
Background	The background color of the graphic object	
Values	The value(s) to send to the PLC	See Note 4, *Notes, page 216*
Reset Values	The value(s) to send to the PLC after the reset delay time has expired. If no reset values are provided, no reset action will occur.	
Reset Delay	The delay time (in milliseconds) that the Push Button should use after sending the value(s) to the PLC before sending the reset value(s).	0-2000
Label	The label to be displayed as part of the graphic object	
Label Color	The color of the label	
Label Font	The font used for the label	
Button Label	The text label for the button	
Button Label Color	The color of the button label	
Button Label Font	The font used for the button label	
OFF Bitmap Choice	The button bitmap when the OFF state is displayed	
ON Bitmap Choice	The button bitmap when the ON state is displayed	
Border Width	The width (in pixels) of the graphic object’s border	0 to 32
Border Color	The color of the graphic object’s border	

Valve

The valve displays the value of a symbol (variable) or direct address in a PLC with two possible states. An input value of 0 is considered off and a non-zero value is considered on. There is a bitmap for the on state and a different one for the off state.

The properties of the valve are listed below:

Property	Description	Limits
Name	The name of the graphic object	
Address	The direct address or the name of a symbol (variable) to monitor	See Note 1, *Notes, page 216*
Data Type	The data type of the direct address or symbol (variable)	See Note 2, *Notes, page 216*
Background	The background color of the graphic object	
Label	The label to be displayed as part of the graphic object	
Label Color	The color of the label	
Label Font	The font used for the label	
OFF Word	The text to display when the input value is off	
OFF Bitmap Choice	The valve bitmap when the OFF word is displayed	
OFF Word Color	The color of the OFF word font	
OFF Word Font	The font used for the OFF word text	
ON Word	The text to display when the input value is ON	
ON Bitmap Choice	The valve bitmap when the ON word is displayed	
ON Word Color	The color of the ON word font	
ON Word Font	The font used for the ON word text	
Flash Interval	The flashing time period (in ms) of the light when the input value is on. Set to 0 for no flashing.	200 to 2000
Border Width	The width (in pixels) of the graphic object’s border	0 to 32
Border Color	The color of the graphic object’s border	
PLC Value	A simulated input value for testing the graphic object	See Note 3, *Notes, page 216*

Notes

These are the notes for the chapter.

<table>
<tr><td>**1.**</td><td colspan="2">If the address property of a graphic object is a direct address and the data type property is set to **UNDEFINED**, a default data type (**BOOL**, **INT**, **DINT**, or **REAL** based on the implied size of the data value) is used. If the address property is a symbol (variable) name, the data type property does not have to be specified and can be set to **UNDEFINED**. However, if the data type property is specified for a symbol (variable), it must match the symbol's (variable's) actual data type exactly.
If the address property is a direct address for a discrete PLC reference (Quantum 0x/1x reference), the data type property must be set to **BOOL**. The data type property may be set to **BOOL** only for a discrete PLC reference.</td></tr>
<tr><td rowspan="15">**2.**</td><td colspan="2">The meaning of the possible values of the Data Type property are:</td></tr>
<tr><td>**Data Type**</td><td>**Meaning**</td></tr>
<tr><td>UNDEFINED</td><td>no data type specified</td></tr>
<tr><td>BOOL</td><td>1-bit discrete (Boolean)</td></tr>
<tr><td>SHORT</td><td>8-bit signed integer</td></tr>
<tr><td>USHORT</td><td>8-bit unsigned integer</td></tr>
<tr><td>INT</td><td>16-bit signed integer</td></tr>
<tr><td>UINT</td><td>16-bit unsigned integer</td></tr>
<tr><td>DINT</td><td>32-bit signed integer</td></tr>
<tr><td>UDINT</td><td>32-bit unsigned integer</td></tr>
<tr><td>REAL</td><td>32-bit IEEE floating point</td></tr>
<tr><td>TIME</td><td>32-bit unsigned integer (in milliseconds)</td></tr>
<tr><td>DATE</td><td>Date (32-bit BCD)</td></tr>
<tr><td>TOD</td><td>Time of Day (32-bit BCD)</td></tr>
<tr><td>DT</td><td>Date and Time (64-bit BCD)</td></tr>
<tr><td>**3.**</td><td colspan="2">The limits for the Maximum PLC Value and Minimum PLC Value properties are the natural limits of the Data Type property that is set. A Data Type setting of UNDEFINED is treated as REAL with respect to its limit values.</td></tr>
<tr><td>**4**</td><td colspan="2">For a Push Button, a minimum of one value must be provided. If the Address property is a symbol (variable) name, then only one value will ever be sent to the PLC, and any additional values will be ignored. If the Address property is a direct address, then all of the values provided will be sent to the PLC as an array, starting at the specified direct address.</td></tr>
</table>

6.5 Graphic Viewer

Graphic Viewer

Overview

Graphic Viewer is a lighter version of the Graphic Editor. Its small size enables you to download it faster. With Graphic Viewer you can only display widgets, you can not edit them.

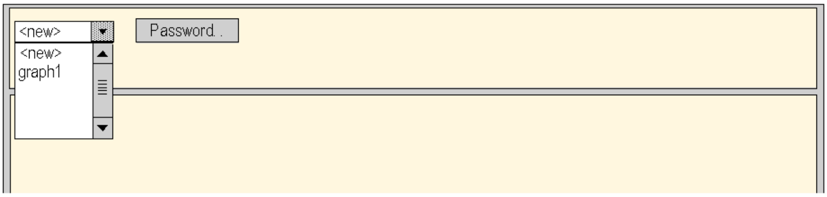
Graphic Viewer is divided into two windows:

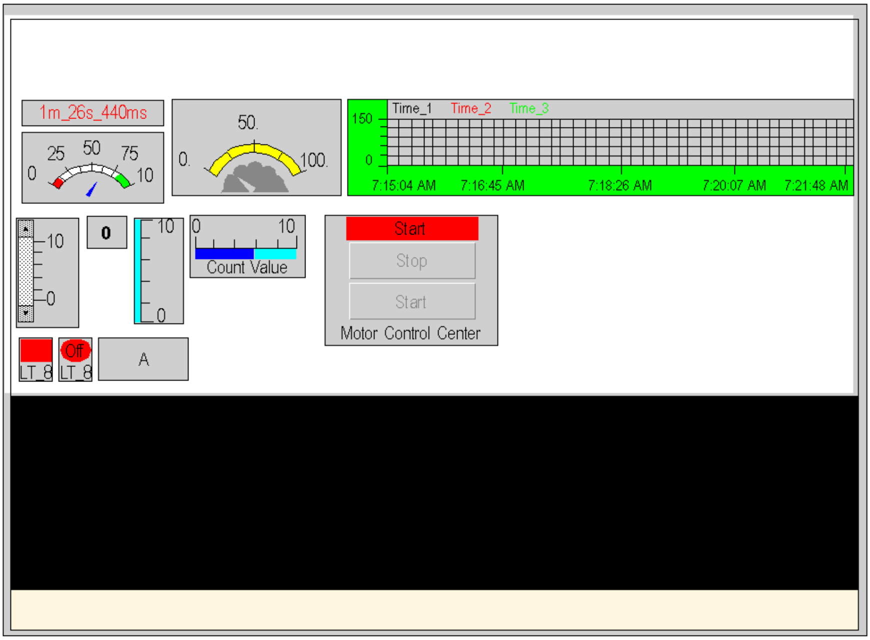
- **Top Window**: This area shows the user controls. This is where you can select a graphic to edit from a drop-down menu.
- **Display Window**: This area displays the selected graphic.

When viewing a graphic in full-screen mode, the **Top Window** is not displayed.

Selecting a Graphic

The following instructions tell you how to select a graphic created by the Graphic Editor:

Step	Action
1	Click on **Graphic Viewer**. The **Top and Display Windows** appear: 

Step	Action
2	Select a graphic from the drop-down menu. **Result**: The selected graphic appears in the **Display Window**:  1m_26s_440ms 0 25 50 75 10 50. 0. 100. 150 0 Time_1 Time_2 Time_3 7:15:04 AM 7:16:45 AM 7:18:26 AM 7:20:07 AM 7:21:48 AM 10 0 0 10 0 0 10 Count Value Start Stop Start Motor Control Center LT_8 Off LT_8 A **Note**: Double-click in the **Display Window** area in the active graphic (unless the active graphic is a command widget) to refresh the **Top Window**.

6.6 Operator Screens

How to Access the Operator Screens

Introduction

An operator screen is a window just as any other editor (configuration, language, data editor). You can import and display operator screens created with UnityPro in the website. This feature enables you to visualize and access operator screens during run time using a simple Web browser.

NOTE: You must import Operator Screens from UnityPro using Web Designer. Refer to the *Web Designer User Manual* for more information.

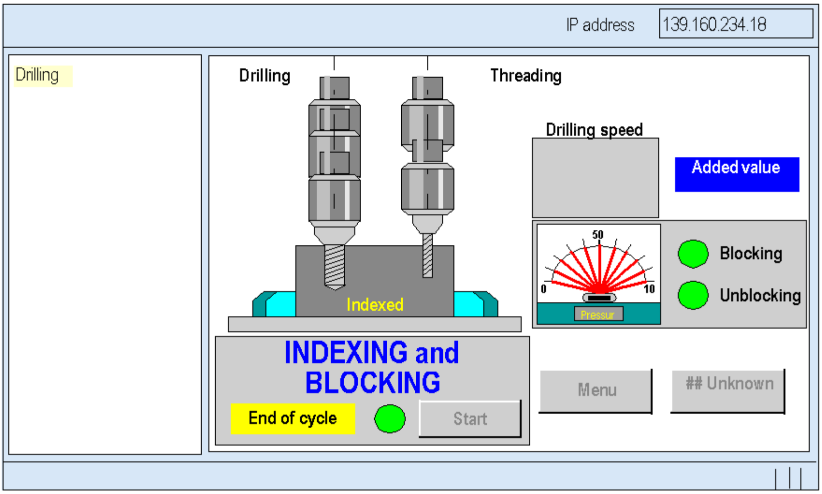
Methodology

The following table lists the operations to perform in order to access operator screens.

Step	Action
1	Launch your Web browser.
2	Connect to the embedded website of the module.
3	Click **Monitoring** in the horizontal menu.
4	Click **Operator Screens** in the vertical menu. **Result**: A list of the imported screen(s) appears.
5	Select a screen in the list by clicking it. **Result**: The selected operator screen appears.

Example of Screen

The following screen shows an example of operator screens.



6.7 PLC Program Viewer

PLC Program Viewer

Presentation

The PLC program viewer feature enables you to visualize and monitor UnityPro programs in run mode using a Web browser. The PLC programs are displayed and animated as they are in UnityPro

PLC programs developed in any languages supported by UnityPro can be visualized:

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

NOTE: You must import PLC programs from UnityPro using Web Designer. Refer to the *Web Designer User Manual* for more information.

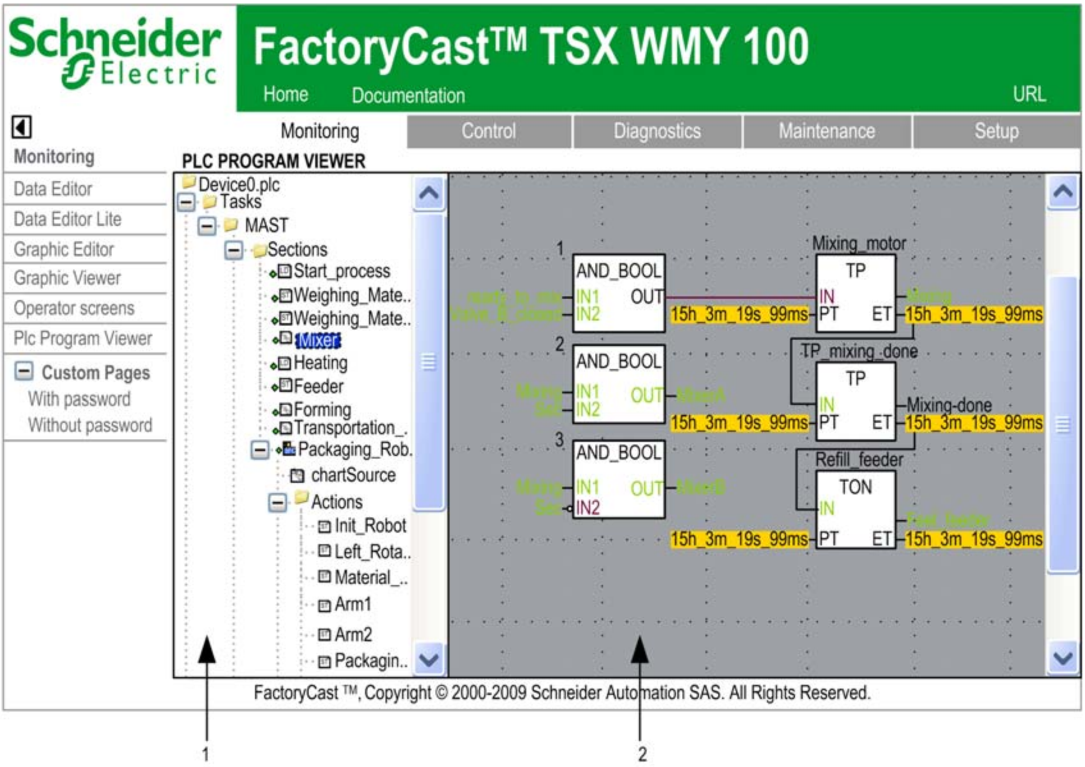
Accessing the PLC Program Viewer

The procedure below shows how to access the PLC Program Viewer page:

Step	Action
1	Launch the website of your module using a Web browser.
2	Click the **Monitoring** link on the Home page of the website.
3	Click the **PLC Program Viewer** link on the Monitoring page.

Representation and Color Convention

The PLC Program Viewer window:



1 Navigation tree: select the section of the PLC program you want to visualize

2 Display window: this zone display the animated PLC program

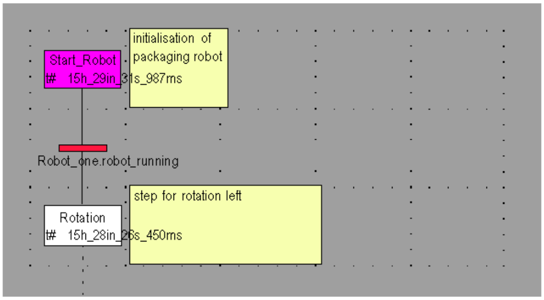
Variables animation:

- Boolean are displayed in:
  - green if its value is true
  - red if its value is false
- Other types display the name or the value of the variable in yellow. Use the tool tip to see more information as the name of the variable, its type, its address and its comment.

Links animation:

- Links connected to boolean variables are displayed in green or red depending on the value of the variable they are connected to (green if true red if false).
- Other links are displayed in black.

SFC animation:



The colors used for the different elements are:

- for steps:
  - green if the step is active,
  - white if the step is inactive,
  - yellow if the activity time of the step is less than the minimum programmed time,
  - pink if the activity time of the step is greater than the minimum programmed time.
- for macro-steps:
  - when a macro-step becomes active the upper half is shown in green,
  - when the OUT step of the macro-step is active the whole of the macro-step is shown in green,
  - when the macro-step becomes inactive it is then shown in white.
- for transitions associated with a Boolean element or a simple Boolean expression:
  - green if the element or the expression is TRUE,
  - red if the element or the expression is FALSE.
- for transitions associated with a section:
  - black as long as the previous step remains inactive,
  - green if the conditions in the section are TRUE,
  - red if the conditions in the section are FALSE.

Tool Tip

The tool tip function is a help bubble which is displayed when you move the cursor over a variable.

It displays information about:

- The value of the variable if only its name is visible in the viewer.
- The type, name, address and comment if only its value is visible in the viewer.

Click on the variable to display the bubble permanently. Right click on it to make it disappear.

Limitations

- Only PLC programs created using UnityPro 4.0 or later can be viewed.
- You can only monitor PLC programs, changes are not allowed.
- The following objects are not animated, they appear in black:
  - Objects for which the result depends on an expression
  - Function blocks without an stance for which there is no information on the input/output variables
  - Standard DFB (i.e. , ALARM_DIA)
  - Multiple dimension tables

NOTE: In case of application signature error due to a limited modification, the PLC program will try to synchronize with the new address of its variables in order to animate the program. This synchronization will be done for each section displayed by the PLC program.

Adding Custom Pages to the Site

7

Overview

You may choose to add your own Web pages to the site on the embedded Server. Web Designer for FactoryCast HMI allows you to protect these pages with the same passwords as the default pages. You can also put them in an unprotected area where anyone can view them without a password.

FactoryCast provides you animated graphical objects. These objects are written in HTML so you can use them in your own Web pages. They enable you to monitor and control PLC variables by associating PLC variables with objects.

This section discusses how to create you own HTML web pages and how to use the real time animated objects provided with FactoryCast.

NOTE: When planning to design Web pages, be sure to keep them within the limits of the memory available for customization.

What's in this Chapter?

This chapter contains the following sections:

7.1 Using FactoryCast’s Java Applets Using HTML Code

Overview

This section describes how to use the animated objects (also called Java applets) that come with FactoryCast. Use these applets to create custom Web pages. To create custom pages, you can use any plain text editor, such as Microsoft Notepad or a dedicated HTML editor, such as Microsoft FrontPage.

Creating custom Web pages with HTML makes it possible to view live PLC data in your browser.

What’s in this Section?

This section contains the following topics:

Inserting Applets on a Web Page

Overview

FactoryCast software includes several graphic objects (or JavaBeans) that are used to visualize live PLC data on a graphic display. Generally, an end-user builds a graphic display using the **Graphic Editor** tool that comes with FactoryCast. However, you can also build a graphic display using graphic objects inserted into a Web page via the LiveBeanApplet. Any of the graphic objects, such as an Analog Meter or a Push Button, can be inserted into a Web page so that dynamic data may be visualized outside the context of the Graphic Editor.

To view live PLC data with a browser, you may opt for one of two methods to insert a FactoryCast applet on a Web page.

- (1) Enter the HTML code found in these sections:
  - Inserting a `LiveBeanApplet` using HTML code
  - Inserting a `LiveLabelApplet` using HTML code
  - Inserting a `LiveTableApplet` using HTML code
- (2) Insert a Java applet and then fill in the dialog boxes using the FactoryCast extension for Microsoft FrontPage or Expression Web.

Inserting LiveBeanApplet

Overview

This topic discusses general concepts about inserting a `LiveBeanApplet`.

NOTE:

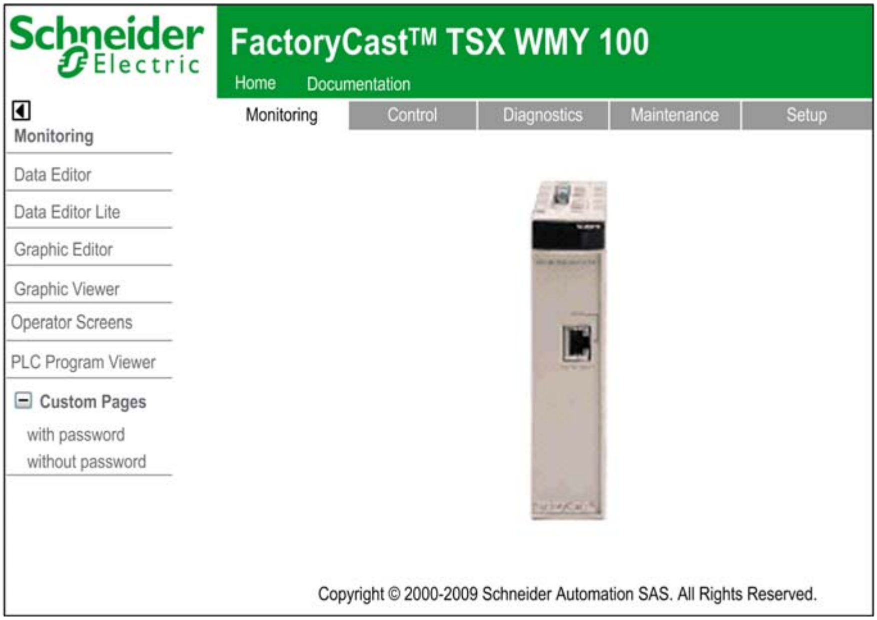
Elsewhere in this guide is information for inserting a `LiveBeanApplet` using:

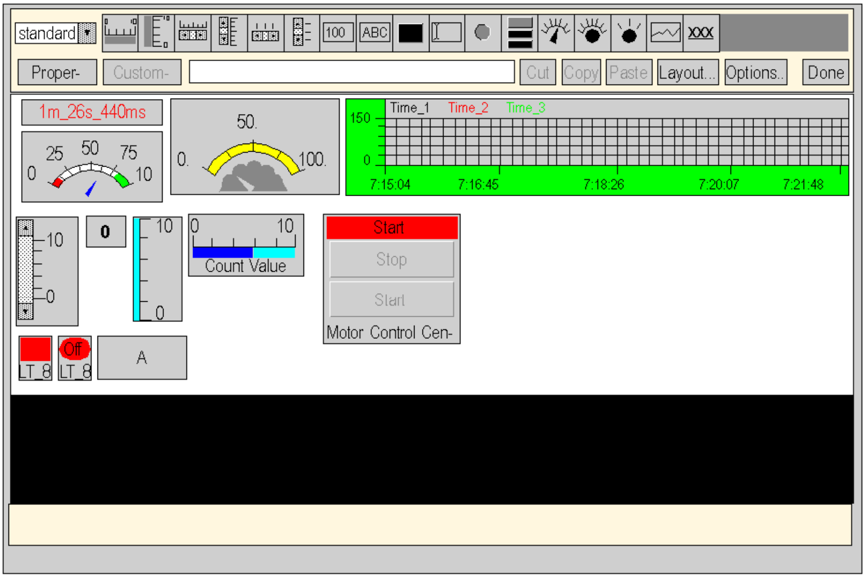
- HTML *(see page 231)*
- Microsoft's FrontPage or Expression Web Extension *(see page 252)*

NOTE: Prior to inserting a `LiveBeanApplet` into a Web page, you must create a JavaBeans library using the Graphic Editor that comes with the FactoryCast software. Generally a user will create a JavaBeans library that has one instance of every object that they would like to use in a Web page. Think of this library as a set of templates that are copied to and customized for your Web pages. For example a library may have one analog meter, one rotary selector, and one push button. Multiple instances of each bean can then be added to a Web page, each with a set of unique parameters such as an address.

Inserting a LiveBeanApplet

To insert a `LiveBeanApplet`:

Step	Action
1	Using your Web browser, enter your Web server module’s name or IP address and go to the Graphic Editor applet found via the Monitoring and Online Editors Web link.  FactoryCast™ TSX WMY 100 Home Documentation Monitoring Control Diagnostics Maintenance Setup Monitoring Data Editor Data Editor Lite Graphic Editor Graphic Viewer Operator Screens PLC Program Viewer Custom Pages with password without password Copyright © 2000-2009 Schneider Automation SAS. All Rights Reserved.

Step	Action
2	Enter the **Edit** mode and create a JavaBeans library you want to use on your Web pages. Customize each bean with a common set of parameters so that you only have to configure unique parameters, such as the register address.  standard 100 ABC XXX Proper- Custom- Cut Copy Paste Layout... Options.. Done 1m_26s_440ms 0 25 50 75 10 50. 0. 100. 150 0 Time_1 Time_2 Time_3 7:15:04 7:16:45 7:18:26 7:20:07 7:21:48 10 0 0 10 0 0 10 Count Value Start Stop Start Motor Control Cen- LT_8 Off LT_8 A **Note:** For help using the Graphic Editor applet, see the section Graphic Editor *(see page 168)*.
3	Once you have created your library, click the **Done** button in the editor window and save your library.

Inserting LiveBeanApplet using HTML Code

Overview

To graphically visualize data, use graphic objects such as Analog Meter or Push Button. Before any beans are inserted into a Web page, the special applet called `LiveBeanMgrApplet` must be inserted into the server.

LiveBeanMgrApplet

The `LiveBeanMgrApplet` allows the Web page to display dynamic data from the controller. This applet **must** be included **once** on the page if any instances of `LiveBeanApplet` are included in the page.

The `LiveBeanMgrApplet` can be included on a Web page in two possible forms.

- Invisible applet—if the Web page is used only to monitor PLC values, then no input is needed from the user
- Icon of a key—if the Web page is used both to send new values and to monitor values to the PLC, then input is needed from the user in order to send new values.

NOTE: If the applet is used as an icon of a key, the user enters a password in order to send values to the PLC. From the Web browser click on the applet (icon of a key), a dialog box appears requesting the user to enter a password; entering the password enables the user to write to the PLC.

Here is the HTML code that you use to include the applet on a Web page that is used only for monitoring:

```
<APPLET codebase="/classes"
archive="SAComm.jar,GDE.jar,Widgets.jar"
code="com.schneiderautomation.gde.LiveBeanMgrApplet"
width=0 height=0>
</APPLET>
```

Here is the HTML code that you use to include the applet on a Web page that is used for sending values to a PLC as well as monitoring:

```
<APPLET codebase="classes"
archive="SAComm.jar,GDE.jar,Widgets.jar"
code="com.schneiderautomation.gde.LiveBeanMgrApplet"
width=32 height=32>
<PARAM name=MODE value="READWRITE">
<PARAM name=AUTO_LOGIN value="FALSE">
</APPLET>
```

In the above example, if the value of the **AUTO_LOGIN** parameter is set to **TRUE**, instead of **FALSE**, then the password is automatically entered and the user is not required to enter the password.

LiveBeanApplet

The `LiveBeanApplet` is included one time for each symbol (variable) or direct address monitored/controlled on the Web page. For instance, if you are monitoring three symbols (variables), you would include the applet three times. `LiveBeanApplet` allows any graphic object/Java Bean that was created with the **Graphic Editor** to be included on a Web page as a separate applet. (See *Data Editor, page 144*, for information on the **Graphic Editor**.) Any graphic object that has been saved as part of a **Graphic Editor** graphic display can be retrieved from the graphic file and presented by the applet.

LiveBeanApplet Parameters

The `LiveBeanApplet` uses parameters that allow you to specify the graphic object to be presented by the applet and to set the applet's background color.

The applet's parameters and their meanings are shown below.

Parameter...	Defines...
LIBRARY	The name of the graphic display which contains the graphic object that is to be presented by the applet. (This will be the same name that was used when the graphic display was saved with the **Graphic Editor**.) **This parameter is required.**
BEAN	The name of the graphic object that is to be retrieved from the graphic display specified by the LIBRARY parameter. (This will be the name that appears as the 'Name' property of the graphic object.) **This parameter is required.**
BACKGRND	The background color for the applet. Acceptable values are WHITE, LT_GRAY, GRAY, DK_GRAY, BLACK, RED, PINK, ORANGE, YELLOW, GREEN, MAGENTA, CYAN, and BLUE. Also, a RGB color value can be entered using the format "0xRRGGBB" where RR, GG, and BB are the hexadecimal values for the red, green, and blue components, respectively. This parameter is optional but is normally set to match the color of the HTML page.

In addition to the above parameters, the <APPLET> tag for a `LiveBeanApplet` must include **width** and **height** attributes. Normally, the size of a `LiveBeanApplet` is set to match the size of the graphic object that it is presenting. To get the size of a graphic object, select the object while the **Graphic Editor** is in editing mode. The selected object's name and size are shown in the **Information Area** at the top of the **Graphic Editor** applet.

LiveBeanApplet Example

All instances of `LiveBeanApplet` that are included in a Web page follow the same pattern, with only the applet's parameters and size varying for each instance.

Here is the HTML code for including a `LiveBeanApplet` that will present the graphic object named **MyMeter** which was saved by the **Graphic Editor** as part of the graphic display, **Library1**.

```
<APPLET codebase="/classes"
archive="SAComm.jar,GDE.jar,Widgets.jar"
code="com.schneiderautomation.gde.LiveBeanApplet"
width=180 height=160>
<PARAM name=LIBRARY value="Library1">
<PARAM name=BEAN value="MyMeter">
<PARAM name=BACKGRND value="0xDDEEFF">
</APPLET>
```

More Information

For more information about creating Java applets and graphic objects to obtain runtime data from a PLC, refer to the Software Developer's Kit included in the FactoryCast installation.

Inserting LiveLabelApplet using HTML Code

Overview

Before any live labels are inserted into a Web page, the special applet called `LiveLabelMgrApplet` must be inserted into the page.

NOTE: However, if a Web page contains both `LiveLabelApplet` and `LiveBeanApplet`, then that page must contain a single instance of `LiveBeanMgrApplet`, not `LiveLabelMgrApplet`.

(`LiveBeanMgrApplet` supports both `LiveLabelApplet` and `LiveBeanApplet`, while `LiveLabelMgrApplet` supports only `LiveLabelApplet`.)

LiveLabelMgrApplet

The `LiveLabelMgrApplet` allows the Web page to display dynamic data from the controller. This applet **must** be included **once** on the page if any instances of `LiveLabelApplet` are included on the page.

Here is the HTML code that you use to include the applet on a page.

```
<APPLET>
codebase="/classes" archive="SAComm.jar"
code="com.schneiderautomation.factorycast.LiveLabelMgrApple"
width=0 height=0>
</APPLET>
```

LiveLabelApplet

Use one `LiveLabelApplet` for every symbol (variable) or direct-address monitored on the Web page used. For example, if you are monitoring three symbols (variables), you would include the applet three times.

This applet displays the following three fields.

Field	Description
Label	Your label for the symbol (variable) or direct address
Value	Run-time value of the symbol (variable) or direct address
Units	The units you specify for the value

Data Parameters

The applet's parameters, their meaning, and the default values are shown below.

Parameter...	Defines...		With Default Value of...
LABEL	A text label to identify the data item		No label
UNITS	A text label to identify the value's engineering units		No units displayed
ADDRESS	The name of Concept/PL7/Unity Pro symbol (variable) or Quantum/Premium direct address		None
DATATYPE	The data type of the symbol (variable) or direct address. Acceptable values for this parameter are		UNDEFINED
	SHORT	8-bit signed integer	
	USHORT	8-bit unsigned integer	
	INT	16-bit signed integer	
	UINT	16-bit unsigned integer	
	DINT	32-bit signed integer	
	UDINT	32-bit unsigned integer	
	REAL	32-bit IEEE floating point	
	TIME	32-bit unsigned integer (in ms)	
	DATE	Date (32-bit BCD)	
	TOD	Time of Day (32-bit BCD)	
	DT	Date and Time 64-bit BCD	
	BOOL	1-bit discrete (boolean)	
NOTES: If the ADDRESS parameter is a direct address, and the DATATYPE parameter is not specified, a default DATATYPE (BOOL ,INT,DINT or REAL based on the implied size of the data value) is used. If ADDRESS is a direct address for a discrete PLC reference (Quantum 0x/1x reference), DATATYPE must be set to BOOL. DATATYPE may be set to BOOL only for discrete PLC references. If the ADDRESS parameter is the name of a Concept, PL7 or Unity Pro symbol (variable), the DATATYPE parameter is optional. If the DATATYPE is specified for a symbol (variable), it must exactly match its actual data type,TIME is not a valid data type for PL7 Premium.			

<table>
<tr><th>Parameter...</th><th colspan="2">Defines...</th><th>With Default Value of...</th></tr>
<tr><td rowspan="8">FORMAT</td><td colspan="2">The display format for the value.
Acceptable values for this parameter are</td><td rowspan="8">DEC for most data types
TIME for data type TIME
BOOL for data type BOOL
DATE for data types DATE, TOD and DT.</td></tr>
<tr><td>DEC</td><td>decimal</td></tr>
<tr><td>HEX</td><td>hexadecimal</td></tr>
<tr><td>BIN</td><td>binary</td></tr>
<tr><td>ASCII</td><td>bytes displayed as ASCII characters</td></tr>
<tr><td>TIME</td><td>‘day_hr_min_sec_ms’</td></tr>
<tr><td>DATE</td><td>‘yyyy-mm-dd-hh and /or hh:mm:ss</td></tr>
<tr><td>BOOL</td><td>ON_WORD or OFF_WORD (see below)</td></tr>
<tr><td colspan="4"><b>NOTE:</b> If DATA TYPE is REAL, a FORMAT other than DEC will give unpredictable results if the value cannot be converted to an integer.</td></tr>
<tr><td>GAIN</td><td colspan="2">The gain (multiplier) used for scaling the retrieved value to engineering units.</td><td>1.0</td></tr>
<tr><td colspan="4"><b>NOTE:</b> Scaling is to be performed only if GAIN or BIAS is set and FORMAT is DEC.
Linear scaling is performed by the formula: SCALED_VALUE=GAINxRAW_VALUE+BIAS</td></tr>
<tr><td>BIAS</td><td colspan="2">The bias (offset) used for scaling the retrieved value to engineering units. See NOTE for GAIN.</td><td>0.0</td></tr>
<tr><td>ON_WORD</td><td colspan="2">A text value to be shown when value is non-zero (Use only if the FORMAT is BOOL).</td><td>ON</td></tr>
<tr><td>OFF_WORD</td><td colspan="2">A text value to be shown when value is zero (Use only if the FORMAT is BOOL).</td><td>OFF</td></tr>
<tr><td>FOREGRND</td><td colspan="2">Foreground color of the applet.
Acceptable values are: WHITE, LT_GRAY, DK_GRAY, BLACK, RED, PINK, ORANGE, YELLOW, GREEN, MAGENTA, CYAN, and BLUE
Also, a RGB color value can be entered using the format "0xRRGGBB" where RR, GG, and BB are the hexadecimal values for the red, green, and blue components, respectively.</td><td>BLACK</td></tr>
<tr><td>BACKGRND</td><td colspan="2">Background color for the applet.
For acceptable values, see FOREGRND.</td><td>LT_GRAY</td></tr>
<tr><td>ERROR_ COLOR</td><td colspan="2">Foreground color of the VALUE field when unable to retrieve the value from the PLC.
For acceptable values, see FOREGRND.</td><td>MAGENTA</td></tr>
<tr><td>LABEL_ ALIGN</td><td colspan="2">Alignment of the text in the LABEL field, if the width of the field is greater than the length of the text.
Acceptable values are: LEFT, CENTER, and RIGHT.</td><td>LEFT</td></tr>
</table>

Parameter...	Defines...	With Default Value of...
VALUE_ ALIGN	Alignment of the text in the VALUE field, if the width of the field is greater than the length of the text. Acceptable values are: LEFT, CENTER, and RIGHT.	LEFT
UNITS_ ALIGN	Alignment of the text in the UNITS field, if the width of the field is greater than the length of the text. Acceptable values are: LEFT, CENTER, and RIGHT.	LEFT
FONT_ NAME	Name of the font used by the applet. Acceptable values are: SERIF, SANSSERIF, and MONOSPACE.	SANSSERIF
FONT_ BOLD	If set, displays all text in the applet as bold. Acceptable values are: TRUE and FALSE.	FALSE
FONT_ ITALIC	If set, displays all text in the applet in italics. Acceptable values are: TRUE and FALSE.	FALSE
FONT_SIZE	Sets the point size of the font used by the applet.	12
LABEL_ WIDTH	The width of the LABEL field.	
UNITS_ WIDTH	The width of the UNITS field.	

Size Parameters

The size of a `LiveLabelApplet` is specified in the width and height attributes of its <APPLET> tag. Unless the width of Label or Units field is set with the LABEL_WIDTH or UNITS_WIDTH parameters, the LABEL and UNITS fields of the applet will always take the width required to display the text values of their associated applet parameters. The remaining width of the applet is given to its VALUE field.

LiveLabelapplet Example #1

The applet example in this section contains almost every applet parameter. Here is the HTML code for this example.

```
<APPLET codebase="/classes" archive="SAComm.jar"
code="com.schneiderautomation.factorycast.LiveLabelApplet"
width=300 height=30>
<PARAM name=LABEL value="Reactor 1 Temperature">
<PARAM name=UNITS value="F">
<PARAM name=ADDRESS value="40101">(ForPremium value="%MW100")
<PARAM name=DATATYPE value="UINT">
<PARAM name=FORMAT value="DEC">
<PARAM name=GAIN value="2.0">
<PARAM name=BIAS value="100.0">
<PARAM name=FOREGRND value="WHITE">
<PARAM name=BACKGRND value="BLACK">
<PARAM name=ERROR_COLOR value="RED">
<PARAM name=FONT_NAME value="SERIF">
<PARAM name=FONT_BOLD value="TRUE">
<PARAM name=FONT_ITALIC value="FALSE">
<PARAM name=FONT_SIZE value="10">
</APPLET>
```

LiveLabelApplet Example #2

This is an example of a minimal applet, using default values for most parameters. Here is the HTML code for this example.

```
<APPLET codebase="/classes" archive="SAComm.jar"
code="com.schneiderautomation.factorycast.LiveLabelApplet"
width=300 height=30>
<PARAM name=LABEL value="Reactor 1 Pressure">
<PARAM name=UNITS value="PSI">
<PARAM name=ADDRESS value="PT_101">
</APPLET>
```

Inserting LiveTableApplet using HTML Code

Overview

`LiveTableApplet` is used to display dynamic, runtime PLC data on a Web page, and `LiveTableApplet` operates in a manner similar to `LiveLabelApplet`. However, there is a difference: `LiveTableApplet` can show multiple input values using a tabular format. Only a single input value can be shown with `LiveLabelApplet`.

LiveTableApplet

LiveTableApplet supports applet parameters for

- Setting the number of rows in its table (N_ROWS)
- Default settings for a row's properties
- Unique settings for each row's properties

The properties (default or unique) that can be set for a row of `LiveTableApplet` are the same properties that can be set for `LiveLabelApplet`. (The parameter names are the same as those for `LiveLabelApplet`, except that **Rx_** prefixes them, where **x** equals the applicable row number. A default setting is specified by not including the **Rx_** prefix.

LiveTableApplet Example

This is an example of `LiveTableApplet` that has two rows and an overall width of 200. The common row properties set the width of the **Label** field to 100 and the width of the **Units** field to 40 (leaving a width of 60 for the **Value** field). Also, every row has a black background with white text of size-10 font. The **Label** text is centered; the **Value** text is right-aligned; and the **Units** text is left-aligned. The address, data type, and the text for the **Label** and **Units** fields are set individually for each of the two rows.

Here is the HTML code for this example.

```
<APPLET codebase="/classes" archive="SAComm.jar"
code="com.schneiderautomation.factorycast.LiveTableApplet"
width="200" height="40">
<PARAM name=N_ROWS value="2">
<PARAM name=LABEL_WIDTH value="100">
<PARAM name=UNITS_WIDTH value="40">
<PARAM name=BACKGRND value="BLACK">
<PARAM name=FOREGRND value="WHITE">
<PARAM name=FONT_SIZE value="10">
<PARAM name=LABEL_ALIGN value="CENTER">
<PARAM name=VALUE_ALIGN value="RIGHT">
<PARAM name=UNITS_ALIGN value="LEFT">

<PARAM name=R1_LABEL value="Reactor Pressure">
<PARAM name=R1_UNITS value="PSIG">
<PARAM name=R1_ADDRESS value="400101">
» (for Premium value="%MW101")
<PARAM name=R1_DATATYPE value="INT">

<PARAM name=R2_LABEL value="Reactor Temperature">
<PARAM name=R2_UNITS value="F">
<PARAM name=R2_ADDRESS value="400102">
» (for Premium value="%MW102")
<PARAM name=R2_DATATYPE value="INT">
</APPLET>
```

7.2 Using FactoryCast Extension for Microsoft FrontPage or Expression Web

Overview

The *Using FactoryCast's Java Applets* section described how to add FactoryCast's Java applets to a HTML document using any text editor. This section describes using an extension for Microsoft's FrontPage or Expression Web application. The extension allows a user of FrontPage or Expression Web to easily insert FactoryCast applets to view real-time PLC data on a Web page.

What's in this Section?

This section contains the following topics:

Installing FactoryCast’s Microsoft Expression Web Extension

Overview

This section describes how to install/remove the FactoryCast extension for Microsoft Expression Web (MS Expression Web).

Installing FactoryCast

MS Expression Web installed before Web Designer for FactoryCast HMI 1.7:

During the installation of Web Designer for FactoryCast HMI 1.7, the FactoryCast Extension for MS Expression Web is automatically installed as a MS Expression Web "Macro File." If this is the case, then proceed to the next section, which provides instructions for adding the extension to MS Expression Web’s menu.

MS Expression Web installed after Web Designer for FactoryCast HMI 1.7:

The extension can be installed by either re-installing Web Designer for FactoryCast HMI 1.7 or manually copying the macro file to MS Expression Web’s macro folder. For manual installation, after installing MS Expression Web, the following will install the extension:

Copy the file "Microsoft Expression Web.wdmacro" from CD-ROM to the following folder (create the final ’Macros’ folder, if it does not already exist):
C:\Documents and Settings\<username>\Application Data\Microsoft\Expression\Macros

NOTE: MS Expression Web uses this address by default. For a custom installation, you can determine this address by typing SET from a Command Prompt.

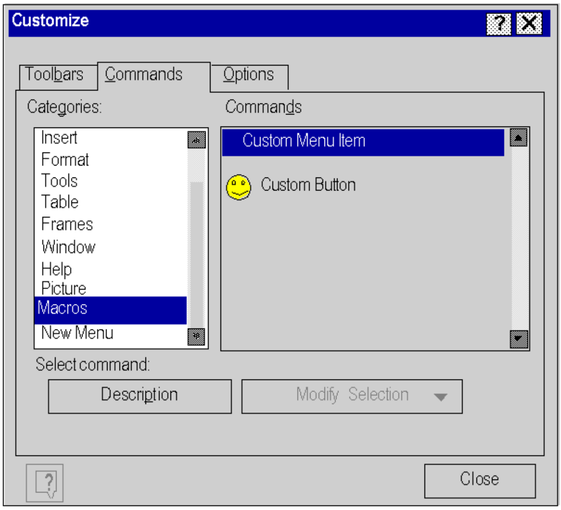
Adding FactoryCast Extension

To add the FactoryCast Extension to the MS Expression Web **Insert** menu, do the following.

Step	Action
1	Start MS Expression Web.
2	Click **Customize** on the **Tools** menu.

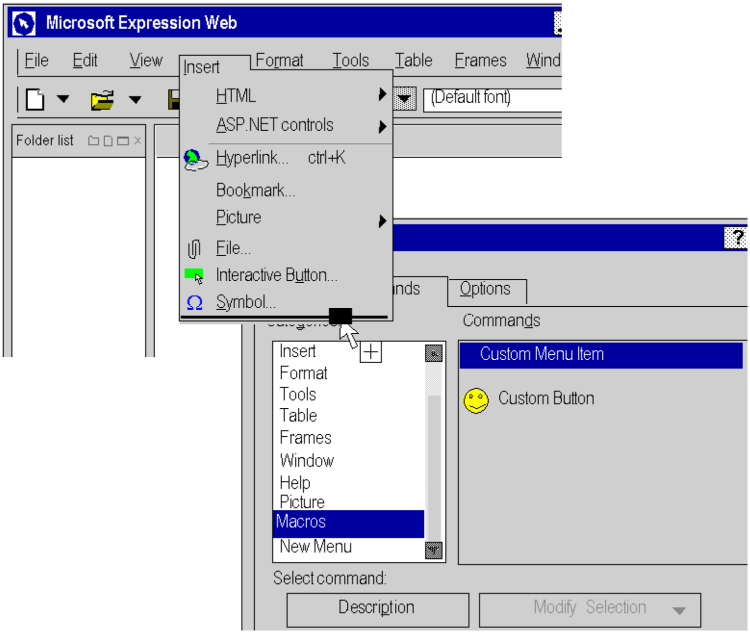
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>

Click the **Commands** tab, and then select **Macros** from the **Categories** list.

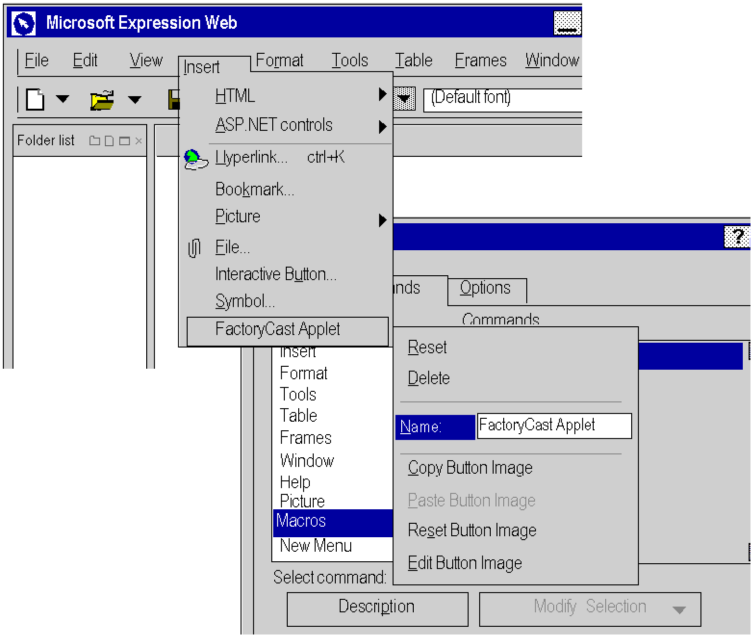
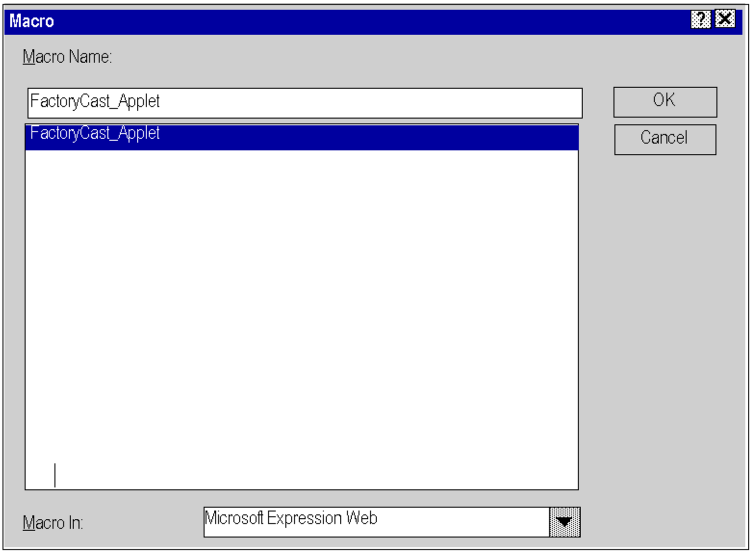


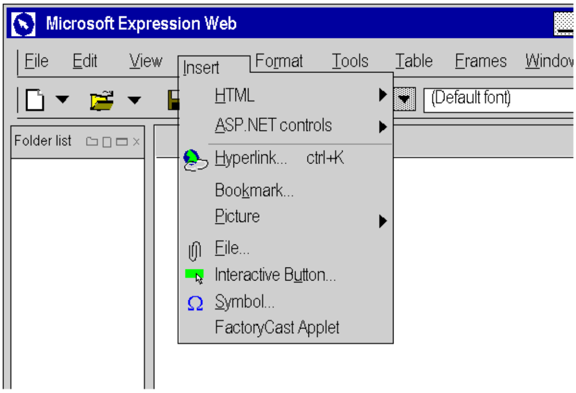
</td></tr>
<tr><td>4</td><td>

Drag the entry **Custom Menu Item** from the **Commands** list to the **Insert** menu, beneath the **Symbol** command (the **Insert** menu will automatically expand when the pointer rolls over it.).



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

Step	Action
5	Click **Modify Selection**, change the name to **FactoryCast Applet**, and then press ENTER key.  Microsoft Expression Web File Edit View Insert Format Tools Table Frames Window (Default font) Folder list HTML ASP.NET controls Hyperlink... ctrl+K Bookmark... Picture File... Interactive Button... Symbol... FactoryCast Applet nds Options Commands Insert Format Tools Table Frames Window Help Picture Macros New Menu Select command: Reset Delete Copy Button Image Paste Button Image Reset Button Image Edit Button Image Description Modify Selection Name: FactoryCast Applet
6	Click **Modify Selection** again, and then choose **Assign Macro**.  Macro Name: FactoryCast_Applet FactoryCast_Applet Macro In: Microsoft Expression Web

Step	Action
7	Choose **FactoryCast_Applet** from the list, and then click **OK**.
8	Click **Close**. **Result:** The **FactoryCast Applet** command is added to the **Insert** menu.  Microsoft Expression Web File Edit View Insert Format Tools Table Frames Windov (Default font) Folder list HTML ASP.NET controls Hyperlink... ctrl+K Bookmark... Picture File... Interactive Button... Symbol... FactoryCast Applet

Removing FactoryCast Extension

To remove the FactoryCast Extension from the MS Expression Web menu, proceed as follow:

Step	Action
1	In MS Expression Web, click **Customize** on the **Tools** menu.
2	Click the **Insert** menu, and then select **FactoryCast Applet**.
3	Right click, and then select **Delete** from the popup menu.

Editing Applets

There are two ways to edit an applet that has been inserted into your Web page:

- You can double-click on the object and make changes via dialog boxes.
- You can switch to the HTML editor in MS Expression Web and do your editing in this environment.

NOTE: It is suggested you edit via the first approach unless you are comfortable programming in the HTML language used to build Web pages.

Installing FactoryCast Extension for Microsoft FrontPage

Overview

This section describes how to install/remove the FactoryCast Extension for Microsoft FrontPage 2000.

Installing FactoryCast

During the installation of Web Designer, if FrontPage 2000 has been installed on the same PC, then the FactoryCast Extension for FrontPage 2000 is automatically installed as a FrontPage "Macro File". If this is the case, then proceed to the next section, which provides instructions for adding the Extension to FrontPage's menu.

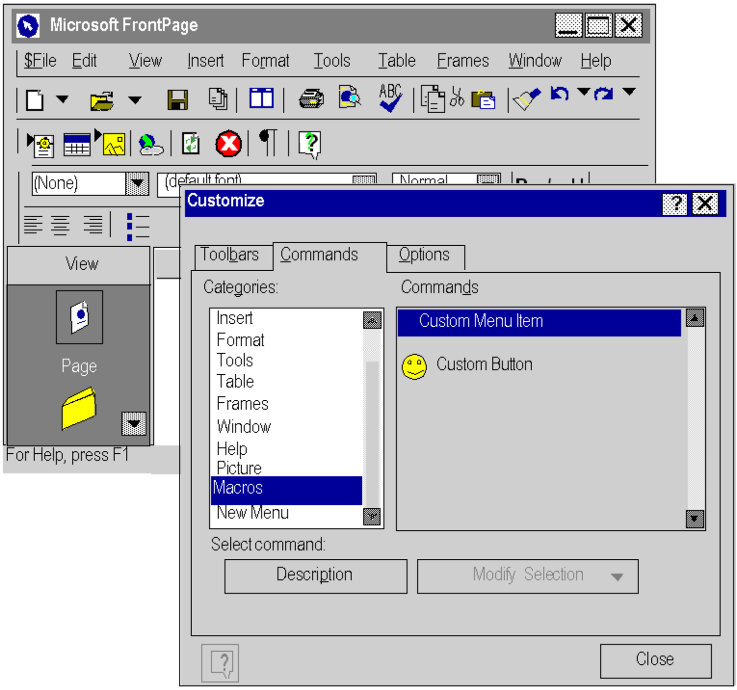
However, if FrontPage 2000 is installed after Web Designer has been installed, then the Extension can be installed by either re-installing Web Designer or manually copying the macro file to FrontPage's macro folder. For manual installation, after installing FrontPage 2000, the following will install the Extension:

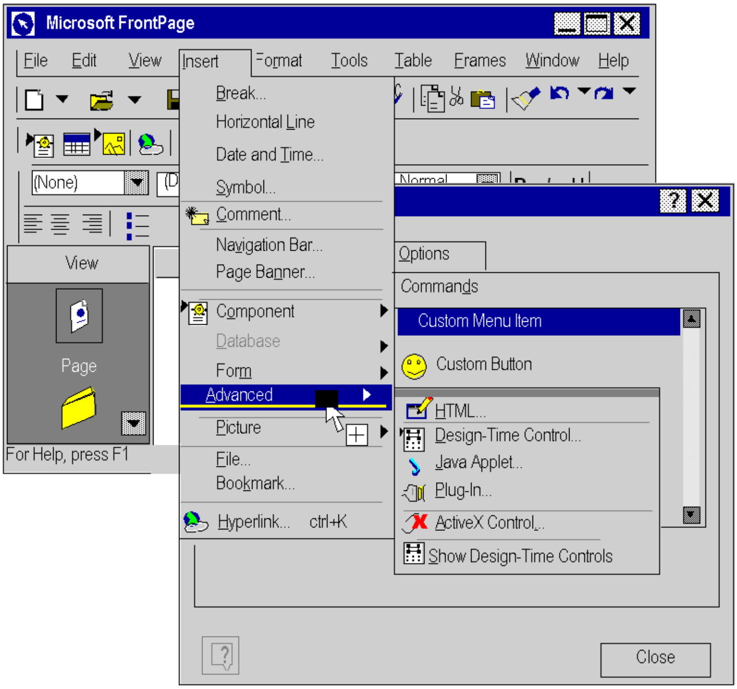
Copy the file *Microsoft FrontPage.fpm* from CD-ROM to the folder *%USERPROFILE%\Application Data\Microsoft\FrontPage\Macros* (create the final *'Macros'* folder, if it does not already exist).

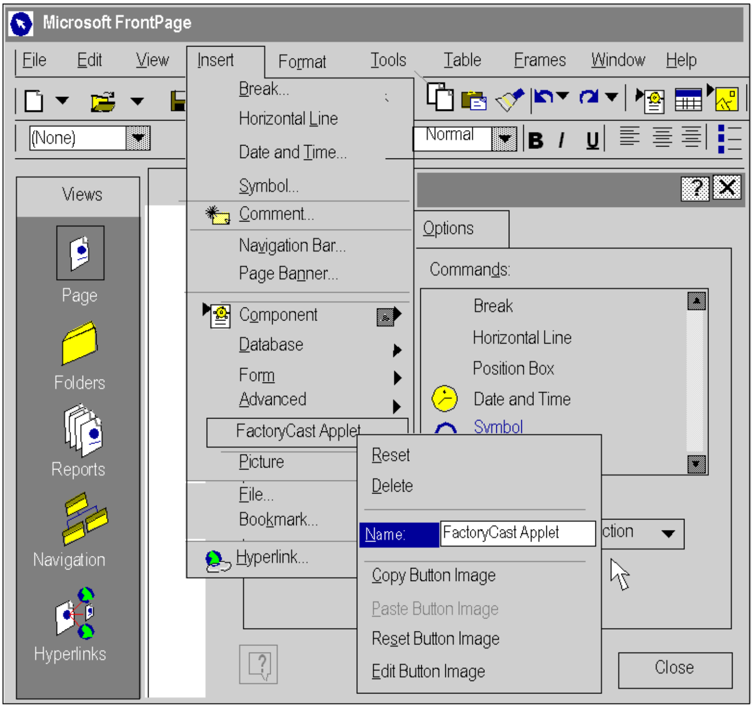
Adding FactoryCast Extension

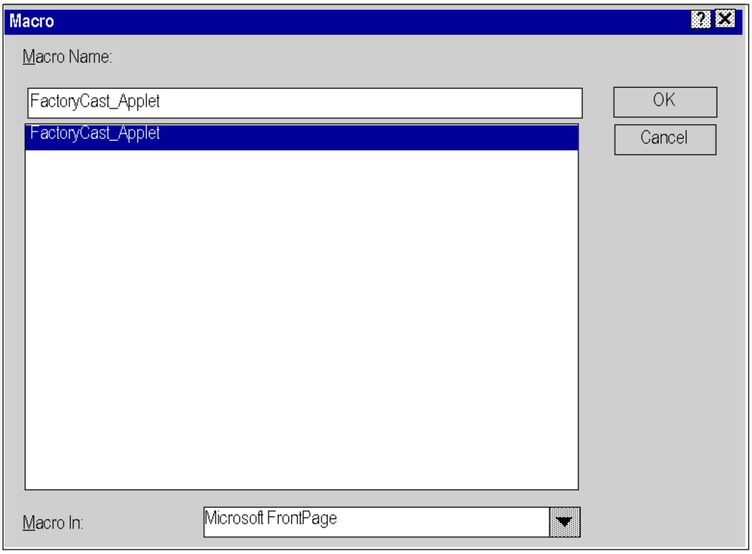
To add the FactoryCast Extension to the FrontPage **Insert** menu, do the following.

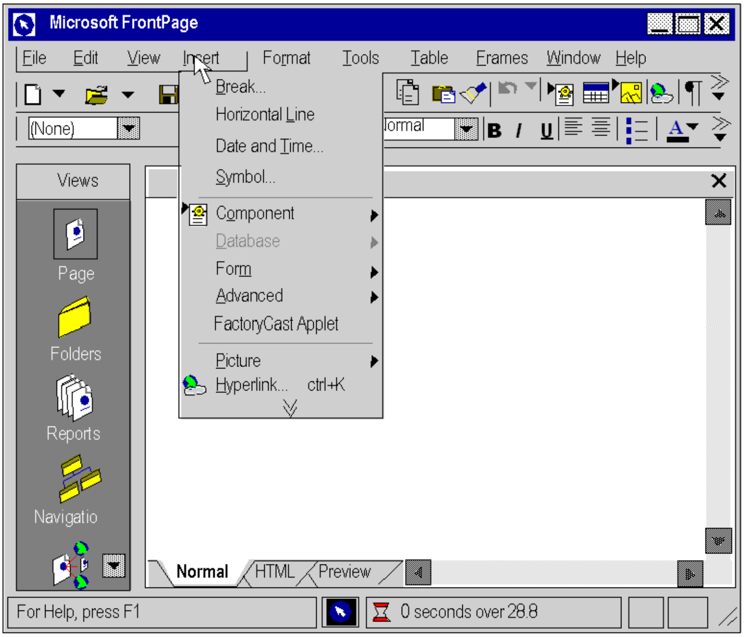
Step	Action
1	Start FrontPage 2000.
2	Click **Customize** on the **Tools** menu.

Step	Action
3	Click the **Commands** tab, and then select **Macros** from the **Categories** list. 

Step	Action
4	Drag and drop the entry **Custom Menu Item** from the **Commands** list to the **Insert** menu, beneath the **Advanced** command. (The **Insert** menu will automatically drop down when you drag over it.)  Microsoft FrontPage File Edit View Insert Format Tools Table Frames Window Help Break... Horizontal Line Date and Time... Symbol... Comment... Navigation Bar... Page Banner... Component Database Form Advanced Picture File... Bookmark... Hyperlink... ctrl+K (None) Normal View Page For Help, press F1 Options Commands Custom Menu Item Custom Button HTML... Design-Time Control... Java Applet... Plug-In... ActiveX Control... Show Design-Time Controls Close

Step	Action
5	Click **Modify Selection**, change the name to **FactoryCast Applet**, and then press ENTER key.  Microsoft FrontPage File Edit View Insert Format Tools Table Frames Window Help (None) Normal Views Page Folders Reports Navigation Hyperlinks Break... Horizontal Line Date and Time... Symbol... Comment... Navigation Bar... Page Banner... Component Database Form Advanced FactoryCast Applet Picture File... Bookmark... Hyperlink... Options Commands: Break Horizontal Line Position Box Date and Time Symbol Reset Delete Copy Button Image Paste Button Image Reset Button Image Edit Button Image ction Close Name: FactoryCast Applet

Step	Action
6	Click **Modify Selection** again, and then choose **Assign Macro**.  Macro Name: FactoryCast_Applet FactoryCast_Applet Macro In: Microsoft FrontPage
7	Choose **FactoryCast_Applet** from the list, and then click **OK**.

Step	Action
8	Click **Close**. The **FactoryCast Applet** command is added to the **Insert** menu. 

Removing FactoryCast Extension

To remove the FactoryCast Extension from the FrontPage menu, do the following.

Step	Action
1	In FrontPage, click **Customize** on the **Tools** menu.
2	Click the **Insert** menu, and then select **FactoryCast Applet**.
3	Right click, and then select **Delete** from the popup menu.

Editing Applets

There are two ways to edit an applet that has been inserted into your Web page. First, you can double-click on the object and make changes via dialog boxes. Or you can switch to the HTML editor in FrontPage and do your editing in this environment. It is suggested you edit via the first approach unless you are comfortable programming in the HTML language used to build Web pages.

Inserting LiveBeanApplet Using FrontPage or Expression Web

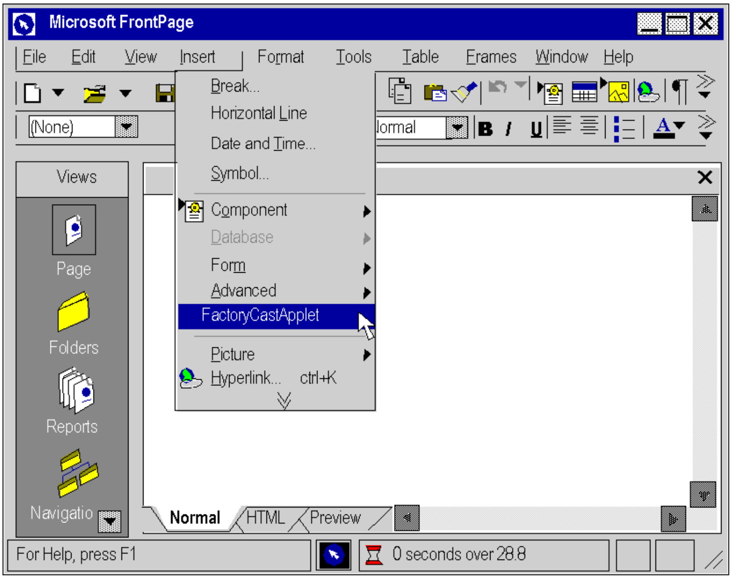
Overview

This section describes inserting `LiveBeanApplet` into a Web page. Before inserting, please read the following note and then follow steps one through sixteen below.

NOTE: The following procedure shows you how inserting LiveBeanApplet using Frontpage. Use the same procedure if you use Expression Web.

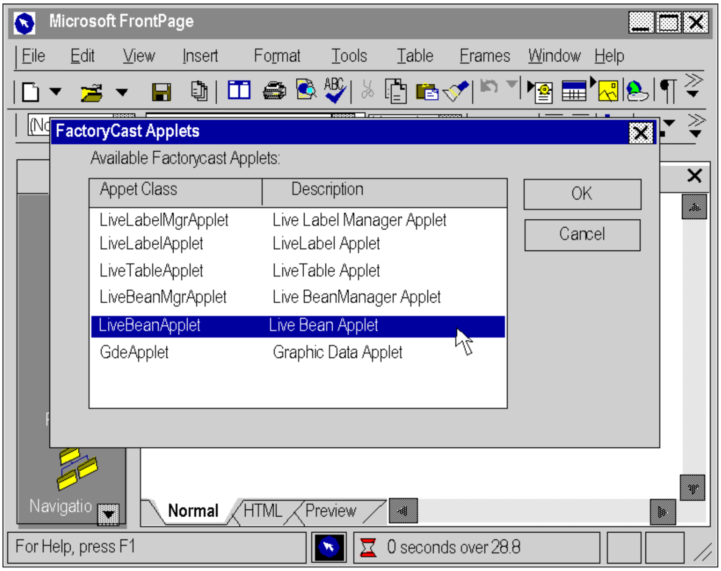
Inserting a LiveBeanApplet

To insert a LiveBeanApplet, do the following.

Step	Action
1	In FrontPage, Select **Insert | FactoryCast Applet** to view the FactoryCast applet selection window. 

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>

Note: The **LiveBeanMgrApplet** is a special applet that is inserted one time only into your Web page.
Select the applet **LiveBeanMgrApplet** and then click the OK button.



FactoryCast Applets

Available Factorycast Applets:

<table>
<tr><th>Appet Class</th><th>Description</th></tr>
<tr><td>LiveLabelMgrApplet</td><td>Live Label Manager Applet</td></tr>
<tr><td>LiveLabelApplet</td><td>LiveLabel Applet</td></tr>
<tr><td>LiveTableApplet</td><td>LiveTable Applet</td></tr>
<tr><td>LiveBeanMgrApplet</td><td>Live BeanManager Applet</td></tr>
<tr><td>LiveBeanApplet</td><td>Live Bean Applet</td></tr>
<tr><td>GdeApplet</td><td>Graphic Data Applet</td></tr>
</table>

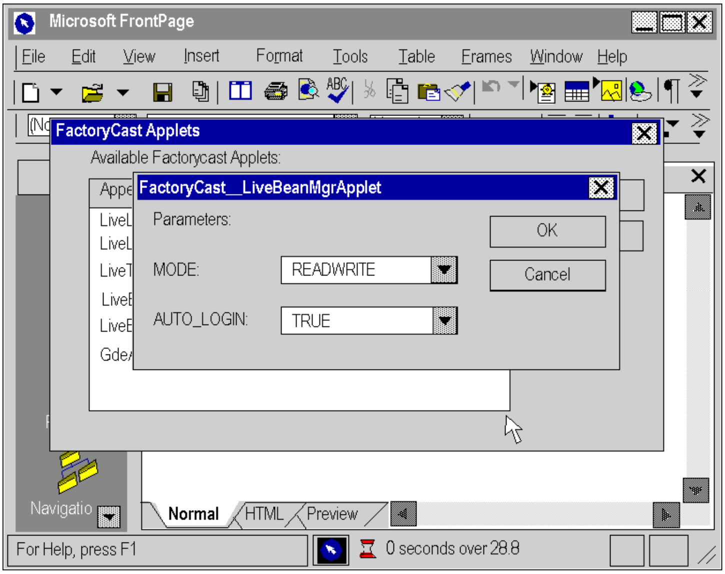
OK

Cancel

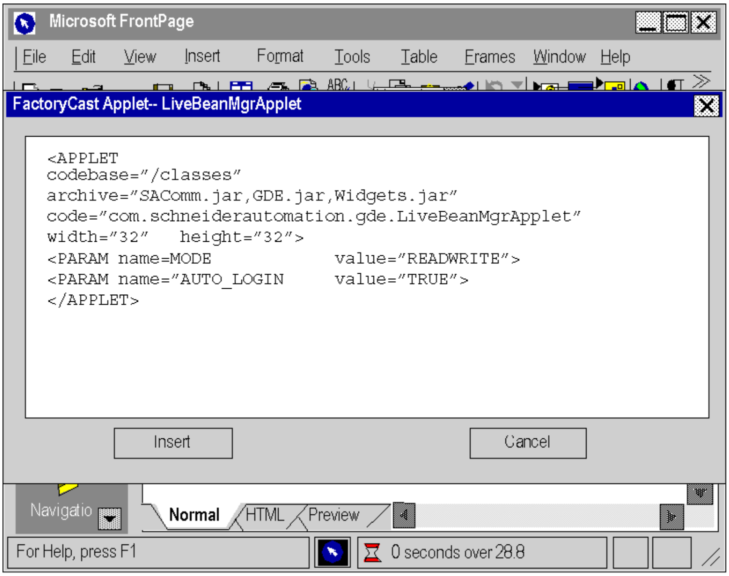
For Help, press F1

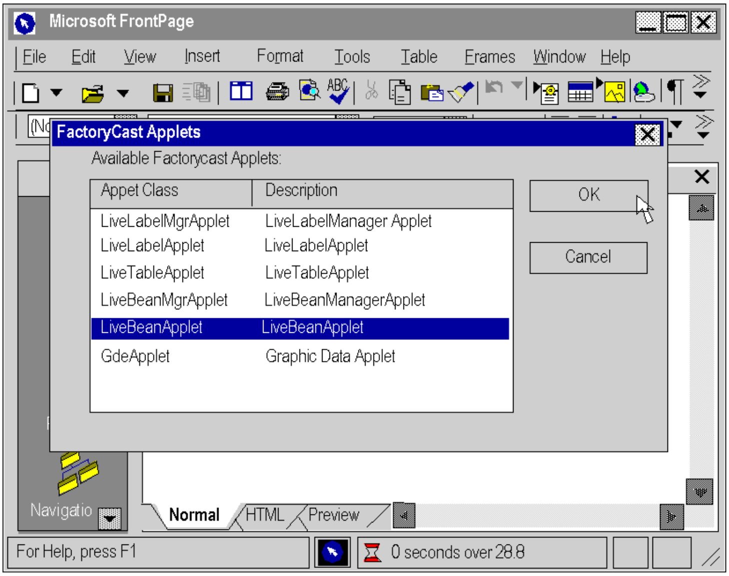
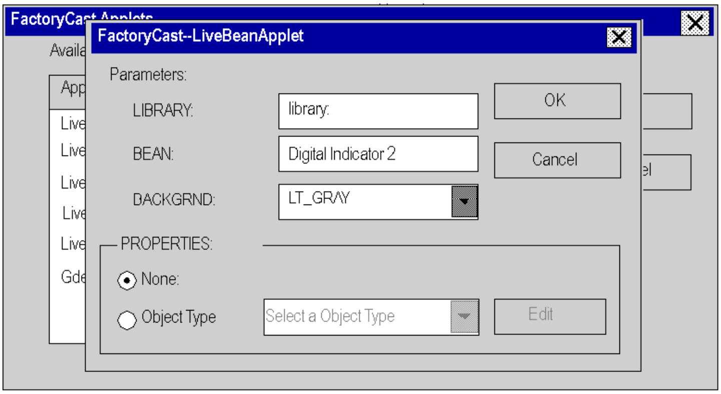
0 seconds over 28.8

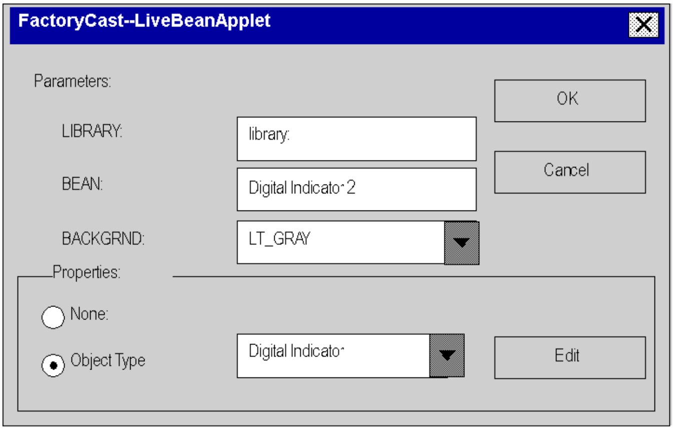
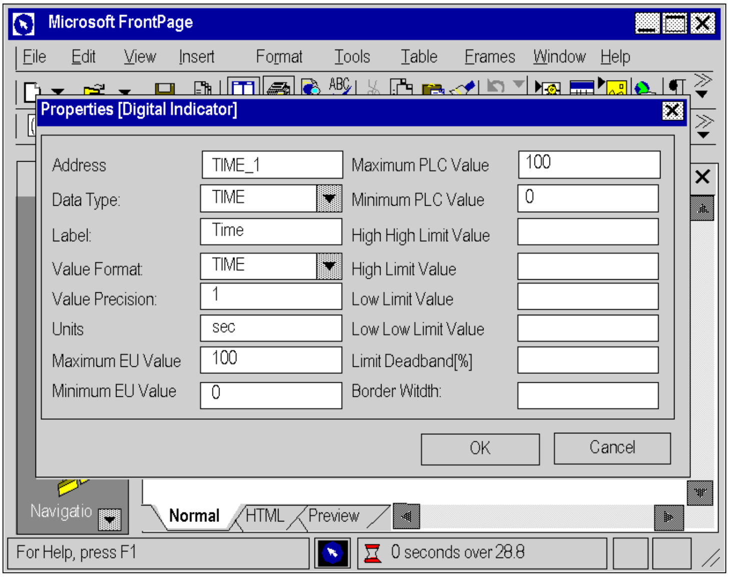
</td></tr>
</table>

Step	Action
3	Configure the **Mode** and **Auto Login** parameters and then click the **OK** button.  FactoryCast__LiveBeanMgrApplet Parameters: MODE: READWRITE AUTO_LOGIN: TRUE

Step	Action

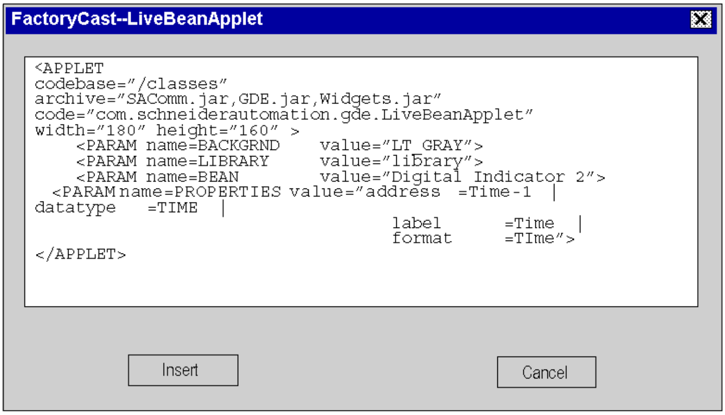
4	A window will appear showing the Java code that will be inserted into your HTML document.
	
	 Microsoft FrontPage
	File Edit View Insert Format Tools Table Frames Window Help
	FactoryCast Applet-- LiveBeanMgrApplet
	Insert
	Cancel
	Navigatio
	Normal
	HTML
	Preview
	For Help, press F1
	0 seconds over 28.8
	`<APPLET`
	`codebase="/classes"`
	`archive="SAComm.jar,GDE.jar,Widgets.jar"`
	`code="com.schneiderautomation.gde.LiveBeanMgrApplet"`
	`width="32"   height="32">`
	`<PARAM name=MODE            value="READWRITE">`
	`<PARAM name="AUTO_LOGIN      value="TRUE">`
	`</APPLET>`
	Select the Insert button to finish inserting this applet.
	Note: Though a gray box with a blue letter "J" will appear on your Web page during editing, this applet will only be visible on your loaded Web page if you configured the Mode parameter as **ReadWrite** (it will appear as a Key when viewed via your browser). Otherwise, if **Mode** *(see page 234)* is set as **ReadOnly** then the width and height of the applet will be set to zero and you will not see it in your browser.

Step	Action
5	Select the **LiveBeanApplet** from the FactoryCast applet selection window and then click the **OK** button.  Microsoft FrontPage FactoryCast Applets Available Factorycast Applets: Appet Class — Description LiveLabelMgrApplet — LiveLabelManager Applet LiveLabelApplet — LiveLabelApplet LiveTableApplet — LiveTableApplet LiveBeanMgrApplet — LiveBeanManagerApplet LiveBeanApplet — LiveBeanApplet GdeApplet — Graphic Data Applet File Edit View Insert Format Tools Table Frames Window Help OK Cancel Navigatio Normal HTML Preview For Help, press F1 0 seconds over 28.8
6	Enter the library name and the name of a particular bean in this library that you want to show. If none of the bean’s properties, such as address, need to be customized, then click the **OK** button.  FactoryCast--LiveBeanApplet Parameters: LIBRARY: library: BEAN: Digital Indicator 2 BACKGRND: LT_GRAY PROPERTIES: ● None: ○ Object Type Select a Object Type

Step	Action
7	Usually at a minimum the address property will need to be customized for your beans. Thus after entering the library name and bean name, select the object type from the **Object Type** box.  FactoryCast--LiveBeanApplet Parameters: LIBRARY: library BEAN: Digital Indicator2 BACKGRND: LT_GRAY Properties: ○ None: ● Object Type: Digital Indicator Click the **Edit** button when finished.
8	In the properties edit window, change only those parameters, such as address, that are unique to this bean. All other parameters will be set to that of the saved bean in your library. Click the **OK** button when finished.  Microsoft FrontPage Properties [Digital Indicator] Address: TIME_1 Data Type: TIME Label: Time Value Format: TIME Value Precision: 1 Units: sec Maximum EU Value: 100 Minimum EU Value: 0 Maximum PLC Value: 100 Minimum PLC Value: 0 High High Limit Value: High Limit Value: Low Limit Value: Low Low Limit Value: Limit Deadband[%]: Border Witdth: For Help, press F1 0 seconds over 28.8 File Edit View Insert Format Tools Table Frames Window Help OK Cancel Navigatio Normal HTML Preview

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>9</td><td>

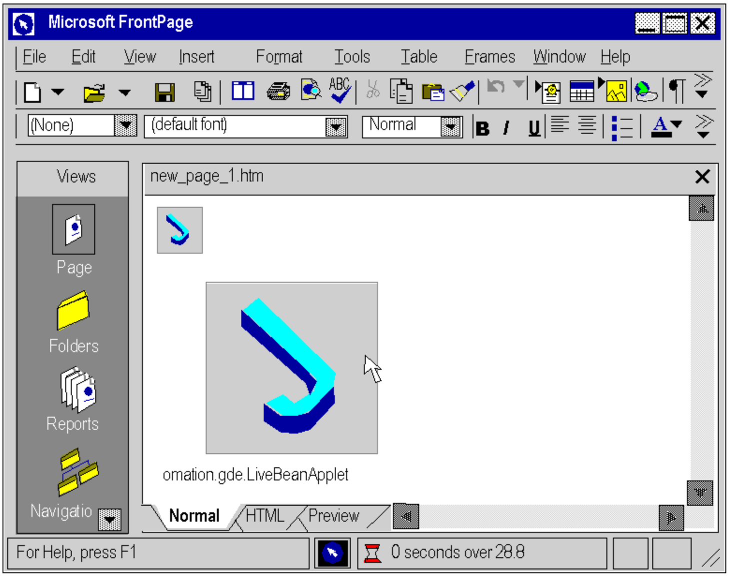
Click the **OK** button. A window will appear showing the Java code that will be inserted into your HTML document. Select the **Insert** button to finish inserting this bean.



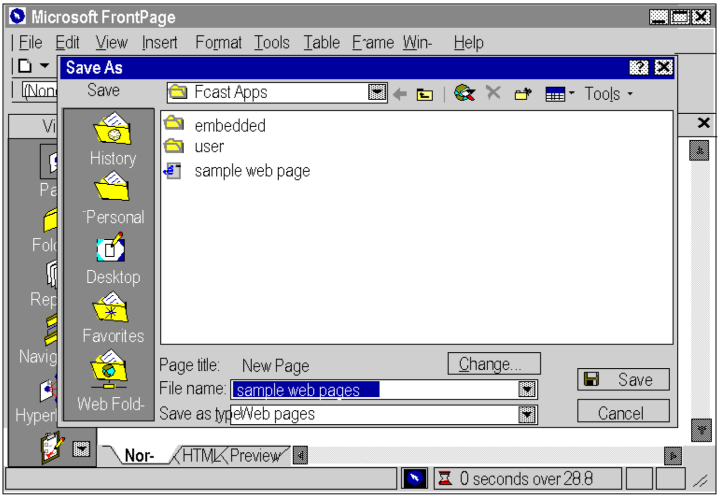
```
<APPLET
codebase="/classes"
archive="SAComm.jar,GDE.jar,Widgets.jar"
code="com.schneiderautomation.gde.LiveBeanApplet"
width="180" height="160" >
    <PARAM name=BACKGRND     value="LT_GRAY">
    <PARAM name=LIBRARY      value="library">
    <PARAM name=BEAN         value="Digital Indicator 2">
  <PARAM name=PROPERTIES value="address  =Time-1  |
datatype   =TIME  |
                                  label      =Time  |
                                  format     =TIme">
</APPLET>
```

</td></tr>
<tr><td>10</td><td>

Continue to add instances of `LiveBeanApplet` to your Web page. Once you have inserted your last applet, click the **Cancel** button on the FactoryCast applet selection window to return to editing your Web page.



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

Step	Action
11	Save the customized Web page.  Microsoft FrontPage File Edit View Insert Format Tools Table Frame Win- Help Save As Save Fcast Apps Tools embedded user sample web page History Personal Desktop Favorites Web Fold- Change... Save Cancel Nor- HTML Preview 0 seconds over 28.8 Page title: New Page File name: sample web pages Save as type: Web pages
12	Transfer the customized Web page to the NOE Web server.
13	Test the application.

Inserting LiveLabelApplet Using FrontPage

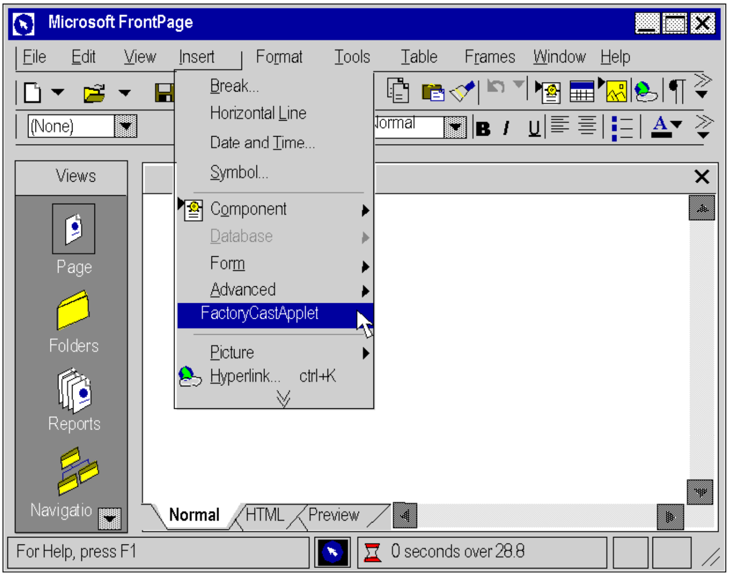
Overview

This section describes inserting LiveLabelApplet into a Web page.

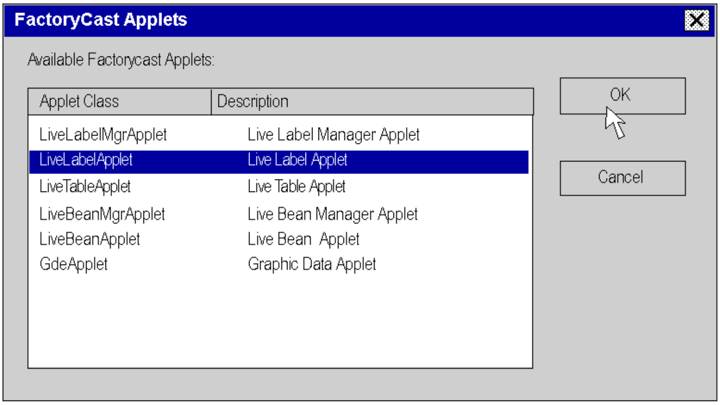
NOTE: The following procedure shows you how inserting LiveTableApplet using Frontpage. Use the same procedure if you use Expression Web.

Inserting LiveLabelApplet

To insert a LiveLabelApplet, do the following.

Step	Action
1	Select **Insert | FactoryCast** to view the FactoryCast applet selection window. 

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Select the applet <b>LiveLabelMgrApplet</b> and then click the <b>OK</b> button.

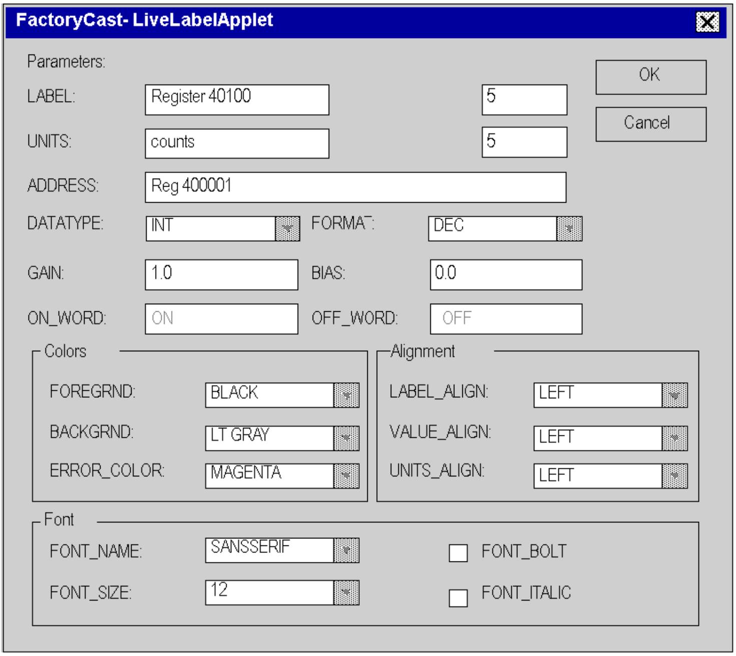
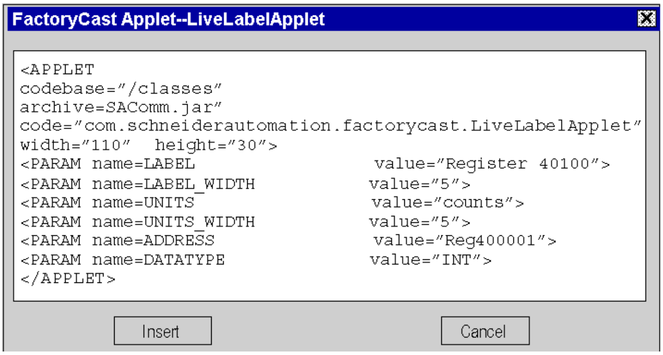


FactoryCast Applets

Available Factorycast Applets:

<table>
<tr><th>Applet Class</th><th>Description</th></tr>
<tr><td>LiveLabelMgrApplet</td><td>Live Label Manager Applet</td></tr>
<tr><td>LiveLabelApplet</td><td>Live Label Applet</td></tr>
<tr><td>LiveTableApplet</td><td>Live Table Applet</td></tr>
<tr><td>LiveBeanMgrApplet</td><td>Live Bean Manager Applet</td></tr>
<tr><td>LiveBeanApplet</td><td>Live Bean Applet</td></tr>
<tr><td>GdeApplet</td><td>Graphic Data Applet</td></tr>
</table>

<b>Note:</b> The <b>LiveLabelMgrApplet</b> is a special applet that is inserted one time only into your Web page. Though a gray box with a blue letter "J" will appear on your Web page during editing, this applet will not be visible when you view your Web page from the Web server module.</td></tr>
<tr><td>3</td><td>Select the applet <b>LiveLabelApplet</b> and then click the <b>OK</b> button. The Parameters edit window appears.</td></tr>
</table>

Step	Action
4	Fill in the parameters for the label and then click the **OK** button.  FactoryCast- LiveLabelApplet Parameters: LABEL: Register 40100 5 UNITS: counts 5 ADDRESS: Reg 400001 DATATYPE: INT FORMAT: DEC GAIN: 1.0 BIAS: 0.0 ON_WORD: ON OFF_WORD: OFF Colors FOREGRND: BLACK BACKGRND: LT GRAY ERROR_COLOR: MAGENTA Alignment LABEL_ALIGN: LEFT VALUE_ALIGN: LEFT UNITS_ALIGN: LEFT Font FONT_NAME: SANSSERIF FONT_SIZE: 12 □ FONT_BOLT □ FONT_ITALIC **Note:** For a description on each parameter, see *Using FactoryCast's Java Applets Using HTML Code, page 226*.
5	A window will appear showing the HTML code that will be inserted into your HTML document. Click the **Insert** button to finish inserting this applet.  FactoryCast Applet--LiveLabelApplet <APPLET codebase="/classes" archive=SAComm.jar" code="com.schneiderautomation.factorycast.LiveLabelApplet" width="110" height="30"> <PARAM name=LABEL value="Register 40100"> <PARAM name=LABEL_WIDTH value="5"> <PARAM name=UNITS value="counts"> <PARAM name=UNITS_WIDTH value="5"> <PARAM name=ADDRESS value="Reg400001"> <PARAM name=DATATYPE value="INT"> </APPLET>

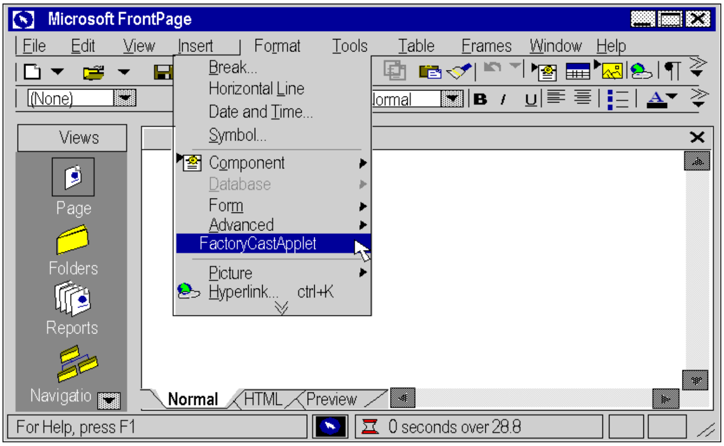
Step	Action
6	Continue to add additional instances of `LiveLabelApplet` to your Web page. Once you have inserted your last applet, click the **Cancel** button on the FactoryCast applet selection window to return to editing your Web page.

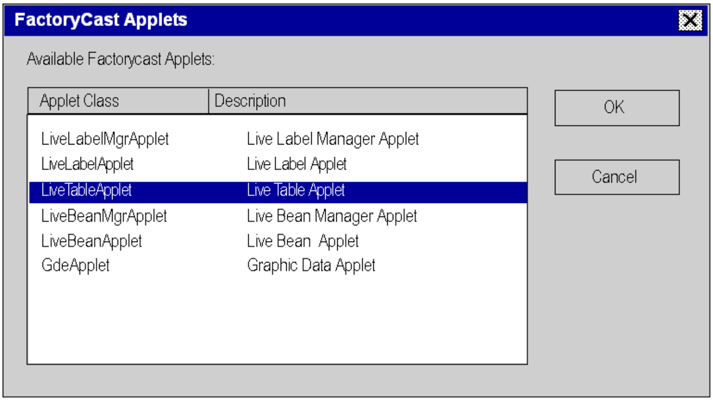
Inserting LiveTableApplet Using FrontPage or Expression Web

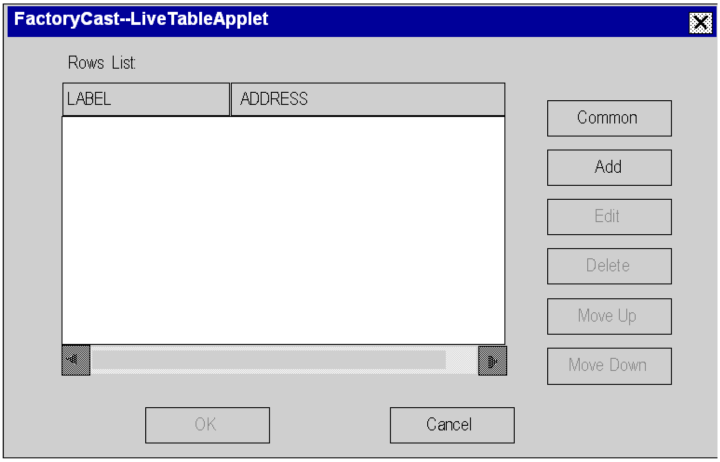
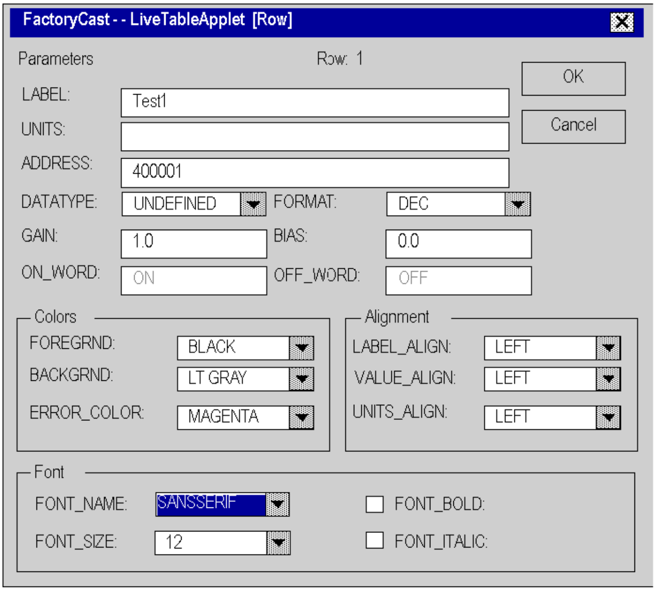
Overview

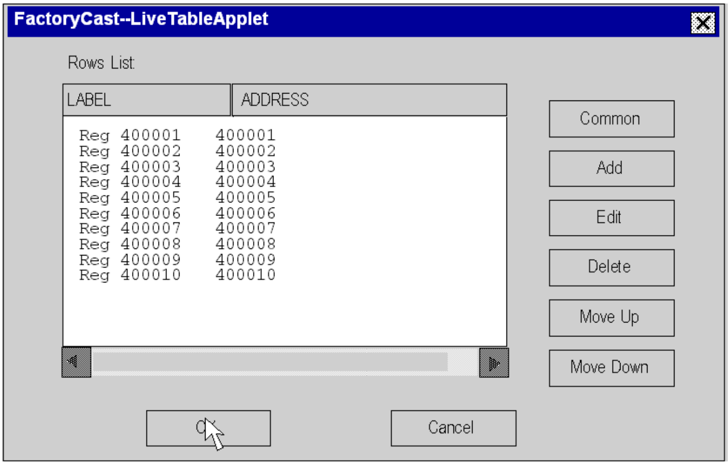
NOTE: The following procedure shows you how inserting **LiveTableApplet** using Frontpage. Use the same procedure if you use Expression Web.

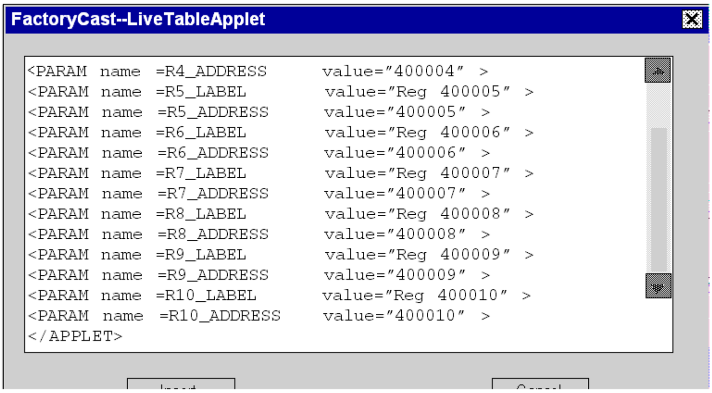
Inserting LiveTableApplet

Step	Action
1	Select **Insert | FactoryCast** to view the FactoryCast applet selection window.  Microsoft FrontPage File Edit View Insert Format Tools Table Frames Window Help (None) Normal Views Page Folders Reports Navigatio Break... Horizontal Line Date and Time... Symbol... Component Database Form Advanced FactoryCastApplet Picture Hyperlink... ctrl+K Normal HTML Preview For Help, press F1 0 seconds over 28.8
2	Select the applet **LiveLabelMgrApplet** and then click the **OK** button. **Note:** The **LiveLabelMgrApplet** is a special applet that is inserted one time only into your Web page.

Step	Action
3	Select the **LiveTableApplet** and then click the **OK** button.  FactoryCast Applets Available Factorycast Applets: Applet Class — Description LiveLabelMgrApplet — Live Label Manager Applet LiveLabelApplet — Live Label Applet LiveTableApplet — Live Table Applet LiveBeanMgrApplet — Live Bean Manager Applet LiveBeanApplet — Live Bean Applet GdeApplet — Graphic Data Applet

Step	Action
4	Click the **Common** button to configure all the parameters that are common to all rows of the table.  FactoryCast--LiveTableApplet Rows List LABEL ADDRESS Common Add Edit Delete Move Up Move Down OK Cancel This procedure speeds configuration by simultaneously entering many values that have the same parameter fields. You many still edit parameters for rows individually during the add process.  Parameters Row: 1 LABEL: Test1 UNITS: ADDRESS: 400001 DATATYPE: UNDEFINED FORMAT: DEC GAIN: 1.0 BIAS: 0.0 ON_WORD: ON OFF_WORD: OFF Colors FOREGRND: BLACK BACKGRND: LT GRAY ERROR_COLOR: MAGENTA Alignment LABEL_ALIGN: LEFT VALUE_ALIGN: LEFT UNITS_ALIGN: LEFT Font FONT_NAME: SANSSERIF FONT_SIZE: 12 □ FONT_BOLD: □ FONT_ITALIC:

Step	Action
5	Click the **Add** button to add a row to your table. The parameters edit dialog window appears. Configure parameters and then click the **Add** button when you are done.  FactoryCast--LiveTableApplet Rows List Common Add Edit Delete Move Up Move Down OK Cancel LABEL ADDRESS Reg 400001 400001 Reg 400002 400002 Reg 400003 400003 Reg 400004 400004 Reg 400005 400005 Reg 400006 400006 Reg 400007 400007 Reg 400008 400008 Reg 400009 400009 Reg 400010 400010
6	Continue to add rows using the **Add** button. You may also select a row and move it up or down the rows of your table using the **Move Up** and **Move Down** buttons. When you are finished adding and editing rows in your table, click the **OK** button from the **Rows List** window.

7	A window will appear showing the HTML code that will be inserted into your HTML document. Click the **Insert** button to finish inserting this applet.
	
	
	`<PARAM name =R4_ADDRESS value="400004" >`
	`<PARAM name =R5_LABEL value="Reg 400005" >`
	`<PARAM name =R5_ADDRESS value="400005" >`
	`<PARAM name =R6_LABEL value="Reg 400006" >`
	`<PARAM name =R6_ADDRESS value="400006" >`
	`<PARAM name =R7_LABEL value="Reg 400007" >`
	`<PARAM name =R7_ADDRESS value="400007" >`
	`<PARAM name =R8_LABEL value="Reg 400008" >`
	`<PARAM name =R8_ADDRESS value="400008" >`
	`<PARAM name =R9_LABEL value="Reg 400009" >`
	`<PARAM name =R9_ADDRESS value="400009" >`
	`<PARAM name =R10_LABEL value="Reg 400010" >`
	`<PARAM name =R10_ADDRESS value="400010" >`
	`</APPLET>`

7.3 Using Graphic Objects Lite

Overview

This section describes how to use the Graphic Objects Lite library. This library uses a modem and allows faster download of the graphical interface by the user. This is a lighter version of the standard graphical library.

What’s in this Section?

This section contains the following topics:

Downloading the Graphic Objects Lite Library

Overview

You must download the Graphic Objects Lite library before you use it. Use the Transfer Web File utility provided with Web Designer, and transfer the file **widgetslite.jar** to the module’s Web site.

NOTE: The **widgetslite.jar** file is located in the installation folder, in the subdirectory **/addons/jar**.

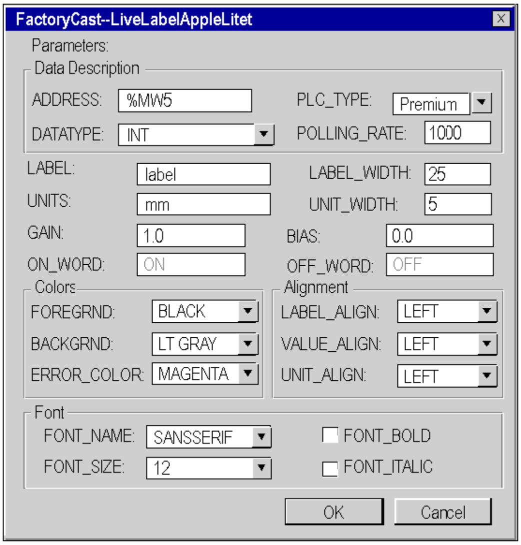
Description of Graphic Objects Lite

Presentation

The set of graphic objects provided in the FactoryCast_Applet can help you create graphic displays similar to the human-machine interface (HMI) screens. All the data control and monitoring objects have integrated communication functions and are designed as standalone graphic objects.

LiveLabel AppletLite Setup

The **LiveLabel AppletLite** window displays the direct address value of a Modbus slave in a text field.



FactoryCast--LiveLabelAppleLitet

Parameters:

Data Description

ADDRESS: %MW5

PLC_TYPE: Premium

DATATYPE: INT

POLLING_RATE: 1000

LABEL: label

LABEL_WIDTH: 25

UNITS: mm

UNIT_WIDTH: 5

GAIN: 1.0

BIAS: 0.0

ON_WORD: ON

OFF_WORD: OFF

Colors

FOREGRND: BLACK

BACKGRND: LT GRAY

ERROR_COLOR: MAGENTA

Alignment

LABEL_ALIGN: LEFT

VALUE_ALIGN: LEFT

UNIT_ALIGN: LEFT

Font

FONT_NAME: SANSSERIF

FONT_SIZE: 12

□ FONT_BOLD

□ FONT_ITALIC

The properties of this widget are as follows:

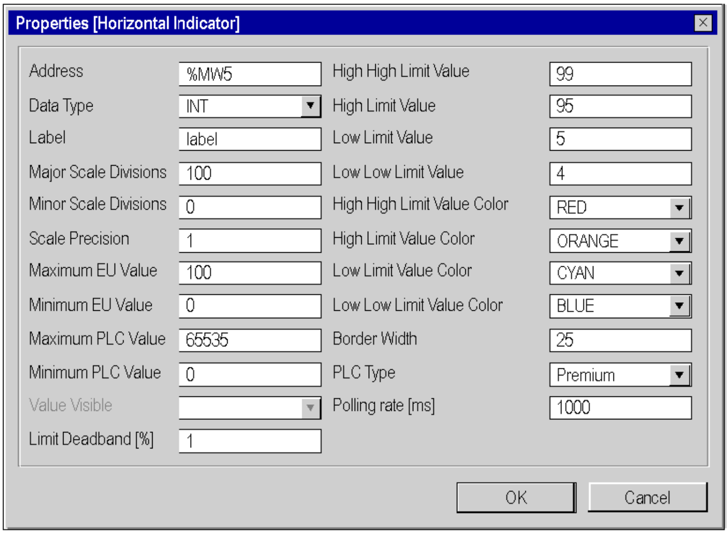
Property	**Description**	**Limits**
Address	Address of the PLC variable	Note 1 *(see page 286)*
Data type	PLC address data type	Note 2 *(see page 286)*
PLC Type	Type of PLC	Premium or Quantum
Label	Label displayed as part of the graphic object	Note 5 *(see page 286)*
Label Width	Width of label	
Unit Width	Width of unit	
Polling rate [ms]	Scanning value	
Gain	The gain (multiplier) is used for the scale of the value retrieved in physical units	1.0
Bias	The bias (multiplier) is used for the scale of the value retrieved in physical units	0.0
ON_Word	Text value displayed when the value is not zero (use if the data type is binary)	ON
OFF_Word	Text value displayed when the value is not zero (use if the data type is binary)	OFF
Foregrnd	Color of the applet foreground	BLACK
Backgrnd	Color of the applet background	LT_GRAY
Error_Color	Color of the applet foreground if the address value cannot be retrieved	MAGENTA
Label_Align	Alignment of text in the Label field if the size is greater than the text length	LEFT
Value_Align	Alignment of text in the Value field if the size is greater than the text length	LEFT
Units_Align	Alignment of text in the Units field if the size is greater than the text length	LEFT
Font_Name	Font name for the applet text	SANSSERIF
Font_Bold	Applet text is bold if configured as TRUE	FALSE
Font_Italic	Applet text is italic if configured as TRUE	FALSE
Font_Size	Applet text size	12

The HTML code with the parameters of the widget above is as follows:

```
<APPLET
codebase="/classes"
archive="mbclient.jar,widgetslite.jar"
code="com.schneiderautomation.factorycast.gateway.widgetsl
ite.LiveLabelApplet"
width="130" height="30">
   <PARAM name="progressbar"    value="true">
   <PARAM name="progresscolor"    value="#000000">
   <PARAM name="ADDRESS"    value="1">
   <PARAM name="UNITID"    value="0">
   <PARAM name="RATE"    value="1000">
   <PARAM name="DATATYPE"    value="REGISTER">
   <PARAM name="LABEL"    value="label">
   <PARAM name="LABEL_WIDTH"    value="25">
   <PARAM name="UNITS"    value="mm">
   <PARAM name="UNITS_WIDTH"    value="5">
</APPLET>
```

Horizontal or Vertical Indicator Setup

The horizontal or vertical indicator gives an analog representation of the value of a direct address of a Modbus slave. It is a horizontal or vertical bar whose length is proportional to the value. It represents a percentage of its range in physical units.



Properties [Horizontal Indicator]

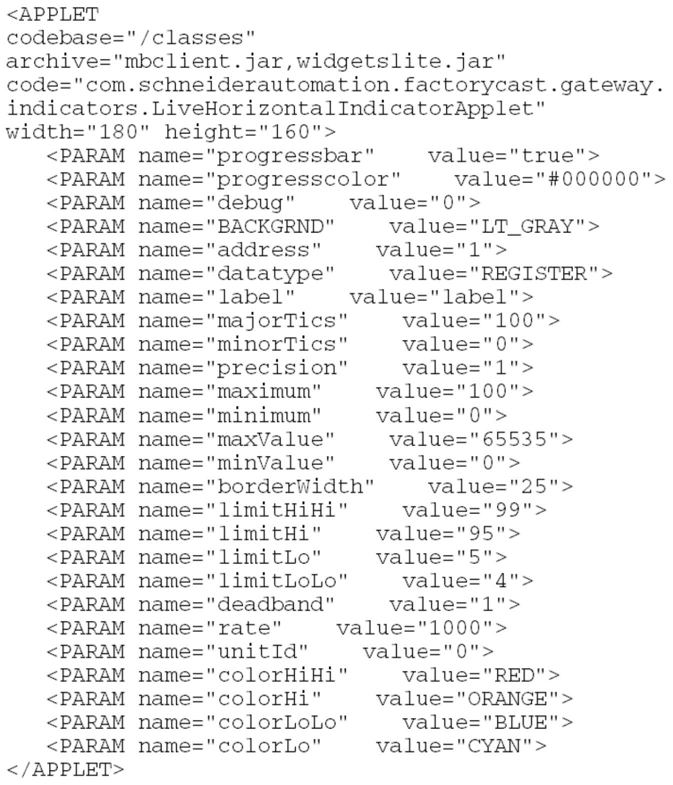
Address	%MW5	High High Limit Value	99
Data Type	INT	High Limit Value	95
Label	label	Low Limit Value	5
Major Scale Divisions	100	Low Low Limit Value	4
Minor Scale Divisions	0	High High Limit Value Color	RED
Scale Precision	1	High Limit Value Color	ORANGE
Maximum EU Value	100	Low Limit Value Color	CYAN
Minimum EU Value	0	Low Low Limit Value Color	BLUE
Maximum PLC Value	65535	Border Width	25
Minimum PLC Value	0	PLC Type	Premium
Value Visible		Polling rate [ms]	1000
Limit Deadband [%]	1		

The properties of the indicator are as follows:

Property	**Description**	**Limits**
Address	Address of the PLC variable	Note 1 *(see page 286)*
Data type	PLC address data type	Note 2 *(see page 286)*
Label	Label displayed as part of the graphic object	Note 5 *(see page 286)*
Major scale gradation	Number of major gradations (marked) in the scale	0 to 100
Minor scale gradation	Number of minor gradations (not marked) in the scale	0 to 100
Scale precision	Number of decimal places shown for the scale gradations (set to -1 to use a general exponential format)	-1 to 6
Maximum EU Value	Maximum value of the direct address for scaling, in physical units	
Minimum EU Value	Minimum value, in physical units, of the direct address for scaling	
Maximum PLC Value	Gross maximum value (without scale) of the direct address in the PLC	Note 3 *(see page 286)*
Minimum PLC Value	Gross minimum value (without scale) of the direct address in the PLC	Note 3 *(see page 286)*
Limit Deadband[%]	Neutral range (as a percentage of the UP range) to apply to verification of the High/Low limit	0 to 10
High High Limit Value	Value expressed in physical units of the "High High" limit	
High Limit Value	Value expressed in physical units of the "High" limit	
Low Limit Value	Value expressed in physical units of the "Low" limit	
Low Low Limit Value	Value expressed in physical units of the "Low Low" limit	
High High Limit Value Color	Color of the indicator bar if the scale value is greater than the "High High" limit	
High Limit Value Color	Color of the indicator bar if the scale value is greater than the "High" limit	
Low Limit Value Color	Color of the indicator bar if the scale value is less than the "Low" limit	
Low Low Limit Value Color	Color of the indicator bar if the scale value is less than the "Low Low" limit	

Property	Description	Limits
Border Width	Width of border	
PLC Type	Type of PLC	Quantum or Premium
Polling rate [ms]	Scanning value	

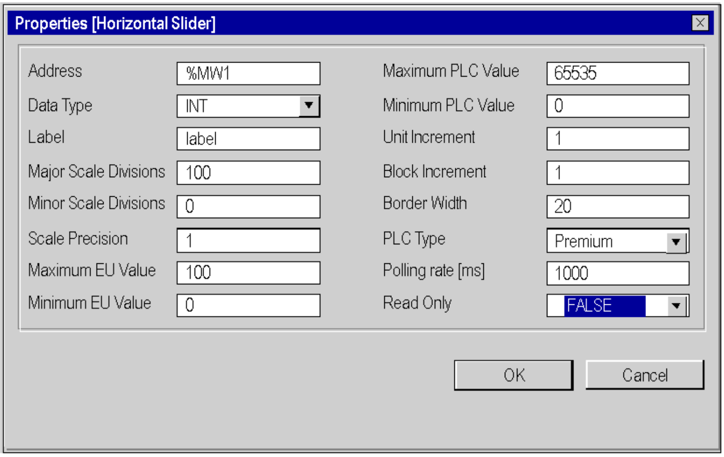
The HTML code with the parameters of the widget above is as follows:



```
<APPLET
codebase="/classes"
archive="mbclient.jar,widgetslite.jar"
code="com.schneiderautomation.factorycast.gateway.
indicators.LiveHorizontalIndicatorApplet"
width="180" height="160">
   <PARAM name="progressbar"    value="true">
   <PARAM name="progresscolor"    value="#000000">
   <PARAM name="debug"    value="0">
   <PARAM name="BACKGRND"    value="LT_GRAY">
   <PARAM name="address"    value="1">
   <PARAM name="datatype"    value="REGISTER">
   <PARAM name="label"    value="label">
   <PARAM name="majorTics"    value="100">
   <PARAM name="minorTics"    value="0">
   <PARAM name="precision"    value="1">
   <PARAM name="maximum"    value="100">
   <PARAM name="minimum"    value="0">
   <PARAM name="maxValue"    value="65535">
   <PARAM name="minValue"    value="0">
   <PARAM name="borderWidth"    value="25">
   <PARAM name="limitHiHi"    value="99">
   <PARAM name="limitHi"    value="95">
   <PARAM name="limitLo"    value="5">
   <PARAM name="limitLoLo"    value="4">
   <PARAM name="deadband"    value="1">
   <PARAM name="rate"    value="1000">
   <PARAM name="unitId"    value="0">
   <PARAM name="colorHiHi"    value="RED">
   <PARAM name="colorHi"    value="ORANGE">
   <PARAM name="colorLoLo"    value="BLUE">
   <PARAM name="colorLo"    value="CYAN">
</APPLET>
```

Horizontal or Vertical Slider Setup

A horizontal or vertical slider gives an analog representation of the direct address of a Modbus device. It is a cursor that is proportional to the value, and it represents a percentage of its range in physical units. Using the mouse, you can change the value of the slider and trigger, and you can send a new value to the Modbus slave.



Properties [Horizontal Slider]

Address	%MW1	Maximum PLC Value	65535
Data Type	INT	Minimum PLC Value	0
Label	label	Unit Increment	1
Major Scale Divisions	100	Block Increment	1
Minor Scale Divisions	0	Border Width	20
Scale Precision	1	PLC Type	Premium
Maximum EU Value	100	Polling rate [ms]	1000
Minimum EU Value	0	Read Only	FALSE

The properties of the slider are as follows:

Property	Description	Limits
Address	Address of the PLC variable	Note 1 *(see page 286)*
Data type	Data type of the PLC address	Note 2 *(see page 286)*
Label	Label displayed as part of the graphic object	Note 5 *(see page 286)*
Major Scale Division	Number of major gradations (marked) in the scale	0 to 100
Minor Scale Division	Number of minor gradations (not marked) in the scale	0 to 100
Scale Precision	Number of decimal places shown for the scale gradations (set to -1 to use a general exponential format)	-1 to 6
Maximum EU value	Maximum value, in physical units, of the direct address for scaling	
Minimum EU Value	Minimum value of the direct address for scaling, in physical units	

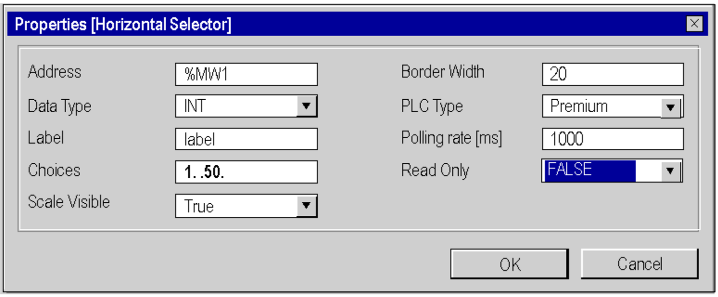
Property	Description	Limits
Maximum PLC Value	Gross maximum value (without scale) of the direct address in the PLC	Note 3 *(see page 286)*
Minimum PLC Value	Gross minimum value (without scale) of the direct address in the PLC	Note 3 *(see page 286)*
Unit Increment	Amount by which the scale value is modified when you click on the slider arrows	
Block Increment	Amount by which the scale value is modified when you click on the slide area of the bar	
Border Width	Width (in pixels) of the border of the graphic object	
PLC Type	Type of PLC	Premium or Quantum
Polling rate [ms]	Scanning value	
Read Only	TRUE = read value, FALSE = read/write value	

The HTML code with the parameters of the widget above is as follows:

```
<APPLET
codebase="/classes"
archive="mbclient.jar,widgetslite.jar"
code="com.schneiderautomation.factorycast.gateway.sliders.
LiveHorizontalSliderApplet"
width="180" height="160">
   <PARAM name="progressbar"    value="true">
   <PARAM name="progresscolor"    value="#000000">
   <PARAM name="debug"    value="0">
   <PARAM name="BACKGRND"    value="LT_GRAY">
   <PARAM name="address"    value="1">
   <PARAM name="datatype"    value="REGISTER">
   <PARAM name="label"    value="label">
   <PARAM name="majorTics"    value="100">
   <PARAM name="minorTics"    value="0">
   <PARAM name="precision"    value="1">
   <PARAM name="maximum"    value="100">
   <PARAM name="minimum"    value="0">
   <PARAM name="maxValue"    value="65535">
   <PARAM name="minValue"    value="0">
   <PARAM name="unitIncrement"    value="1">
   <PARAM name="blockIncrement"    value="1">
   <PARAM name="rate"    value="1000">
   <PARAM name="unitId"    value="0">
   <PARAM name="borderWidth"    value="20">
   <PARAM name="readOnly"    value="False">
</APPLET>
```

Horizontal or Vertical Selector

A horizontal or vertical selector allows the user to choose from a number of options. Once the selection has been made, the value corresponding to the choice is sent to the PLC. The choices are represented by marks on a "scale", the current selection being indicated by the position of the cursor on a slider.



Properties [Horizontal Selector]

Address: %MW1
Data Type: INT
Label: label
Choices: 1..50.
Scale Visible: True
Border Width: 20
PLC Type: Premium
Polling rate [ms]: 1000
Read Only: FALSE

The properties of the slider are as follows:

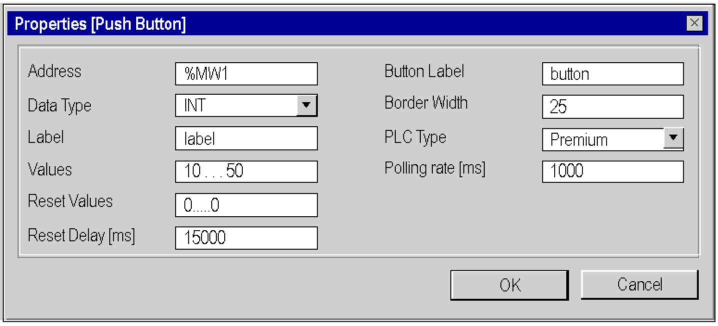
Property	Description	Limits
Address	Address of the PLC variable	Note 1 *(see page 286)*
Data type	PLC address data type	Note 2 *(see page 286)*
Label	Label displayed as part of the graphic object	Note 5 *(see page 286)*
Choices	Setpoints to apply to the address value	
Border Width	Width (in pixels) of the border of the graphic object	
PLC Type	Type of PLC	Premium or Quantum
Polling rate [ms]	Scanning value	
Read Only	True = read value, False = read/write value	

The HTML code with the parameters of the widget above is as follows:

```
<APPLET
codebase="/classes"
archive="mbclient.jar,widgetslite.jar"
code="com.schneiderautomation.factorycast.gateway.selectors
.LiveHorizontalSelectorApplet"
width="180" height="160">
   <PARAM name="progressbar"    value="true">
   <PARAM name="progresscolor"    value="#000000">
   <PARAM name="debug"    value="0">
   <PARAM name="BACKGRND"    value="LT_GRAY">
   <PARAM name="address"    value="1">
   <PARAM name="datatype"    value="REGISTER">
   <PARAM name="label"    value="label">
   <PARAM name="scaleVisible"    value="True">
   <PARAM name="choices"    value="1=1000,9=9000,50=50000">
   <PARAM name="rate"    value="1000">
   <PARAM name="unitId"    value="0">
   <PARAM name="borderWidth"    value="20">
   <PARAM name="readOnly"    value="False">
</APPLET>
```

Push Button Setup

You can send a preset value to one or more Modbus slaves with a push button. A push button is activated with the mouse.



Properties [Push Button]

Address	%MW1	Button Label	button
Data Type	INT	Border Width	25
Label	label	PLC Type	Premium
Values	10 . . . 50	Polling rate [ms]	1000
Reset Values	0.....0		
Reset Delay [ms]	15000		

The properties of the push button are as follows:

Property	Description	Limits
Address	Address of the PLC variable	Note 1 *(see page 286)*
Data type	PLC address data type	Note 2 *(see page 286)*
Label	Label displayed as part of the graphic object	Note 5 *(see page 286)*
Values	Setpoints to apply to the address value when the button is pressed	Note 4 *(see page 286)*
Reset Values	Value to apply when the Reset Delay ends	
Reset Delay [ms]	Time in ms counted down after the button is pressed	
Button Label	Label to display on the button	
Border Width	Width (in pixels) of the border of the graphic object	
PLC Type	Type of PLC	Premium or Quantum
Polling rate [ms]	Scanning value	

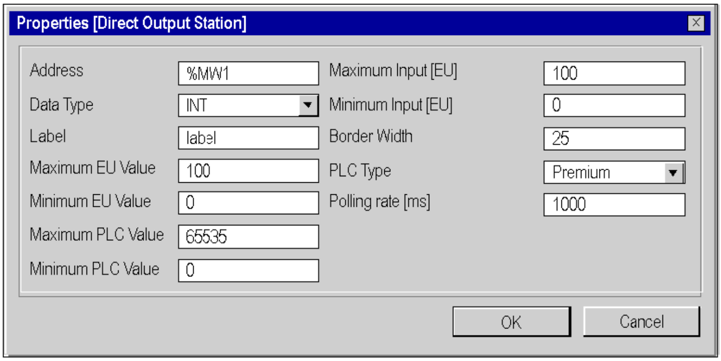
The HTML code with the parameters of the widget above is as follows:

```
<APPLET
codebase="/classes"
archive="mbclient.jar,widgetslite.jar"
code="com.schneiderautomation.factorycast.gateway.pushButton
.LivePushButtonApplet"
width="180" height="160">
   <PARAM name="progressbar"    value="true">
   <PARAM name="progresscolor"    value="#000000">
   <PARAM name="debug"    value="0">
   <PARAM name="BACKGRND"    value="LT_GRAY">
   <PARAM name="address"    value="1">
   <PARAM name="datatype"    value="REGISTER">
   <PARAM name="label"    value="label">
   <PARAM name="values"    value="10,50">
   <PARAM name="resetValues"    value="0">
   <PARAM name="resetDelay"    value="15000">
   <PARAM name="borderWidth"    value="25">
   <PARAM name="buttonLabel"    value="button">
   <PARAM name="rate"    value="1000">
   <PARAM name="unitID"    value="0">
</APPLET>
```

NOTE: In this example, when you press the button, the value 10 is applied to address 1 and the value 50 is applied to address 2 for 15,000 ms. After 15,000 ms, the two addresses are reset to 0.

Direct Output Window

With the Direct Output Window applet, you can enter a value in a text input field directly from the keyboard. If you enter a numerical value between the upper and lower preset limits, an OK button is activated. The value is sent to the Modbus slave each time you click OK or press the ENTER key (if the input field is active for keyboard input).



Properties [Direct Output Station]

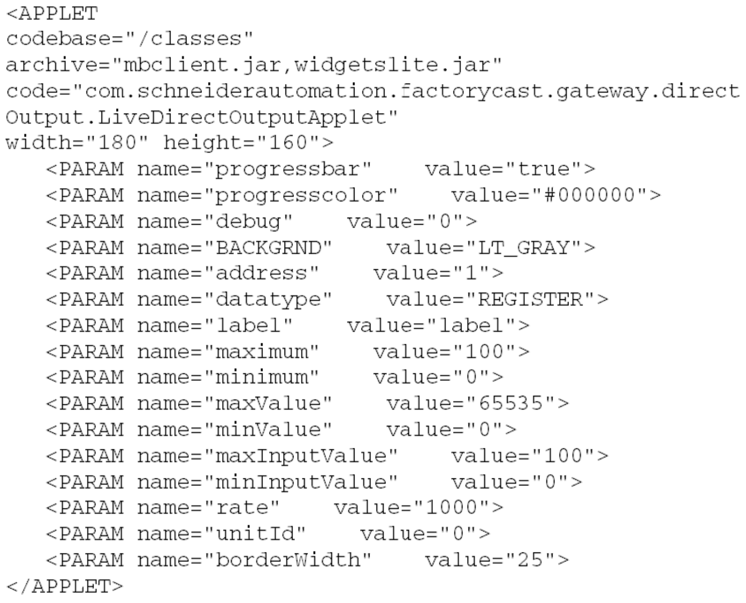
Address	%MW1	Maximum Input [EU]	100
Data Type	INT	Minimum Input [EU]	0
Label	label	Border Width	25
Maximum EU Value	100	PLC Type	Premium
Minimum EU Value	0	Polling rate [ms]	1000
Maximum PLC Value	65535		
Minimum PLC Value	0		

The direct output properties are as follows:

Property	Description	Limits
Address	Address of the PLC variable	Note 1 *(see page 286)*
Data type	PLC address data type	Note 2 *(see page 286)*
Label	Label displayed as part of the graphic object	Note 5 *(see page 286)*
Maximum EU Value	Maximum value, in physical units, of the direct address for scaling	
Minimum EU Value	Minimum value, in physical units, of the direct address for scaling	
Maximum PLC Value	Gross maximum value (without scale) of the direct address in the PLC	Note 3 *(see page 286)*
Minimum PLC Value	Gross minimum value (without scale) of the direct address in the PLC	Note 3 *(see page 286)*
Maximum Input [EU]	Maximum setpoint for the input	

Property	Description	Limits
Minimum Input [EU]	Minimum setpoint for the input	
Border Width	Width (in pixels) of the border of the graphic object	
PLC Type	Type of PLC	Premium or Quantum
Polling rate [ms]	Scanning value	

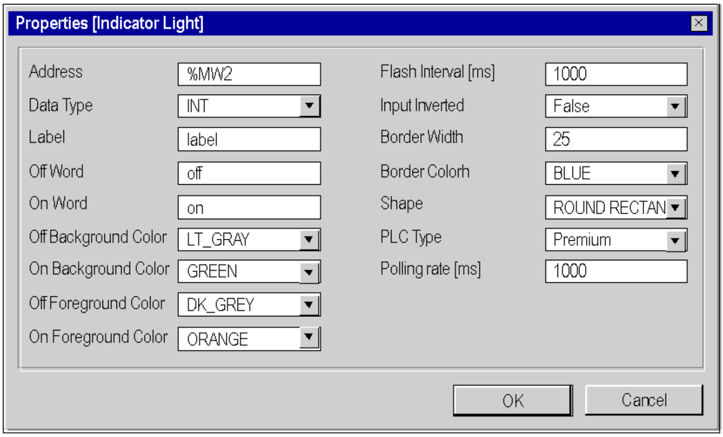
The HTML code with the parameters of the widget above is as follows:



```
<APPLET
codebase="/classes"
archive="mbclient.jar,widgetslite.jar"
code="com.schneiderautomation.factorycast.gateway.direct
Output.LiveDirectOutputApplet"
width="180" height="160">
   <PARAM name="progressbar"    value="true">
   <PARAM name="progresscolor"    value="#000000">
   <PARAM name="debug"    value="0">
   <PARAM name="BACKGRND"    value="LT_GRAY">
   <PARAM name="address"    value="1">
   <PARAM name="datatype"    value="REGISTER">
   <PARAM name="label"    value="label">
   <PARAM name="maximum"    value="100">
   <PARAM name="minimum"    value="0">
   <PARAM name="maxValue"    value="65535">
   <PARAM name="minValue"    value="0">
   <PARAM name="maxInputValue"    value="100">
   <PARAM name="minInputValue"    value="0">
   <PARAM name="rate"    value="1000">
   <PARAM name="unitId"    value="0">
   <PARAM name="borderWidth"    value="25">
</APPLET>
```

Indicator Light Setup

The indicator light provides a dual indication of the value of a direct address in a PLC. If the Input inverted property is not set to TRUE, a 0 input value is declared OFF, and a non-zero value is declared ON. If the Flash Interval property is set to a positive value, the indicator light flashes when the input value is equivalent to ON.



Properties [Indicator Light]

Address	%MW2	Flash Interval [ms]	1000
Data Type	INT	Input Inverted	False
Label	label	Border Width	25
Off Word	off	Border Colorh	BLUE
On Word	on	Shape	ROUND RECTAN
Off Background Color	LT_GRAY	PLC Type	Premium
On Background Color	GREEN	Polling rate [ms]	1000
Off Foreground Color	DK_GREY		
On Foreground Color	ORANGE		

The properties of the indicator light are as follows:

Property	Description	Limits
Address	Address of the PLC variable	Note 1 *(see page 286)*
Data type	PLC address data type	Note 2 *(see page 286)*
Label	Label displayed as part of the graphic object	Note 5 *(see page 286)*
OFF Word	Text displayed when the input value is OFF	
ON Word	Text displayed when the input value is ON	
OFF Background Color	Background color of the indicator light when OFF Word is displayed	
ON Background Color	Background color of the indicator light when ON Word is displayed	
OFF Foreground Color	OFF Word text color	
ON Foreground Color	ON Word text color	

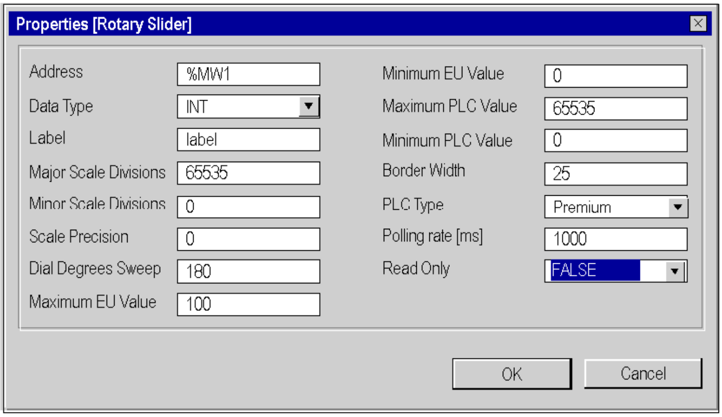
Property	Description	Limits
Flash Interval	The flashing time for the indicator light (expressed in ms) when the input value is ON. Set to zero for no flashing	200 to 2000
Input inverted	On TRUE, inverts the input value (the indicator light displays the OFF Word when the input value is ON)	
Border Width	Width (in pixels) of the border of the graphic object	
Border Color	Color of the border	
Shape	Shape (circular, rectangular, etc.) of the indicator light	
PLC Type	Type of PLC	Premium or Quantum
Polling rate [ms]	Scanning value	

The HTML code with the parameters of the widget above is as follows:

```
<p><APPLET
codebase="/classes"
archive="mbclient.jar,widgetslite.jar"
code="com.schneiderautomation.factorycast.gateway.indica-
torLight.LiveIndicatorLightApplet"
width="180" height="160">
   <PARAM name="progressbar"    value="true">
   <PARAM name="progresscolor"    value="#000000">
   <PARAM name="debug"    value="0">
   <PARAM name="BACKGRND"    value="LT_GRAY">
   <PARAM name="address"    value="2">
   <PARAM name="datatype"    value="REGISTER">
   <PARAM name="label"    value="label">
   <PARAM name="offWord"    value="off">
   <PARAM name="onWord"    value="on">
   <PARAM name="offWordBackground"    value="LT_GRAY">
   <PARAM name="onWordBackground"    value="GREEN">
   <PARAM name="offWordForeground"    value="DK_GRAY">
   <PARAM name="onWordForeground"    value="ORANGE">
   <PARAM name="flashInterval"    value="1000">
   <PARAM name="inputInverted"    value="False">
   <PARAM name="borderWidth"    value="25">
   <PARAM name="borderColor"    value="BLUE">
   <PARAM name="shape"    value="ROUND RECTANGLE">
   <PARAM name="rate"    value="1000">
   <PARAM name="unitId"    value="0">
</APPLET>
```

Rotary Slider Setup

A rotary slider gives an analog representation of the direct address of a Modbus device. It draws a position proportional to the value of the address and represents a percentage of its range in physical units on a circular dial. The size of the circular dial (cycle in degrees of a circle) and the button color can be configured. Using the mouse, the user can change the value of the rotary slider and trigger sending a new value to the PLC.



Properties [Rotary Slider]

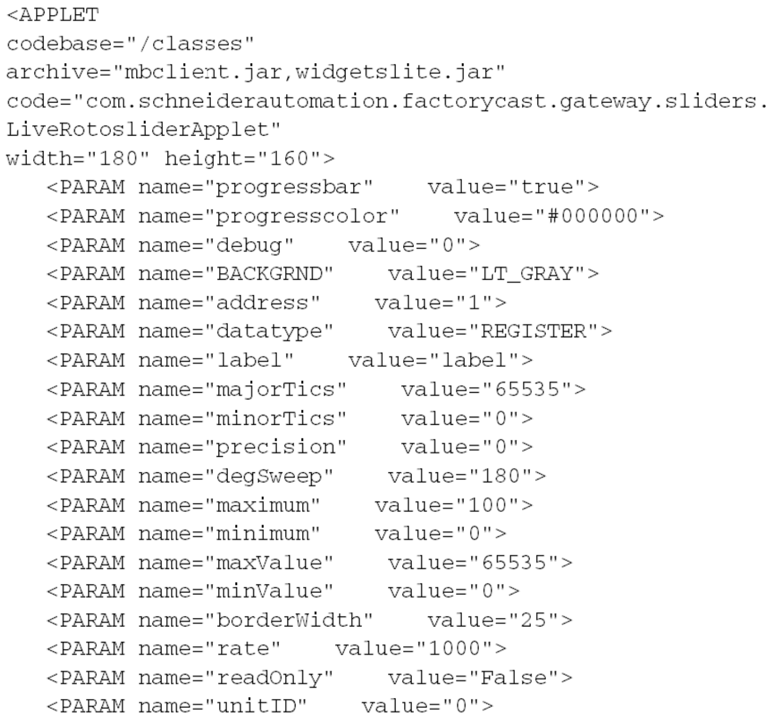
Address	%MW1	Minimum EU Value	0
Data Type	INT	Maximum PLC Value	65535
Label	label	Minimum PLC Value	0
Major Scale Divisions	65535	Border Width	25
Minor Scale Divisions	0	PLC Type	Premium
Scale Precision	0	Polling rate [ms]	1000
Dial Degrees Sweep	180	Read Only	FALSE
Maximum EU Value	100		

The properties of the slider are as follows:

Property	Description	Limits
Address	Address of the PLC variable	Note 1 *(see page 286)*
Data type	PLC address data type	Note 2 *(see page 286)*
Label	Label displayed as part of the graphic object	Note 5 *(see page 286)*
Major scale gradation	Number of major gradations (marked) in the scale	0 to 100
Minor scale gradation	Number of minor gradations (not marked) in the scale	0 to 100
Scale precision	Number of decimal places in the scale gradations (set to -1 to use a general exponential format)	-1 to 6
Dial Degrees Sweep	Portion of circular dial used to draw the scale	60 to 300
Maximum EU Value	Maximum value of the direct address for scaling, in physical units	
Minimum EU Value	Minimum value of the direct address for scaling, in physical units	

Property	Description	Limits
Maximum PLC Value	Gross maximum value (without scale) of the direct address in the PLC	Note 3 *(see page 286)*
Minimum PLC Value	Gross minimum value (without scale) of the direct address in the PLC	Note 3 *(see page 286)*
Border Width	Width (in pixels) of the border of the graphic object	
PLC Type	Type of PLC	Premium or Quantum
Polling rate [ms]	Scanning value	
Read Only	True = read value, False = read/write value	

The HTML code with the parameters of the widget above is as follows:



```
<APPLET
codebase="/classes"
archive="mbclient.jar,widgetslite.jar"
code="com.schneiderautomation.factorycast.gateway.sliders.
LiveRotosliderApplet"
width="180" height="160">
   <PARAM name="progressbar"    value="true">
   <PARAM name="progresscolor"    value="#000000">
   <PARAM name="debug"    value="0">
   <PARAM name="BACKGRND"    value="LT_GRAY">
   <PARAM name="address"    value="1">
   <PARAM name="datatype"    value="REGISTER">
   <PARAM name="label"    value="label">
   <PARAM name="majorTics"    value="65535">
   <PARAM name="minorTics"    value="0">
   <PARAM name="precision"    value="0">
   <PARAM name="degSweep"    value="180">
   <PARAM name="maximum"    value="100">
   <PARAM name="minimum"    value="0">
   <PARAM name="maxValue"    value="65535">
   <PARAM name="minValue"    value="0">
   <PARAM name="borderWidth"    value="25">
   <PARAM name="rate"    value="1000">
   <PARAM name="readOnly"    value="False">
   <PARAM name="unitID"    value="0">
```

Notes

1.	The address can be:	
	%MW	IEC internal word
	%MD	IEC double word
	%M	IEC internal bit
	400000	Concept integer
	100000	Concept boolean
2.	The various values of the Data type property have the following meaning:	
	Data type	Meaning
	`INT`	`integer`
	`DINT`	`double integer`
	`BOOL`	`boolean`
3.	The limits of the Maximum PLC Value and Minimum PLC Value properties are the natural limits of the configured Data type property.	
4.	Specify at least one value for a push button. If several values are entered, they will be assigned to an address table starting with the direct address indicated.	
5.	If you specify `param name="label" value="$data$’` in the HTML code, the applet displays the numerical value of the data in place of a label.	

Alarm Viewer

8

Overview

The Alarm Viewer is a Java applet designed to monitor Premium and Quantum PLCs. It is a Client Runtime Function and will run in the JVM of internet client browser (Internet Explorer, Mozilla Firefox, etc.).

What's in this Chapter?

This chapter contains the following topics:

Display

Access

Select **Diagnostics | Alarm Viewer**.

NOTE: Alarm Viewer is a plug-in. You must download it before you can use it.

The display is composed of

- Button bar (in the work area)
- List of alarms
- Status frame

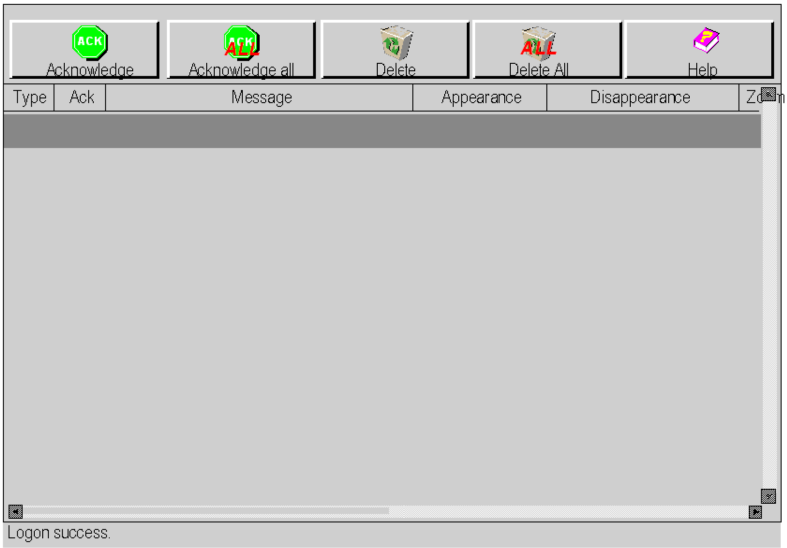
Button Bar

The button bar contains five buttons.

ACK	This button is used to acknowledge a selected alarm in the list. A request is sent to the PLC (Diagnostic Buffer).
ACK ALL	This button is used to acknowledge all alarms in the list that can be acknowledged. A request for each alarm is sent to the PLC.
🗑	This button is used to delete a selected alarm in the list. There is no request sent to the PLC. This command affects only the alarm list in Alarm Viewer.
🗑 ALL	This button is used to delete all alarms in the list that can be deleted. There is no request sent to the PLC. This command affects only the alarm list in Alarm Viewer.
?	This button displays a frame that contains help.

List of Alarms

Alarms in the list are displayed in historical order. The last alarm in runtime is added to the end of the list.



Logon success.

Each line displayed in the list corresponds to an alarm and contains the following information

Type: An icon that represents the alarm type. For each type of alarm there is a different icon.

DFB DFB alarm

Grafcet alarm

System alarm

ASi A-SI alarm

Ack: An icon that represents the acknowledgement status.

STOP This alarm must be acknowledged by the user.

ACK Alarm has been acknowledged.

OK Alarm doesn’t need to be acknowledged.

Message: Alarm text.

Appearance: Date and time when alarm occurs.

Disappearance: Date and time when alarm disappears.

Zone or area number: Area or geographical zone from which the alarm comes (common area: 0).

NOTE: Consider the following.

- You can change the column width by using the mouse.
- The number of alarms that can be displayed in the list is limited to 1000. When this limit is reached, an information message is displayed in the status frame. Alarm Viewer recognizes subsequent alarms, but doesn’t display them. To see the next alarms, the user must purge the list of alarms displayed by Alarm Viewer.
- Alarm Viewer displays alarms from all zones. The zone contains values from 0...15.

NOTE: An alarm that appears is displayed in red. An alarm that disappears is displayed in green.

Status Frame

This frame is used to display the alarm, an information message, or throughput messages. For example a message such as, "Diagnostic Application is not configured in this application."

Operation and Management of Alarms

Browsing

Use the UP, DOWN, PAGEUP, or PAGEDOWN keys or the mouse to select the alarms in the list. Use the scroll bar if the list contains more alarms than can be displayed in the dialog.

Acknowledg-ment

To acknowledge an alarm that requires acknowledgment, select the alarm and use the appropriate toolbar button.

Several alarms can be acknowledged at one time by using the **Ack All** button.

NOTE: An alarm can be acknowledged by another Alarm Viewer. In this case, Alarm Viewer is notified and the alarm is displayed as acknowledged.

Deleting an Alarm from the List

- An alarm that requires acknowledgment or that has not disappeared cannot be deleted.
- DELETE and DELETE ALL buttons can be used to delete only those alarms that have disappeared and have been acknowledged (if acknowledgment is required).

Limitations

Overview

For each alarm, there is additional information stored in the diagnostic buffer. For example, many DFBs have outputs named STATUS (word) where the cause is coded. Alarm Viewer doesn't use this information - it displays only basic information about alarms.

Alarm Viewer Functionality

The Alarm Viewer works only under the conditions listed below:

Product	Firmware Platform	Software Platform
TSX WMY 100	Premium PLC TSX57/PCX57/PMX57 V3.3	PL7 PRO PL-7 Junior V3.3 Unity Pro
TSX NWM 100	Quantum Unity Platform	Unity Pro

Description of the TSX WMY 100 and 140 NWM 100 00 modules

II

Subject of this Part

This part describes the Premium TSX WMY 100 module and the Quantum 140 NWM 100 00 module.

What's in this Part?

This part contains the following chapters:

Compatibility with the different software workshops

9

FactoryCast HMI module compatibility with different software workshops

At a Glance

To implement FactoryCast HMI applications, you need to configure the module in the appropriate software workshop (i.e. Unity Pro, Concept or PL7).

Compatibility

The table below shows which FactoryCast HMI modules are compatible with which software workshop versions:

	Unity Pro	Concept	PL7
TSX WMY 100	V2.0		V4.4 with TSX WMY 100 update
140 NWM 100 00	V2.0	V2.6	

NOTE: Earlier software versions are upwardly compatible with later versions.

TSX WMY 100 module

10

Subject of this Chapter

This chapter covers the implementation of hardware for a **TSX WMY 100** HMI module on a Premium PLC.

What's in this Chapter?

This chapter contains the following sections:

10.1 At a Glance

At a Glance

General

The **TSX WMY 100** HMI module uses FactoryCast HMI software. It is made up of a communication channel whose main features are as follows:

- Connection to a TCP/IP network.
- Communication in Half and Full Duplex mode by automatic recognition.
- Transmission speed from 10 or 100 Mbits/s by automatic recognition.
- Connection to network by copper cable via an RJ45 connector.

This module is used to carry out the following functions:

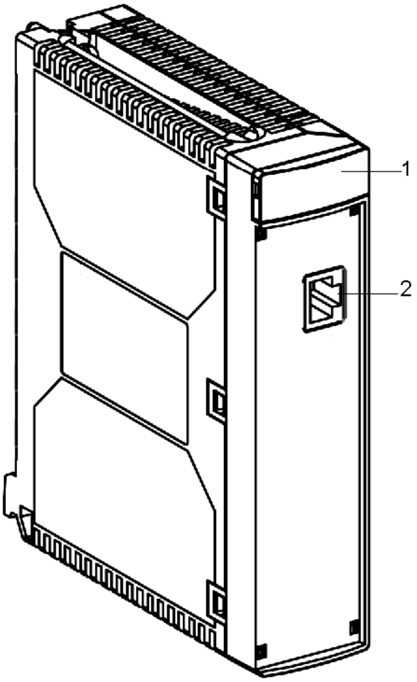
- X-WAY UNI-TE and Modbus messaging service on TCP/IP.
- SNMP Service,
- Web server,
- E-mail service,
- Database.

10.2 Description

Physical Description

View of the Module

The **TSX WMY 100** HMI module is a standard format module which is inserted in a slot on the main or extension rack of a Premium PLC station.

1. Display panel indicating module status: ● One green **RUN** LED. ● Two red LEDs: **ERR** and **COL**. ● Three yellow LEDs: **STS**, **TX** and **RX**. **2.** Standard connector for 10baseT (RJ45) interface.	Illustration: 

10.3 Ethernet Channel Characteristics

Communication Channel Characteristics

General

The **TSX WMY 100** HMI module has a standard interface for connecting to a 10/100BASE-T network. On the front panel is a RJ45 connector for a point-to-point link via a link cable comprising two independent twisted pairs.

Services and functions supported by the modules include:

Support	Service	Protocol	Functions
TCP-IP Services	Messaging	UNI-TE	● Client/server mode. ● Synchronous requests of 256 bytes. ● Asynchronous requests of 1 Kbyte.
		Modbus	● Data exchange.
	Network management	SNMP	● SNMP agent, (MIB II, Schneider MIB).
	Web	HTTP	● Website which can be modified and increased by increments within the limit of 7.5Mb.

Maximum Capacity of the Module

The maximum frame size depends on the type of transaction:

- In synchronous messaging, the maximum frame size is 256 bytes.
- In asynchronous messaging, the maximum frame size is 1 KByte.

The TSX WMY 100 module manages 64 TCP connections using the port 502 messaging service.

In conjunction with a dedicated processor, the module can be used:

- for synchronous X-WAY messaging on TCP/IP (UNI-TE server):
  - with a TSX P57-453 processor: 800 messages per second,
  - with a TSX P57-303 processor: 490 messages per second,
- for asynchronous X-WAY messaging on TCP/IP:
  - between 600 and 1200 messages per second (number varies depending on the size of the messages, the number of clients and the application execution time).

Faulty Device Replacement

The module reserves a 512 KBytes zone for this function.

Compatibility

The **TSX WMY 100** module can be configured only in a TSX Premium with a V5.1 or higher processor.

The functional level of the **TSXWMY 100** module requires version V4.4 of the PL7 software.

10.4 Installing the Module

Subject of this Section

This section covers the installation of a **TSX WMY 100** HMI module on a Premium PLC.

What's in this Section?

This section contains the following topics:

At a Glance

General

The **TSX WMY 100** HMI module is mounted in the rack slot of a Premium/Atrium PLC station. It can be installed in any available slot (except in an offset X Bus rack), on condition that the supply constraints of the rack are observed.

Selecting the Type of Processor

Selection Guide

Selecting the processor to control the PLC station will depend on the number of network connections required.

Processors	Number of network connections	Number of TSX WMY 100 modules
TSX P57 1••/2•• **PCX 57 253**	1	1
TSX P57 3•• **PCX 57 353**	3	3
TSX P57 4••	4	4
TSX P57 2•23/2634	1	0
TSX P57 3623/3634	3	2
TSX P57 4823	4	3
TSX P57 5634	5	4
(*) assuming that the power consumption breakdown on the 5V supply is compatible with the power supply selected.		

Connection/Disconnection with Power Switched On

The Module

The **TSX WMY 100** HMI module can be connected or disconnected when switched on without disrupting the operation of the station.

The modules do not have an internal RAM backup memory function: this will be erased when power is switched off.

The modules reset when switched on. A communication break can be expected during these interventions.

The link

The 10/100BASE-T interface RJ45 connector can be connected or disconnected when power is on. A communication break can therefore be expected in the application in progress.

10.5 10/100BASE-T Interface

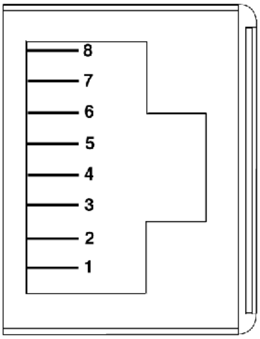
10/100BASE-T Interface

Presentation

This interface has a standard type RJ45 connector.

Refer to the ETHERNET reference manual for the connection accessories that comply with environmental circumstances the PLC requires in an industrial setting.

Pin Assignment



Pin	Signal
1	TD+
2	TD-
3	RD+
4	not connected
5	not connected
6	RD-
7	not connected
8	not connected

NOTE: If there is a connection via a shielded cable, the connector casing on the module is linked up to the ground connection.

Speed Line

The choice of different line speeds for the TSX WMY 100 module are as follows:

- 100 Mb in Half Duplex
- 10 Mb in Half Duplex
- 10 Mb in Full Duplex

Speed Adaptation

You cannot configure the speed line. The line adapts itself as follows:

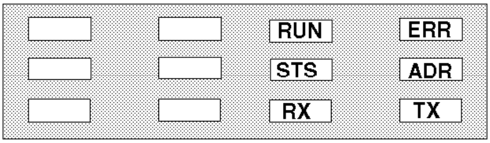
1	Each entity broadcasts its possibilities on the line.
2	The chosen speed is the fastest of the entity possibilities on the line. In other words, speed is limited by the slowest entity on the line of which the speed possibility is the weakest.

10.6 Display, Diagnostics

Display Panel, Diagnostics

General

The display panel conforms to the Premium standard



The COL, RX and TX LEDs are managed by the line’s electronics; they indicate:

- COL: a collision.
- RX: a reception
- TX: a transmission.

Diagnostics

Meaning of the diagnostics LEDs:

RUN	ERR	STS	COL	TX	RX	Meaning
OFF	OFF	OFF	ns	ns	ns	No supply to module.
OFF	OFF	ON	ns	ns	ns	Module running self-test.
ON	OFF	OFF	ns	ns	ns	Module ready.
OFF	ON	OFF	ns	ns	ns	Module not operational.
OFF	ON	ON	ns	ns	ns	Temporary state causing module reinitialization.
OFF	F	ON, F	ns	ns	ns	Module not configured or configuration in progress.
ON	OFF	ON	ns	ns	ns	Module configured, operational.
ns	ns	F	ns	ns	ns	Module configured. Diagnostics according to how the LEDs are flashing: ● 2 flashes: module has no MAC address. ● 3 flashes: ETHERNET cable not connected on the module or Hub side ● 4 flashes: the module IP address is duplicated by another IP address on the network. Conflicting remote device flashing in the same way. ● 5 flashes: module configured as a BOOTP client and is waiting for a BOOTP server response.
ON	OFF	ON	OFF	F	OFF	Sending ETHERNET communication.
ON	OFF	ON	OFF	OFF	F	Receiving ETHERNET communication.
ON	OFF	ON	OFF	F	F	Sending/Receiving ETHERNET communication.
ON	OFF	ON	F	F	OFF	Module has detected collision.
ON = Permanently on, F = flashing, OFF = Off, ns = not significant						

10.7 Electrical Characteristics

Electrical Characteristics

Table of Consumption

The **TSX WMY 100** HMI module can be inserted in any rack slot of a Premium/Atrium station (except in the X Bus offset rack).

At 5 V, the TSX WMY 100 consumes power as follows:

Power consumption		**Power dissipation**	
Typical	Maximum	Typical	Maximum
360 mA	400 mA	1.8 W	2.1 W

10.8 Standards

Norms and Standards

Compliance with Standards

The **TSX WMY 100** HMI module complies with the following standards:

- UL 508
- CSA
- IEC 1121-2
- Marine classification

10.9 Operating Conditions

Operating Conditions

Applicable Conditions

- **Configuration software:**
  - **PL7 version** ≥ **V4.4**: recognizes processors including TSX WMY 100 module.
- **Conditions of use:**
  - Temperature from 0 to +60° C
  - Relative humidity between 10% and 95% (without condensation) at 60° C
  - Altitude of between 0 and 4500 meters
  - Immunity to vibrations complies with the IEC 68-2-6 standard, Fc test
  - Immunity to shocks complies with the IEC 68-2-27 standard, Ea test
  - Immunity to free fall, hardware packaged as per IEC 68-2-32 standard, method 1
  - IP 20 protection rating
- **Storage conditions:**
  - Temperature from -40° C to +85° C
  - Relative humidity between 0% and 95% (without condensation) at 60° C

Quantum module 140 NWM 100 00

11

Subject of this Chapter

This chapter covers the implementation of hardware for a 140 NWM 100 00 Quantum HMI module.

What's in this Chapter?

This chapter contains the following sections:

11.1 140 NWM 100 00 Product Description

Overview

This section provides an overview of the 140 NWM 100 00 primary features and LED indicators.

What’s in this Section?

This section contains the following topics:

140 NWM 100 00 Module Overview

Presentation

The following information provides an overview of the Quantum 140 NWM 100 00 module.

General Description

The Qauntum 140 NWM 100 00, 10/100 Ethernet module, is the latest model in a line of Quantum Ethernet TCP/IP modules. The module is designed to make it possible for a Quantum Programmable Logic Controller (PLC) to communicate with devices over an Ethernet network. The electronics for the 140 NWM 100 00 module are contained in a standard Quantum single width case that takes up one slot in a Quantum backplane.

The 140 NWM 100 00 provides real-time peer-to-peer communications and a Modbus/TCP server. The included HTTP services provide maintenance and configuration utilities to the module.

The following table shows the different features of the 140 NWM 100 00 Ethernet module. A key new feature of this module is that it supports embedded Java Virtual Machine.

Key Features

This is a module that provides a PLC with access to an Ethernet network. The module can plug into any available slot with a Quantum backplane and is capable of being hot swapped. The key features are listed below.

Feature	140 NWM 100 00
Java Virtual Machine	yes
FTP Server	yes
Flash File System	yes
HTTP Server	yes
SNMP v1	yes
Schneider Private MIB	yes
FactoryCast HMI	yes
User Programmable Web Pages	yes
Modbus Messaging	yes

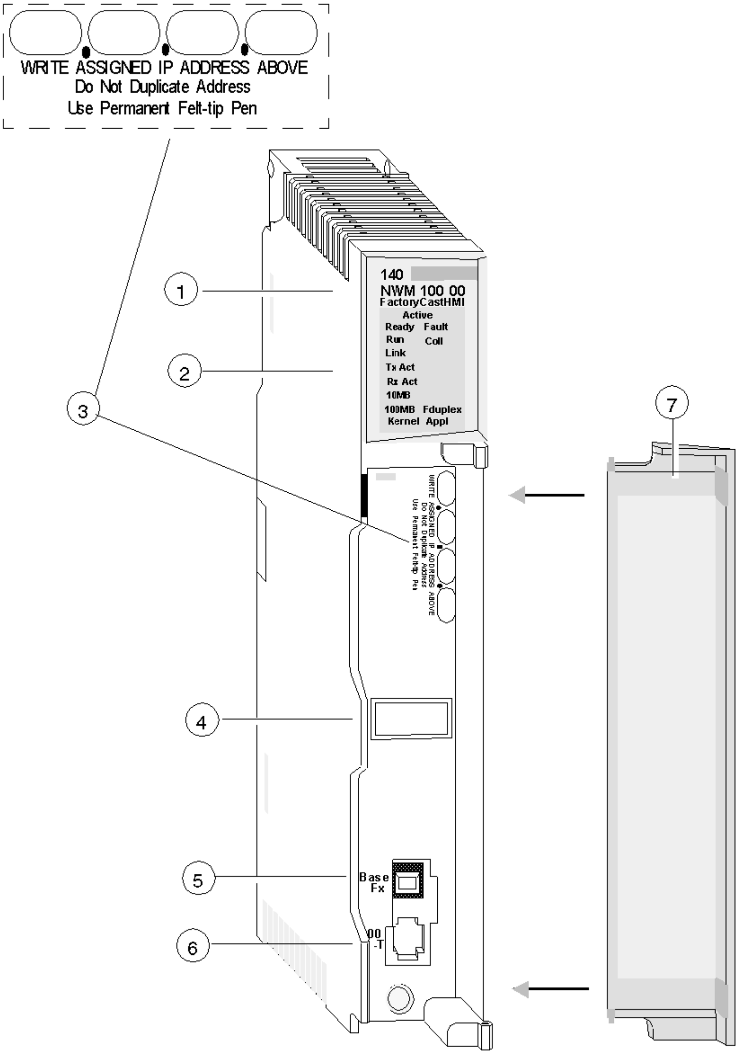
Front Panel Components

The front panel of the 140 NWM 100 00 module contains identification markings, color codes, and LED displays. A writable area for an IP address, a global address label, and two Ethernet cable connectors are located behind the removable front panel door.

The following table provides a description of the front panel components, which are shown in front view figure.

Component	Description
LED indicator Panel	Indicates the operating status of the module and the fiber optic or Ethernet communications network to which it is connected. (See LED Indicators in this chapter.)
IP Address Area	Provides a writable area to record the module’s assigned IP address.
Global Address Label	Indicates the module’s global Ethernet MAC address assigned at the factory.
100 BASE-FX Connector	Provides an MT-RJ socket for connection to a 100 megabit fiber optic Ethernet cable.
10/100BASE-T Connector	Provides an RJ-45 socket for connection to a shielded, twisted pair Ethernet cable. Connects to both 10 and 100 mbps networks.

Front View



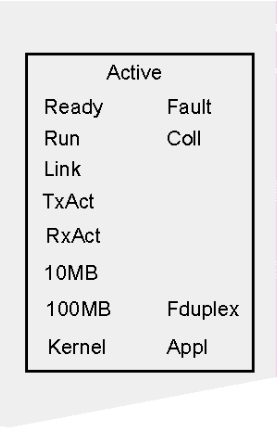
1 Model number, module description, color code
2 LED display
3 IP Address writable area
4 Global address label
5 100 Base Fx MT-RJ cable connector
6 10/100BASE-T RJ-45 cable connector
7 Removable door

LED Indicators

Presentation

The LED indicator panel provides continuous operating information about the 140 NWM 100 00 module and its connection to the network.

LED	Color	Description
Active	Green	Indicates the backplane is operating.
Ready	Green	Indicates module is healthy.
Fault	Red	Indicates when the NWM is inoperative.
Run	Green	Flashes to indicate diagnostic code, as described in "Run LED Status" (below).
Coll	Red	Flashes when Ethernet collisions occur.
Link	Green	On when Ethernet link is active.
TxAct	Green	Flashes to indicate Ethernet transmission.
RxAct	Green	Flashes to indicate Ethernet reception.
10MB	Green	On when the module is connected to a 10 Megabit network.
100MB	Green	On when the module is connected to a 100 Megabit network.
Fduplex	Green	On when Ethernet is operating in the full duplex mode.
Kernel	Amber	On when in Kernel Mode. Flashing while in download mode.
Appl	Green	On when crash log entry exists.



Run LED Status

Indicator State	Status
On (steady)	Normal operation: The NWM module is ready for network communication.
Number of flashes in sequence	
one	Not used
two	Not used
three	No Link: the network cable is not connected or is broken.
four	Duplicate IP address: The module will stay offline.
five	No IP address: The module is attempting to obtain an IP address from a BOOTP server.
six	Using default IP address
seven	No valid executive NWM present
eight	Invalid IP configuration (Likely cause: Default gateway is not on the same subnet mask.)
nine	Flash file system is corrupted.

11.2 140 NWM 100 00 Installation

Installing the Module

Overview

The following information describes how to install the 140 NWM 100 00 module.

Before You Begin

Locate the backplane where you will mount the140 NWM 100 00 module. Ensure that an open slot is available to mount the module.

NOTE: The 140 NWM 100 00 module can be installed only in a local backplane.

NOTE: Power requirements

- When installing the 140 NWM 100 00, ensure that it does not exceed the Quantum backplane requirements. See the specifications section.

Backplane Slot Placement

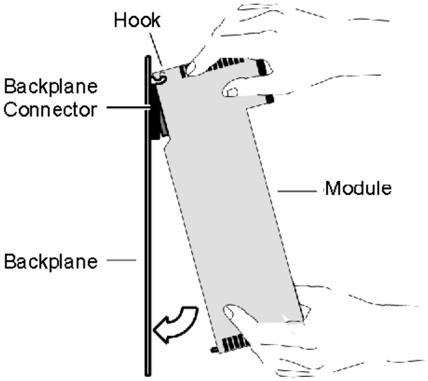
The modules may be placed in any slot on the backplane. They do not have to be placed next to each other.

Tools Required

You will need one medium-size, Phillips-head screw driver.

Mounting the Module in the Backplane

Perform the following steps to mount the 140 NWM 100 00 module on a Quantum backplane.

Step	Action
1	Holding the module at an angle, mount it on the two hooks located near the top of the backplane. The following figure shows the correct way to hold the module. 
2	Swing the module down so its connector engages the backplane connector.
3	Using a Phillips-head screw driver, tighten the screw at the bottom of the module between 2 and 4 in-lbs or between .22 and .45 Newton meters of torque.

11.3 140 NWM 100 00 Ethernet and Communications

Overview

This section contains information pertaining to the Ethernet and communications aspects of the 140 NWM 100 00 module.

What's in this Section?

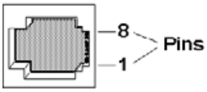
This section contains the following topics:

10/100BASE-T and 100BASE-FX Ethernet

10/100BASE-T Twisted Pair Connector

The 140 NWM 100 00 modules' 10/100BASE-T connector (shown below) is a standard RJ-45 twisted pair socket.

The following figure shows the 10/100BASE-T connector.



Schneider Electric recommends that you use Category 5 STP cabling, which is rated to 100 Mbps, with an RJ-45 connector.

The eight pins are arranged vertically and numbered in order from the bottom to the top. The RJ-45 pinout used by this module is:

- Receive Data (+) 3
- Receive Data (-) 6
- Transmit Data (+) 1
- Transmit Data (-) 2

100 BASE-FX

The140 NWM 100 00 modules' 100BASE-FX connector consists of a MT-RJ socket, or uses a mating fiber optic cable connector.

For the140 NWM 100 00, you may need an MT-RJ to SC (Duplex) multimode fiber optic cable assembly 62.5/125mm. Schneider Electric recommends cable number 490NOC00005 to connect to fiber hubs/switches.

NOTE: The 140 NWM 100 00 is a one channel device. It is capable of communicating over either a 10/100BASE-T or a 100BASE-FX Ethernet network at any given time, but not over both at the same time.

HTTP Server

Overview

The following information describes the services provided by the HTTP Server.

HTTP Server

The NWM's HyperText Transport Protocol (HTTP) Server is available as soon as the module receives an IP address. It can be used with version 4.0 or greater of either the Internet Explorer or Netscape browser.

The NWM HyperText Transport Protocol (HTTP) Server allows you to view the following information.

- Module's Ethernet statistics
- Controller and I/O information

The HTTP Server's HTML pages allow you to configure the module's SNMP Agent.

The HTTP Server is protected with a default name and password. The default name and password are both USER, and both are case sensitive.

For the NWM module, the user name and password can be changed via Web Designer for FactoryCast HMI.

The NWM supports a maximum of 32 HTTP simultaneous connections.

NOTE: Browsers may open multiple connections so 32 HTTP connections does not indicate 32 simultaneous users.

Modbus Messaging

Introduction - Client

The 140 NWM 100 00 Quantum Ethernet TCP/IP module provides the user with the capability of transferring data to and from nodes on a TCP/IP network through the use of a communication instruction. All PLCs that support networking communication capabilities over Ethernet can use the MSTR Ladder Logic instruction to read or write controller information or can also use IEC communication blocks.

Introduction - Server

The 140 NWM 100 00 Quantum Ethernet TCP/IP module provides the user with the ability to access data from the controller using the standard Modbus/TCP protocol. Any device: PC, HMI package, another PLC, or any Modbus/TCP compliant device can access data from the PLC. The Modbus/TCP Server also allows programming panels to log into the controller over Ethernet.

Limitations

The 140 NWM 100 00 supports up to 64 simultaneous Modbus/TCP Server connections. The 140 NWM 100 00 allows only one Programming Panel to be logged in at a time to guarantee consistency of changes to the controller configuration.

The following Modbus/TCP commands are supported by the 140 NWM 100 00.

- Read Data
- Write Data
- Read/Write Data
- Get Remote Statistics
- Clear Remote Statistics
- Modbus 125 Commands (used by programming panels to download a new Exec to the NWM)

Performance

The following table shows the performance characteristics of the NWM’s Modbus/TCP Server.

Parameter	Value
Typical Response Time (ms)	0.6
Number of Modbus connections (Client and Server) There can be a maximum of 64 connections of which there can be a maximum of 16 client connections.	64
Number of simultaneous login channels	1

NOTE: 140 NWM 100 00 Modbus/TCP performance measurements are made with Quantum 140 CPU 534 14 PLC.

11.4 140 NWM 100 00 Specifications

Specifications

Overview

The following information describes the main specifications for the Quantum 140 NWM 100 00 Ethernet Module.

Specification Table

The main specifications for the Quantum 140 NWM 100 00 Ethernet Module are described in the following table.

Item	Specification
Communication Ports	One auto-sensing 10/100Base-T shielded twisted pair (RJ-45 connector) port and one 100Base-FX (MT-RJ connector) port. Both ports transmit and receive Modbus commands encapsulated in TCP/IP protocol
Bus Current Required	900 mA
Power Dissipation	4.5 W
Fuse	None
Programming Software	
Type and version	Concept, Ver. 2.6
Firmware	
CPU Type and Executive version	Quantum CPU113/213 Ver 2.4 Quantum CPU113/213 Ver 2.31 (stripped) Quantum 424 Firmware Ver 2.19 Quantum 486/586 Firmware Ver 1.15 Quantum 486A/586A Firmware Ver 1.2
NWM Upgradeable	Field Upgradeable via FTP or Programming Panel.
Operating Conditions	
Temperature	0 to +60° C
Humidity	0 to 95% Rh non condensing @ 60° C
Altitude	6500 ft (2000 m)
Vibration	10-57 Hz @ 0.0075 mm d.a
	57-150 Hz @ 1 g
Processor	
Processor Speed	80 MHz
Expanded RAM size	32 MB

Item	Specification
Flash	16 MB)
Storage Conditions	
Temperature	-40 to +85°C
Humidity	0 to 95% Rh non condensing @ 60°C
Free Fall	1 m unpackaged
Shock	3 shocks / axis, 15 g, 11 ms
International Standard	ANSI/IEEE Std
US Standard	UL508, CEI 1131-2 for Immunity and 50082-1 for emissions.
Canadian Standard	CSA C22.2 / 142
European Standard	Conformity to FCC-B for Emission (50082-1 CE Mark IEC 61131-2, EN66631-2
Agency Approvals	UL 508, 94 Specifications CSA 22.2-142 IEC 1131-2 CE Factory Mutual Class 1 Division 2

Configuration of HMI modules using Unity Pro, Concept and PL7

III

Subject of this Part

This part describes configuration of Quantum and Premium modules using Unity Pro, Concept and PL7 software workshops.

What's in this Part?

This part contains the following chapters:

Configuration of the TSX WMY 100 and 140 NWM 100 00 modules using Unity Pro

12

Subject of this Chapter

This chapter describes installation of the Premium **TSX WMY 100** and Quantum **140 NWM 100 00** HMI modules in the Unity Pro software workshop.

NOTE: The IO scanning, address server, and Global Data services can be configured using Unity Pro. But project generation will not be possible because these services are not available on the HMI TSX WMY 100 and 140 NWM 100 00 modules.

NOTE: Configuration of the HMI TSX WMY 100 and 140 NWM 100 00 modules is identical in Unity Pro.

What’s in this Chapter?

This chapter contains the following sections:

12.1 General

Subject of this Section

This section introduces ETHERNET communication from a Premium TSX WMY 100 module or a Quantum 140 NWM 100 00 module.

What's in this Section?

This section contains the following topics:

General

At a Glance

The communication channel of the TSX WMY 100 or the 140 NWM 100 00 module offers connection to a TCP/IP network supporting the UNI-TE and Modbus messaging service on a TCP/IP profile.

The module also offers the following services:

- Thanks to their SNMP agent functionality, the modules can be supervised by one or two SNMP managers.
- They are able to perform the role of DHCP and BOOTP client.
- They integrate an embedded HTTP server.
- The Replace inoperative device service allows the device configuration to be saved in the module. In case of breakdown, another installed blank module can be restarted with the configuration parameters of the previous one.

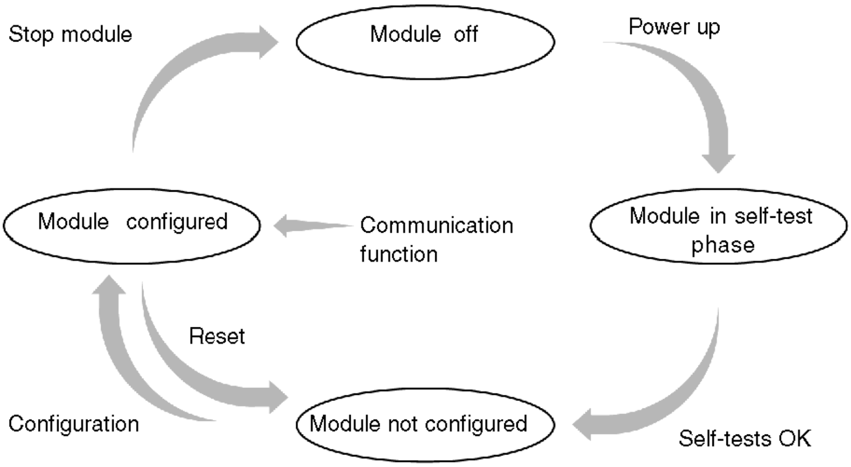
The module offers the option of creating user Web and TCP Open pages.

Operating modes of the HMI modules

At a Glance

The following diagram describes the operating modes of the Premium and Quantum HMI modules.

General Diagram



Operation

- After power-up, the module carries out self-testing. During this phase, the STS LED is lit. After the self tests, the RUN LED lights up.
- The module does not operate with a default configuration. It must be sent this configuration by the application of the local PLC.
  The configuration values are given in the list of language objects %KW and in particular provide the IP address and the X-Way address of the module.
  The configuration is retransmitted upon every PLC restart (warm or cold).
- When the configuration is received, the module resets the current communication to zero before configuring itself (terminates current exchanges, shuts down TCP connections).
  The module is now operational. The RUN and STS LEDs are lit up.

Special Cases

If a module is not configured in the Unity Pro application (RUN LEDs go out and ERR LEDs flash), it takes the IP address constructed from its MAC address:

085.016.xxx.yyy where xxx and yyy are the last two numbers of the MAC address.

Example

The module’s MAC address is (in hexadecimal): 00 80 F4 01 12 20

In this case the default IP address is (in decimal): 085.016.018.032

The module also transmits BOOTP/DHCP requests to obtain another IP address. These requests are sent until a BOOTP/DHCP server responds or until configuration by Unity Pro.

If the module detects a duplicated IP address, the STS LED flashes 4 times during a minute (as does that of the module having the same IP address).

12.2 Configuration of TSX WMY 100 & 140 NWM 100 modules running Unity Pro

Subject of this Section

This section describes configuration of the TSX WMY 100 and 140 NWM 100 00 modules running Unity Pro.

What's in this Section?

This section contains the following topics:

Module configuration screen

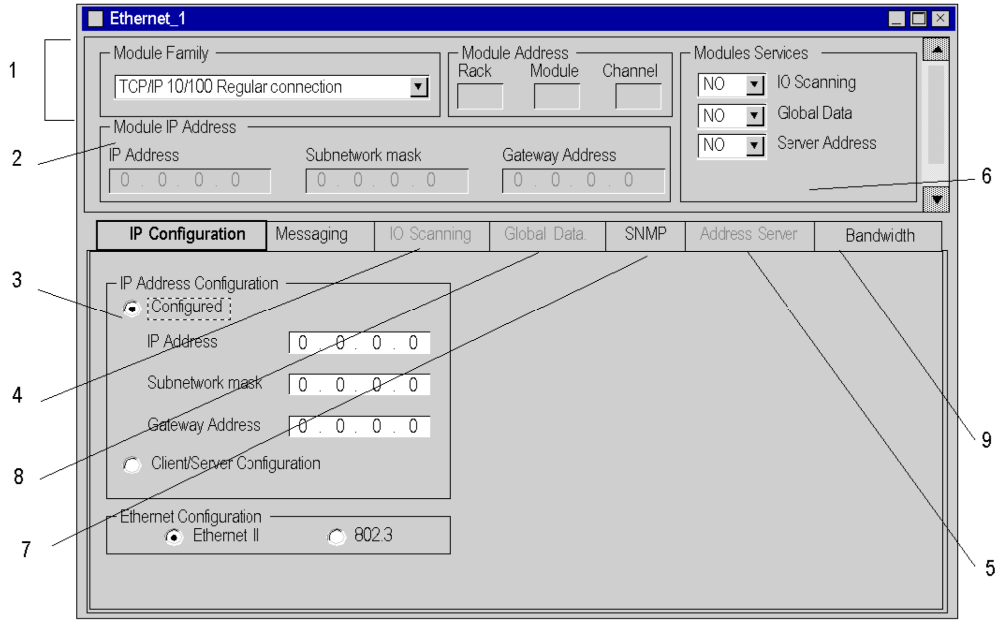
At a Glance

This screen, separated into two zones, is used to declare the communication channel and to configure the necessary parameters for an Ethernet link.

NOTE: Configuration of the HMI TSX WMY 100 and 140 NWM 100 00 modules is identical in Unity Pro.

Illustration

The screen dedicated to Ethernet communication is displayed as follows:



Ethernet_1

Module Family: TCP/IP 10/100 Regular connection

Module Address: Rack / Module / Channel

Modules Services:
- NO IO Scanning
- NO Global Data
- NO Server Address

Module IP Address:
- IP Address 0 . 0 . 0 . 0
- Subnetwork mask 0 . 0 . 0 . 0
- Gateway Address 0 . 0 . 0 . 0

IP Configuration | Messaging | IO Scanning | Global Data | SNMP | Address Server | Bandwidth

IP Address Configuration
- ● Configured
  - IP Address 0 . 0 . 0 . 0
  - Subnetwork mask 0 . 0 . 0 . 0
  - Gateway Address 0 . 0 . 0 . 0
- ○ Client/Server Configuration

Ethernet Configuration
- ● Ethernet II
- ○ 802.3

Elements and Functions

This table describes the various zones that make up the configuration screen:

Zone	Number	Function
common	1	common part of the communication configuration screens.
specific	2	allows the display of the module address.
	3	allows configuration of the TCP/IP services. See *Configuration Parameters Linked to TCP/IP Services, page 467*.
	4	IO scanning service, see note below.
	5	Address server service, see note below.
	6	allows selection of the services used by the module.
	7	allows configuration of the SNMP service. See *Configuration Parameters Linked to the SNMP Service, page 476*.
	8	Global Data service, see note below.
	9	allows confirmation that the services configured are compatible with the Ethernet channel processing capacity.

NOTE: The IO scanning, address server and Global Data services can be configured using Unity Pro. But project generation will not be possible because these services are not available on the HMI TSX WMY 100 and 140 NWM 100 00 modules.

Type of Communication According to Connection Configuration

At a Glance

Depending on the connection configuration of the HMI modules, you can carry out messaging:

- in mono-connection
- in multi-connection.

According to the type of protocol and access control management, multi-connection mode requires a specific configuration of the correspondence table.

NOTE: In the following, only examples of multi-connection will be dealt with. Furthermore, the emphasis is placed on communication between the module and a remote PC device containing several applications.

Configuration Rules

Several connections can be configured with the same IP address. They must be defined with the same protocol, the same access rights, and the same connection mode.

In the case of a Modbus connection, only two connections can be configured with the same IP address. Addresses associated with these connections are:

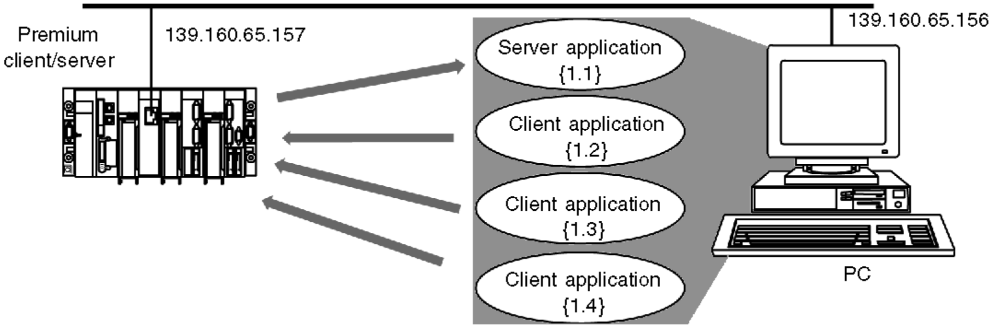
- an X-Way address (for example: {2.103}) for the Modbus client connection,
- an X-Way address equal to {x.x} for Modbus server connections.

Multi-connection in UNI-TE Protocol with Access Management

In this example, the PC contains four applications with a unique X-Way address and one single IP address:

- The module opens a connection and communicates with the server application with the address {1.1}.
- Each of the other client applications with the address {1.2}, {1.3}, {1.4} is authorized to open a connection and communicate with the module.

Exchanges are carried out according to the UNI-TE protocol and the module's access control service is enabled.



In order to implement this example, you must:

- activate connection management by checking the module's **Access control** box,
- configure communication from the module to the PC's server application in the table
  - **X-Way address**: 1.1
  - **IP address**: 139.160.65.156
  - **Protocol**: UNI-TE
  - **Access**: authorized (cell checked)
  - **Mode**: Multi
- configure communication of each of the PC's client applications to the server module in the table:
  - **X-Way address**: 1.2 and 1.3 and 1.4
  - **IP address**: 139.160.65.156
  - **Protocol**: UNI-TE
  - **Access**: authorized (cell checked)
  - **Mode**: Multi

Multi-connection in UNI-TE protocol without access management

This example is the same as the previous one, except that access control is not configured.

In this case, if the application {1.1} is the only server application, the module's correspondence table only contains the following line:

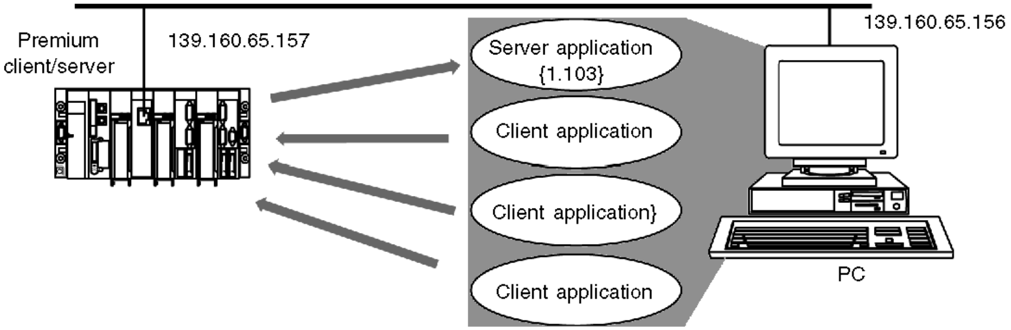
- **X-Way address**: 1.1
- **IP address**: 139.160.65.156
- **Protocol**: UNI-TE
- **Access**: the cell is grayed out
- **Mode**: Multi

Multi-connection in Modbus Protocol with Access Management

In this example, the PC contains four applications. The server application has an imaginary X-Way address and the other client applications do not need an X-WAY address:

- The module opens a connection and communicates with the server application with the imaginary address {1.103}.
- Each of the client applications has no address but is authorized to open a connection and communicates with the module.

Exchanges are carried out according to the Modbus protocol and the module's access control service is enabled.



In order to implement this example, you must:

- activate connection management by checking the module's **Access control** box,
- configure communication from the module to the PC's server application in the table
  - **X-Way address**: 1.103 (address between 100 and 163)
  - **IP address**: 139.160.65.156
  - **Protocol**: Modbus
  - **Access**: the cell is grayed out
  - **Mode**: Multi

- configure communication of all the PC’s client applications to the server module in the table:
  - **X-Way address**: x.x (this address allows you to specify that other applications exist for the same IP address)
  - **IP address**: 139.160.65.156
  - **Protocol**: Modbus
  - **Access**: authorized (allows you to specify that other applications are authorized to connect and communicate)
  - **Mode**: Multi

NOTE: If you want to prevent connection by client applications you must configure

- **X-Way address**: x.x (this address allows you to specify that other applications exist for the same IP address)
- **IP address**: 139.160.65.156
- **Protocol**: Modbus
- **Access**: not authorized (cell unchecked)
- **Mode**: Multi

Multi-connection in Modbus protocol without access management

This example is the same as the previous one, except that access control is not configured.

In this case, if the application {1.103} is the only server application, the module’s correspondence table only contains the following line:

- **X-Way address**: 1.103
- **IP address**: 139.160.65.156
- **Protocol**: Modbus
- **Access**: the cell is grayed out
- **Mode**: Multi

Configuration of TCP/IP Messaging

Presentation

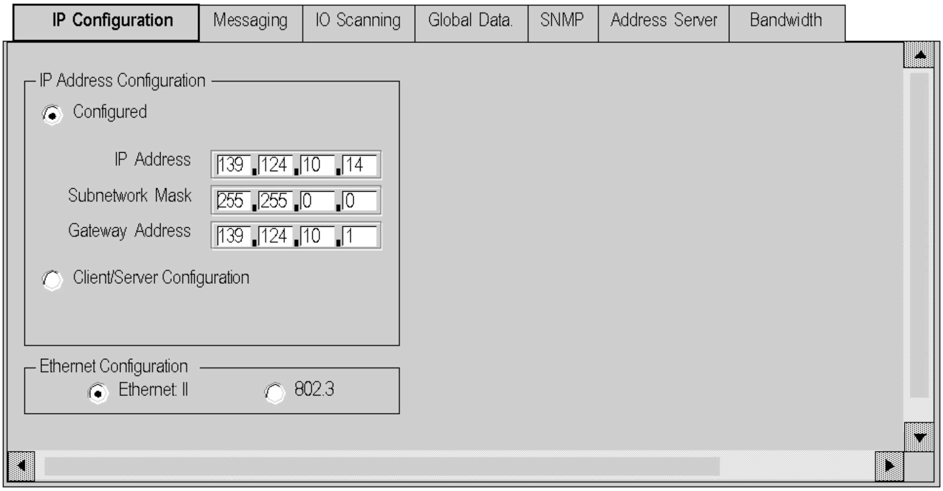
In order to use the TSX WMY 100 and 140 NWM 100 00 modules to communicate on Ethernet, it is necessary to adjust the messaging configuration parameters.

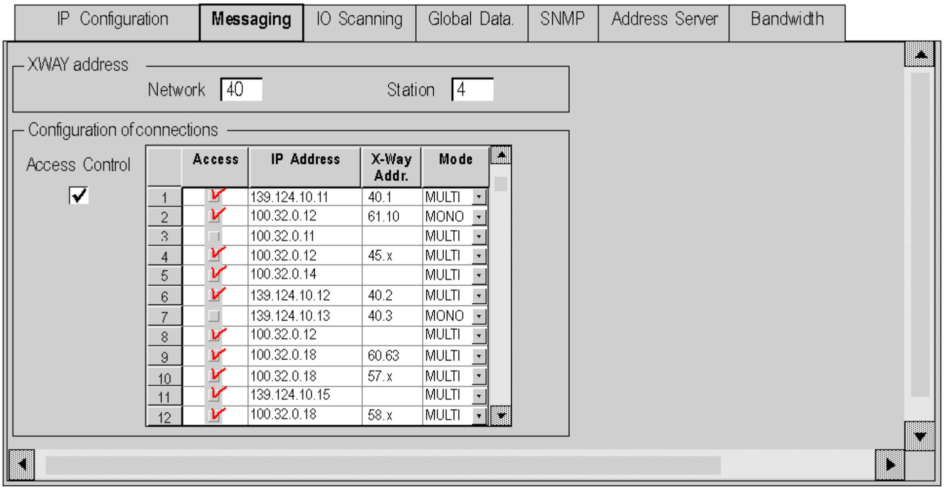
Messaging is configured in the two windows accessible by the tabs: **IP Configuration** and **Messaging**.

Having two devices with the same IP address can cause unpredictable operation of your network.

⚠ WARNING
UNINTENTIONAL OPERATION — DUPLICATE IP ADDRESS ● Use a method to assign a unique IP address to each device on the network. ● Obtain your IP addresses from your system administrator to avoid the possibility of duplicate addresses. **Failure to follow these instructions can result in death, serious injury, or equipment damage.**

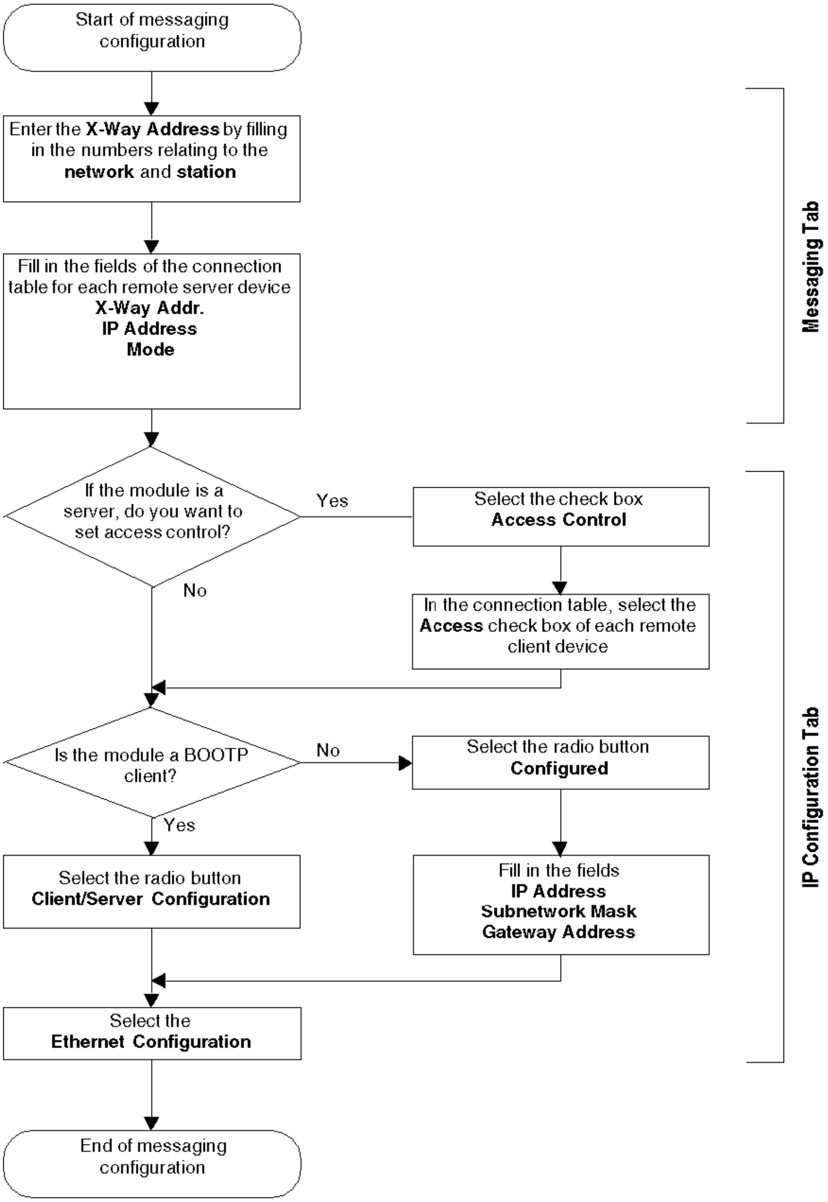
How to Access the Messaging

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Access the module configuration screen.</td></tr>
<tr><td>2</td><td>Select the <b>IP Configuration</b> tab.
<b>Result</b>


IP Address Configuration
● Configured
IP Address: 139.124.10.14
Subnetwork Mask: 255.255.0.0
Gateway Address: 139.124.10.1
○ Client/Server Configuration
Ethernet Configuration
● Ethernet II ○ 802.3
</td></tr>
<tr><td>3</td><td>Select the <b>Messaging</b> tab.
<b>Result</b>


XWAY address
Network: 40 Station: 4
Configuration of connections
Access Control ☑
<table>
<tr><th></th><th>Access</th><th>IP Address</th><th>X-Way Addr.</th><th>Mode</th></tr>
<tr><td>1</td><td>✓</td><td>139.124.10.11</td><td>40.1</td><td>MULTI</td></tr>
<tr><td>2</td><td>✓</td><td>100.32.0.12</td><td>61.10</td><td>MONO</td></tr>
<tr><td>3</td><td>□</td><td>100.32.0.11</td><td></td><td>MULTI</td></tr>
<tr><td>4</td><td>✓</td><td>100.32.0.12</td><td>45.x</td><td>MULTI</td></tr>
<tr><td>5</td><td>✓</td><td>100.32.0.14</td><td></td><td>MULTI</td></tr>
<tr><td>6</td><td>✓</td><td>139.124.10.12</td><td>40.2</td><td>MULTI</td></tr>
<tr><td>7</td><td>□</td><td>139.124.10.13</td><td>40.3</td><td>MONO</td></tr>
<tr><td>8</td><td>✓</td><td>100.32.0.12</td><td></td><td>MULTI</td></tr>
<tr><td>9</td><td>✓</td><td>100.32.0.18</td><td>60.63</td><td>MULTI</td></tr>
<tr><td>10</td><td>✓</td><td>100.32.0.18</td><td>57.x</td><td>MULTI</td></tr>
<tr><td>11</td><td>✓</td><td>139.124.10.15</td><td></td><td>MULTI</td></tr>
<tr><td>12</td><td>✓</td><td>100.32.0.18</td><td>58.x</td><td>MULTI</td></tr>
</table>
</td></tr>
</table>

How to Configure the Messaging


Start of messaging configuration
Enter the **X-Way Address** by filling in the numbers relating to the **network** and **station**
Fill in the fields of the connection table for each remote server device **X-Way Addr.** **IP Address** **Mode**
Messaging Tab
If the module is a server, do you want to set access control?
Yes
Select the check box **Access Control**
In the connection table, select the **Access** check box of each remote client device
No
Is the module a BOOTP client?
No
Select the radio button **Configured**
Yes
Select the radio button **Client/Server Configuration**
Fill in the fields **IP Address** **Subnetwork Mask** **Gateway Address**
IP Configuration Tab
Select the **Ethernet Configuration**
End of messaging configuration

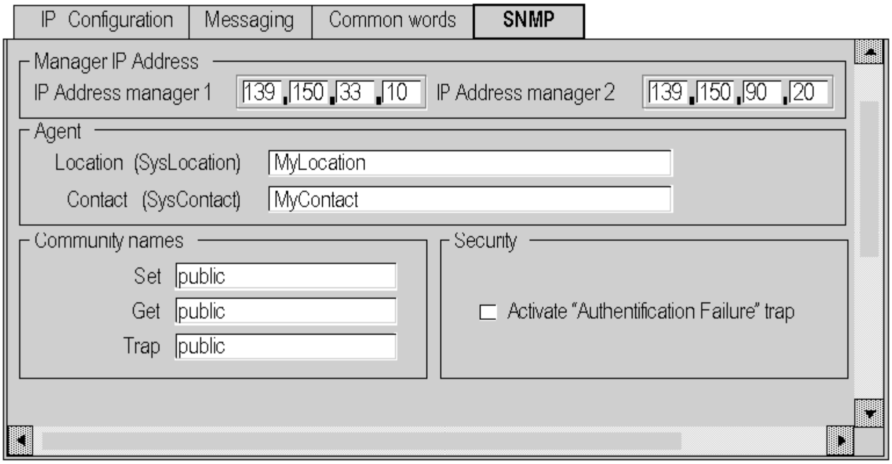
Configuration of the SNMP Service

At a Glance

In order to use the TSX WMY 100 and 140 NWM 100 00 module as an SNMP agent, it is necessary to adjust the configuration parameters of the SNMP service.

How to Access the SNMP Service

The procedure for accessing the configuration parameters of the SNMP service is as follows.

Step	Action
1	Access the module configuration screen.
2	Select the **SNMP** tab. **Result**  Manager IP Address IP Address manager 1 139.150.33.10 IP Address manager 2 139.150.90.20 Agent Location (SysLocation) MyLocation Contact (SysContact) MyContact Community names Set public Get public Trap public Security □ Activate “Authentification Failure” trap

How to Configure SNMP

The following procedure gives the configuration principle for the SNMP service.

Step	Action
1	Enter the SNMP manager addresses: ● **Manager 1 IP addresses** ● **Manager 2 IP addresses**
2	Fill in the fields: ● **Location (SysLocation)** ● **Contact (SysLocation)**. Or alternatively check the **SNMP Manager** box to indicate that the information will be completed by the SNMP Manager.
3	If you want to define access rights, fill in the community names: ● **Set** ● **Get** ● **Trap**
4	If you want to activate transmission of an event to the module, check the **Activate "Authentication Failure" trap** box.

Configuration of the Bridge Function

At a Glance

The TSX WMY 100 and 140 NWM 100 00 modules may be used as an X-WAY bridge station. This guarantees transparent communication between various networks.

12.3 Debugging HMI modules in Unity Pro

Subject of this Section

This section provides information about debugging Premium and Quantum HMI modules in the Unity Pro software workshop.

What's in this Section?

This section contains the following topics:

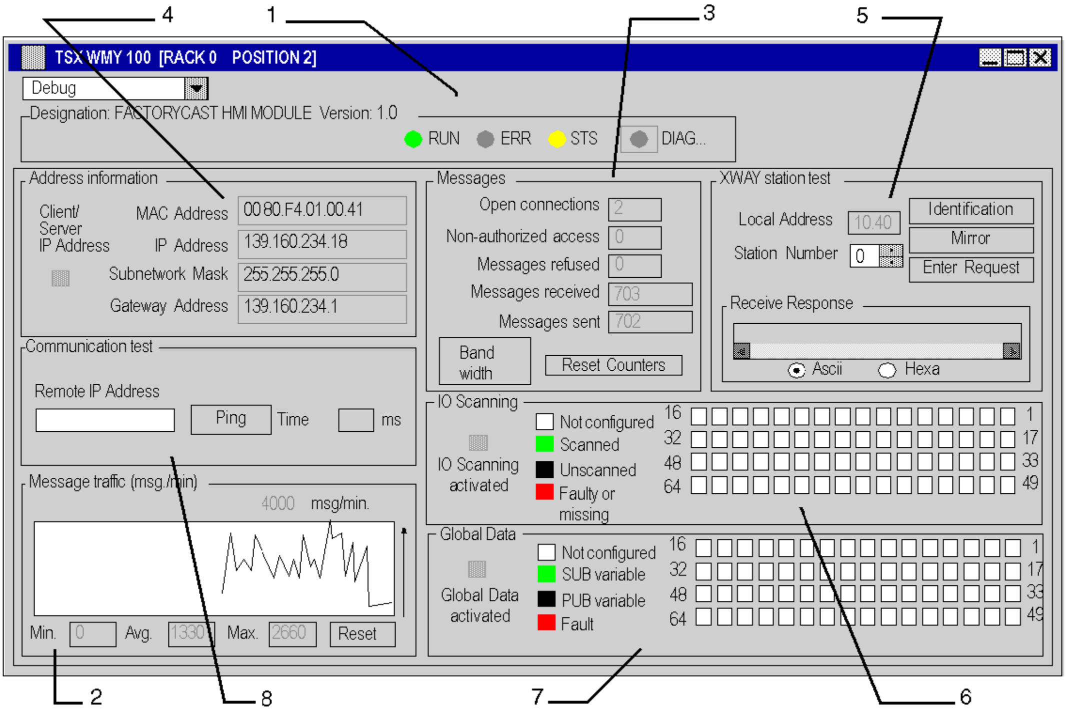
Module debug screen

At a Glance

This screen, separated into eight zones, allows debugging via an Ethernet link.

Illustration

The screen dedicated to Ethernet communication is displayed as follows:



TSX WMY 100 [RACK 0 POSITION 2]

Debug

Designation: FACTORYCAST HMI MODULE Version: 1.0

● RUN ● ERR ● STS ● DIAG...

Address information

Client/ Server IP Address

MAC Address: 00.80.F4.01.00.41

IP Address: 139.160.234.18

Subnetwork Mask: 255.255.255.0

Gateway Address: 139.160.234.1

Messages

Open connections: 2

Non-authorized access: 0

Messages refused: 0

Messages received: 703

Messages sent: 702

Band width | Reset Counters

XWAY station test

Local Address: 10.40

Station Number: 0

Identification | Mirror | Enter Request

Receive Response

● Ascii ○ Hexa

Communication test

Remote IP Address

Ping | Time | ms

Message traffic (msg./min)

4000 msg/min.

Min. 0 Avg. 1330 Max. 2660 Reset

IO Scanning

IO Scanning activated

□ Not configured

🟩 Scanned

■ Unscanned

🟥 Faulty or missing

16 □□□□□□□□□□□□□□□□ 1

32 □□□□□□□□□□□□□□□□ 17

48 □□□□□□□□□□□□□□□□ 33

64 □□□□□□□□□□□□□□□□ 49

Global Data

Global Data activated

□ Not configured

🟩 SUB variable

■ PUB variable

🟥 Fault

16 □□□□□□□□□□□□□□□□ 1

32 □□□□□□□□□□□□□□□□ 17

48 □□□□□□□□□□□□□□□□ 33

64 □□□□□□□□□□□□□□□□ 49

Elements and Functions

This table describes the different zones that make up the debug screen:

Number	Zone	Function
1	common	common part of the communication debug screens
2	Message traffic	allows the graphical display of the number of messages processed by the module:
3	Messages	allows the display of the number of connections and unacknowledged or refused messages. The counter values can be reinitialized using the **Reset Counters** button. A **Bandwidth** button is used to access bandwidth diagnostics.
4	TCP/IP services	allows: ● display of TCP/IP services configuration, ● communication testing of the TCP/IP profile.
5	X-WAY test station	allows UNI-TE communication testing on the TCP/IP profile.
6	IO Scanning	unavailable.
7	Global Data.	unavailable.
8	Communication test	is used to perform a communication test

General Debugging Parameters

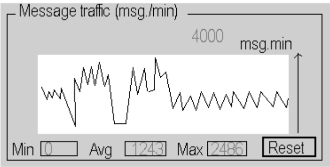
At a Glance

The general debugging parameters are grouped together in two windows:

- the **Message traffic** window
- the **Messages** window.

Message traffic

The window is displayed as below:


Message traffic (msg./min)

4000 msg.min

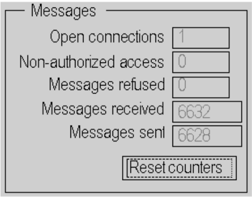
Min 0 Avg 1243 Max 2486

It shows the number of messages processed by the module per minute (transmission and reception) in graphic form. To provide a better display, the scale is automatically adapted according to change in flow.

The **Reset** button sets the three counters **Min.**, **Avg.** and **Max.** to zero.

Messages

The window is displayed as below:


Messages

Open connections 1

Non-authorized access 0

Messages refused 0

Messages received 6632

Messages sent 6628

This window is related to TCP/IP messaging:

- number of TCP connections open
- number of non-authorized accesses
- number of messages refused, received and sent

You can reset these counters to zero by clicking on the **Reset counter** button.

Debugging parameters for TCP/IP services

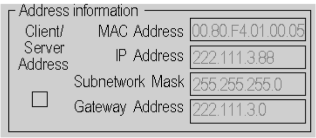
At a Glance

The debugging parameters for the TCP/IP services are grouped together in two windows:

- the **Address information** window,
- the **Communication test** window.

Address information

The window is displayed as below:



Client/ Server Address ☐

MAC Address 00.80.F4.01.00.05

IP Address 222.111.3.88

Subnetwork Mask 255.255.255.0

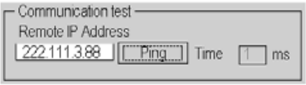
Gateway Address 222.111.3.0

It specifies the configuration data of the IP address:

- **MAC address**: unique fixed address for a module
- **IP Address**
- **Subnetwork mask**
- **Gateway Address**: address of the gateway

Communication test

Once the **Ping** button has been pressed, the window is displayed as follows:



Remote IP Address 222.111.3.88

Time 1 ms

This window is used to test IP communication towards another station.

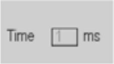
How to Test TCP/IP communication with the Ping request

At a Glance

This page indicates the procedure for testing TCP/IP communication from a **Ping** request.

Procedure

The following procedure is used to send the Ping request and therefore to test the correct routing of information between two devices.

Step	Action
1	From the main screen, click on the **Ping** button.
2	Enter the address of the station to be queried using the **Remote IP address** field.
3	Press the **Ping** button. **Result** The response appears in the **Time** field:  Time 1 ms The returned time corresponds to the time it takes the message to be sent and return in ms.

Response type

The following table groups together the various types of possible responses to the Ping request.

If the response is	then
positive	The windows records the time it takes the message to be sent and return in ms. a window appears with the message **Exchange successful**.
negative	a window with the message **Exchange incorrect** specifies that the remote device was not reached in the network architecture.

Available requests for testing a communication channel

Presentation

This page describes the various options for testing a communication channel from the debug screen.

Test conditions

When a request is sent to a station that is not connected, an message appears.

The test applies to stations belonging to the same local area network.

Available requests

The **X-Way station test** window allows the following requests:

- **Identification**: causes the Identification request to be sent to the designated remote station,
- **Mirror**: is used to send a Mirror request to the designated station. If this function is chosen, a screen appears allowing you to select the length of the string of characters to be sent (maximum of 80 characters). The PLC then sends this string of characters (ABCD, etc.) to the destination device. This device automatically returns the string of characters received to the sender,
- **Enter Request** : is used to send a UNI-TE request, other than those sent by the command buttons, to the designated station. If this function is chosen, a screen appears allowing you to select the parameters specific to the request (the request code must be coded in hexadecimal).

NOTE: The first two requests are sent to the UNITE server of the TSX WMY 100 module. The third is sent to the server of the processor.

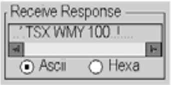
How to Test a Channel with the Identification and Mirror requests

At a Glance

This page indicates the procedure for testing a communication channel using the Identification and Mirror requests.

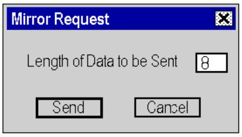
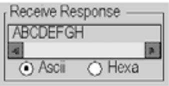
How to Identify a Station

The following procedure is used to identify a designated station.

Step	Actions
1	Select the address of the station to be queried using the **Station number** field.
2	Press the **Identification** button. **Result** The response appears in the **Receive Response** window:  Receive Response ..’TSX WMY 100.!...... ● Ascii ○ Hexa

How to Send the Mirror Request

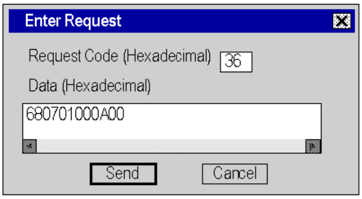
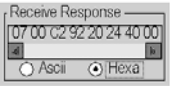
The following procedure is used to send the Mirror request and therefore to test the correct routing of information between two devices.

Step	Action
1	Select the address of the station to be queried using the **Station** field.
2	Press the **Mirror** button. **Result** The following window appears:  Mirror Request Length of Data to be Sent 8
3	Enter the length of the data to be sent (maximum 80 characters).
4	Press the **Send** button. **Result** The response appears in the **Receive Response** window:  Receive Response ABCDEFGH ● Ascii ○ Hexa The response contains: ● the character string ABCDEFGH corresponding to a length of data sent of 8.

How to test a channel with requests

How to Send a Request

The following procedure is used to send a request, other than those sent by the command buttons, to a designated station. In this example, the sent request enables 10 words to be read (from %MW1 to %MW10).

Step	Action
1	Select the address of the station to be queried using the **Station** field.
2	Press the **Enter Request** button. **Result** The following window appears:  Enter Request Request Code (Hexadecimal) 36 Data (Hexadecimal) 680701000A00 Send Cancel The data transmitted in this example is coded on 6 bytes.
3	Enter the function code (coded in hexadecimal on a byte), corresponding to the request that you want to send. In this example, the read request code is 16#36.
4	Enter the data to be transmitted by coding each data item in hexadecimal. The data is then entered one after the other with no spaces between data items. When the data is coded on a word, the most significant bytes and the least significant bytes are inverted. In this example, the data is as follows: ● 16#68: on a byte, define the segment (internal data), ● 16#07: on a byte, define the type of object (words), ● 16#0100: on a word, define the first word to be read, ● 16#0A00: on a word, define the number of words to be read.
5	Press the **Send** button. **Result** The response appears in the **Receive Response** window:  Receive Response 07 00 C2 92 20 24 40 00 ○ Ascii ● Hexa The response for the example contains the data on 21 bytes. ● 16#07: corresponds to the type of object (words), ● 16#00C2: corresponds to the value of the first word (the most significant bytes and the least significant bytes are inverted; its value is 16#C200), ● ...

12.4 Language objects associated with HMI modules

Subject of this Section

This section describes the language objects associated with the Premium and Quantum HMI modules, as well as the IODDTs.

What’s in this Section?

This section contains the following topics:

Creating an IODDT Type Data Instance

At a Glance

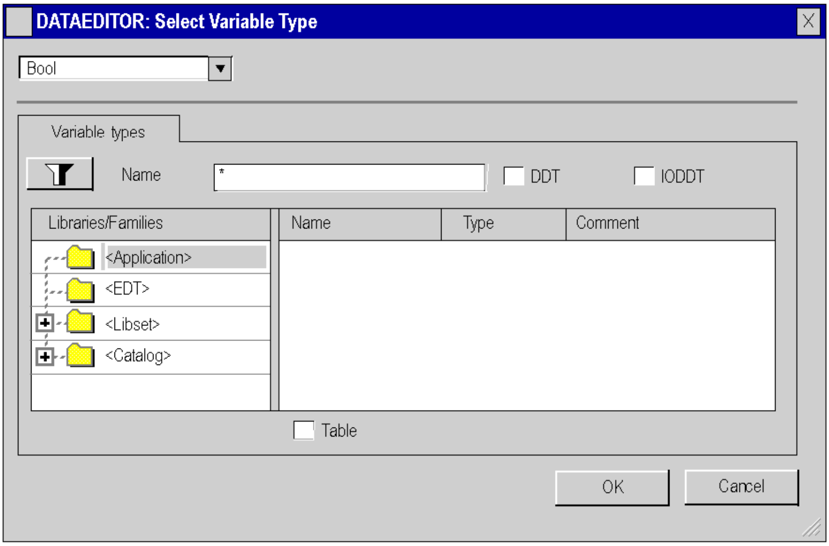
In the software installation principle the following must be carried out in order:

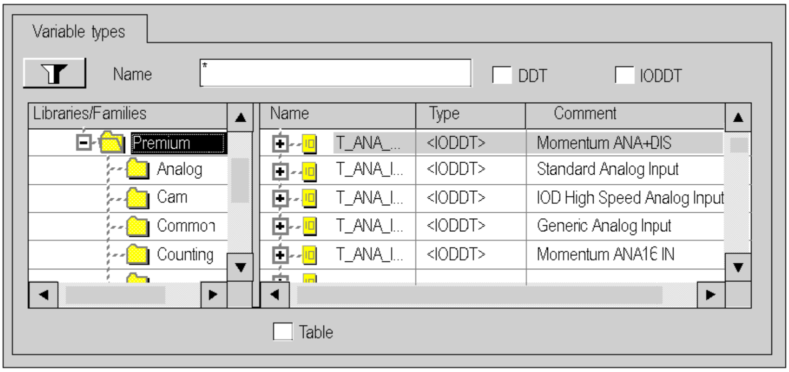
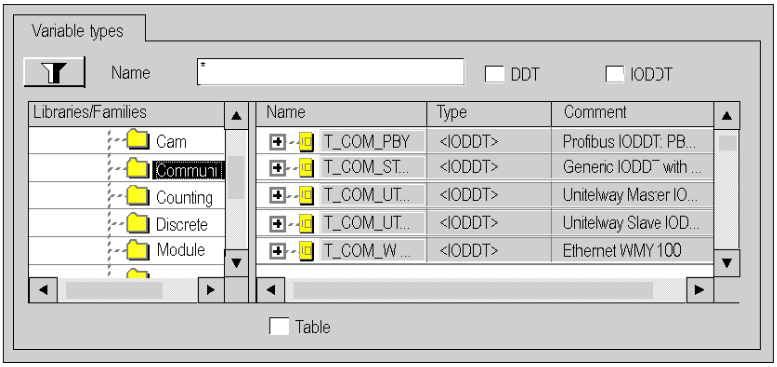
- create an IODDT type instance,
- associate the IODDT instance with the module,
- generate the project.

The following examples present the creation and association with a channel of an IODDT instance of the type **T_COM_WMY_100**.

Creating an IODDT Type Instance

To create an IODDT type instance, carry out the following actions:

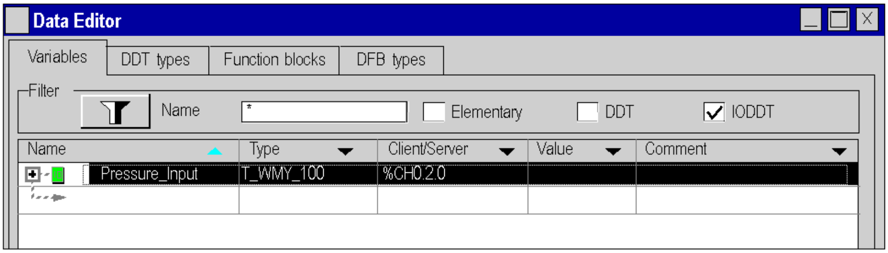
Step	Action
1	Using the variable editor, select the **Variables** tab.
2	Double-click on an empty **Name** cell, and enter the name of the instance.
3	Double-click on the corresponding **Type** cell, and press on the button ... . The following screen is displayed:  DATAEDITOR: Select Variable Type Bool Variable types Name * DDT IODDT Libraries/Families Name Type Comment <Application> <EDT> <Libset> <Catalog> Table OK Cancel

Step	Action
4	In the zone **Libraries/Families** deploy the **Premium** subdirectory of the **Catalog** directory. **Result**: the following screen is displayed.  Variable types Name * □ DDT □ IODDT Libraries/Families Premium Analog Cam Common Counting Name Type Comment T_ANA_... <IODDT> Momentum ANA+DIS T_ANA_I... <IODDT> Standard Analog Input T_ANA_I... <IODDT> IOD High Speed Analog Input T_ANA_I... <IODDT> Generic Analog Input T_ANA_I... <IODDT> Momentum ANA16 IN □ Table
5	Select the **Module** directory in the zone **Libraries/Families**. **Result**: the IODDT type appears.  Variable types Name * □ DDT □ IODDT Libraries/Families Cam Communi Counting Discrete Module Name Type Comment T_COM_PBY <IODDT> Profibus IODDT: PB... T_COM_ST... <IODDT> Generic IODDT with ... T_COM_UT... <IODDT> Unitelway Master IO... T_COM_UT... <IODDT> Unitelway Slave IOD... T_COM_W ... <IODDT> Ethernet WMY 100 □ Table
	In the **Type** column, select the **IODDT type** desired (in our example T_WMY_100). Validate with **OK**.

IODDT instance link with the application-specific module

To link an IODDT type instance to the channel of an application-specific module, carry out the following actions:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Using the variable editor, select the <b>Variables</b> tab.</td></tr>
<tr><td>2</td><td>Check the <b>IO Derived</b> box, so that only IODDT type instances are displayed.</td></tr>
<tr><td>3</td><td>Select the IODDT instance from which the link is to be created.</td></tr>
<tr><td>4</td><td>In the <b>Address</b> column, enter the address corresponding to the module or to its future slot (the module need not be configured at this stage).
Example:



Name: *   □ Elementary   □ DDT   ☑ IODDT

Name	Type	Client/Server	Value	Comment
Pressure_Input	T_WMY_100	%CH0.2.0		
</td></tr>
</table>

Rules to observe

The rules are as follows:

- an IODDT data type cannot be nested in a DDT data type,
- an IODDT data type cannot be nested in another IODDT data type,
- the public or private variable of a DFB cannot be of the type IODDT,
- the input/output parameter of a DFB can be of the type IODDT,
- it is not possible to access an IODDT type instance from a DFB section.

Implicit Exchange Language Objects Associated with the Application-Specific Function

Presentation

An integrated application-specific interface or the addition of a module automatically enhances the language objects application used to program this interface or module.

These objects correspond to the input/output images and software data of the module or integrated application-specific interface.

Reminders

The module inputs (%I and %IW) are updated in the PLC memory at the start of the task, the PLC being in RUN or STOP mode.

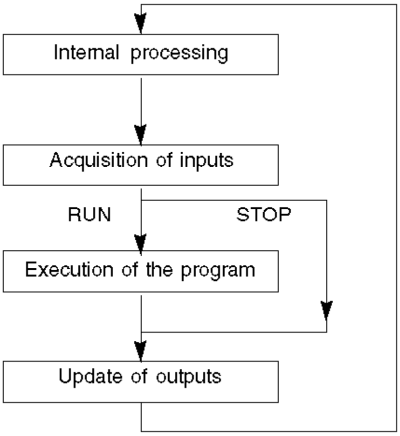
The outputs (%Q and %QW) are updated at the end of the task, only when the PLC is in RUN mode.

NOTE: When the task occurs in STOP mode, either of the following are possible, depending on the configuration selected:

- Outputs are set to fallback position (fallback mode),
- Outputs are maintained at their last value (maintain mode).

Cyclic Execution

The following diagram shows the operating cycle of a PLC task.



Explicit exchange language objects associated with the application-specific function

Presentation

Explicit exchanges are exchanges performed at the user program’s request, and using instructions:

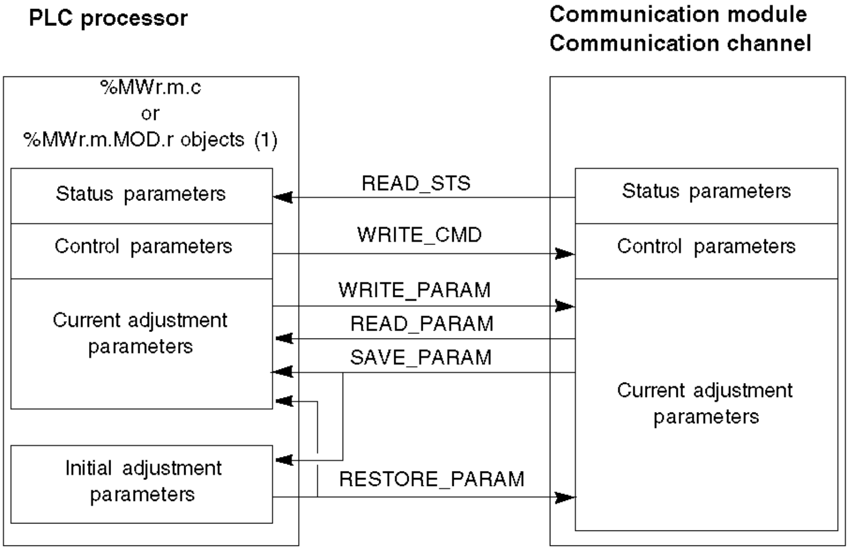
- READ_STS (read status words),
- WRITE_CMD (write command words),
- WRITE_PARAM (write adjustment parameters),
- READ_PARAM (read adjustment parameters),
- SAVE_PARAM (save adjustment parameters),
- RESTORE_PARAM (restore adjustment parameters).

These exchanges apply to a set of %MW objects of the same type (status, commands or parameters) that belong to a channel.

NOTE: These objects provide information about the module (e.g.: type of channel detected fault, etc.), can be used to command them (e.g.: switch command) and to define their operating modes (save and restore adjustment parameters in the process of application).

General principle for using explicit instructions

The diagram below shows the different types of explicit exchanges that can be made between the processor and module.



(1) Only with the instructions READ_STS and WRITE_CMD.

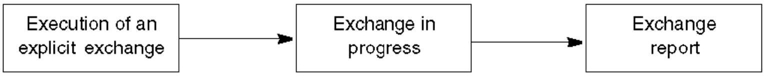
Managing exchanges

During an explicit exchange, it is necessary to check its performance in order that data is only taken into account when the exchange has been correctly executed.

To do this, two types of information is available:

- information concerning the exchange in progress,
- the exchange report.

The following diagram describes the management principle for an exchange



Language Objects Associated with Configuration

At a Glance

This page describes all the configuration language objects for Ethernet communication with the TSX WMY 100, which can be displayed by the application program.

Internal Constants

The following table describes the internal constants:

Object	Function	Meaning
%KWxy.i.0	Type	Byte 0 = 11 for Ethernet communication Byte 1: reserved
%KWxy.i.1	Physical layer	Byte 0: fixed at 16#01 (corresponds to GPX2) Byte 1: reserved
%KWxy.i.2	Reserved	-
%KWxy.i.3	Reserved	-
%KWxy.i.4	Reserved	-
%KWxy.i.5	X-Way network address	Byte 0: network number (0 by default) Byte 1: reserved
%KWxy.i.6	Type of Ethernet driver for TCP/IP	Byte 0: fixed at 16#01 (corresponds to RJ45) ● = 16#01: RJ45 Byte 1: ● = 16#00: Ethernet II (default value) ● = 16#01: 802.3
%KWxy.i.7	TCP/IP configuration: address type	Byte 0: reserved Byte 1: inherited address ● = 16#01: from the application-specific function ● = 16#03: from a server
%KWxy.i.8 and %KWxy.i.9	Local IP address	Example with the address 139.160.650.109 Byte 0 = 109 Byte 1 = 65 Byte 2 = 160 Byte 3 = 139
%KWxy.i.10 and %KWxy.i.11	IP address of gateway	Example with the address 139.160.65.1 Byte 0 = 1 Byte 1 = 65 Byte 2 = 160 Byte 3 = 139

Object	Function	Meaning
%KWxy.i.12 and %KWxy.i.13	Subnetwork mask	Example with the address 255.255.255.0 Byte 0 = 0 (least significant) Byte 1 = 255 Byte 2 = 255 Byte 3 = 255 (most significant)
%KWxy.i.14	Services used	Byte 0: ● x0 = 1: the IOScanner service is used ● x1 = 1: BOOTP server service is used ● x2 = 1: the Global Data service is used ● x3 = 1: the service to adjust bandwidth is used Byte 1: reserved
%KWxy.i.15	TCP services used	Byte 0: ● x0 = 1: Modbus communication on TCP/IP is used ● x1 = 1: access control is activated Byte 1: reserved

Details of Explicit Exchange Objects of the IODDT of type T_COM_WMY_100

At a Glance

This part introduces explicit exchange objects of the `T_COM_WMY_100` IODDT which apply to the TSX WMY 100 module (there are no implicit exchange objects in this IODDT). It groups word-type objects whose bits have a particular meaning. These objects are described in detail below.

Example of declaring a variable: `IODDT_VAR1` of type `T_COM_WMY_100`

Notes

- The meaning of a bit is generally given for the status of the bit when set to 1. In specific cases an explanation is given for each status of the bit.
- Not all bits are used.

Execution flags of an explicit exchange: EXCH_STS

The table below presents the exchange control bits of the channel `EXCH_STS` (%MWr.m.c.0).

Standard symbol	Type	Access	Meaning	Address
STS_IN_PROGR	BOOL	R	Reading of status words of the channel in progress.	%MWr.m.c.0.X0
CMD_IN_PROGR	BOOL	R	Exchange of parameters of command in progress	%MWr.m.c.0.X1
ADJ_IN_PROGR	BOOL	R	Exchange of parameters of adjustment in progress	%MWr.m.c.0.X2

Explicit exchange report: EXCH_RPT

The table below presents the meaning of the exchange report bits `EXCH_RPT` (%MWr.m.c.1).

Standard symbol	Type	Access	Meaning	Address
STS_ERR	BOOL	R	Detected error in reading status words of the channel	%MWr.m.c.1.X0
CMD_ERR	BOOL	R	Detected error when exchanging command parameters	%MWr.m.c.1.X1
ADJ_ERR	BOOL	R	Detected error when exchanging adjustment parameters	%MWr.m.c.1.X2

Standard channel detected faults, CH_FLT

The table below presents the meaning of the status word CH_FLT (%MWr.m.c.2). Reading is carried out by a READ_STS(IODDT_VAR1).

Standard symbol	Type	Access	Meaning	Address
INTERNAL_FLT	BOOL	R	Internal detected fault or channel self-test	%MWr.m.c.2.4
APPLI_FLT	BOOL	R	Application detected fault (adjustment or configuration)	%MWr.m.c.2.7

Network diagnostics

The table below shows the status words (or double words) CH_FLT (%MWr.m.c.3 to 6, %MWr.m.c.11 to 15 and %MDr.m.c.7 and 9) used for diagnosing the network. Reading is carried out by a READ_STS(IODDT_VAR1).

Standard symbol	Type	Access	Meaning	Address
NB_P502_CNX	INT	R	Number of Port 502 connections and bridge configuration information	%MWr.m.c.3
NB_DENIED_CNX	INT	R	Number of denied Port 502 connections)	%MWr.m.c.4
NB_P502_REF	INT	R	Number of refused messages on Port 502	%MWr.m.c.5
XWAY_ADDR	INT	R	X-Way address (Network, station)	%MWr.m.c.6
NB_SENT_MSG	DINT	R	Number of sent messages on Port 502	%MDr.m.c.7
NB_RCV_MSG	DINT	R	Number of received messages on Port 502	%MDr.m.c.9
BW_OTHER_MSG	INT	R	Loading of messaging service and other services	%MWr.m.c.15

Configuration of 140 NWM 100 00 module using Concept

13

Subject of this Chapter

This chapter describes configuration of the 140 NWM 100 00 module using Concept.

What's in this Chapter?

This chapter contains the following sections:

13.1 Configuration With Concept

Overview

This section explains the procedure in configuring your module using Concept software.

What’s in this Section?

This section contains the following topics:

Selecting Your PLC

Overview

The following information describes how to start to configure the 140 NWM 100 00 using Concept.

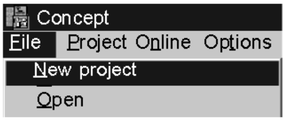
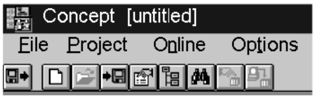
Initial Setup

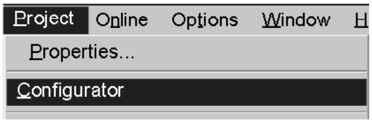
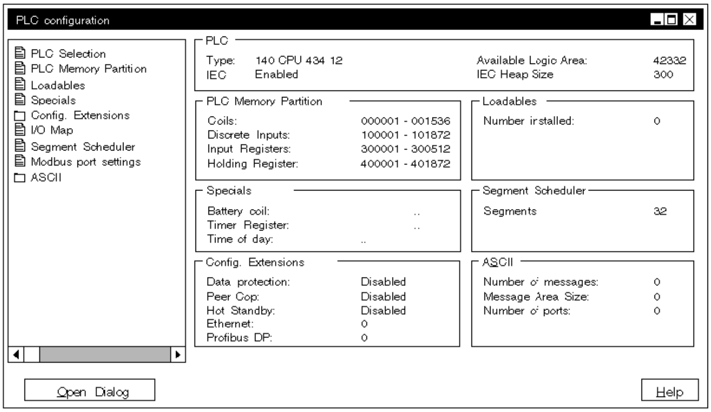
After the NWM module has been installed in a Quantum backplane (refer to the section *Installing the Module*), you can configure it using Concept. To configure the NWM, first select your CPU (PLC).

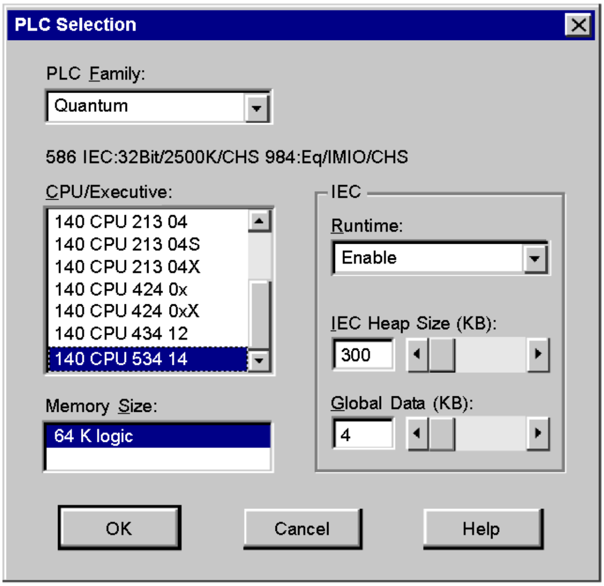
NOTE: For detailed information about how to use Concept, refer to the set of manuals shipped with that software.

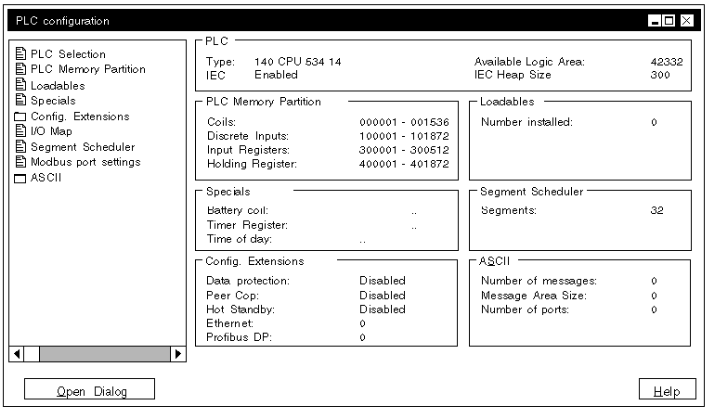
Procedure for Selecting a CPU

Perform the following steps to select a CPU.

Step	Action
1	Open Concept on your programming panel (PC).
2	As shown in the following figure, from the File menu, select **New project**.  Concept File Project Online Options New project Open **Result**: As shown in the following figure, a new project is opened and the file name (untitled) appears over the menu bar. 

Step	Action
3	Or, as shown in the following figure, from the Project menu, select **Configurator**.  Project Online Options Window H Properties... Configurator **Result**: As shown in the following figure, the PLC Configuration screen displays.  PLC configuration PLC Selection PLC Memory Partition Loadables Specials Config. Extensions I/O Map Segment Scheduler Modbus port settings ASCII Open Dialog Help PLC Type: 140 CPU 434 12 IEC Enabled Available Logic Area: 42332 IEC Heap Size 300 PLC Memory Partition Coils: 000001 - 001536 Discrete Inputs: 100001 - 101872 Input Registers: 300001 - 300512 Holding Register: 400001 - 401872 Loadables Number irstalled: 0 Specials Battery coil: .. Timer Register: .. Time of day: .. Segment Scheduler Segments 32 Config. Extensions Data protection: Disabled Peer Cop: Disabled Hot Standby: Disabled Ethernet: 0 Profibus DP: 0 ASCII Number of messages: 0 Message Area Size: 0 Number of ports: 0

Step	Action
4	Double click on the **Type** field in the PLC section of the Configure menu. **Result**: As shown in the following figure, the PLC Selection dialog box displays. The default selection is Quantum.  PLC Selection PLC Family: Quantum 586 IEC:32Bit/2500K/CHS 984:Eq/IMIO/CHS CPU/Executive: 140 CPU 213 04 140 CPU 213 04S 140 CPU 213 04X 140 CPU 424 0x 140 CPU 424 0xX 140 CPU 434 12 140 CPU 534 14 Memory Size: 64 K logic IEC Runtime: Enable IEC Heap Size (KB): 300 Global Data (KB): 4
5	From the **CPU/Executive** scroll box, select the CPU that is installed in your Quantum backplane NOTE: Depending on the CPU selected, you may need to select the correct memory size applicable to it in the **Memory Size** dialog box.

Step	Action
6	Click the **<OK>** button. **Result:** As shown in the following figure, your PLC type and default configuration parameters are displayed in the PLC Configuration screen. 

PLC configuration

PLC

Type:	140 CPU 534 14	Available Logic Area:	42332
IEC	Enabled	IEC Heap Size	300

PLC Memory Partition

Coils:	000001 - 001536
Discrete Inputs:	100001 - 101872
Input Registers:	300001 - 300512
Holding Register:	400001 - 401872

Loadables

Number installed:	0

Specials

Battery coil:	..
Timer Register:	..
Time of day:	..

Segment Scheduler

Segments:	32

Config. Extensions

Data protection:	Disabled
Peer Cop:	Disabled
Hot Standby:	Disabled
Ethernet:	0
Profibus DP:	0

ASCII

Number of messages:	0
Message Area Size:	0
Number of ports:	0

Next

Next, you must configure the number of Ethernet modules that your system will contain, as shown in the *Cabling Schemes* procedure.

Setting the Number of NWMs

Overview

The following information describes how many NWMs you can configure in a single controller and how to configure that number.

Introduction

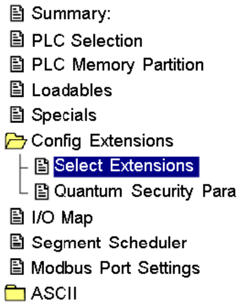
You may configure a mamimum of two to six Ethernet modules in a single controller, depending on the model. A 140 CPU 113 or 213 will accept a total of two network option modules, including NWM, NOE, NOM, NOP, and CRP 811.
A 140 CPU 424, 434, 534. 434A, or 534A will accept six.

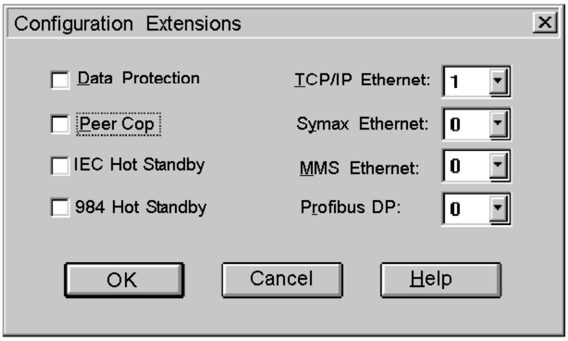
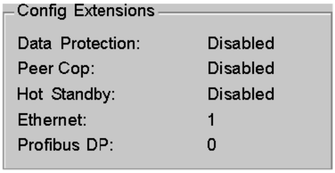
Memory Requirements

The first Ethernet TCP/IP module configured requires 20 words of memory. Each additional module requires an additional 16 words of memory.

Procedure for Setting the Number of NWMs

From the PLC Configuration screen, follow the steps below to select the number of 140 NWM 100 00 modules.

Step	Action
1	As shown in the following figure, from the Configure menu, select Config Extensions, or double-click anywhere in the **Config Extensions** region of the screen. Next, select the Select Extensions prompt.  Summary: PLC Selection PLC Memory Partition Loadables Specials Config Extensions Select Extensions Quantum Security Para I/O Map Segment Scheduler Modbus Port Settings ASCII **Result**: The Configuration Extension dialog box displays.

Step	Action
2	As shown in the following figure, in the TCP/IP Ethernet scroll box, select the number of 140 NWM 100 00 modules to be configured.  Configuration Extensions ☐ Data Protection ☐ Peer Cop ☐ IEC Hot Standby ☐ 984 Hot Standby TCP/IP Ethernet: 1 Symax Ethernet: 0 MMS Ethernet: 0 Profibus DP: 0
3	Click on the <**OK**> button **Result**: As shown in the following figure, the Ethernet status in the Config Extensions section of the PLC Configuration screen, changes from 0 to the number selected in Step 2.  Config Extensions Data Protection: Disabled Peer Cop: Disabled Hot Standby: Disabled Ethernet: 1 Profibus DP: 0

Next

Next, you need to create an I/O map for the NWMs in your configuration, as shown in the *Accessing and Editing the I/O Map (see page 379)* section.

Accessing and Editing the I/O Map

Overview

The following information describes how to create an I/O map for the NWMs in your system.

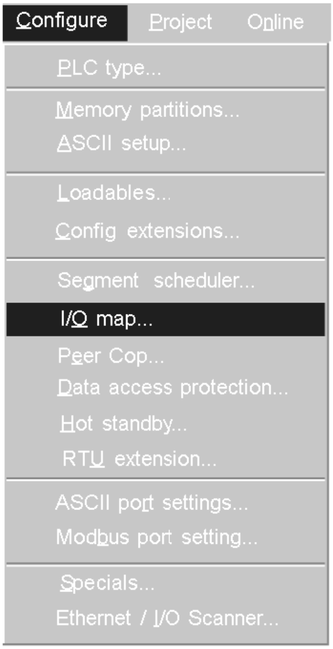
Introduction

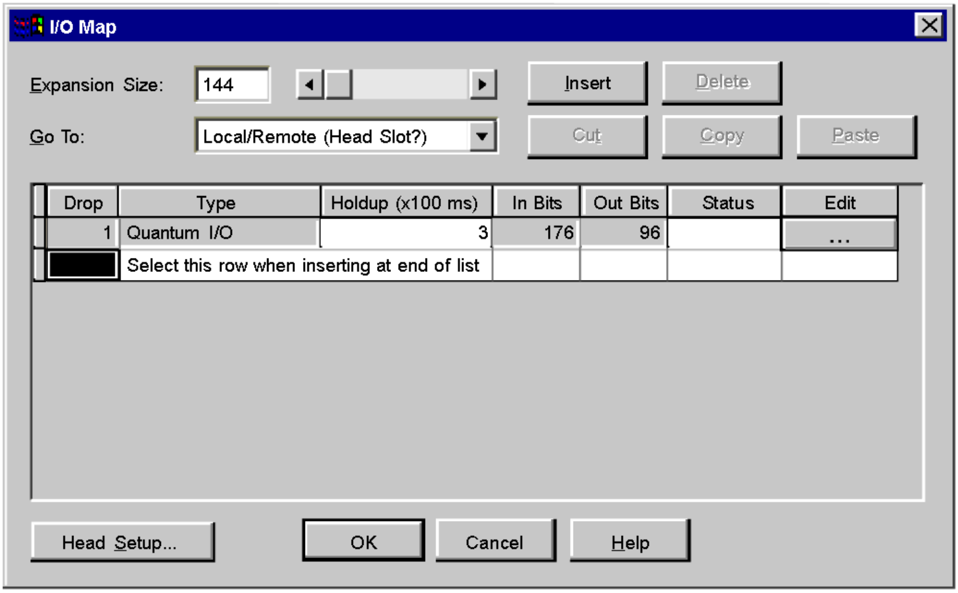
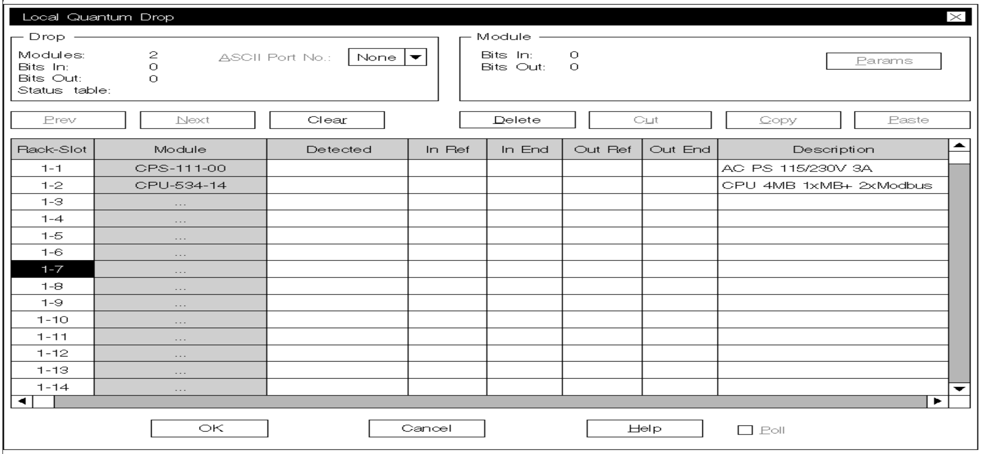
This procedure is required to determine the slot location of NWMs in the system.

As part of the configuration process, you need to create an I/O Map for the local backplane including the 140 NWM 100 00 module.

Procedure for Accessing and Editing an I/O Map

Perform the following steps to access and edit an I/O Map from the PLC Configuration screen.

Step	Action
1	As shown in the following figure, from the Configure menu, select **I/O map**.  Configure Project Online PLC type... Memory partitions... ASCII setup... Loadables... Config extensions... Segment scheduler... I/O map... Peer Cop... Data access protection... Hot standby... RTU extension... ASCII port settings... Modbus port setting... Specials... Ethernet / I/O Scanner... **Result**: As shown in the following figure, the I/O Map dialog box displays.

Step	Action
2	Click the <**Edit**> button at the end of the Quantum I/O row.  The Local Quantum Drop I/O box is displayed. Note: there may be no modules yet listed under the Module column. Continue to the next step. 

I/O Map

Expansion Size: 144

Go To: Local/Remote (Head Slot?)

Drop	Type	Holdup (x100 ms)	In Bits	Out Bits	Status	Edit
1	Quantum I/O	3	176	96		...
	Select this row when inserting at end of list					

Local Quantum Drop

Drop

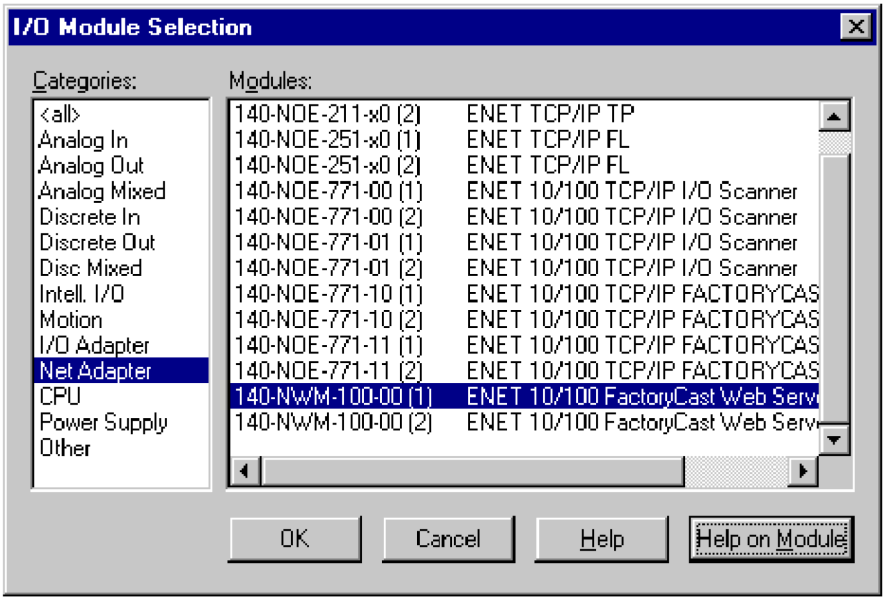
Modules: 2
Bits In: 0
Bits Out: 0
Status table:

ASCII Port No.: None

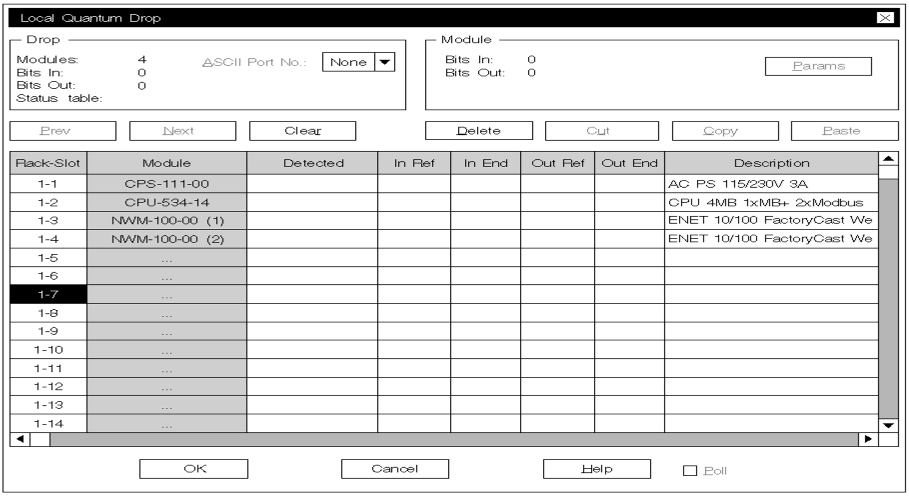
Module

Bits In: 0
Bits Out: 0

Rack-Slot	Module	Detected	In Ref	In End	Out Ref	Out End	Description
1-1	CPS-111-00						AC PS 115/230V 3A
1-2	CPU-534-14						CPU 4MB 1xMB+ 2xModbus
1-3	...						
1-4	...						
1-5	...						
1-6	...						
1-7	...						
1-8	...						
1-9	...						
1-10	...						
1-11	...						
1-12	...						
1-13	...						
1-14	...						

Step	Action
3	Click on the **...** button under **Module**. **Result**: As shown in the following figure, the I/O Module Selection drop-down menu displays. 

140-NOE-211-x0 (2)	ENET TCP/IP TP
140-NOE-251-x0 (1)	ENET TCP/IP FL
140-NOE-251-x0 (2)	ENET TCP/IP FL
140-NOE-771-00 (1)	ENET 10/100 TCP/IP I/O Scanner
140-NOE-771-00 (2)	ENET 10/100 TCP/IP I/O Scanner
140-NOE-771-01 (1)	ENET 10/100 TCP/IP I/O Scanner
140-NOE-771-01 (2)	ENET 10/100 TCP/IP I/O Scanner
140-NOE-771-10 (1)	ENET 10/100 TCP/IP FACTORYCAS
140-NOE-771-10 (2)	ENET 10/100 TCP/IP FACTORYCAS
140-NOE-771-11 (1)	ENET 10/100 TCP/IP FACTORYCAS
140-NOE-771-11 (2)	ENET 10/100 TCP/IP FACTORYCAS
140-NWM-100-00 (1)	ENET 10/100 FactoryCast Web Serv
140-NWM-100-00 (2)	ENET 10/100 FactoryCast Web Serv

Step	Action
4	Click on **140 NWM 100 00**, and then click the <**OK**> button. **Result**: The Local Quantum Drop I/O dialog box updates and the **NWM** is now listed under Module and described in the Description column. 
5	Repeat Steps 3 and 4 if other modules need to be added to the I/O map.
6	Click the <**OK**> buttons to return to the PLC Configuration screen.

Local Quantum Drop

Drop

Modules: 4
Bits In: 0
Bits Out: 0
Status table:

ASCII Port No.: None

Module

Bits In: 0
Bits Out: 0

Rack-Slot	Module	Detected	In Ref	In End	Out Ref	Out End	Description
1-1	CPS-111-00						AC PS 115/230V 3A
1-2	CPU-534-14						CPU 4MB 1xMB+ 2xModbus
1-3	NWM-100-00 (1)						ENET 10/100 FactoryCast We
1-4	NWM-100-00 (2)						ENET 10/100 FactoryCast We
1-5	...						
1-6	...						
1-7	...						
1-8	...						
1-9	...						
1-10	...						
1-11	...						
1-12	...						
1-13	...						
1-14	...						

Next

Next, you will configure the Ethernet address parameters from the Ethernet/ I/O Scanner screen as shown in the *Configuring the Ethernet Address Parameters (see page 383)* section.

Configuring the Ethernet Address Parameters

Overview

The following information describes how to configure Ethernet address parameters for the 140 NWM 100 00 with Concept.

Introduction

The 140 NWM 100 00 module’s Ethernet address parameters, consisting of IP address, Subnet mask, and Gateway addresses, are accessible from the Ethernet/ I/O Scanner dialog box. Prior to performing the following procedure, consult your system administrator to determine if you must configure new Ethernet address parameters, or whether the module will obtain them from the BOOTP server.

NOTE: Concept has to be disconnected from the CPU to configure the 140 NWM 100 00 module.

Having two devices with the same IP address can cause unpredictable operation of your network.

Procedure for Configuring Ethernet Address Parameters

⚠ WARNING

UNINTENTIONAL OPERATION — DUPLICATE IP ADDRESS

- Use a method to assign a unique IP address to each device on the network.
- Obtain your IP addresses from your system administrator to avoid the possibility of duplicate addresses.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Perform the following steps to configure the Ethernet Address Parameters.

Step	Action
1	From the Config Extensions menu, select Ethernet I/O Scanner. **Result**: The Ethernet/ I/O Scanner dialog box displays.
2	Click on the **Specify IP Address** radio button.
3	Type in the new IP Address, Subnet Mask, and Gateway address in the applicable text boxes.
4	Select the correct Internet frame type from the **Frame Type** scroll box.
5	If the module's BOOTP server will assign Ethernet address parameters, click on the Use BOOTP Server radio button. Note that if you select this option, the address parameter text boxes will be grayed out and will not display the addresses.

How the Module Derives Its IP Address

During initialization, the 140 NWM 100 00 module attempts to read the address parameter information from the PLC and determines its IP Address in the following fashion.

- If the PLC has the IP Address and the BOOTP server is not selected, the module will use the configured IP address that you assigned in Step 2 of the above procedure.
- If the BOOTP server was selected in Step 5 of the above procedure, the module will send BOOTP requests to receive its IP Address.
- If no Configuration Extension exists, the 140 NWM 100 00 sends out BOOTP requests. If the module does not receive its IP Address from the BOOTP server after 2 minutes, it will then use the IP Address derived from its MAC address.

NOTE: The MAC address is assigned at the factory and is recorded on a label on the front panel, above the cable connector. This is a unique 48-bit global assigned address. It is set in PROM. The Ethernet address is recorded on the label in hexadecimal, in the form 00.00.54.xx.xx.xx.

13.2 Transfer Data Using 984 LL

Overview

This section explains how to transfer data using 984 Ladder Logic.

What's in this Section?

This section contains the following topics:

MSTR Description

Overview

The following information describes MSTR operations.

Introduction

All 140 NWM 100 00 Quantum Ethernet TCP/IP modules provide the user with the capability of transferring data to and from nodes on a TCP/IP network through the use of a special MSTR (master instruction). All PLCs that support networking communication capabilities over Modbus Plus and Ethernet can use the MSTR ladder logic instruction to read or write controller information.

MSTR Operations

The following table lists each of the possible MSTR network communications operations and indicates whether a TCP/IP Ethernet network supports it. Each operation is designated by a code. The following table lists the operations and indicates those that are supported on an Ethernet TCP/IP network.

MSTR Operation	Operation Type	TCP/IP Ethernet Support
Write data	1	supported
Read Data	2	supported
Get local statistics	3	supported
Clear local statistics	4	supported
Get remote statistics	5	supported
Clear remote statistics	6	supported
Reset Option Module	7	supported
Read CTE (config extension)	8	supported
Write CTE (config extension)	9	supported

Number of MSTR Instructions Allowed

Up to16 MSTR instructions can be simultaneously serviced in a ladder logic program per NWM. More than 16 MSTRs may be programmed to be enabled by the logic flow as one active MSTR block releases the resources it has been using and becomes deactivated, the next MSTR operation encountered in logic can be activated.

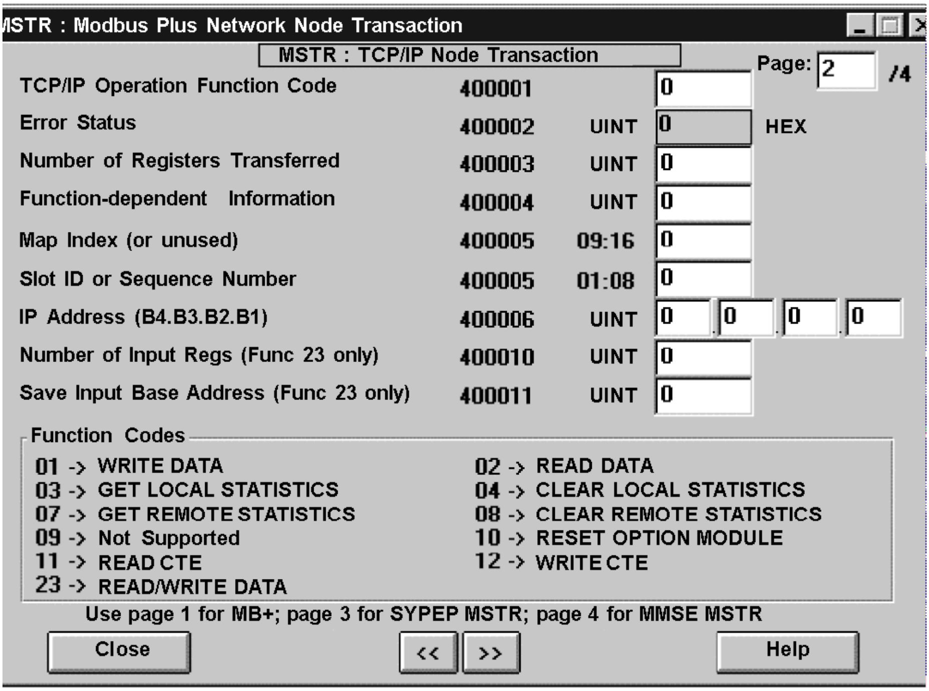
MSTR Block for TCP/IP in Concept

Overview

This following information describes how to complete installing the MSTR block in a TCP/IP network for the 140 NWM 100 00 module.

Procedure for Installing the MSTR Block in TCP/IP

This is the MSTR Block as used for TCP/IP in Concept Ladder Logic. After the MSTR Block is inserted in the network, perform the following steps.

Step	Action
1	Place cursor on the MSTR block.
2	Configure the control block and data area by a double click
3	Press <Ctrl D> or **Edit** → **DX Zoom** (DX Zoom screen)
4	Go to page 2 for TCP/IP specific MSTR block and complete the required information. 

ISTR : Modbus Plus Network Node Transaction

MSTR : TCP/IP Node Transaction

Page: 2 /4

TCP/IP Operation Function Code	400001		0
Error Status	400002	UINT	0 HEX
Number of Registers Transferred	400003	UINT	0
Function-dependent Information	400004	UINT	0
Map Index (or unused)	400005	09:16	0
Slot ID or Sequence Number	400005	01:08	0
IP Address (B4.B3.B2.B1)	400006	UINT	0 . 0 . 0 . 0
Number of Input Regs (Func 23 only)	400010	UINT	0
Save Input Base Address (Func 23 only)	400011	UINT	0

Function Codes

01 -> WRITE DATA	02 -> READ DATA
03 -> GET LOCAL STATISTICS	04 -> CLEAR LOCAL STATISTICS
07 -> GET REMOTE STATISTICS	08 -> CLEAR REMOTE STATISTICS
09 -> Not Supported	10 -> RESET OPTION MODULE
11 -> READ CTE	12 -> WRITE CTE
23 -> READ/WRITE DATA	

Use page 1 for MB+; page 3 for SYPEP MSTR; page 4 for MMSE MSTR

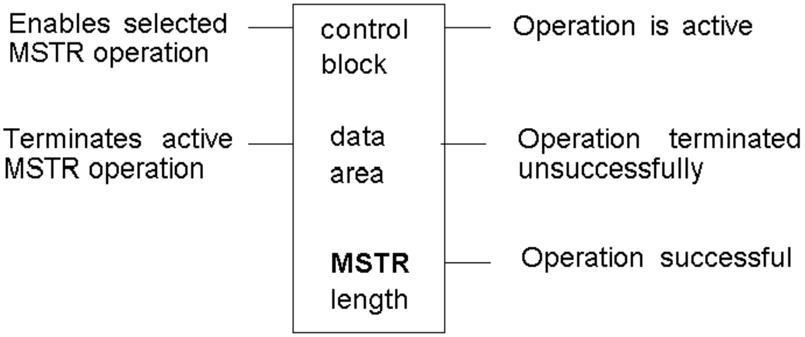
MSTR Ladder Logic Representation

Overview

The following information describes the Ladder Logic representation for MSTR.

Ladder Logic Diagram

As the following figure shows, the MSTR Block is represented in Ladder Logic diagrams.


Enables selected MSTR operation
Terminates active MSTR operation
control block
data area
MSTR length
Operation is active
Operation terminated unsuccessfully
Operation successful

Inputs

The MSTR instruction has the following two control inputs.

- the input to the top node enables the instruction if it is ON
- the input to the middle node terminates the active operation if it is ON

Outputs

The MSTR instruction can produce the following three possible outputs.

- the output from the top node echoes the state of the top input - it goes ON while the instruction is active
- the output from the middle node echoes the state of the middle input - it goes ON if the MSTR operation is terminated prior to completion or if an anomaly occurs in completing the operation
- the output from the bottom node goes ON if an MSTR operation has been completed successfully

All outputs are zero is an indication that four MSTR instructions are already in progress.

Top Node Content

The 4x register entered in the top node is the first of several (network dependent) holding registers that comprise the network *control block*. The *control block* structure differs according to the network in use.

In the case of the Ethernet Read and Write CTE operations the middle node stores the contents of the Ethernet configuration extension table in a series of registers.

The following table shows the *control block* structure for the TCP/IP Ethernet network.

Register	Content
Displayed	Identifies one of ten MSTR operations legal for TCP/IP (1... 4 and 7... 12).
First implied	Displays status.
Second implied	Displays length (number of registers transferred).
Third implied	Displays MSTR operation-dependent information.
Fourth implied	Low byte: Destination index.
	High byte: Quantum backplane slot address of the NWM module.
Fifth implied	Byte 4 (MSB) of the 32-bit destination IP Address.
Sixth implied	Byte 3 of the 32-bit destination IP Address.
Seventh implied	Byte 2 of the 32-bit destination IP Address
Eight implied	Byte 1 (LSB) of the 32-bit destination IP Address

Middle Node Content

The 4x register entered in the middle node is the first in a group of contiguous holding registers that comprise the *data area*. For operations that provide the communication processor with data such as a Write operation, the *data area* is the source of the data. For operations, such as Read, that get data from the communication processor, the *data area* is the destination for the data

In the case of the Ethernet Read and Write CTE operations, the middle node stores the contents of the Ethernet configuration extension table in a series of registers.

Bottom Node Content

The integer value entered in the bottom node specifies the *length* - the maximum number of registers in the *data area*. The *length* must be in the range 1... 100.

MSTR Function Error Codes

Presentation

The following information describes the detected error codes for MSTR operations for the NWM module.

Where Displayed

If an anomaly occurs during an MSTR operation, a hexadecimal code is displayed in the first implied register in the *control block* (the top node). Function codes are network-specific.

TCP/IP Ethernet Codes

The following table describes the anomalies that can occur in the MSTR *control block* if the MSTR routine runs over TCP/IP Ethernet.

Hex Code	Meaning
1001	User has aborted the MSTR element.
2001	An unsupported operation type has been specified in the *control block*.
2002	One or more *control block* parameters has been changed while the MSTR element is active (applies only to operations that take multiple scans to complete). *Control block* parameters may be changed only when the MSTR element is not active.
2003	Invalid value in the length field of the *control block*.
2004	Invalid value in the offset field of the *control block*.
2005	Invalid values in the length and offset fields of the *control block*.
2006	Invalid slave device data area.
2008	Invalid slave device network routing
3000	Generic Modbus detected error code.
30ss*	Modbus slave exception response.
4001	Inconsistent Modbus slave response.
F001	Option Module not responding
* ss = subfield	

The following table lists the ss subfield values in code 30ss.

ss Hex value	Meaning
01	Slave device does not support the requested operation.
02	Nonexistent slave device registers requested.
03	Invalid data value requested.
04	Reserved
05	Slave has accepted long-duration program command.
06	Function cannot be performed now; a long-duration command is in effect.
07	Slave rejected long-duration program command.

TCP/IP Ethernet Network Anomalies

The following table describes the codes that can occur in the MSTR *control block* as a result of an anomaly on the TCP/IP Ethernet network.

Hex Error Code	Meaning
5004	Interrupted system call
5005	I/O detected error
5006	No such address
5009	The socket descriptor is invalid
500C	Not enough memory
500D	Permission denied
5011	Entry exists
5016	An argument is invalid
5017	An internal table has run out of space
5020	The connection is broken
5028	Destination address required
5029	Protocol wrong type for socket
502A	Protocol not available
502B	Protocol not supported
502C	Socket type not supported
502D	Operation not supported on a socket
502E	Protocol family not supported
502F	Address family not supported
5030	Address already in use
5031	Cannot assign requested address
5032	Socket operation on a non-socket

5033	Network is unreachable
5034	Network dropped connection on reset
5035	Network caused connection abort
5036	Connection reset by peer
5037	No buffer space available
5038	Socket is already connected
5039	Socket is not connected
503A	Cannot send after socket shutdown
503B	Too many references, cannot splice
503C	Connection timed-out (See note below.)
503D	Connection refused
503E	Network is down
503F	Text file busy
5040	Too many levels of links
5041	No route to host
5042	Block device required
5043	Host is down
5044	Operation now in progress
5045	Operation already in progress
5046	Operation would block
5047	Function not implemented
5050	No Network Resource
5051	Length detected error
5052	Addressing detected error
5053	Application detected error
5054	Client in Bad State for Request
5055	No Remote Resource (Note: May indicate no path to Remote Device) (See note below.)
5056	Non-Operational TCP Connection
5057	Incoherent Configuration

NOTE: Codes 503C and 5055:

- Code 5055 can occur before a 503C code.

No remote device takes precedence over a timeout.

CTE Codes

The following table lists the codes that are returned if there is a problem with the Ethernet configuration extension table (CTE) is inoperative in your program configuration.

Hex Error Code	Meaning
7001	There is no Ethernet configuration extension.
7002	The CTE is not available for access.
7003	The offset is invalid.
7004	The offset + length is invalid.
7005	Bad data field in the CTE.

Read and Write MSTR Operations

Presentation

The MSTR Write operation (operation type 1 in the displayed register of the top node) transfers data from a master source device to a specified slave destination device on the network. The MSTR Read operation (operation type 2 in the displayed register of the top node) transfers data from a specified slave source device to a master destination device on the network. Read and Write use one data master transaction path and may be completed over multiple scans.

NOTE: TCP/IP Ethernet routing must be accomplished via standard third-party Ethernet IP router products.

Control Block Utilization

The following table describes the registers in the MSTR *control block* (the top node) that contain the Read or Write information.

Register	**Function**	**Content**	
Displayed	Operation Type	1 = Write, 2 = Read	
First implied	Status	Displays a hex value indicating an MSTR detected error.	
		Exception response, where response size is incorrect.	Exception code +3000
		Exception response where response size is incorrect.	4001
		Read Write	
Second implied	Length	Write = number of registers to be sent to slave. Read = number of registers to be read from slave.	
Third implied	Slave device data area	Specifies starting 4x register in the slave to be read from or written to (1 = 4001, 49 =40049).	
Fourth implied	Low byte	Quantum backplane slot address of the NWM module.	
Fifth ... Eighth implied	Destination	Each register contains one byte of the 32-bit IP address.	

Get Local Statistics MSTR Operation

Presentation

The Get Local Statistics operation (operation type 3 in the display register of the top node) obtains information related to the local node where the MSTR has been programmed. Refer to TCP/IP Ethernet Statistics for information recieved in the Get Local Statistics operation.

Control Block Utilization

The following table describes the registers in the MSTR *control block* (the top node). These registers contain the Get Local Statistics information.

Register	Function	Content
Displayed	Operation Type	3
First implied	Status	Displays a hex value indicating an MSTR detected error, if relevant.
Second implied	Length	Starting from *offset*, the number of words of statistics from the local processor’s statistics table; the *length* must be > 0 < *data area*.
Third implied	Offset	An offset value relative to the first available word in the local processor’s statistics table. If the offset is specified as 1, the function obtains statistics starting with the second word in the table.
Fourth implied	Low byte	Quantum backplane slot address of the NWM module.
Fifth ... Eighth implied	Not applicable	

Clear Local Statistics MSTR Operation

Presentation

The Clear Local Statistics operation (operation type 4 in the displayed register of the top node) clears statistics relative to the local node where the MSTR has been programmed.

Control Block Utilization

The following table describes the registers in the MSTR *control block* (the top node). These registers contain the Clear Local Statistics information.

Register	Function	Content
Displayed	Operation Type	4
First implied	Status	Displays a hex value indicating an MSTR detected error, if relevant.
Second implied	Not applicable	
Third implied	Not applicable	
Fourth implied	Low byte	Quantum backplane slot address of the NWM module.
Fifth ... Eighth implied	Not applicable	

Get Remote Statistics MSTR Operation

Presentation

The Get Remote Statistics operation (operation type 7 in the displayed register of the top node) obtains information relative to remote nodes on the network. This operation may require multiple scans to complete and does not require a master data transaction path.

The remote Ethernet module always returns its complete statistics table if a request is made, even if the request is for less than the full table. The MSTR instruction then copies only the amount of words you have requested to the designated 4x registers.

NOTE: TCP/IP Ethernet routing must be accomplished via standard third-party Ethernet IP router products.

Control Block Utilization

The following table describes the registers in the MSTR *control block* (the top node). These registers contain the Get Remote Statistics information.

Register	Function	Content
Displayed	Operation Type	7
First implied	Status	Displays a hex value indicating an MSTR detected error, if relevant.
Second implied	Length	Starting from an *offset*, the number of words of statistics from the local processor's statistics table; the length must be > 0 < *data area*.
Third implied	Offset	Specifies an offset value relative to the first available word in the local processor's statistics table. If the *offset* is specified as 1, the function obtains statistics starting with the second word in the table.
Fourth implied	High byte	Destination index
Fifth ... Eighth implied	Destination	Each register contains one byte of the 32-bit IP address.

Clear Remote Statistics MSTR Operation

Presentation

The Clear Remote Statistics operation (operation type 8 in the displayed register of the top node) clears statistics relative to a remote network node from the *data area* in the local node. This operation may require multiple scans to complete and uses a single data master transaction path.

Control Block Utilization

The following table describes the registers in the MSTR *control block* (the top node). These registers contain the Clear Remote Statistics information.

Register	Function	Content
Displayed	Operation Type	8
First implied	Status	Displays a hex value indicating an MSTR detected error, if relevant.
Second implied	Not applicable	
Third implied	Not applicable	
Fourth implied	High byte	Destination index
Fifth ... Eighth implied	Destination	Each register contains one byte of the 32-bit IP address.

Reset Option Module MSTR Operation

Presentation

The Reset Option Module operation (operation type 10 in the displayed register of the top node) causes a Quantum 140 NWM 100 00 option module to enter a reset cycle to reset its operational environment.

Control Block Utilization

The following table describes the registers in the MSTR *control block* (the top node). These registers contain the Reset Option Module information.

Register	Function	Content
Displayed	Operation Type	10
First implied	Status	Displays a hex value indicating a MSTR detected error, if relevant.
Second implied	Not applicable	
Third implied	Not applicable	
Fourth implied	Low byte	Quantum backplane slot address of the NWM module.
Fifth ... Eighth implied	Not applicable	

Read CTE (Config Extension Table) MSTR Operation

Presentation

The Read CTE operation (operation type 11 in the displayed register of the top node) reads a given number of bytes from the Ethernet configuration extension table to the indicated buffer in PLC memory. The bytes to be read begin at a byte offset from the beginning of the CTE. The content of the Ethernet CTE table is displayed in the middle node of the MSTR block.

Control Block Utilization

The following table describes the registers in the MSTR *control block* (the top node). These registers contain the Read CTE information.

Register	**Function**	**Content**
Displayed	Operation Type	11
First implied	Status	Displays a hex value indicating a MSTR detected error, when relevant.
Second implied	Not applicable	
Third implied	Not applicable	
Fourth implied	Low byte	Quantum backplane slot address of the NWM module.
Fifth ... Eighth implied	Not applicable	

CTE Display Implementation

The values in the Ethernet configuration extension table (CTE) are displayed in a series of registers in the middle node of the MSTR instruction when a Read CTE operation is implemented. The middle node contains the first of 11 contiguous 4x registers.

The following table describes the CTE data contained in the registers.

Parameter	**Register**	**Content**	
Frame type	Displayed	1 = 802.3	
		2 = Ethernet	
IP Address	First implied	Byte 4 (MSB) of the 32-bit IP address	
	Second implied	Byte 3 of the 32-bit IP address	
	Third implied	Byte 2 of the 32-bit IP address	
	Fourth implied	Byte 1 (LSB) of the 32-bit IP address	
Subnetwork mask	Fifth implied	Hi word	
	Sixth implied	Low word	
Gateway	Seventh implied	Byte 4 (MSB) of the 32-bit gateway address	
	Eighth implied	Byte 3 of the 32-bit gateway address	
	Ninth implied	Byte 2 of the 32-bit gateway address	
	Tenth implied	Byte 1 (LSB) of the 32-bit gateway address	
	Eleventh implied	**High byte**	**Low byte**

NOTE: Module type only used by the p-unit during an upload to determine module.

Write CTE (Config Extension Table) MSTR Operation

Presentation

The Write CTE operation writes an indicated number of bytes from PLC memory, starting at a specified byte address, to an indicated Ethernet configuration extension table at a specified offset. The content of the Ethernet CTE table is contained in the middle node of the MSTR block.

Network Implementation

The Write CTE operation (type 12 in the displayed register of the top node) can be implemented for TCP/IP Ethernet networks via the appropriate network adapter.

NOTE: Modbus Plus networks do not use this operation.

Control Block Utilization

In a Write CTE operation, the registers in the MSTR *control block* (the top node) differ according to the network in user.

The following table describes the registers in the MSTR *control block* (the top node). These registers contain the Write CTE information.

Register	Function	Content
Displayed	Operation Type	12
First implied	Status	Displays a hex value indicating a MSTR detected error, if relevant.
Second implied	Not applicable	
Third implied		
Fourth implied	Low byte	Either a value displayed in the high byte of the register or not used.
	Slot Index	Number displayed in the low byte, in a range 1 ... 16 indicating the slot in the local backplane where the option resides.
Fifth ... Eighth implied	Not applicable	

CTE Display Implementation

The values in the Ethernet configuration extension table (CTE) are displayed in a series of registers in the middle node of the MSTR instruction if a Write CTE operation is implemented. The middle node contains the first of 11 contiguous 4x registers.

The following table describes the CTE data contained in the registers.

Parameter	**Register**	**Content**	
Frame type	Displayed	1 = 802.3	
		2 = Ethernet	
IP Address	First implied	First byte of the IP address	
	Second implied	Second byte of the IP address	
	Third implied	Third byte of the IP address	
	Fourth implied	Fourth byte of the IP address	
Subnetwork mask	Fifth implied	Hi word	
	Sixth implied	Low word	
Gateway	Seventh implied	First byte of the gateway	
	Eighth implied	Second byte of the gateway	
	Ninth implied	Third byte of the gateway	
	Tenth implied	Fourth byte of the gateway	
	Eleventh implied	**High byte**	**Low byte**

TCP/IP Ethernet Statistics

Presentation

A TCP/IP Ethernet board responds to the "Get Local Statistics" and "Set Local Statistics" commands with the following information.

Word	Meaning
00 ... 02	MAC address
03	Board Status (see the Board Status Bit Definition table in this map)
04 and 05	Number of receiver interrupts
06 and 07	Number of transmitter interrupts
08 and 09	Transmit _ timeout count
10 and 11	Collision_detect count
12 and 13	Missed packets
14 and 15	Memory detected error
16 and 17	Number of times driver has restarted
18 and 19	Receive framing error
20 and 21	Receiver overflow error
22 and 23	Receive CRC error
24 and 25	Receive buffer error
26 and 27	Transmit buffer error
28 and 29	Transmit silo underflow
30 and 31	Late collision
32 and 33	Lost carrier
34 and 35	Number of retries
36 and 37	IP address

Board Status Word Bit Definition

Bit #	Definition
15	NWM Link On=1, Off=0
14	NWM Appl LED On-1, Off=0
13	NWM Fiber/TP)Fiber=1, TP=0)
12	NWM 100Mbit (100=1, 10=0
11-4	Module Type (see table below)
3	UNUSED
2	NWM Full Duplex (0=Half, 1=Full)
1	NWM Configured
0	NWM Runningt

Board Status Word Bit Definition by Module Type

Value of Bits 11-4	Module Type
0	NOE 2x1
1	170 ENT 110 00
2	M1E
3	NOE 771 00
4	ETY 410
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	Reserved
10	NOE 771 10
11	NOE 771 01
12	NOE 771 11
13	NWM 100 00
14	170 ENT 111 01

For bit level detail for the Momentum 170ENT11001 and Momentum 170ENT11000 see the *Momentum Ethernet Communication Adapter 170ENT11001 and 170ENT11000 User Guide*, 870USE11400.

13.3 Transfer Data Using IEC

Overview

This section explains how to transfer data blocks using IEC.

What’s in this Section?

This section contains the following topics:

CREAD_REG

Function Description

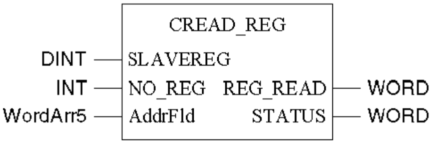
This function block reads the register area continuously. It reads data from an addressed node via TCP/IP-Ethernet.

EN and ENO can be projected as additional parameters.

NOTE:

- When programming a CREAD_REG function, you must be familiar with the routing procedures used by your network.
- For technical reasons, this function block does not allow the use of programming languages ST and IL.

Representation



Parameter Description

Parameter	Data Type	Meaning
SLAVEREG	DINT	Offset address of the first 4x register in the slave to be read from
NO_REG	INT	Number of registers to be read from slave
AddrFld	WordArr5	Data structure describing the TCI/IP address
REG_READ	WORD	First 4x area register for read values
STATUS	WORD	See Runtime detected errors

Elementary Description for WordArr5 with TCP/IP Ethernet

Element	Data Type	Meaning
WordArr5[1]	WORD	Low value byte: MBP on Ethernet Transporter (MET) mapping index High value byte: Slot of the NWM module
WordArr5[2]	WORD	Byte 4 (MSB) of the 32-bit destination IP address
WordArr5[3]	WORD	Byte 3 of the 32-bit destination IP address
WordArr5[4]	WORD	Byte 2 of the 32-bit destination IP address
WordArr5[5]	WORD	Byte 1 (LSB) of the 32-bit destination IP address

Function Mode of the CREAD_REG Block

Although a large number of CREAD_REG function blocks can be programmed, only four read operations may be active at the same time. In such a case it is insignificant whether they are the result of this function block or others (e.g. MBP_MSTR, MSTR, READ_REG). All function blocks use one data transaction path and require multiple cycles to complete a job.

NOTE: A TCP/IP communication between a Quantum PLC (NWM) and a Momentum PLC (all TCP/IP CPUs and all TCP/IP I/O modules) is only possible, when only one read or write job is carried out in every cycle. If several jobs are sent per PLC cycle, the communication stops without generating a message in the status register of the function block.

The entire routing information is contained in data structure WordArr5 of input AddrFld. The type of function block connected to this input and thus the contents of the data structure depends on the network used.

Please use:
- TCP/IP Ethernet the function block TCP_IP_ADDR

NOTE: For experts: The WordArr5 data structure can be used with constants as well.

NOTE: This function block puts a heavy load on the network; therefore the network load must be carefully monitored. If the network load is too high, the program logic should be reorganized in order to work with the READ_REG function block, a variation of this function block that does not operate in a continuous mode, but under command control.

SLAVEREG

Start of the area in the addressed slave from which the source data is read. The source area always resides within the 4x register area. SLAVEREG expects the source reference as offset within the 4x area. The leading "4" must be omitted (e.g. 59 (contents of the variables or value of the literal) = 40059).

The parameter can be specified as direct address, located variable, unlocated variable, or literal.

NO_REG

Number of registers to be read from the addressed slave (1 ... 100). The parameter can be entered as a Direct address, Located variable, Unlocated variable, or Literal. The parameter can be entered as a Direct address, Located variable, or Unlocated variable.

REG_READ

This word parameter addresses the first register in a series of NO_REG registers, listed one after the other, which are used as a destination data area. The parameter must be entered as a Direct address or located Variable.

STATUS

See Runtime detected errors.

The parameter can be specified as direct address, located variable or unlocated variable.

CWRITE_REG

Function Description

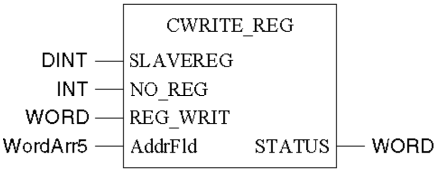
The purpose of this function block is to write the register area continuously. It transfers data from the PLC via TCP/IP Ethernet to an addressed slave.

EN and ENO can be configured as additional parameters.

NOTE:

- You must be familiar with the routing procedures of the network when programming a CWRITE_REG function.
- For technical reasons, this function block does not allow the use of ST and IL programming languages.

Symbol



Parameter Description

Parameter	Data Type	Meaning
SLAVEREG	DINT	Offset address of the first 4x register in the slave to be written to
NO_REG	INT	Number of registers to be written to slave
REG_WRIT	WORD	First 4x register of the source data area
AddrFld	WordArr5	Data structure for transferring the TCI/IP address
STATUS	WORD	MSTR code, see Runtime detected errors

Elementary Description for WordArr5 with TCP/IP Ethernet

Element	Data Type	Meaning
WordArr5[1]	WORD	Low value byte: MBP on Ethernet Transporter (MET) mapping index High value byte: Slots of the NWM module
WordArr5[2]	WORD	Byte 4 (MSB) of the 32-bit destination IP address
WordArr5[3]	WORD	Byte 3 of the 32-bit destination IP address
WordArr5[4]	WORD	Byte 2 of the 32-bit destination IP address
WordArr5[5]	WORD	Byte 1 (LSB) of the 32-bit destination IP address

CWRITE_REG Block Function Mode

Although a large number of CWRITE_REG function blocks can be programmed, only four write operations may be active at the same time. It makes no difference whether these operations are performed using this function block or others (e.g. MBP_MSTR, MSTR, WRITE_REG). All function blocks use one data transaction path and require multiple cycles to complete a job.

If several CWRITE_REG function blocks are used within an application, they must at least differ in the values of their NO_REG or REG_WRITE parameters.

NOTE: A TCP/IP communication between a Quantum PLC (NWM) and a Momentum PLC (all TCP/IP CPUs and all TCP/IP I/O modules) is only possible, when only one read or write job is carried out in every cycle. If several jobs are sent per PLC cycle, the communication stops without generating a message in the status register of the function block.

The entire routing information is contained in data structure WordArr5 of input AddrFld. The type of function block connected to this input and thus the contents of the data structure depend on the network used.

Please use:

- TCP/IP Ethernet: the function block TCP_IP_ADDR

NOTE: For experts: The WordArr5 data structure can also be used with constants.

NOTE: This function block puts a heavy load on the network. The network load must therefore be carefully monitored. If the network load is too high, the program logic should be reorganized to work with the WRITE_REG function block, which is a variant of this function block that does not operate in continuous mode but is command driven.

SLAVEREG

Start of the area in the addressed slave to which the source data are written. The destination area always resides within the 4x register area. SLAVEREG expects the destination address as offset within the 4x area. The initial "4" must be omitted (e.g. 59 (contents of the variables or value of the literal) = 40059).

The parameter can be specified as direct address, located variable, unlocated variable, or Literal.

NO_REG

Number of registers to be written to slave processor (1 ... 100). The parameter can be specified as direct address, located variable, unlocated variable, or Literal.

STATUS

MSTR code, see Runtime detected errors.

The parameter can be specified as direct address, located variable or unlocated variable.

REG_WRIT

This word parameter addresses the first register in a series of NO_REG Successive registers used as source data area.

The parameter must be entered as a direct address or located variable.

READ_REG

Function Description

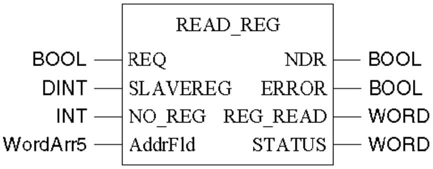
If requested, this function block will read a register area once (rising edge of the REQ input). It reads data from an addressed slave via TCP/IP-Ethernet.

EN and ENO can be projected as additional parameters.

NOTE:

- You must be familiar with the routing procedures of your network when programming a READ_REG function.
- For technical reasons, this function block does not allow use of the programming languages ST and IL.

Symbol



Parameter Description

Parameter	Data Type	Meaning
REQ	BOOL	Start read operation once
SLAVEREG	DINT	Offset address of the first 4x register in the slave to be read from
NO_REG	INT	Number of registers to be read from slave
AddrFld	WordArr5	Data structure describing the TCP/IP address
NDR	BOOL	Set to "1" for one cycle after reading new data
ERROR	BOOL	Set to "1" for one scan in case of error
STATUS	WORD	See Runtime detected errors
REG_READ	WORD	First 4x area register for read values

Elementary Description for WordArr5 with TCP/IP Ethernet

Element	Data Type	Meaning
WordArr5[1]	WORD	Low value byte: MBP on Ethernet Transporter (MET) mapping index High value byte: Slot of the NWM module
WordArr5[2]	WORD	Byte 4 (MSB) of the 32-bit destination IP address
WordArr5[3]	WORD	Byte 3 of the 32-bit destination IP address
WordArr5[4]	WORD	Byte 2 of the 32-bit destination IP address
WordArr5[5]	WORD	Byte 1 (LSB) of the 32-bit destination IP address

Function Mode of READ_REG Blocks

Although a large number of READ_REG function blocks can be programmed, only four read operations may be active at the same time. In such a case it is insignificant whether they are the result of this function block or of other read operations (e.g. MBP_MSTR, MSTR, CREAD_REG). All function blocks use one data transaction path and require multiple cycles to complete a job.

NOTE: A TCP/IP communication between a Quantum PLC (NWM) and a Momentum PLC (all TCP/IP CPUs and all TCP/IP I/O modules) is only possible, when only one read or write job is carried out in every cycle. If several jobs are sent per PLC cycle, the communication stops without generating a message in the status register of the function block.

The entire routing information is contained in data structure WordArr5 of input AddrFld. The type of function block connected to this input and thus the contents of the data structure depends on the network used.

Please use:

- TCP/IP Ethernet the function block TCP_IP_ADDR

NOTE: For experts: The WordArr5 data structure can be used with constants as well.

REQ

A rising edge triggers the read transaction.

The parameter can be specified as direct address, located variable, unlocated variable, or Literal.

SLAVEREG

Start of the area in the addressed slave from which the source data is read. The source area always resides within the 4x register area. SLAVEREG expects the source reference as offset within the 4x area. The leading "4" must be omitted (e.g. 59 (contents of the variables or value of the literal) = 40059).

The parameter can be specified as direct address, located variable, unlocated variable, or Literal.

NO_REG

Number of registers to be read from the addressed slave (1 ... 100).

The parameter can be specified as direct address, located variable, unlocated variable, or Literal.

NDR

Transition to ON state for one program cycle signifies receipt of new data ready to be processed.

The parameter can be specified as direct address, located variable, or unlocated variable.

ERROR

Transition to ON state for one program cycle signifies detection of a new anomaly.

The parameter can be specified as direct address, located variable, or unlocated variable.

REG_READ

This word parameter addresses the first register in a series of NO_REG registers lying in series used as destination data area.

The parameter must be entered as a direct address or located variable.

STATUS

See Runtime detected errors.

The parameter can be specified as direct address, located variable or unlocated variable.

WRITE_REG

Function Description

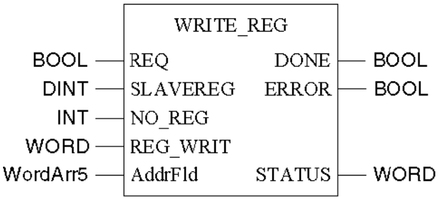
If requested, this function block will write a register area once (rising edge of the REQ input). It transfers data from the PLC via TCP/IP Ethernet to an addressed slave.

EN and ENO can be configured as additional parameters.

NOTE:

- You must be familiar with the routing procedures of your network when programming a WRITE_REG function.
- For technical reasons, this function block does not allow the use of ST and IL programming languages.

Symbol



Parameter Description

Parameter	Data Type	Meaning
REQ	BOOL	Start write operation once
SLAVEREG	DINT	Offset address of the first 4x register in the slave to be written to
NO_REG	INT	Number of registers to be written from slave
AddrFld	WordArr5	Data structure transferring the TCP/IP address
REG_WRIT	WORD	First 4x register of the source data area
DONE	BOOL	Set to "1" for one scan after writing data
ERROR	BOOL	Set to "1" for one scan in case of error
STATUS	WORD	See Runtime detected errors

Elementary Description for WordArr5 with TCP/IP Ethernet

Element	Data Type	Meaning
WordArr5[1]	WORD	High value byte: Slot of the NWM module Low value byte: MBP on Ethernet Transporter (MET) mapping index
WordArr5[2]	WORD	Byte 4 (MSB) of the 32-bit destination IP address
WordArr5[3]	WORD	Byte 3 of the 32-bit destination IP address
WordArr5[4]	WORD	Byte 2 of the 32-bit destination IP address
WordArr5[5]	WORD	Byte 1 (LSB) of the 32-bit destination IP address

Function Mode of the WRITE_REG Module

Although a large number of WRITE_REG function blocks can be programmed, only four write operations may be active at the same time. In such a case it is insignificant whether they are the result of this function block or of other write operations (e.g. MBP_MSTR, MSTR, CWRITE_REG). All function blocks use one data transaction path and require multiple cycles to complete a job.

If several WRITE_REG function blocks are used within an application, they must at least differ in the values of their NO_REG or REG_WRITE parameters.

NOTE: A TCP/IP communication between a Quantum PLC (NWM) and a Momentum PLC (all TCP/IP CPUs and all TCP/IP I/O modules) is only possible, when only one read or write job is carried out in every cycle. If several jobs are sent per PLC cycle, the communication stops without generating a message in the status register of the function block.

The status signals DONE and ERROR report the function block state to the user program.

The entire routing information is contained in data structure WordArr5 of input AddrFld. The type of function block connected to this input and thus the contents of the data structure depend on the network used.

Please use:

- TCP/IP Ethernet: the function block TCP_IP_ADDR

NOTE: For experts: The WordArr5 data structure can also be used with constants.

REQ

A rising edge triggers the read transaction.

The parameter can be specified as direct address, located variable or unlocated variable.

SLAVEREG

Start of the area in the addressed slave from which the source data is read. The source area always resides within the 4x register area. SLAVEREG expects the source reference as offset within the 4x area. The leading "4" must be omitted (e.g. 59 (contents of the variables or value of the literal) = 40059).

The parameter can be specified as direct address, located variable, unlocated variable, or Literal.

NO_REG

Number of registers to be read from the addressed slave (1 ... 100).

The parameter can be specified as direct address, located variable, unlocated variable, or Literal.

REG_WRIT

This word parameter addresses the first register in a series of NO_REG registers used as source data area.

The parameter must be entered as a direct address or located variable.

DONE

Transition to ON state for one program scan signifies data have been transferred.

The parameter can be specified as direct address, located variable or unlocated variable.

ERROR

Transition to ON state for one program scan signifies detection of a new error.

The parameter can be specified as direct address, located variable or unlocated variable.

STATUS

See Runtime detected errors.

The parameter can be specified as direct address, located variable, or unlocated variable.

TCP_IP_ADDR

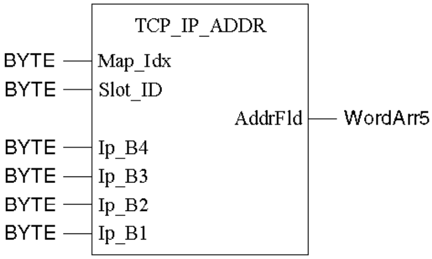
Function Description

This function block enables the input of TCP/IP addresses for the READ_REG, CREAD_REG, WRITE_REG and CWRITE_REG Function Blocks. The address is transferred in the form of a data structure.

The parameters EN and ENO can additionally be projected.

NOTE: You must be familiar with your network when programming the TCP_IP_ADDR Function Block.

Symbol



Parameter Description

Parameter	Data Type	Meaning
Map_Idx	BYTE	Map-Index MBP on Ethernet Transporter (MET) mapping index
Slot_ID	BYTE	Slot ID Slot of the NWM module
Ip_B4	BYTE	Byte 4 (MSB) of the 32-bit destination IP address
Ip_B3	BYTE	Byte 3 of the 32-bit destination IP address
Ip_B2	BYTE	Byte 2 of the 32-bit destination IP address
Ip_B1	BYTE	Byte 1 (LSB) of the 32-bit destination IP address
AddrFld	WordArr5	Data structure used to transfer the TCP/IP address

Elementary Description for WordArr5

Element	Data Type	Meaning
WordArr5[1]	WORD	High value byte: Slot of the NWM module Low value byte: MBP on Ethernet Transporter (MET) mapping index
WordArr5[2]	WORD	Byte 4 (MSB) of the 32-bit destination IP address
WordArr5[3]	WORD	Byte 3 of the 32-bit destination IP address
WordArr5[4]	WORD	Byte 2 of the 32-bit destination IP address
WordArr5[5]	WORD	Byte 1 (LSB) of the 32-bit destination IP address

Map_Idx

The MBP on Ethernet Transporter (MET) mapping index is given at the Map_Idx input, i.e. if MET is 6, the value appears as follows.

0	0	0	0	0	1	1	0

Slot_ID

If an NWM in the rack of a Quantum controller is addressed as destination node, the value at the Slot_ID input represents the physical NWM slot, i.e. if the NWM is plugged in at Slot 7 of the rack, the value appears as follows.

0	0	0	0	0	1	1	1

AddrFld

If an NWM in the rack of a Quantum controller is addressed as a destination node, the value in the High value byte represents the physical slot of the NWM and the Low value byte represents the MBP on Ethernet Transporter (MET) mapping index, i.e. if the NWM is inserted in slot 7 of the rack and the MET mapping index is 6, the first element of the data structure looks as follows.

High value byte								Low value byte							
0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	0

High value byte Slots 1 ... 16

Low value byte MBP on Ethernet Transporter (MET) mapping index

MBP_MSTR

Function Block

With this function block, it is possible to select one of 12 available network communication operations.

Although a large number of MBP_MSTR function blocks can be programmed, only four of them can be active at the same time. All function blocks use one data transaction path and require multiple cycles to complete a job.

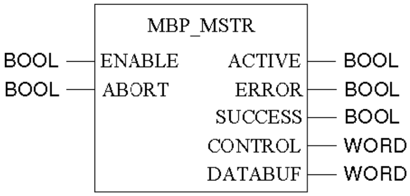
NOTE:

- A TCP/IP communication between a Quantum PLC (NWM) and a Momentum PLC (all TCP/IP CPUs and all TCP/IP I/O modules) is only possible, when only one read or write job is carried out in every cycle. If several jobs are sent per PLC cycle, the communication stops without generating an error message in the status register of the function block.
- In FBD and LD sections, the function block can only be used on the program level, i.e. not in Derived Function Blocks (DFBs).
- For technical reasons, this function block does not allow the use of programming languages ST and IL.

EN and ENO can be configured as additional parameters.

NOTE: You must be familiar with the routing procedures of your network when programming an MSTR function.

Symbol



Parameter Description

Parameter	Data Type	Meaning
ENABLE	BOOL	Enable MSTR function
ABORT	BOOL	Cancel active MSTR operation
ACTIVE	BOOL	Operation is active
ERROR	BOOL	Operation not completed
SUCCESS	BOOL	Operation completed successfully
CONTROL	WORD	First 4x register of the MSTR control block
DATABUF	WORD	First 4x register of the data field

Function Mode of MBP_MSTR Blocks

Using the MBP_MSTR block, one of 12 available network communication operations can be triggered via the network. Each operation receives a code. Whether the operations are available depends on the type of network used.

Valid Function Codes

Code	Function	TCP/IP Ethernet
1	Write Data	X
2	Read Data	X
3	Get Local Statistics	X
4	Clear Local Statistics	X
7	Get Remote Statistics	X
8	Clear Remote Statistics	X
10	Reset optional module	X
11	Read CTE (Config extension)	X
12	Read CTE (Config extension	X
23	Read/Write register	X

Legend

X	Yes
-	No

ENABLE

When ON, the operation specified in the first CONTROL register is enabled.

ABORT

When ON, the currently active operation is aborted.

ACTIVE

ON, if the operation is active.

ERROR

ON, if the operation was aborted without success.

SUCCESS

ON, if the operation concluded successfully.

DATABUF

The 4x register specified is the first in a group of successive output/marker words, making up the data field. For operations providing data, e.g. the write operation, the data field is the data source. For operations receiving data, e.g. the read operation, the data field is the data sink.

In the case of Ethernet CTE Read and Write operations, the middle input stores the contents of the Ethernet configuration extension table in a series of registers.

CONTROL

This word parameter addresses the first of several successive 4x registers. The control block is contained in these registers. The first register displayed contains a number from 1 to 12, which provides the operation code of the MODBUS operation to be performed. The contents of the sequence registers are determined by the operation.

The structure of the control block differs according to the network used:

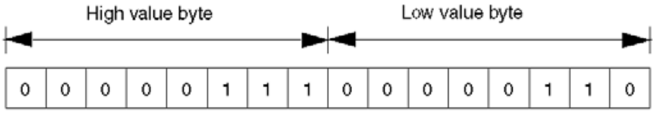
- TCP/IP Ethernet

Control Block for TCP/IP Ethernet

Register	Contents
4x	Indicates one of the Operations which are valid for TCP/IP
4x + 1	Indicates the Error status
4x + 2	Indicates the length (number of registers transferred)
4x + 3	Indicates MSTR operation-dependent information
4x + 4	Routing register Low value byte: MBP on Ethernet Transporter (MET) mapping index High value byte: Slot of the NWM module
4x + 5	Byte 4 (MSB) of the 32-bit destination IP address
4x + 6	Byte 3 of the 32-bit destination IP address
4x + 7	Byte 2 of the 32-bit destination IP address
4x + 8	Byte 1 (LSB) of the 32-bit destination IP address

Routing Register (4x + 4) in TCP/IP Ethernet

If a NWM in the rack of a Quantum controller is addressed as destination node, the value in the high value byte represents the physical NWM slot and the value in the low value byte represents the MBP on Ethernet (MET) mapping index, i.e. if the NWM is plugged in at slot 7 of the rack and MET mapping index is 6, the first element of the data structure appears as follows.



0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	0

High value byte Slots 1 to 16

Low value byte MBP on Ethernet Transporter (MET) mapping index

13.4 SNMP

Overview

This section explains how to set up SNMP service.

What's in this Section?

This section contains the following topics:

SNMP

Presentation

This following information describes the Simple Network Management Protocol (SNMP), which is configured on your NWM.

Network management software allows a network manager to

- Monitor and control network components
- Isolate difficulties and find their causes
- Query devices such as a host computer, routers, switches, and bridges to determine their status
- Obtain statistics about the networks to which they attach

Manager/Agent Paradigm

Network management software follows the conventional client-server model.

To avoid confusion with other network communication protocols that use the client/server terminology, network management software uses the following terms:

- *Manager*
  For the client application that runs on the manager’s computer
- *Agent*
  For the application that runs on a network device

The manager uses conventional transport protocols (e.g., TCP or UDP) to establish communication with the agent. Managers and agents then exchange requests and responses according to the network management protocol.

Simple Network Management Protocol

Your 140 NWM 100 00 module is configured with the Simple Network Management Protocol (SNMP), which is the standard protocol used to manage a local area network (LAN). SNMP defines exactly how a *manager* communicates with an *agent*.

The SNMP defines the format of the requests that a manager sends to an agent and the format of the replies that the agent returns to the manager.

The MIB

Each object SNMP has access to has to be defined and given a unique name. Both the manager and agent program must agree on the names and the meanings of the fetch and store operations. The set of all objects SNMP can access is known as a *Management Information Base (MIB*).

The Private MIB

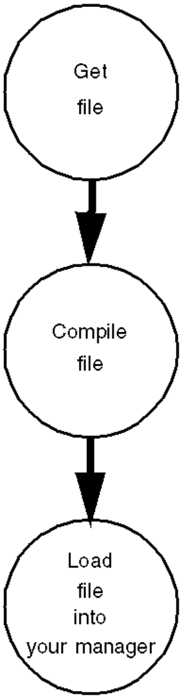
Schneider obtained a private MIB, Groupe_Schneider (3833). Under the Groupe Schneider private MIB is a Transparent Factory Ethernet (TFE) private MIB. The Transparent Factory SNMP embedded component controls the Schneider private MIB function.

Choosing a SNMP Manager

If you have a SNMP Manager already working, you may continue to use that SNMP Manager. If you are selecting a SNMP Manager, there are many SNMP Managers on the market, and you may use any of these managers. You must use a SNMP Version 1 compliant manager.

Using a SNMP Manager

Step	Action
1	Get Schneider `.mib` file from the NWM Web page. You are going to find the `.mib` file as a packed file under **/wwwroot/SchneiderTFE.zip** on your NWM module.
2	Compile `.mib` file in the compiler that comes with the NWM.
3	Load compiled `.mib` file to the SNMP manager.
4	When you are done, you will see the Schneider private MIB manager in your manager.



More SNMP Information

SNMP and related subjects are well documented on Web sites and in many books.

- As of this writing, a useful description appears on Microsoft's *Technet* pages. Browse to *http://www.microsoft.com/technet*. Use the **Search** function to find "Network Management for Microsoft Networks Using SNMP."
- Use an Internet search engine to search for a SNMP introduction, a SNMP tutorial, and other topics on SNMP.
- The SNMP FAQ from the news group `comp.protocols.snmp` appear on many `.com` and `.org` Web pages. Search for the combination of "`comp.protocols.snmp`" and "`FAQ`."
- A list of print books about SNMP appears in the SNMP FAQs. In addition, a search of most online retail book sites will yield a substantial list of titles.

ASN.1 Naming Scheme

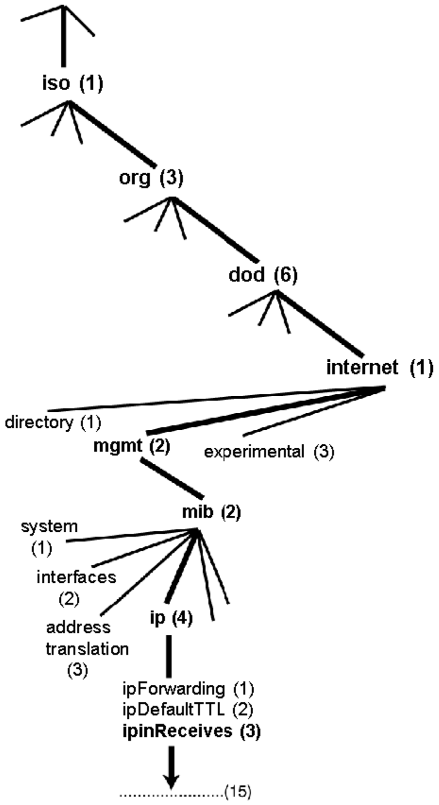
ASN.1 Overview

Abstract Syntax Notation One (ASN.1) is a formal language for abstractly describing messages to be exchanged between distributed computer systems.

An Example

Objects in a MIB are defined with the ASN.1 naming scheme that assigns each object a long prefix that guarantees that the name will be unique. For example, an integer that counts the number of IP datagrams that a device has received is named: *iso.org.dod.internet.mgmt.mib.ip.ipinReceives*.

The following figure depicts the ASN.1 Naming Scheme example.



This object name is represented in an SNMP message by assigning each part an integer. So, the above message would appear as 1.3.6.1.2.2.4.3.

Each integer has the following meaning.

- 1 = ISO (International Organization for Standardization)
- 3 = identified organization — one of branches under the ISO root
- 6 = U. S. Department of Defense (DOD) — one of the children under branch1.3
- 1 = the Internet subtree under 1.3.6
- 2 = the mgmt branch — (one of seven) of the Internet subtree. It is managed by the Internet Assigned Numbers Authority, and includes the standard MIBs
- 2 = mib-2(1) group of managed objects
- 4 = ip — the mib-2(1) IP group (one of 11)
- 3 = ipinReceives — the MIB object

Configuring the NWM with SNMP

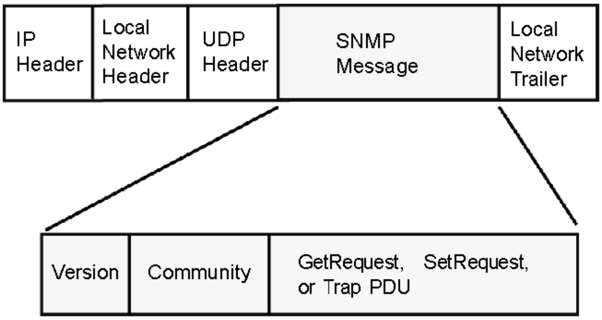
The Object Identifier (OID)

In the ASN.1 Naming Scheme example, the MIB object identified by the notation 1.3.6.1.2.2.4.3 is referred to as the Object Identifier or OID. All OIDs can be envisioned as part of a tree structure which begins at the root (ISO) and branches out with each subtree identified by an integer.

SNMP Protocol Data Units

SNMP uses Protocol Data Units (PDUs) to carry the requests and responses, between the manager and the agents, for the information contained in an OID.

As the following figure shows, the SNMP message is the innermost part of a typical network transmission frame.



The PDUs within the SNMP initiate the communication between the manager and the agents.

The SNMP installed on your NWM module uses the following three PDUs.
- GetRequest
- SetRequest
- Trap

GetRequest PDU

The GetRequest (shortened to Get) PDU is used by the SNMP manager to retrieve the value of one or more objects (OIDs) from an agent.

SetRequest PDU

The SetRequest (shortened to Set) PDU is used by the SNMP manager to assign a value to one or more objects (OIDs) residing in an agent.

Trap PDU

The Trap PDU is used by the agent to alert the manager that a predefined event has occurred.

Version & Community Identifiers

The version identifies the version number of the SNMP software being used by the manager and the agent. Your NWM supports Version 1 of the SNMP. The community is an identifier that you assign to your SNMP network. If community names for the manager and the agent do not agree, the agent will send a message to the manager. If the community names and version number agree, the SNMP PDU will be processed.

What Can Be Configured

Your NWM module can be configured to send an authentication trap to two SNMP managers if it receives a community name in a Get/Set request that does not match the configured name. Also, you can configure the SysContact and SysLocation via the configuration page in the module’s Embedded Web pages. After making changes in the SNMP Configuration Web page and to set those changes, reboot the module using hot swap.

Configuring the NWM with TFE Private MIB

Introduction

An MIB, Management Information Base, is an element used in network management. Network management services are based on the need to monitor and manage the following.

- Performance
- Detected fault occurrences
- Security

Each MIB contains a finite number of objects. Manage your MIB with a management station running an SNMP management application. The management application uses **GETs** and **SETs** to retrieve system information and to set system environment variables.

Schneider Private MIB

Schneider Automation obtained a Private Enterprise Number (PEN) from the Internet Assigned Numbers Authority (IANA). That number represents a subtree in the SNMP MIB, a number that is a unique identifier used for Groupe Schneider.

The object identifier for the root of the Groupe Schneider subtree is **`1.3.6.1.4.1.3833`** and represents a path to the subtree as follows.

```
iso(1)
        org(3)
                dod(6)
                        internet(1)
                                private(4)
                                        enterprise(1)
                                                GroupeSchneider(3833)
                                                        Transparent_Factory_Ethernet(1)
```

Under the GroupeSchneider private MIB is a Transparent Factory Ethernet (TFE) private MIB, **Transparent_Factory_Ethernet(1)**.

TFE Private MIB

The Transparent Factory SNMP-embedded component controls the Schneider private MIB function. The Schneider private MIB, and associated services, perform Network Management on all components of the system. The Transparent Factory private MIB provides the data to manage the main Transparent Factory communication services for all the communication components of the Transparent Factory architecture. The Transparent Factory private MIB does not define the specific management applications and policies.

The **Groupe_Schneider (3833)** subtree is the root of Groupe Schneider's private MIB in the Structure of Management Information (SMI) used by SNMP and defined in RFC-1155, which is a specification that defines the structure and identification of management information for TCP/IP-based Internets.

Transparent Factory Ethernet Subtree

The **Transparent_Factory_Ethernet** subtree defines groups that support the Transparent Factory Ethernet services and devices.

Service	Description
Port502_Messaging (2)	Subtree defines objects for managing explicit client / server communications supporting applications, such as HMI, SCADA, or programming tools
Web (5)	Subtree defines objects for managing the activity of the embedded Web servers
Equipment_Profiles)	Subtree identifies objects for each type of device in Transparent Factory Ethernet's product portfolio

Port502 Messaging Subtree

The Port502_Messaging subtree, or group, provides connection management and data flow services. The following list describes the function of each object.

Service	Description
port502Status(1)	Indicates the status of the service (Idle, Operational)
port502SupportedProtocol(2)	Indicates the supported protocols (MODBUS, Xway)
port502IpSecurity(3):	Indicates the status of the Port502 IP Security service (enabled/disabled)
port502MaxConn(4)	Indicates the maximum TCP connection number supported by the Port502 entity
port502LocalConn(5)	Indicates the TCP connection number currently opened by the local Port502 entity
port502RemConn(6)	Indicates the TCP connection number currently opened by the remote entity to the local Port502 entity
port502IpSecurityTable(7)	Indicates a table containing the number of unsuccessful TCP connection open tries from a remote TCP entity
port502ConnTable(8)	Indicates a table containing Port502 TCP specific information (MsgIn, MsgOut)
port502MsgIn(9)	Indicates the total number of Port502 messages received from the network
port502MsgOut(10)	Indicates the total number of Port502 messages sent from the network
port502MsgOutErr(11)	Indicates the total number of messages built by the Port502 messaging entity and sent to the network
port502AddStackStat(12)	Indicates the support of additional port502 stack statistics 1 - Disabled 2 - Enabled
port502AddStackStatTable(13)	Indicates additional stack statistics for Port502 (optional)

Web Subtree

The Web subtree, or group, contains the objects related to the Web server service.

Service	Description
webStatus(1)	Indicates the global status of the Web service 1 - Idle 2 - Operational
webPassword (2)	Indicates a switch to enable or disable the use of Web passwords 1 - Disabled 2 - Enabled
webSuccessfullAccess (3)	Indicates the total number of successful accesses to the Web site
webFailedAttempts (4)	Indicates the total number of unsuccessful accesses to the Web site

Equipment Profile Subtree

The Equipment_Profiles subtree contains a set of common objects.

Service	Description
profileProductName(1)	Displays the commercial name of the NWM 100 00
profileVersion(2)	Displays the software version of the communication product in a string form (for example: Vx.y or V1.1)
profileCommunicationServices(3)	Displays a list of the communication services supported by the profile (Port502Messaging and Web.)
profileConfigMode(5)	Indicates the IP configuration mode of the communication module 1 - Local: The IP configuration is created locally 2 - dhcpServed: The IP configuration is created by a remote DHCP server
profileRoleName(6)	Indicates the role name for the IP address management if it exists (Empty string if there is none)
profileLedDisplayTable(9)	Displays a table giving the name and the state of each module’s LEDs
profileSlot(10)	Indicates the position of the communication module inside the rack if there is one. If there is no rack, the profileSlot value will be zero
profileCPUType(11)	Indicates that if the CPU type exists, this variable identifies the host for which that communication module is a part. If there is no host, the string is empty
profileTrapTableEntriesMax(12)	Indicates the maximum numbers of entries in the Trap Table. This entry equals the number of possible remote managers

Service	Description
profileTrapTable(13)	Displays a table allowing you to enable or disable the private traps for each of the communication services
profileSpecificId(14)	Indicates a unique Profile Specific Identification inside the equipmentProfile object of the Schneider Transparent Factory MIB. (For example the PLC Premium family is 100)
profileIpAddress(15)	Indicates the IP address of the SNMP agent
profileIpNetMask(16)	Indicates the subnet mask associated with the IP address of the SNMP agent. The value of the mask is an IP address with all the network bits set to 1 and all the host bits set to 0
profileIpGateway(17)	Indicates the default Gateway IP address of the SNMP agent
profileMacAddress(18)	Indicates the Ethernet media-dependent address of the SNMP agent

Private Traps and MIB Files

Traps are used to signal Status Changes to the manager. Using traps helps to avoid unnecessary network traffic.

The four status changes signaled by the trap are for:
- LEDs
- Communication Ports

The following list describes the characteristics of private traps, which means that they can:
- Send messages to the two managers whose IP addresses are configured in the SNMP configuration (either the PL7, Unity Pro, or the Web page)
- Use the community name given to this configuration
- Enable or disable each of the Transparent Factory Ethernet Private MIB groups: Switch (1), Port502_Messaging (2), Web (3), Equipment_Profiles(4)

Private traps are described in the MIB ASN.1 description, which is contained in a **`.mib`** text file.

13.5 Maintenance

Overview

This section deals with performing maintenance on the NWM.

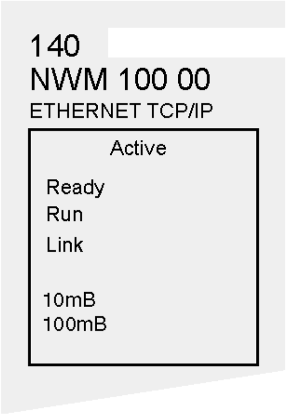
What’s in this Section?

This section contains the following topics:

Responding to Detected Errors

Detecting Errors

When anomalies occur, the 140 NWM 100 00 LED display can help you determine what went wrong. The following figure shows the pattern that the LEDs should display during normal operation.



The **Run** indicator will be solid. The **Coll** LED may flash, indicating that collisions are occurring on the Ethernet network. Such collisions are normal.

If an anomaly occurs, the normal LEDs may be extinguished or other indicators may light. This section will discuss informations reported by the **Active**, **Ready**, **Coll**, **Link**, **Kernel**, **Appl** and **Fault** indicators.

For each type, try the suggested remedies in the order given. If no remedy suggested here works, call your local service representative or Schneider Electric customer service for further directions.

Procedure for Responding to an Active LED Indicator

If the **Active** LED is inoperative, the 140 NWM 100 00 module is not communicating with the backplane. The following procedure describes the steps to perform to respond to an Active LED detected error.

Step	Action
1	Make sure the NNWM module and the controller are installed properly.
2	Verify that the controller is working; if it is not, replace it.
3	If neither the new controller nor the NWM module functions, replace the backplane.
4	Make sure that no more than two network option modules -- including NWM, NOE, NOP, and CRP 811 modules -- have been installed in the backplane with a 140 CPU 113 or 213; not more than six network option modules with a 140 CPU 424, 534 or 140 CPU 6xx.
5	Check the version of the controller executive. You must have version 2.0 or greater to support the Ethernet module. Earlier versions do not recognize the module.
6	If the module stays inoperative after steps 4 and 5, replace the NWM module.

Procedure for Responding to a Ready LED Indicator

If the **Ready** LED is inoperative, the 140 NWM 100 00 module did not pass internal diagnostic tests. The following procedure describes the steps to perform.

Step	Action
1	Make sure that power has been applied to the backplane.
2	If step 1 checks out ok, replace the NWM module.

Procedure for Responding to a Link LED Indicator

If the **Link** LED is inoperative, the 140 NWM 100 00 module is not communicating with the Ethernet hub/switch. The following procedure describes the steps to perform to respond to a **Link** LED detected error.

Step	Action
1	Make sure that the cable has been installed correctly and the module is functioning properly.
2	Verify that the hub/switch is working properly.
3	If steps 1 and 2 check ok, replace the NWM module.

Kernel LED

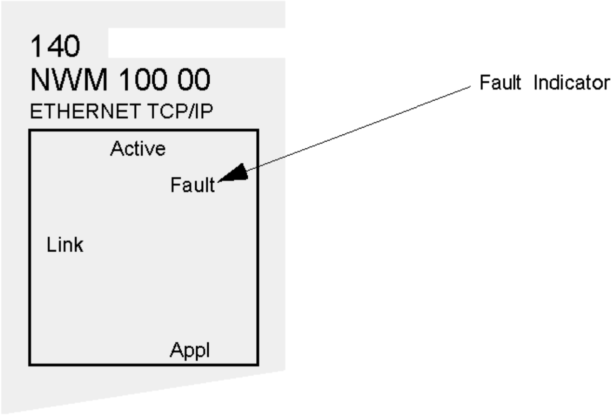
Kernel LED anomalies that may occur and how to respond to them.

If	Then
The **Ready** LED is on and the **Kernel** LED is flashing	the module has detected an invalid software image.
The **Ready** LED is on and the **Kernel** LED is shining steadily,	the module is not in kernal mode; it is inoperative and customer support should be contacted.
Either of the above conditions exists.	download ftp and a new NWM Exec.

Fault LED

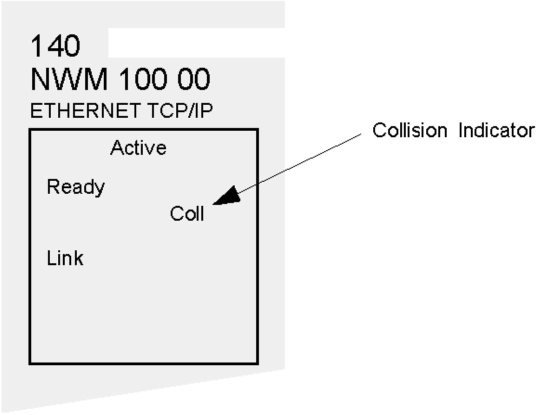
The **Fault** LED can flash briefly following a detected error as the module attempts to recover.

The following figure shows the **Fault** LED.



Collision LED

The following figure shows the Collision LED.



Procedure for Responding to a Collision LED

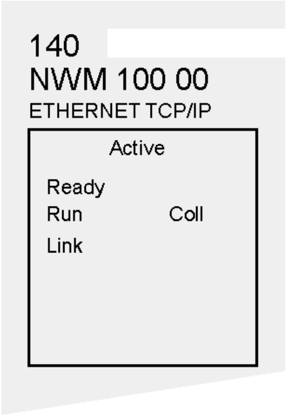
If the Collision LED is inoperative, use the following procedure.

Step	Action
1	Make sure that the cable has been installed and is working properly.
2	Verify that the Ethernet Hub/Switch is working properly.

Collision LED Normal Condition

If the **Coll** LED is flashing, the module is reporting collisions on the Ethernet network. While such collisions are normal, the frequency of the flashes is an indication of the volume of traffic on the network. The flashes may be so frequent that the LED appears to be shining steadily. Heavy collisions will slow communications. If response time is important to your application, you should consider segmenting your network to reduce the frequency of collisions.

The following figure shows the Collision LED under normal conditions.



Run LED

The following table describes the action to be taken if the **Run** LED is flashing. The action depends on the number of flashes in sequence.

# of Flashes in Sequence	Action
One	Not used
Two	Not used
Three	No Link:The network cable is not connected or is broken.
Four	Duplicate IP address
Five	No IP address: The module is attempting to obtain an IP address from the BOOTP server.
Six	Using default IP address
Seven	No valid executive NWM present
Eight	Invalid IP configuration (Likely cause; Default gateway is not on the same subnet mask as the module.
Nine	Flash file system is corrupted

Application LED

If the module crashes, it will note the reason in a log. If the module is able to recover, the **Appl** LED will light, indicating that an entry has been made in the crash log.

Reading and Clearing the Crash Log

Presentation

The crash log provides you with the ability to capture conditions that lead to an anomalous condition. By providing the crash log to Schneider Electric technical support, you can facilitate their assistance to answer your questions.

NOTE: The crash log is provided with the understanding that, with a complex product in thousands of customer applications, there may be conditions that require advance diagnostics. The crash log is one of the tools used to solve complex anomalies.

The Crash Log

If the **Appl** indicator is on, entries have been made in the crash log. The log may hold up to 64K of entries.

Reading the Crash Log

The crash log can be read from the Embedded Web Pages or via FTP.

Procedure for Reading the Crash Log via FTP

The following procedure describes the steps to perform to access the crash log via FTP.

Step	Action
1	Log into the module’s FTP server with a login ID and default password.
2	Change the directory to *wwwroot/conf/diag*
3	Perform an FTP to get the crash log file: `get crash.log`

Clearing the Crash Log

The crash log can be cleared from the Embedded Web Pages or via FTP.

Procedure for Clearing the Crash Log via FTP

The following procedure describes the steps to perform to access the crash log via FTP.

Step	Action
1	Log into the module’s FTP Server.
2	Change the directory to *wwwroot/conf/diag.*
3	Perform an FTP to delete the crash log file: `rm crash.log`

Downloading a New NWM Exec

Introduction

The following tools can be used to download a new 140 NWM 100 00 Exec.

- Schneider Electric programming packages (see 840USE49300 Concept User Manual)
- FTP

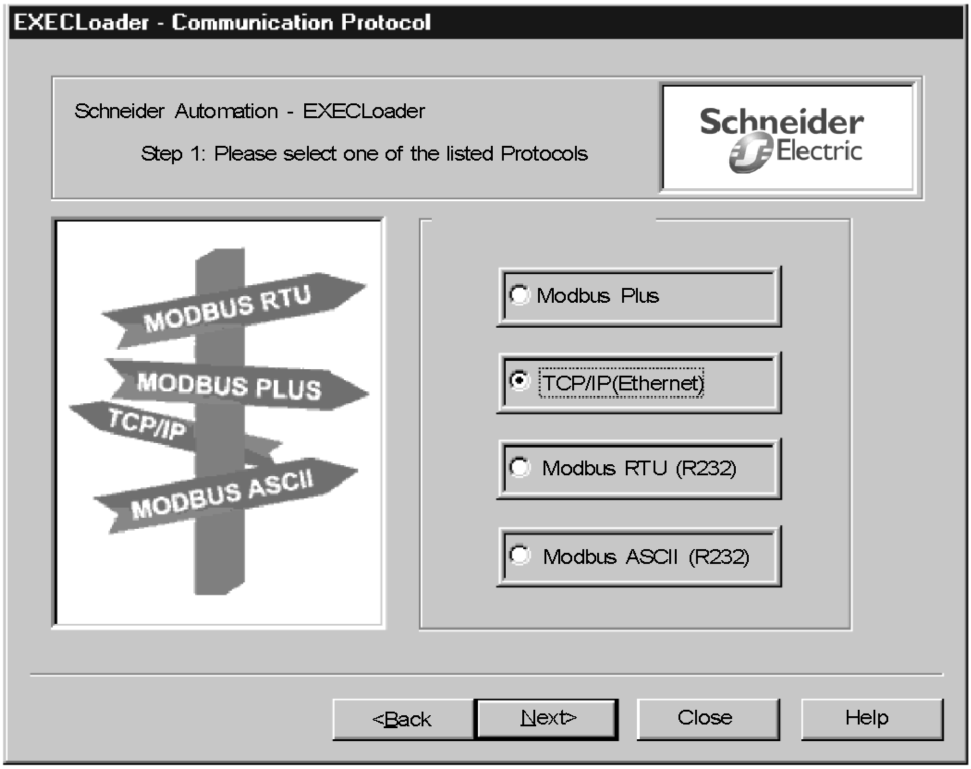
The Concept EXECLoader

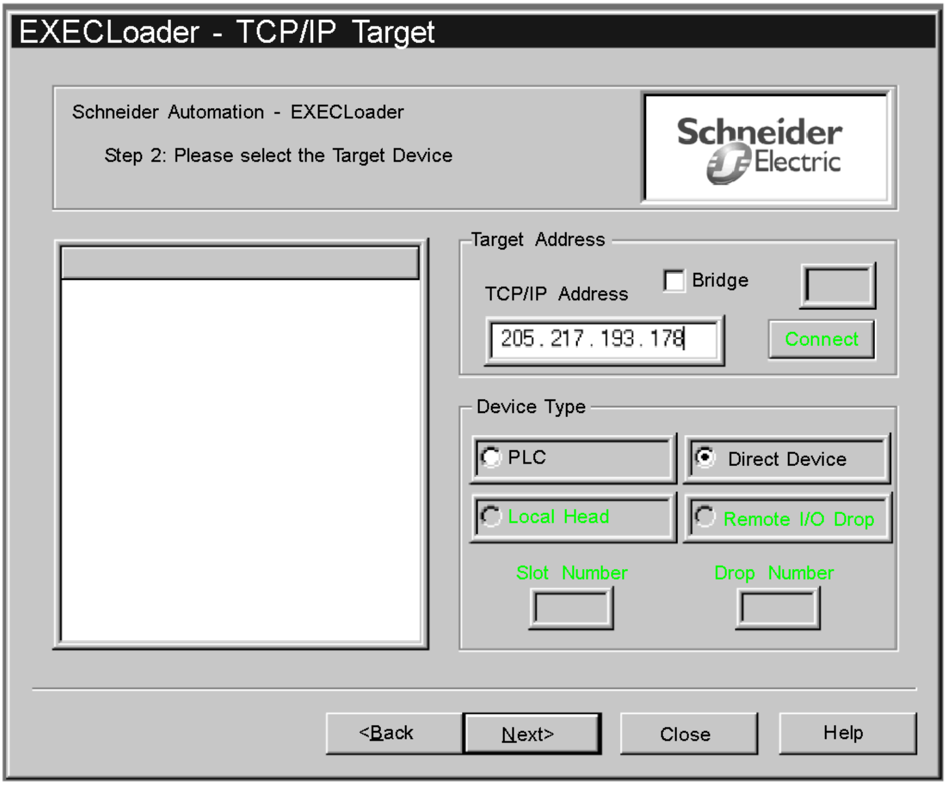
Overview

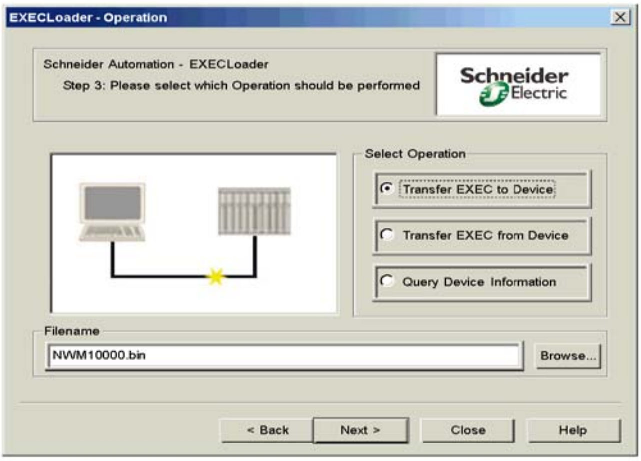
This section describes the use of the Concept EXECLoader, for downloading a new 140 NWM 100 000 Exec.

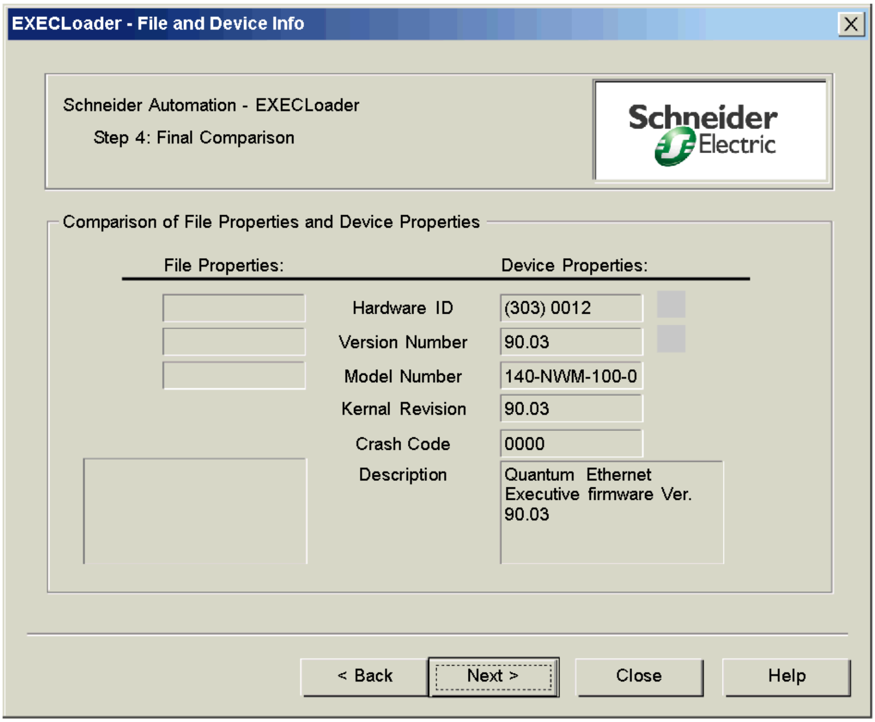
Procedure for Downloading NWM Exec

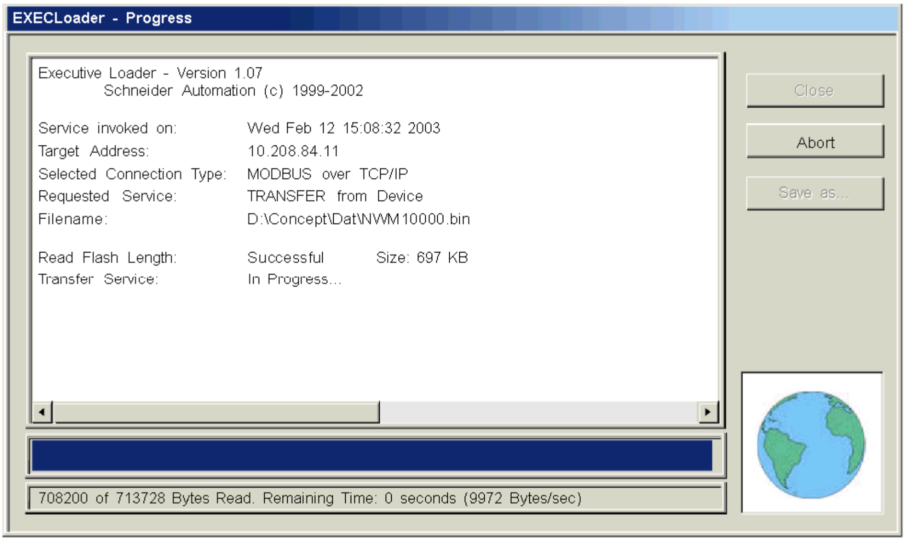
The following procedure provides the steps involved in downloading a new 140 NWM 100 00 Exec, using the Concept EXECLoader.

Step	Action
1	Activate the Exec Loader program.
2	Click on the **Next** button to start the Exec Loader process. **Result:** As shown in the following figure, the EXECLoader - Communication Protocol screen is displayed.  EXECLoader - Communication Protocol Schneider Automation - EXECLoader Step 1: Please select one of the listed Protocols Schneider Electric MODBUS RTU MODBUS PLUS TCP/IP MODBUS ASCII ○ Modbus Plus ● TCP/IP(Ethernet) ○ Modbus RTU (R232) ○ Modbus ASCII (R232) <Back Next> Close Help Click on **TCP/IP [Ethernet**.}

Step	Action
3	Click on the **Next** button **Result**: As shown in the following figure, the EXECLoader - TCP/IP Target screen is displayed.  EXECLoader - TCP/IP Target Schneider Automation - EXECLoader Step 2: Please select the Target Device Target Address TCP/IP Address 205.217.193.178 □ Bridge Device Type ○ PLC ● Direct Device ○ Local Head ○ Remote I/O Drop Slot Number Drop Number Schneider Electric Connect <Back Next> Close Help Click on **Direct Device**.

Step	Action
4	Click on the **Next** button. **Result**: As shown in the following figure, the EXECLoader - Operation screen is displayed.  EXECLoader - Operation Schneider Automation - EXECLoader Step 3: Please select which Operation should be performed Select Operation ● Transfer EXEC to Device ○ Transfer EXEC from Device ○ Query Device Information Filename: NWM10000.bin Browse... < Back · Next > · Close · Help Click on **Transfer EXEC to Device**.
5	Use Browser to select the file name.

Step	Action
6	Click on the **Next** button. **Result**: The EXECLoader - File and Device Info screen is displayed.  EXECLoader - File and Device Info Schneider Automation - EXECLoader Step 4: Final Comparison Comparison of File Properties and Device Properties File Properties: Device Properties: Hardware ID (303) 0012 Version Number 90.03 Model Number 140-NWM-100-0 Kernal Revision 90.03 Crash Code 0000 Description Quantum Ethernet Executive firmware Ver. 90.03

Step	Action
7	Click on the **Next** button. **Result**: As shown in the following figure, The EXECLoader - Progress screen is displayed.  EXECLoader - Progress Executive Loader - Version 1.07 Schneider Automation (c) 1999-2002 Service invoked on: Wed Feb 12 15:08:32 2003 Target Address: 10.208.84.11 Selected Connection Type: MODBUS over TCP/IP Requested Service: TRANSFER from Device Filename: D:\Concept\Dat\NWM10000.bin Read Flash Length: Successful Size: 697 KB Transfer Service: In Progress... 708200 of 713728 Bytes Read. Remaining Time: 0 seconds (9972 Bytes/sec) When the process is completed, you can click on the **Close** button.

Configuration of the TSX WMY 100 module running PL7

14

Subject of this Chapter

This chapter describes installation of the TSX WMY 100 module in the PL7 software workshop.

NOTE: The functional level of the TSXWMY 100 module requires version V4.4 of the PL7 software.

What's in this Chapter?

This chapter contains the following sections:

14.1 General

Subject of this Section

This section introduces ETHERNET communication from a **TSX WMY 100** HMI module and its properties.

What’s in this Section?

This section contains the following topics:

General

Presentation

The ETHERNET communication channel of the **TSX WMY 100** HMI module provides connection to a TCP/IP network supporting the UNI-TE messaging service and Modbus on a TCP/IP profile.

It also offers the following services:

- FTP and SNMP messaging
- Embedded HTTP server
- DHCP Client
- The Replace inoperative device service, which allows the device configuration to be saved in the module. In case of breakdown, another installed blank module can be restarted with the configuration parameters of the previous one.

In addition, the **TSX WMY 100** module offers the option of creating user Web pages.

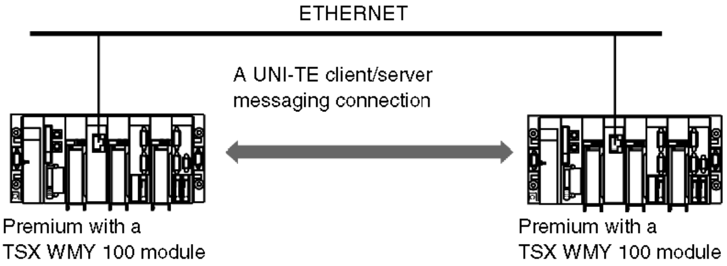
Type of supported connections

At a Glance

The **TSX WMY 100** HMI module authorizes a maximum of 16 parallel HTTP connections and a maximum of 64 parallel TCP/IP connections.

Connection to a TSX Premium

Between two TSX Premium PLCs using UNI-TE messaging on the TCP/IP profile, the **TSX WMY 100** module only allows one connection in client/server mode.

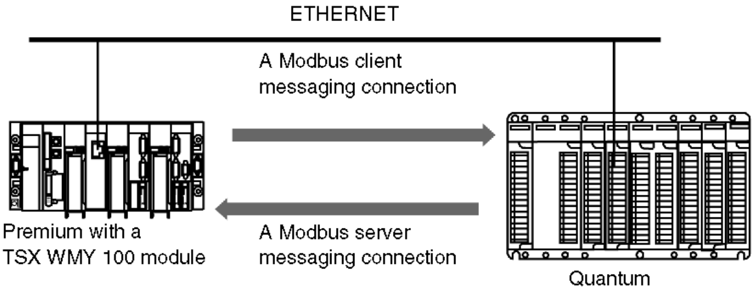

ETHERNET
A UNI-TE client/server messaging connection
Premium with a TSX WMY 100 module
Premium with a TSX WMY 100 module

Connection to a TSX Quantum

Between a TSX Premium PLC and a TSX Quantum PLC using Modbus messaging on the TCP/IP profile, the **TSX WMY 100** module allows:

- a single connection in Modbus client mode,
- a single connection in Modbus server mode.

Example

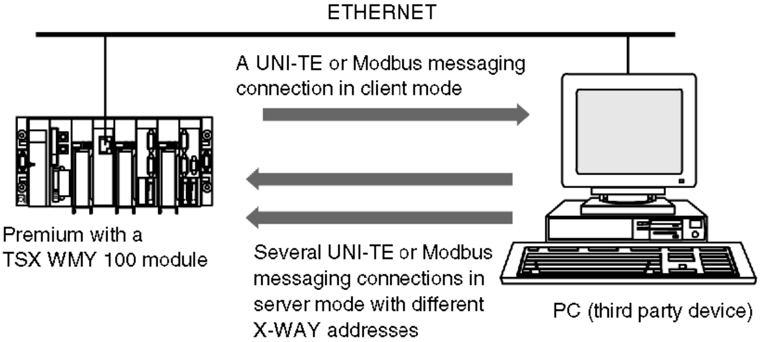

ETHERNET
A Modbus client messaging connection
Premium with a TSX WMY 100 module
A Modbus server messaging connection
Quantum

Connection to a Third Party Device

Between a TSX Premium PLC and a third party device, the **TSX WMY 100** module allows:

- a single connection in UNI-TE or Modbus client mode,
- several connections in UNI-TE or Modbus server mode.

Example

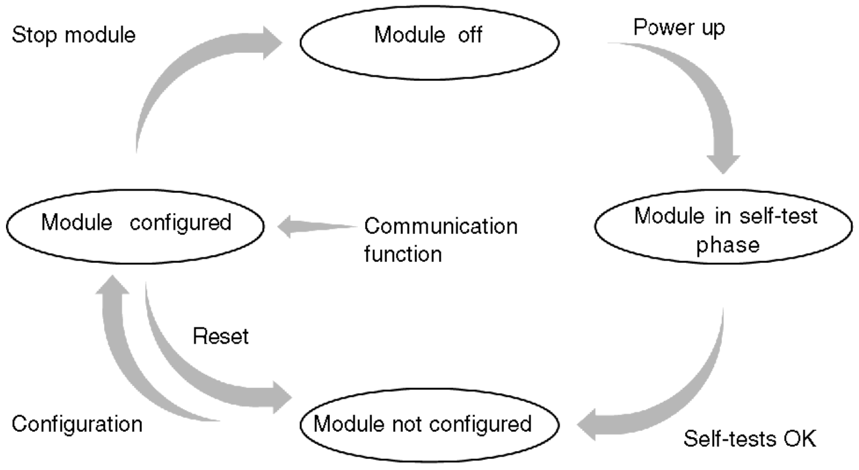

ETHERNET
A UNI-TE or Modbus messaging connection in client mode
Premium with a TSX WMY 100 module
Several UNI-TE or Modbus messaging connections in server mode with different X-WAY addresses
PC (third party device)

Operating modes of the TSX WMY 100 module

At a Glance

The following diagram describes the **TSX WMY 100** HMI module operating modes.

General Diagram



Operation

- After power-up, the module carries out self-testing. During this phase, the STS LED is lit. After the self tests, the RUN LED lights up.
- The module does not operate with a default configuration. This has to be transmitted to it by the local PLC’s PL7 application.
  The configuration values are given in the %KW language object list and in particular provide the IP address and the X-WAY address of the module.
  The configuration is retransmitted upon every PLC restart (warm or cold).
- When the configuration is received, the module resets the current communication to zero before configuring itself (terminates current exchanges, shuts down TCP connections).
  The module is now operational. The RUN and STS LEDs are lit up.
- The IO Scanning function starts when the PLC transfers to RUN mode. It stops when it transfers to STOP mode.

Special Cases

If the module is not configured in the PL7 application (RUN LEDs go out and ERR LEDs flash), it takes the IP address constructed from its MAC address:

085.016.xxx.yyy where xxx and yyy are the last two numbers of the MAC address.

Example

The module’s MAC address is (in hexadecimal): 00 80 F4 01 12 20

In this case the default IP address is (in decimal): 085.016.018.032

The module also transmits BOOTP/DHCP requests to obtain another IP address. These requests are sent until a BOOTP/DHCP server responds or until configured via PL7.

If the module detects a duplicated IP address, the STS LED flashes 4 times during a minute (as does that of the module having the same IP address).

14.2 Configuration

Subject of this Section

This section describes installation of the **TSX WMY 100** HMI module during its configuration using PL7.

What's in this Section?

This section contains the following topics:

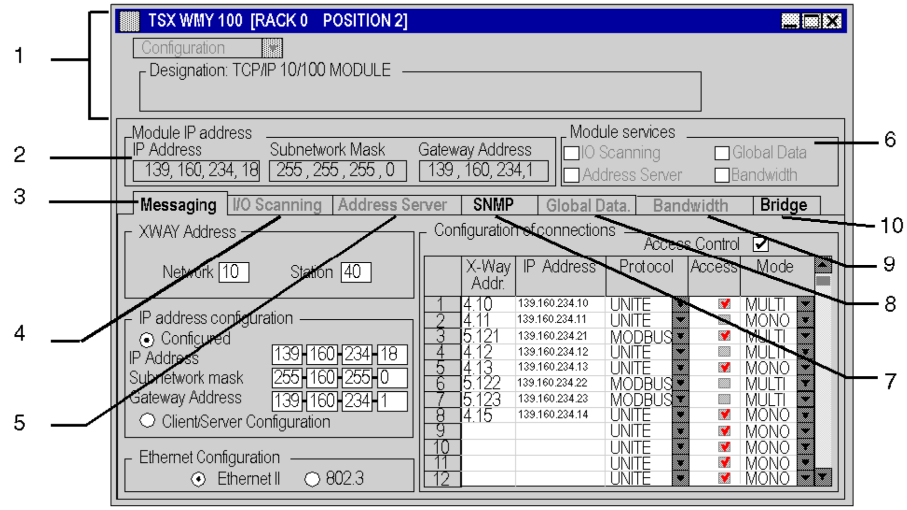
Module configuration screen

At a Glance

This screen, separated into two zones, allows the declaration of the communication channel and the configuration of the necessary parameters for an ETHERNET link.

Illustration

The screen dedicated to ETHERNET communication is displayed as follows:



TSX WMY 100 [RACK 0 POSITION 2]

Configuration

Designation: TCP/IP 10/100 MODULE

Module IP address
IP Address: 139, 160, 234, 18
Subnetwork Mask: 255 , 255 , 255 , 0
Gateway Address: 139 , 160, 234,1

Module services
□ IO Scanning □ Global Data
□ Address Server □ Bandwidth

XWAY Address
Network: 10 Station: 40

IP address configuration
● Configured
IP Address: 139.160.234.18
Subnetwork mask: 255.160.255.0
Gateway Address: 139.160.234.1
○ Client/Server Configuration

Ethernet Configuration
● Ethernet II ○ 802.3

Configuration of connections
Access Control ☑

	X-Way Addr.	IP Address	Protocol	Access	Mode
1	4.10	139.160.234.10	UNITE	✓	MULTI
2	4.11	139.160.234.11	UNITE	□	MONO
3	5.121	139.160.234.21	MODBUS	✓	MULTI
4	4.12	139.160.234.12	UNITE	□	MULTI
5	4.13	139.160.234.13	UNITE	✓	MONO
6	5.122	139.160.234.22	MODBUS	□	MULTI
7	5.123	139.160.234.23	MODBUS	□	MULTI
8	4.15	139.160.234.14	UNITE	✓	MONO
9			UNITE	✓	MONO
10			UNITE	✓	MONO
11			UNITE	✓	MONO
12			UNITE	✓	MONO

Elements and Functions

This table describes the various zones that make up the configuration screen:

Zone	Number	Function
common	1	common part of the communication configuration screens.
specific	2	used to display the module address.
	3	used to configure the TCP/IP services. See *page 465*.
	4	not available on the TSX WMY 100 module.
	5	not available on the TSX WMY 100 module.
	6	used to select the services used by the module.
	7	used to configure of the SNMP service. See *page 472*.
	8	not available on the TSX WMY 100 module.
	9	not available on the TSX WMY 100 module.
	10	allows configuration of the module in Bridge.

Type of Communication According to Connection Configuration

At a Glance

Depending on the connection configuration of the **TSX WMY 100** HMI module, you can carry out messaging:

- in mono-connection
- in multi-connection.

According to the type of protocol and access control management, multi-connection mode requires a specific configuration of the correspondence table.

NOTE: In the following, only examples of multi-connection will be dealt with. Furthermore, the emphasis is placed on communication between the module and a remote PC device containing several applications.

Configuration Rules

Several connections can be configured with the same IP address. They must be defined with the same protocol, the same access rights, and the same connection mode.

In the case of a Modbus connection, only two connections can be configured with the same IP address. Addresses associated with these connections are:

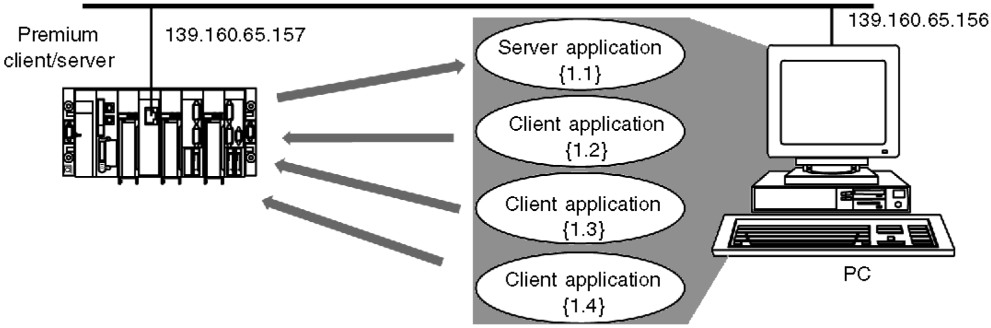
- an X-WAY address (for example: {2.103}) for the Modbus client connection,
- an X-WAY address equal to {x.x} for Modbus server connections.

Multi-connection in UNI-TE Protocol with Access Management

In this example, the PC contains four applications with a unique X-WAY address and one single IP address:

- The module opens a connection and communicates with the server application with the address {1.1}.
- Each of the other client applications with the address {1.2}, {1.3}, {1.4} is authorized to open a connection and communicate with the module.

Exchanges are carried out according to the UNI-TE protocol and the module’s access control service is enabled.



In order to implement this example, you must:

- activate connection management by checking the module’s **Access control** box,
- configure communication from the module to the PC’s server application in the table
  - **X-Way address**: 1.1
  - **IP address**: 139.160.65.156
  - **Protocol**: UNI-TE
  - **Access**: authorized (cell checked)
  - **Mode**: Multi
- configure communication of each of the PC’s client applications to the server module in the table:
  - **X-Way address**: 1.2 and 1.3 and 1.4
  - **IP address**: 139.160.65.156
  - **Protocol**: UNI-TE
  - **Access**: authorized (cell checked)
  - **Mode**: Multi

Multi-connection in UNI-TE protocol without access management

This example is the same as the previous one, except that access control is not configured.

In this case, if the application {1.1} is the only server application, the module’s correspondence table only contains the following line:

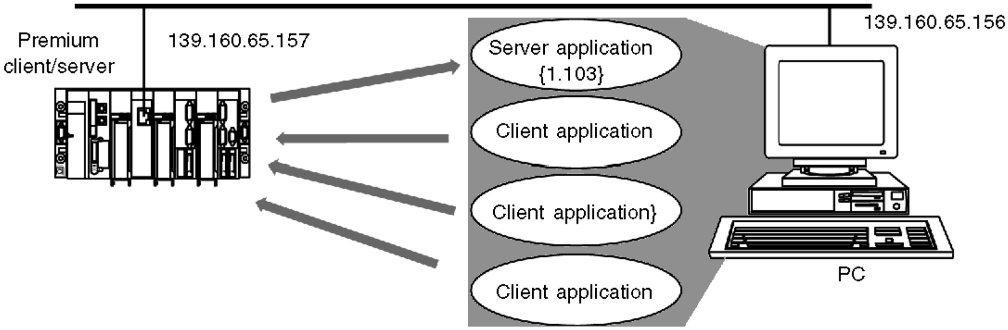
- **X-Way address**: 1.1
- **IP address**: 139.160.65.156
- **Protocol**: UNI-TE
- **Access**: the cell is grayed out
- **Mode**: Multi

Multi-connection in Modbus Protocol with Access Management

In this example, the PC contains four applications. The server application has an imaginary X-Way address and the other client applications do not need an X-Way address:

- The module opens a connection and communicates with the server application with the imaginary address {1.103}.
- Each of the client applications has no address but is authorized to open a connection and communicates with the module.

Exchanges are carried out according to the Modbus protocol and the module's access control service is enabled.



In order to implement this example, you must:

- activate connection management by checking the module's **Access control** box,
- configure communication from the module to the PC's server application in the table
  - **X-Way address**: 1.103 (address between 100 and 163)
  - **IP address**: 139.160.65.156
  - **Protocol**: Modbus
  - **Access**: the cell is grayed out
  - **Mode**: Multi

- configure communication of all the PC’s client applications to the server module in the table:
  - **X-Way address**: x.x (this address allows you to specify that other applications exist for the same IP address)
  - **IP address**: 139.160.65.156
  - **Protocol**: Modbus
  - **Access**: authorized (allows you to specify that other applications are authorized to connect and communicate)
  - **Mode**: Multi

NOTE: If you want to prevent connection by client applications you must configure

- **X-Way address**: x.x (this address allows you to specify that other applications exist for the same IP address)
- **IP address**: 139.160.65.156
- **Protocol**: Modbus
- **Access**: not authorized (cell unchecked)
- **Mode**: Multi

Multi-connection in Modbus protocol without access management

This example is the same as the previous one, except that access control is not configured.

In this case, if the application {1,103} is the only server application, the module’s correspondence table only contains the following line:

- **X-Way address**: 1.103
- **IP address**: 139.160.65.156
- **Protocol**: Modbus
- **Access**: the cell is grayed out
- **Mode**: Multi

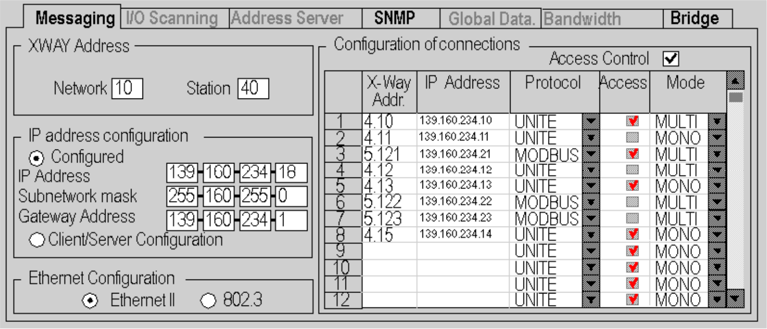
Configuration of TCP/IP Messaging

At a Glance

In order to use the **TSX WMY 100** HMI module to communicate on ETHERNET, it is necessary to adjust the messaging configuration parameters *(see page 467)*.

How to Access the Messaging

The procedure for accessing the configuration parameters for TCP/IP services is as follows.

Step	Action
1	Access the module configuration screen.
2	Select the **Messaging** tab. **Result** 

XWAY Address

Network 10 Station 40

IP address configuration

● Configured

IP Address 139 160 234 18

Subnetwork mask 255 160 255 0

Gateway Address 139 160 234 1

○ Client/Server Configuration

Ethernet Configuration

● Ethernet II ○ 802.3

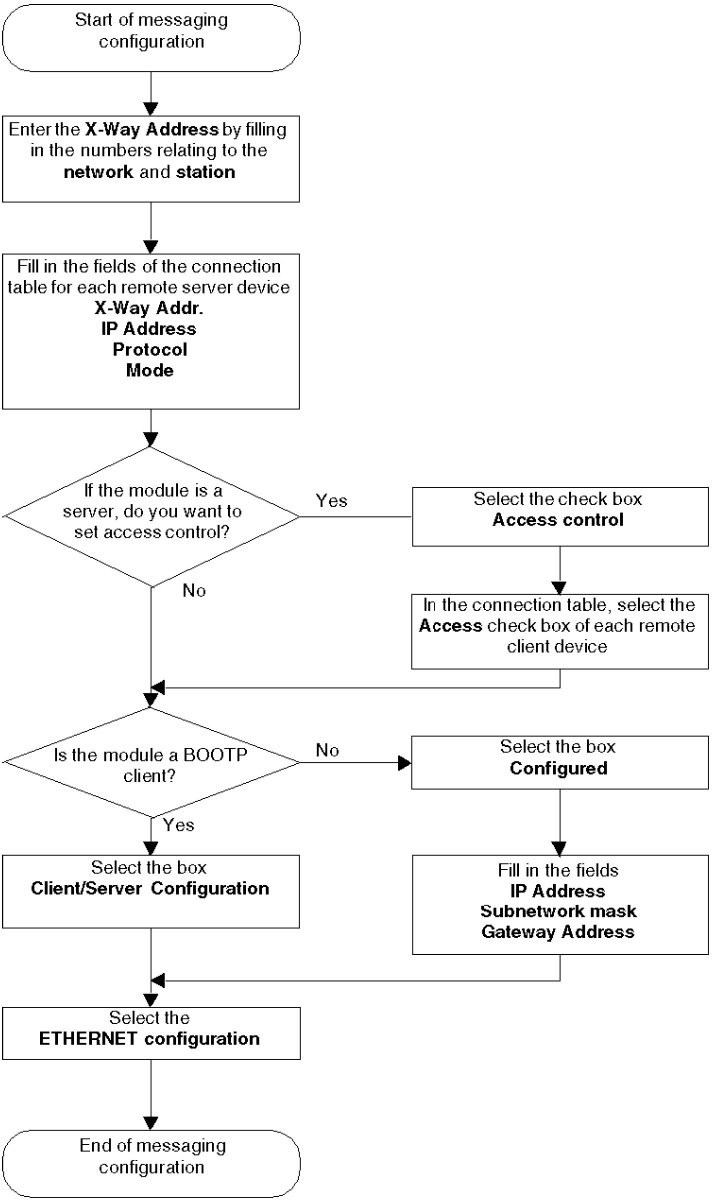
Configuration of connections

Access Control ☑

	X-Way Addr.	IP Address	Protocol	Access	Mode
1	4.10	139.160.234.10	UNITE	☑	MULTI
2	4.11	139.160.234.11	UNITE	□	MONO
3	5.121	139.160.234.21	MODBUS	☑	MULTI
4	4.12	139.160.234.12	UNITE	□	MULTI
5	4.13	139.160.234.13	UNITE	☑	MONO
6	5.122	139.160.234.22	MODBUS	□	MULTI
7	5.123	139.160.234.23	MODBUS	□	MULTI
8	4.15	139.160.234.14	UNITE	☑	MONO
9			UNITE	☑	MONO
10			UNITE	☑	MONO
11			UNITE	☑	MONO
12			UNITE	☑	MONO

How to Configure the Messaging

The following procedure gives the configuration principle.


Start of messaging configuration
Enter the **X-Way Address** by filling in the numbers relating to the **network** and **station**
Fill in the fields of the connection table for each remote server device **X-Way Addr.** **IP Address** **Protocol** **Mode**
If the module is a server, do you want to set access control?
Yes
Select the check box **Access control**
In the connection table, select the **Access** check box of each remote client device
No
Is the module a BOOTP client?
No
Select the box **Configured**
Fill in the fields **IP Address** **Subnetwork mask** **Gateway Address**
Yes
Select the box **Client/Server Configuration**
Select the **ETHERNET configuration**
End of messaging configuration

Configuration Parameters Linked to TCP/IP Services

At a Glance

The **TSX WMY 100** HMI module has configuration parameters linked to TCP/IP services.

Availability of Parameters

The following table shows, for the**TSX WMY 100** HMI module, the parameters that can be accessed from the configuration screen and the values that they can be assigned.

Parameters			**TSX WMY 100**
X-WAY Address			By PL7
IP Address	Default IP address		X
	Configured	IP Address	X
		Subnetwork mask	X
		Gateway address	X
	Client/Server Configuration		X
Connections	Connections which can be opened		-
	Access control		X
	Function table	X-WAY Address	X
		IP Address	X
		Protocol	UNI-TE or Modbus
		Access	X
		Mode	Mono- or Multi-connection
ETHERNET	ETHERNET II		X
	802.3		X
Legends:			
X	Accessible		
-	Not accessible		

Configuration Parameters for IP Addresses

Presentation

This zone allows the IP address of a module to be defined in three distinct ways:

- by manual configuration,
- or by using a configuration supplied by a BOOTP/DHCP server device.

Configured

Selecting the **Configured** field allows manual configuration according to your own requirements:

- module IP address,
- the subnetwork mask, which defines the part allocated to the subnetwork identifier in the IP address,
- the IP address of the default gateway, to which messages for other networks are transmitted.

NOTE: If the module is connected to an existing TCP/IP network, the IP addresses are administered globally, therefore the IP parameters must be configured. Otherwise there is a risk of disruption on the existing network caused by possible double allocation of the IP addresses.

⚠ **WARNING**

UNINTENTIONAL OPERATION — DUPLICATE IP ADDRESS

- Use a method to assign a unique IP address to each device on the network.
- Obtain your IP addresses from your system administrator to avoid the possibility of duplicate addresses.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Client/Server Configuration

Selecting the **Client/Server configuration** field allows the module’s IP address to be configured from a remote device acting as a BOOTP server.

In this instance, the **IP address**, **Subnetwork mask** and **Gateway Address** fields are not significant and are grayed out.

Connection Configuration Parameters

At a Glance

This field enables you to :

- configure the number of connections that can be opened by the module,
- activate an access check service,
- list the remote devices which can be connected to the module according to the communication protocol.

Connections which can be opened

The **Connections that can be opened** field gives the maximum number of devices that can be connected in parallel to the module:

- The default value is 8 connections.
- The value is between 1 and 32 connections.

NOTE: You are recommended to set this parameter to the useful value in order to optimize the communication resources for each connection point.

Access control

The **Access check box** is used to activate or deactivate the check on remote devices that want to open a TCP connection to the module.

- If the box is checked, access check management is activated and the **Access column** in the table is de-grayed (active).
  If the module is operating in server mode, only the remote devices selected by the **Access box** in the table are authorized to be connected as a client and then to communicate.
- If the box is unchecked, check management does not operate and the **Access column** in the table is grayed out (not active).
  If the module is operating in server mode, third party remote devices can be connected as a client and then communicate with the module without having to be declared in the table.

NOTE: The access check only works on the TCP/IP profile and is used for the module to operate in server mode.

Function table

This table enables you:

- to list the remote devices with which the local module wishes to open a TCP connection and then transmit (module operating in client mode),
- to provide a cross reference between the **X-WAY address** {network, station} and the **IP address**,

  - the network number must be less than or equal to 127,
  - the station number is between 0 and 63 for a connection in UNITE or between 100 and 163 for a connection in Modbus.

- to specify the communication protocol for each remote device during the connection using a scrolling menu:
  - **UNI-TE** (default value),
  - **Modbus**,

- when managing access checks, to nominate the remote devices authorized to open a TCP connection and then to send to the local module (module operating in server mode),
- for each remote IP address, to choose the **Connection mode** using a scrolling menu :
  - **Monoconnection** : the module only allows one connection with the same remote IP address,
  - **Multiconnection** : the module allows only one connection with the same remote IP address in client mode and several connections with the same remote IP address in server mode.

Ethernet Configuration Parameters

At a Glance

This zone enables you to define the format of the Ethernet frame via TCP/IP:

- Ethernet II format: as per RFC 894 (Most commonly used format),
- 802.3 format: as per RFC 1042. This format should be selected when remote devices use this format.

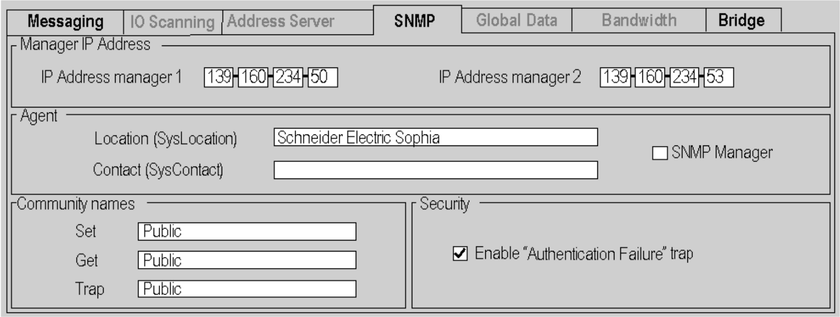
Configuring the SNMP Service

At a Glance

In order to use the **TSX WMY 100** HMI module as an SNMP agent, it is necessary to adjust the configuration parameters of the SNMP service *(see page 474)*.

How to Access the SNMP Service

The procedure for accessing the configuration parameters of the SNMP service is as follows.

Step	Action
1	Access the module configuration screen.
2	Select the **SNMP** tab. **Result**  Messaging IO Scanning Address Server SNMP Global Data Bandwidth Bridge Manager IP Address IP Address manager 1 139.160.234.50 IP Address manager 2 139.160.234.53 Agent Location (SysLocation) Schneider Electric Sophia Contact (SysContact) □ SNMP Manager Community names Set Public Get Public Trap Public Security ☑ Enable "Authentication Failure" trap

How to Configure SNMP

The following procedure gives the configuration principle for the SNMP service.

Step	Action
1	Enter the SNMP manager addresses: ● **Manager 1 IP addresses** ● **Manager 2 IP addresses**
2	Fill in the fields: ● **Location (SysLocation)** ● **Contact (SysLocation)**. Or alternatively check the **SNMP Manager** box to indicate that the information will be completed by the SNMP Manager.
3	If you want to define access rights, fill in the community names: ● **Set** ● **Get** ● **Trap**
4	If you want to activate transmission of an event to the module, check the **Activate "Authentication Failure" trap** box.

Description of SNMP Service

Presentation

Parameters linked to the SNMP module are divided into 4 categories:

- the IP addresses of the SNMP module administrative devices,
- the agents,
- the community names,
- security.

NOTE: Only 7-bit ASCII characters can be used in the character string entry fields.

Administrators IP Addresses

This zone allows the completion of the SNMP administrators IP addresses. The modules authorize a maximum of two administrators.

These addresses are used during possible transmission of events (TRAP).

Agent

This zone allows the localization and identification of an agent from the SNMP administrator.

It comprises two fields:

- The **Location (SysLocation)** field: indicates the physical location of the device (32 characters maximum).
- The **Contact (SysLocation)** field: indicates the personnel to contact for device management and the method of contact (32 characters maximum).
- It is possible to select information assignment by a "SNMP Manager" tool for network management, rather than by PL7 software. To do this, check the **SNMP Manager** box.

Community Name

This zone allows the definition of a community name by service family Set, Get and Trap.

It comprises three fields:

- The **Set** field defines the community name for the Set service (16 characters maximum). The default value of the field is **Public**.
- The **Get** field defines the community name for the Get service (16 characters maximum). The default value of the field is **Public**.
- The **Trap** field defines the community name for the Trap service (16 characters maximum). The default value of the field is **Public**.

The purpose of these fields is to define the access rights for SNMP agent MIB objects (local module) in relation to requests transmitted by the administrator.

Example

If the administrator transmits a SetRequest request with the community name **Test** and the module has the community name **Public**, the request will not be executed.

Security

This zone contains the **Enable "Authentication Failure" trap** check box.

Enabling this box allows the transmission of an event (TRAP) of an unsuccessful authentication to be confirmed from the SNMP agent to the administrator that originally posted the request.

In this way, the agent warns the administrator that the request has been refused following an unsuccessful identification (community name configured in the administrator is different to the one configured in the agent).

NOTE:

- The SNMP administrator is able to modify the value of certain parameters, which can be configured by PL7 (Enabling "Authentication failure", Location, Contact. etc.)
- When there is a cold start, warm restart or application download, the values initially configured in PL7 are then restored.

Configuration Parameters Linked to the SNMP Service

At a Glance

The **TSX WMY 100** HMI module has configuration parameters linked to the SNMP service.

Availability of Parameters

The following table shows, for each ETHERNET module, the parameters that can be accessed from the configuration screen.

Parameters		**TSX WMY 100**
Manager IP Address		X
Agent	Location (SysLocation)	X
	Contact (SysContact)	X
	SNMP Manager	X
Community names	Set	X
	Get	X
	Trap	X
Security	Enable "Authentication Failure" trap	X
Legends:		
X	Accessible	
-	Not accessible	

Configuration of the Bridge Function

Presentation

TheTSX WMY 100HMI module may be used as an X-Way bridge station for transparent communication between networks.

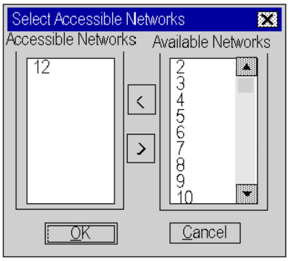
How to Access the Bridge Function

The procedure for accessing the configuration parameters for the bridge function is as follows.

Step	Action
1	Access the module configuration screen.
2	Select the **Bridge** tab.

Configuring the Bridge Function

The following procedure allows you to configure the **TSX WMY 100** HMI module of the station as a bridge.

Step	Action
1	Double click on the zone in reverse video. **Result** The **Select Accessible Networks** window appears.  Select Accessible Networks Accessible Networks Available Networks 12 2 3 4 5 6 7 8 9 10 < > OK Cancel If the list of accessible networks for the module is empty, the window appears automatically (without double clicking).
2	Double click on the network number in the **Available Networks** scroll down list. **Result** The network number is assigned in the **Accessible Networks** scroll down list.
3	Repeat operation 3 as many times as needed to define all networks that can be accessed by the module. Then go to step 5.
4	Confirm the selection by clicking on the **OK** button.
5	Confirm the bridge configuration by clicking on the **Save** button.

How to Delete the Bridge Function

The module bridge function can be deleted.

Step	Action
1	Access the **Bridge Configuration** screen.
2	Click on the **Delete** button.
3	Confirm the deletion by clicking on the **Save** button.

How to Delete Access to a Network

Access to a single network can be deleted.

Step	Action
1	Access the **Bridge Configuration** screen.
2	Double click on the zone in reverse video.
3	Double click on the network number in the **Accessible Networks** scroll down list. **Result** The network number is reassigned in the **Available Networks** scroll down list.
4	Confirm the selection by clicking on the **OK** button.
5	Confirm the bridge configuration by clicking on the **Save** button.

14.3 Debugging

Subject of this Section

This section describes installation of the **TSX WMY 100** HMI module running PL7 during debugging.

What’s in this Section?

This section contains the following topics:

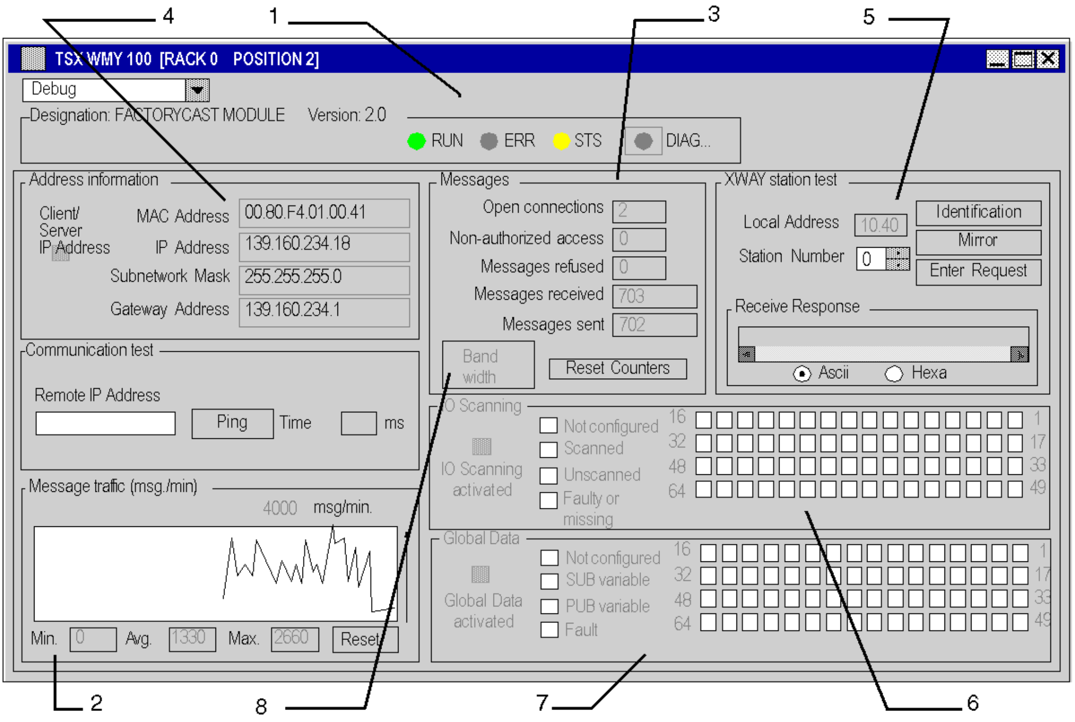
Module Debug Screen

At a Glance

This screen, separated into six zones, is used for debugging through an ETHERNET link.

Illustration

The screen dedicated to ETHERNET communication is displayed as follows:



TSX WMY 100 [RACK 0 POSITION 2]

Debug

Designation: FACTORYCAST MODULE Version: 2.0

● RUN ● ERR ● STS ● DIAG...

Address information

Client/ Server IP Address

MAC Address 00.80.F4.01.00.41

IP Address 139.160.234.18

Subnetwork Mask 255.255.255.0

Gateway Address 139.160.234.1

Messages

Open connections 2

Non-authorized access 0

Messages refused 0

Messages received 703

Messages sent 702

XWAY station test

Local Address 10.40

Station Number 0

Receive Response

● Ascii ○ Hexa

Communication test

Remote IP Address

Time ms

Message trafic (msg./min)

4000 msg/min.

Min. 0 Avg. 1330 Max. 2660

IO Scanning

□ Not configured

□ Scanned

□ Unscanned

□ Faulty or missing

Global Data

□ Not configured

□ SUB variable

□ PUB variable

□ Fault

Elements and Functions

This table describes the different zones that make up the debug screen:

Number	Zone	Function
1	common	common part of the communication debug screens.
2	Message traffic	allows the graphical display of the number of messages processed by the module:
3	Messages	allows the display of the number of unacknowledged or refused messages and connections.
4	TCP/IP services	allows: ● display of TCP/IP services configuration, ● communication testing of the TCP/IP profile.
5	X-WAY test station	allows UNI-TE communication testing on the TCP/IP profile.
6	IO Scanning	not accessible on this module.
7	Global Data.	not accessible on this module.
8	Bandwidth	not accessible on this module.

General debugging parameters

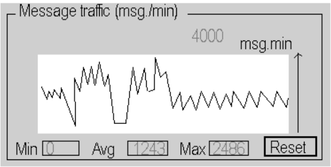
At a Glance

The general debugging parameters are grouped together in two windows:

- the **Message traffic** window
- the **Messages** window.

Message traffic

The window is displayed as below:



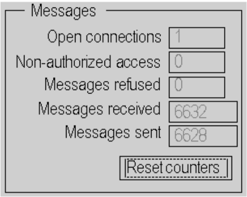
Min 0 Avg 1243 Max 2486

It shows the number of messages processed by the module per minute (transmission and reception) in graphic form. To provide a better display, the scale is automatically adapted according to change in flow.

The **Reset** button sets the three counters **Min.**, **Avg.** and **Max.** to zero.

Messages

The window is displayed as below:



Open connections	1
Non-authorized access	0
Messages refused	0
Messages received	6632
Messages sent	6628

This window is related to TCP/IP messaging:

- number of TCP connections open
- number of non-authorized accesses
- number of messages refused, received and sent

You can reset these counters to zero by clicking on the **Reset counter** button.

Debugging parameters for TCP/IP services

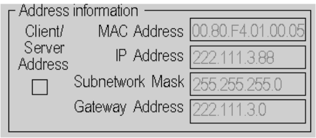
At a Glance

The debugging parameters for the TCP/IP services are grouped together in two windows:

- the **Address information** window,
- the **Communication test** window.

Address information

The window is displayed as below:



Address information

Client/ Server Address □

MAC Address: 00.80.F4.01.00.05

IP Address: 222.111.3.88

Subnetwork Mask: 255.255.255.0

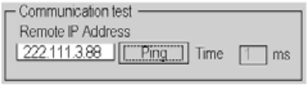
Gateway Address: 222.111.3.0

It specifies the configuration data of the IP address:

- **MAC address**: unique fixed address for a module
- **IP Address**
- **Subnetwork mask**
- **Gateway Address**: address of the gateway

Communication test

The window is displayed as below:



Communication test

Remote IP Address: 222.111.3.88

Time: 1 ms

This window is used to test IP communication towards another station.

How to Test TCP/IP communication with the Ping request

At a Glance

This page indicates the procedure for testing TCP/IP communication from a **Ping** request.

Procedure

The following procedure is used to send the Ping request and therefore to test the correct routing of information between two devices.

Step	Action
1	Enter the address of the station to be queried using the **Remote IP address** field.
2	Press the **Ping** button. **Result** The response appears in the **Time** field: Time 1 ms The returned time corresponds to the time it takes the message to be sent and return in ms.

Response type

The following table groups together the various types of possible responses to the Ping request.

If the response is	then
positive	The windows records the time it takes the message to be sent and return in ms.
negative	a window with the message **Exchange incorrect** specifies that the remote device was not reached in the network architecture.

Available requests for testing a communication channel

Presentation

This page describes the various options for testing a communication channel from the debug screen.

Test conditions

When a request is sent to a station that is not connected, a message appears.

The test applies to stations belonging to the same local area network.

Available requests

The **X-Way station test** window allows the following requests:

- **Identification**: causes the Identification request to be sent to the designated remote station,
- **Mirror**: is used to send a Mirror request to the designated station. If this function is chosen, a screen appears allowing you to select the length of the string of characters to be sent (maximum of 80 characters). The PLC then sends this string of characters (ABCD, etc.) to the destination device. This device automatically returns the string of characters received to the sender,
- **Enter Request** : is used to send a UNI-TE request, other than those sent by the command buttons, to the designated station. If this function is chosen, a screen appears allowing you to select the parameters specific to the request (the request code must be coded in hexadecimal).

NOTE: The first two requests are sent to the UNI-TE server of the TSX WMY module. The third is sent to the server of the processor.

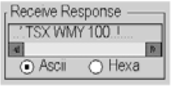
How to Test a Channel with the Identification and Mirror requests

At a Glance

This page indicates the procedure for testing a communication channel using the Identification and Mirror requests.

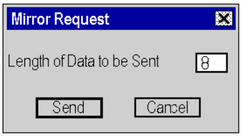
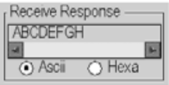
How to Identify a Station

The following procedure is used to identify a designated station.

Step	Actions
1	Select the address of the station to be queried using the **Station number** field.
2	Press the **Identification** button. **Result** The response appears in the **Receive Response** window:  Receive Response ..' TSX WMY 100.!.... ● Ascii ○ Hexa

How to Send the Mirror Request

The following procedure is used to send the Mirror request and therefore to test the correct routing of information between two devices.

Step	Action
1	Select the address of the station to be queried using the **Station** field.
2	Press the **Mirror** button. **Result** The following window appears:  Mirror Request Length of Data to be Sent: 8
3	Enter the length of the data to be sent (maximum 80 characters).
4	Press the **Send** button. **Result** The response appears in the **Receive Response** window:  Receive Response ABCDEFGH ● Ascii ○ Hexa The response contains: - the character string ABCDEFGH corresponding to a length of data sent of 8.

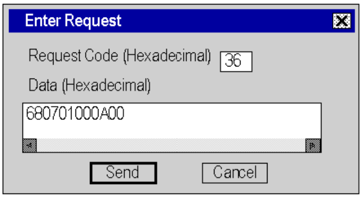
How to test a channel with requests

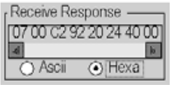
At a Glance

This page indicates the procedure for testing a communication channel from the debug screen through different requests.

How to Send a Request

The following procedure is used to send a request, other than those sent by the command buttons, to a designated station. In this example, the sent request enables 10 words to be read (from %MW1 to %MW10).

Step	Action
1	Select the address of the station to be queried using the **Station** field.
2	Press the **Enter Request** button. **Result** The following window appears:  Enter Request Request Code (Hexadecimal) 36 Data (Hexadecimal) 680701000A00 The data transmitted in this example is coded on 6 bytes.
3	Enter the function code (coded in hexadecimal on a byte), corresponding to the request that you want to send. In this example, the read request code is 16#36.
4	Enter the data to be transmitted by coding each data item in hexadecimal. The data is then entered one after the other with no spaces between data items. When the data is coded on a word, the most significant bytes and the least significant bytes are inverted. In this example, the data is as follows: ● 16#68: on a byte, define the segment (internal data), ● 16#07: on a byte, define the type of object (words), ● 16#0100: on a word, define the first word to be read, ● 16#0A00: on a word, define the number of words to be read.

Step	Action
5	Press the **Send** button. **Result** The response appears in the **Receive Response** window:  Receive Response 07 00 C2 92 20 24 40 00 ○ Ascii ◉ Hexa The response for the example contains the data on 21 bytes. ● 16#07: corresponds to the type of object (words), ● 16#00C2: corresponds to the value of the first word (the most significant bytes and the least significant bytes are inverted; its value is 16#C200), ● ...

14.4 Language objects associated with the TSX WMY 100 module

Subject of this Section

This section presents the different language objects specific to the **TSX WMY 100** HMI module.

What's in this Section?

This section contains the following topics:

Implicit Exchange Language Objects

Presentation

This page describes all the implicit exchange *(see page 359)* language objects for ETHERNET communication with the TSX WMY 100 module that can be displayed or modified by the application program.

Bit Objects

The table below shows the various implicit exchange bit objects.

Object (1)	**Function**	**Meaning**
%Ixy.MOD.ERR	Module detected error bit	This bit set to 1 indicates a module anomaly (at least one of the channels is inoperative, for instance)
%Ixy.i.ERR	Channel detected error bit	This bit set to 1 indicates a detected line fault.
Legend		
(1)	Address xy.i ● x : corresponds to the rack number ● y : corresponds to the module number ● i : corresponds to the channel number	

Word Objects

The table below shows the various implicit exchange word objects.

Object (1)	**Function**	**Meaning**
%IWxy.i.0 to %IWxy.i.9	Reserved	-
Legend		
(1)	Address xy.i ● x : corresponds to the rack number ● y : corresponds to the module number ● i : corresponds to the channel number	

Explicit Exchange Language Objects

Presentation

This page describes all the explicit exchange *(see page 364)* language objects for ETHERNET communication with the TSX WMY 100 module that can be displayed or modified by the application program.

Internal Words

The following table describes the internal words:

Object (1)	**Function**	**Meaning**
%MWxy.MOD.2	Module status	● x0 = 1: inoperative module ● x1 = 1: functional detected error (anomaly between the processor and the module, adjustment or configuration anomaly, etc.) ● x2 = 1: terminal block detected fault (not connected) ● x3 = 1: self-tests running ● x4 = 1: reserved ● x5 = 1: hardware or software configuration anomaly (the module present is not that declared in the configuration, the sub-modules are not compatible) ● x6 = 1: module missing ● x7 = 1: detected error in one of the sub-modules
%MWxy.i.2	Channel standard status	● x0 to x3 = 0: reserved ● x4 = 1: module detected error or self-tests running ● x5 and x6 = 0: reserved ● x7 = 1: application detected error (check the configuration)
%MWxy.i.3	Specific channel status	Byte 0: ● x0 = 1: if X-WAY bridge (0 if not bridge) Byte 1: number of TCP connections open
%MWxy.i.4	Rejected requests counters	Number of connection requests refused
%MWxy.i.5	Rejected messages counters	Number of messages refused
%MWxy.i.6	Network/station address	Byte 0: station number Byte 1: network number
%MWxy.i.7 and %Mwxy.i.8	Message counters on the link layer	Number of messages sent on the link layer

Object (1)	**Function**	**Meaning**
%MWxy.i.9 and %Mwxy.i.10	Message counters on the link layer	Number of messages received on the link layer
%MWxy.i.11	reserved	-
%MWxy.i.12	reserved	-
%MWxy.i.13	reserved	-
%MWxy.i.14	reserved	-
%Mwxy.i.15	Module loading rate	Byte 0: ● Messaging loading rate Byte 1: ● Other services loading rate
Legend		
(1)	Address xy.i ● x: corresponds to the rack number ● y: corresponds to the module number ● i: corresponds to the channel number	

Explicit exchange management and report

At a Glance

This page describes all the language objects that manage explicit exchanges.

Word objects

The table below shows the different word objects for the management of explicit exchanges.

Object (1)	Function	Meaning
%MWxy.MOD.0	Module exchanges in progress	● x0 = 1: status reading in progress ● x1 = 1: sending of command parameters to the communication module ● x2 = 1: sending of adjustment parameters to the communication module
%MWxy.MOD.1	Module report	● x1 = 0: command parameters received and accepted by the module ● x2 = 0: adjustment parameters received and accepted by the module
%MWxy.i.0	Channel exchanges in progress	● x0 = 1: status reading in progress ● x1 = 1: sending of command parameters to the communication channel ● x2 = 1: sending of adjustment parameters to the communication channel
%MWxy.i.1	Channel report	● x1 = 0: command parameters received and accepted by the communication channel ● x2 = 0: adjustment parameters received and accepted by the communication channel
Legend		
(1)	Address xy.i ● x : corresponds to the rack number ● y : corresponds to the module number ● i : corresponds to the channel number	

Language Objects Associated with Configuration

At a Glance

This page describes all the configuration language objects for ETHERNET communication with the TSX WMY 100 module that can be displayed by the application program.

Internal Constants

The following table describes the internal constants:

Object	Function	Meaning
%KWxy.i.0	Type	Byte 0 = 11 for ETHERNET communication Byte 1: reserved
%KWxy.i.1	Physical layer	Byte 0: fixed at 16#01 (corresponds to GPX2) Byte 1: reserved
%KWxy.i.2	Reserved	-
%KWxy.i.3	Reserved	-
%KWxy.i.4	Reserved	-
%KWxy.i.5	X-Way network address	Byte 0: network number (0 by default) Byte 1: reserved
%KWxy.i.6	Type of ETHERNET driver for TCP/IP	Byte 0: fixed at 16#01 (corresponds to RJ45) ● = 16#01: RJ45 Byte 1: ● = 16#00: ETHERNET II (default value) ● = 16#01: 802.3
%KWxy.i.7	TCP/IP configuration: address type	Byte 0: reserved Byte 1: inherited address ● = 16#01: from the application-specific function ● = 16#03: from a server
%KWxy.i.8 and %KWxy.i.9	Local IP address	Example with the address 139.160.65.109 Byte 0 = 109 Byte 1 = 65 Byte 2 = 160 Byte 3 = 139
%KWxy.i.10 and %KWxy.i.11	IP address of gateway	Example with the address 139.160.65.1 Byte 0 = 1 Byte 1 = 65 Byte 2 = 160 Byte 3 = 139

Object	Function	Meaning
%KWxy.i.12 and %KWxy.i.13	Subnetwork mask	Example with the address 255.255.255.0 Byte 0 = 0 (least significant) Byte 1 = 255 Byte 2 = 255 Byte 3 = 255 (most significant)
%KWxy.i.14	Services used	Byte 0: ● x0 = 1: IO Scanning service is used ● x1 = 1: BOOTP server service is used ● x2 = 1: global data service is used ● x3 = 1: the bandwidth adjustment service is used Byte 1: reserved
%KWxy.i.15	TCP services used	Byte 0: ● x0 = 1: Modbus communication on TCP/IP is used ● x1 = 1: access control is activated Byte 1: reserved

14.5 Questions/Answers

Questions/Answers

General

Below you will find a list of the most frequently asked questions and answers regarding communication by Ethernet network.

What should I do if the TSX WMY 100 module displays the LED status "Not configured" (Flashing ERR)?

- Check that it is configured in PL7.
- Check that there is no other module with the same X-WAY network number in the PLC configuration.
- If the module is configured in "Client/Server address", make sure that a BOOTP server is operational and accessible.

What should I do if the module displays a correct LED status (STS=RUN=ON) but does not respond to requests?

Try a Ping command to this IP address, from a device on the same network:

- No response:
  - Check the network configuration: wiring, "hubs" status, IP addresses, masks and gateways if there are routers in the configuration.
- Correct response to the Ping:
  - Check the module’s X_WAY address.
  - Check whether access control is activated or not.
  - Try to connect to the Rack Viewer page by http. If this connection works, check the X-WAY address of the client application.

Can I change the MAC address of the module?

No. This is fixed for every Ethernet device to guarantee its uniqueness.

How can I find out the speed with which the module communicates?

Via an internet browser, by connecting to the Ethernet Module Statistics page.

What should I do if the messaging EFs do not get a response?

- Check that the X-WAY destination address is part of the IP/X-WAY connections table.
- Check that the configuration of the destination module does not prohibit this connection (access control).
- In debug mode, try the "Ping" command towards the IP address of the destination PLC, then try an "Identification" request towards the destination address, if it is part of the same X-Way network.

What should I do if I receive the 0xFF12 refusal code on the messaging EFs?

This code indicates that the module is full. Reduce the transition rate.

What should I do if the READ_ASYNC, WRITE_ASYN messaging EFs do not get a response (time out)?

Check that the "mast" tasks of the client and server PLCs are configured to "periodic", with a sufficient period to ensure that the PLCs are not overrun.

When should I configure my connections to "Mono-Connection"?

When this connection has to be established to a TSX ETY 110, MODBUS protocol.

The destination PLC of my messages is a bridge. Which Xway destination address should I configure in the connections table?

The X-WAY address of the first communication module of the PLC bridge (it is the address of the processor’s PCMCIA card, if there is one present).

In the module configuration screens, the last entry inserted into one of the grids has not been taken into account at the validation. Why?

You need to remove the focus (the scale indicator) from the last cell of the grid in which new data has been entered in order to take this modification into account.

Why is the Bridge tab grayed out when I use a TSX / PCX processor of type 571xx and 572xx?

These processors do not manage bridge data.

Why is the Bridge tab grayed out when I have already changed the position of the processor?

After this type of modification, you have to validate the new hardware configuration to access and modify the bridge data managed globally at the level of the processor.

Appendices

Overview

This is the appendice for FactoryCast HMI. It describes SOAP services.

What’s in this Appendix?

The appendix contains the following chapters:

SOAP Web Services

A

Purpose of this chapter

This chapter describes the various SOAP Web services implemented in FactoryCast Web servers. SOAP Web services are fully compliant with the W3C WS-I Web services standards. They provide a new efficient and standard means of making controller devices interact directly with IT/management level applications.

⚠ WARNING

LOSS OF PROCESS ALERTS AND CONTROL

- Do not rely upon the SOAP Web Services to retrieve and display or control events that could affect human, material or equipment integrity.
- You must provide an appropriate and independent protection via your own application or process.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Web services are based on standards such as:

- SOAP (Single Object Access Protocol), the exchange protocol carried out via the HTTP (HyperText Transfer Protocol) channel.
- WSDL (Web Services Description Language), in XML format.
- XML (eXtensible Markup Language), the universal data exchange standard.

FactoryCast SOAP Web services act as SOAP server interfaces. They allow developers to easily design client applications that can exchange data directly with FactoryCast Web servers. Applications such as Microsoft.NET, SQL Server, Microsoft Office, IBM (WebSphere), SUN (Java, Eclipse), Lotus, Oracle, SAP, MES, ERP and so forth can be interfaced directly with FactoryCast using SOAP Web services.

Three types of web services are provided in FactoryCast modules as SOAP server interfaces:

- ModbusXMLDA: Web service to implement data access to Modbus variables
- SymbolicXMLDA: Web service to implement Symbolic data access
- ExtendedSymbolicXMLDA: Web service to extend Symbolic data access to Unity Pro.

The Web services provided by FactoryCast are compatible with the WS-I basic profile 1.1.

NOTE: For more information on Web services, please refer to specialized publications on the subject.

What's in this Chapter?

This chapter contains the following topics:

Principles for Designing a SOAP Client Interface

Overview

An interface enables a SOAP client application to communicate directly with a FactoryCast Web server module. Exchanges are initiated by the SOAP client application. The FactoryCast server responds to these requests.

A SOAP Client Interface

The following table describes the process of designing a SOAP client interface:

Stage	Action
1	Create the client application: The development environment (for example, Visual Studio.net) connects to a FactoryCast Web server module where it can access a list of available Web services. The Web server returns descriptions of the requested services as WSDL objects.
2	Develop the client application: The developer integrates the Web service APIs using the code retrieved in the previous stage as a Web reference and generates the client application.
3	Execute the client application: In run mode, the client application communicates in real time with the FactoryCast Web server module using the SOAP protocol.

ModBusXMLDA SOAP Services

Overview

The ModbusXMLDA Web Service is implemented to provide data access to Modbus variables using a syntax similar to the Modbus protocol.

This Web service offers the following APIs:

- ReadDeviceIdentification,
- ReadMultipleRegisters,
- WriteMultipleRegisters,
- ReadCoils,
- WriteMultipleCoils,
- ReadInt32,
- WriteInt32.

Data amounts for API parameters

This table presents the maximum amount of data that can be used for API parameters:

APIs	**BMX NOE 0110**
ReadMultipleRegisters	124 (Register)
WriteMultipleRegisters	96 (Register)
ReadCoils	255 (Coils)
WriteMultipleCoils	254 (Coils)
ReadInt32	61 (Int32)
WriteInt32	48 (Int32)

ReadDevice-Identification

The ReadDeviceIdentification request returns the entire device identification of the destination specified by the ID provided in the request.

This API is mapped on Modbus Function Code 43 14.

Parameters

Input: int UnitID

Output: string VendorName, ProductCode, MajorMinorRevision, VendorURL, ProductName, ModelName, UserApplicationName, TRImplementationClass, TRCommunicationServices

Example

The following is an example of a SOAP request and its response:

Request

```
POST /ws/ModbusXmlDa
HTTP/1.0 Host: 139.160.65.83:8080
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/ModbusXmlDa/ReadDeviceIdentification"
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <ReadDeviceIdentification xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
       <UnitID>int</UnitID>
     </ReadDeviceIdentification>
   </soap:Body>
 </soap:Envelope>
```

Response

```
HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length <?xml version="1.0" encoding="utf-8"
?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <ReadDeviceIdentificationResponse
xmlns="http://www.schneider-electric.com/ws/ModbusXmlDa/">
       <ReadDeviceIdentificationResult>
         <DeviceIdentification>
                 <VendorName>string</VendorName>
                 <ProductCode>string</ProductCode>
           <MajorMinorRevision>string</MajorMinorRevision>
           <VendorUrl>string</VendorUrl>
           <ProductName>string</ProductName>
           <ModelName>string</ModelName>
           <UserApplicationName>string</UserApplicationName>
           <TRImplementationClass>string</TRImplementa-
tionClass>
           <TRCommunicationServices>string</TRCommunication-
Services>
         </DeviceIdentification>
       </ReadDeviceIdentificationResult>
     </ReadDeviceIdentificationResponse>
   </soap:Body>
</soap:Envelope>
```

ReadMultiple-Registers

This is the ReadMultipleRegisters API used to implement a read multiple registers action.

The request provides three parameters:

- the destination of the read request,
- the starting address,
- the number of values to read (base index is provided by starting address).

In response, the list of values is provided.

This API is mapped on Modbus Function Code 03.

Parameters

Input: int UnitID, Address, Quantity

Output: array of int Result

Example

The following is an example of a SOAP request and its response:

Request

```
POST /ws/ModbusXmlDa
 HTTP/1.0 Host: 139.160.65.83:8080
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/ModbusXmlDa/ReadMultipleRegisters"
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <ReadMultipleRegisters xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
       <UnitID>int</UnitID>
       <Address>int</Address>
       <Quantity>int</Quantity>
     </ReadMultipleRegisters>
   </soap:Body>
 </soap:Envelope>
```

Response

```
HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length <?xml version="1.0" encoding="utf-8" ?>
 <soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <ReadMultipleRegistersResponse
xmlns="http://www.schneider-electric.com/ws/ModbusXmlDa/">
       <ReadMultipleRegistersResult>
         <int>int</int>
         <int>int</int>
       </ReadMultipleRegistersResult>
     </ReadMultipleRegistersResponse>
   </soap:Body>
 </soap:Envelope>
```

WriteMultiple-Registers

This is the WriteMultipleRegisters API used to implement a write multiple registers action.

The request provides three parameters:

- the destination of the write request,
- the starting address,
- the values to write.

This API is mapped on Modbus Function Code 16.

Parameters

Input: int UnitID, Address, array of int Value

Output: none

Example

The following is an example of a SOAP request and its response:

Request

```
POST /ws/ModbusXmlDa
HTTP/1.0 Host: 139.160.65.83:8080
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/ModbusXmlDa/WriteMultipleRegisters"
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <WriteMultipleRegisters xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
       <UnitID>int</UnitID>
       <Address>int</Address>
       <Value>
         <int>int</int>
         <int>int</int>
       </Value>
     </WriteMultipleRegisters>
 </soap:Body>
</soap:Envelope>
```

Response

```
HTTP/1.0 200 OKContent-Type: text/xml; charset=utf-8
Content-Length: length <?xml version="1.0" encoding="utf-8" ?>
 <soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
     <WriteMultipleRegistersResponse
xmlns="http://www.schneider-electric.com/ws/ModbusXmlDa/">
     </WriteMultipleRegistersResponse>
  </soap:Body>
 </soap:Envelope>
```

ReadCoils

This is the ReadCoils API used to implement a read multiple coils action.

The request provides three parameters:

- the destination of the read request,
- the starting address,
- the number of values to read (base index is provided by starting address).

In response, the list of values is provided.

This API is mapped on Modbus Function Code 01.

Parameters

Input: int UnitID, Address, Quantity

Output: array of int Result

Example

The following is an example of a SOAP request and its response:

Request

```
POST /ws/ModbusXmlDa
HTTP/1.0 Host: 139.160.65.83:8080
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/ModbusXmlDa/ReadCoils"
<?xml version="1.0" encoding="utf-8" ?>
 <soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <ReadCoils xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
       <UnitID>int</UnitID>
       <Address>int</Address>
       <Quantity>int</Quantity>
     </ReadCoils>
   </soap:Body>
</soap:Envelope>
```

Response

```
HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8" ?>
 <soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
     <ReadCoilsResponse xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
       <ReadCoilsResult>
         <int>int</int>
         <int>int</int>
       </ReadCoilsResult>
     </ReadCoilsResponse>
  </soap:Body>
</soap:Envelope>
```

WriteMultiple-Coils

This is the WriteMultipleCoils API used to implement a write multiple coils action.

The request provides three parameters:

- the destination of the write request,
- the starting address,
- the values to write.

This API is mapped on Modbus Function Code 15.

Parameters

Input: int UnitID, Address, array of int Value

Output: none

Example

The following is an example of a SOAP request and its response:

Request

```
POST /ws/ModbusXmlDa HTTP/1.0
Host: 139.160.65.83:8080
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/ModbusXmlDa/WriteMultipleCoils"
<?xml version="1.0" encoding="utf-8" ?>
 <soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <WriteMultipleCoils xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
       <UnitID>int</UnitID>
       <Address>int</Address>
       <Value>
         <int>int</int>
         <int>int</int>
       </Value>
     </WriteMultipleCoils>
   </soap:Body>
 </soap:Envelope>
```

Response

```
HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8" ?>
 <soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <WriteMultipleCoilsResponse xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
     </WriteMultipleCoilsResponse>
   </soap:Body>
</soap:Envelope>
```

ReadInt32 SOAP request/response

The ReadInt32 request provides three parameters:

- The destination of the read request,
- the starting address,
- the number of values to read (base index is provided by starting address).

Parameters

Input: int UnitID, Address, Quantity

Output: array of int Result

Example

In response, the list of values is provided. The following is an example of a SOAP request and response:

Request

```
POST /ws/ModbusXmlDa
HTTP/1.0 Host: 139.160.65.83:8080
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/ModbusXmlDa/ReadMultipleRegisters"
<?xml version="1.0" encoding="utf-8" ?>
 <soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <ReadInt32 xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
       <UnitID>int</UnitID>
       <Address>int</Address>
       <Quantity>int</Quantity>
     </ ReadInt32 >
   </soap:Body>
 </soap:Envelope>
```

Response

```
HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8" ?>
 <soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     < ReadInt32Response xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
       < ReadInt32Result>
         <int>int</int>
         <int>int</int>
       </ ReadInt32Result>
     </ ReadInt32Response>
   </soap:Body>
 </soap:Envelope>
```

WriteInt32 SOAP request/response

The WriteInt32 request provides three parameters:

- The destination of the write request,
- the starting address,
- the values to write.

Parameters

Input: int UnitID, Address, array of int Value

Output: none

Example

The following is an example of a SOAP request and response:

Request

```
POST /ws/ModbusXmlDa
HTTP/1.0 Host: 139.160.65.83:8080
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/ModbusXmlDa/WriteMultipleRegisters"
<?xml version="1.0" encoding="utf-8" ?>
 <soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <WriteInt32 xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
       <UnitID>int</UnitID>
       <Address>int</Address>
       <Value>
         <int>int</int>
         <int>int</int>
       </Value>
     </ WriteInt32 >
   </soap:Body> </soap:Envelope>
```

Response

```
HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8" ?>
 <soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
   < WriteInt32Response xmlns="http://www.schneider-
electric.com/ws/ModbusXmlDa/">
      </ WriteInt32Response>
   </soap:Body>
 </soap:Envelope>
```

SymbolicXMLDA SOAP Services

Overview

The SymbolicXMLDA is the Web Service provided to implement a Symbolic Data Access based on a FactoryCast namespace. The services are:

- read,
- write,
- browse.

Limitation on symbolic access

As symbolic access relies on low-level Modbus access for actual communication, only symbols mapped on data accessible through native Modbus requests (Read/Write registers and Read/Write Coils) are supported. Consequently, only data of these types are supported:

- bit
- 16-bit
- 32-bit double word

Unlocated data of all types are not supported.

Read service

The aim of this request is to read one or several symbols. The symbols passed as request arguments are translated using the namespace to get the address of the value to read. When all addresses are known, a Modbus request is issued by contiguous addresses (i.e., if registers 10, 11, and 13 are requested, two requests are issued: one for 10 and 11, and one for 13) and by type of variable desired (coils, registers). Once all the values are retrieved, the SOAP response is built using the variable name, type and values.

Parameters

Input: list of string ItemName

Output: array of Item ReadResult

Example

The following is a example SOAP request and response:

Request

```
POST /ws/SymbolicXmlDa.asmx  HTTP/1
Host: 139.160.65.83:8080
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/SymbolicXmlDa/1.0/Read"
  <?xml version="1.0" encoding="utf-8"?>
  <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
      <Read xmlns="http://www.schneider-
electric.com/ws/SymbolicXmlDa/1.0/">
        <ItemList>
          <Items>
            <ItemName>string</ItemName>
          </Items>
          <Items>
            <ItemName>string</ItemName>
          </Items>
        </ItemList>
      </Read>
    </soap:Body>
  </soap:Envelope>
```

Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
      <WriteResponse xmlns="http://www.schneider-
electric.com/ws/SymbolicXmlDa/">
      </WriteResponse>
    </soap:Body>
</soap:Envelope>
```

Write service

The aim of this request is to write one or several symbols. The symbols, their types, and values are passed as request arguments. The names of the variables are translated using the namespace to get the address of the value to read. When all addresses are known, a Modbus request is issued by contiguous addresses (i.e., if register 10, 11, and 13 are requested, two requests are issued: one for 10 and 11, and one for 13) and by type of variable desired (coils, registers).

Once all the values are written, the SOAP response is built.

Parameters

Input: list of (string ItemName, Value, Type)

Output: none

Example

The following is a example SOAP request and response:

Request

```
POST /Recipe/ws/SymbolicXmlDa.asmx HTTP/1.1
Host: 139.160.65.83:8080
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/SymbolicXmlDa/1.0/Write"
<?xml version="1.0" encoding="utf-8"?>
 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
     <Write xmlns="http://www.schneider-
electric.com/ws/SymbolicXmlDa/1.0/">
       <ItemList>
         <Items>
           <ItemName>string</ItemName>
           <Value>string</Value>
           <Type>string</Type>
         </Items>
         <Items>
           <ItemName>string</ItemName>
           <Value>string</Value>
           <Type>string</Type>
         </Items>
      </ItemList>
     </Write>
   </soap:Body>
 </soap:Envelope>
```

Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <ReadResponse xmlns="http://www.schneider-
electric.com/ws/SymbolicXmlDa/">
       <ReadResult>
       <Items>
          <Name>string</Name
          <SymbolType>string</SymbolType>
          <IntValue>int</IntValue>
       </Items>
       <Items>
          <Name>string</Name>
          <SymbolType>string</SymbolType>
          <IntValue>int</IntValue>
     </Items>
     </ReadResult>
     </ReadResponse>
   </soap:Body>
</soap:Envelope>
```

Browse service

The aim of this request is to read a list of symbolic variables present on the namespace of the FactoryCast module.

This functionality does not interface with the Modbus layer since its only action is to return all the namespace items. The response contains all the symbols and their types.

Parameters

Input: list of (string ItemName, Type)

Output: list of Descriptions BrowseResult

Example

The following is a example SOAP request and response:

Request

```
POST /Recipe/ws/SymbolicXmlDa.asmx
HTTP/1.1 Host: 139.160.65.83:8080
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/SymbolicXmlDa/1.0/Browse"
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <Browse xmlns="http://www.schneider-
electric.com/ws/SymbolicXmlDa/1.0/" />
   </soap:Body> </soap:Envelope>
```

Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length <?xml version="1.0" encoding="utf-8"?>
 <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <BrowseResponse xmlns="http://www.schneider-
electric.com/ws/SymbolicXmlDa/1.0/">
       <Elements>
        <BrowseElement>
           <ItemName>string</ItemName>
           <Type>string</Type>
         </BrowseElement>
         <BrowseElement>
          <ItemName>string</ItemName>
           <Type>string</Type>
         </BrowseElement>
      </Elements>
     </BrowseResponse>
   </soap:Body>
 </soap:Envelope>
```

ExtendedSymbolicXMLDA SOAP Services

Presentation

The ExtendedSymbolicXMLDA extends the SymbolicXmlDA SOAP services for UnityPro. It allows you to access direct addresses data and unlocated data . The services are:

- read,
- write,
- browse.

Limitation on variables access

As variables access relies on low-level Unity protocol access for actual communication, only symbols and direct addresses mapped on data accessible through native Unity requests (Read/Write registers and Read/Write Coils) are supported. Consequently, only data of these types are supported:

Type	Syntax
Internal Variables	%Mi, %MXi, %MWi, %MWi.j
Constants	%KWi
System Variables	%Si, %SXi, %SWi
Simulated	%MDi, %MFi, %KDi, %KFi

Supported types are all basic UnityPro types including elements of arrays.

For direct addressing, supported syntax are:

Syntax	Type	Comment
%Mi or %MXi	EBOOL	mapped on XML/W3C type unsignedByte
%MWi	INT	mapped on XML/W3C type short
%MWi.j	BOOL	mapped on XML/W3C type boolean
%MDi	DINT	mapped on XML/W3C type int
%MFi	REAL	mapped on XML/W3C type float
%KWi	INT	mapped on XML/W3C type short
%KWi.j	BOOL	mapped on XML/W3C type boolean
%KDi	DINT	mapped on XML/W3C type int
%KFi	REAL	mapped on XML/W3C type float
%Si or %SXi	BOOL	mapped on XML/W3C type boolean
%SWi	INT	mapped on XML/W3C type short
%SWi.j	BOOL	mapped on XML/W3C type boolean

For Unity symbols, supported types are:

Type	Comment
BOOL	mapped on XML/W3C type boolean
EBOOL	mapped on XML/W3C type unsignedByte
INT	mapped on XML/W3C type short
DINT	mapped on XML/W3C type int
UINT	mapped on XML/W3C type unsignedShort
UDINT	mapped on XML/W3C type unsignedInt
TIME	mapped on XML/W3C type unsignedInt
DATE	mapped on XML/W3C type date or dateTime (dateTime is mandatory for write, date part is ignored)
TOD	mapped on XML/W3C type time or dateTime (dateTime is mandatory for write, time part is ignored)
DT	mapped on XML/W3C type dateTime
REAL	mapped on XML/W3C type float or double (double is only for write request, cast the value in float)
BYTE	mapped on XML/W3C type unsignedByte
WORD	mapped on XML/W3C type unsignedShort
DWORD	mapped on XML/W3C type unsignedInt
STRING	mapped on XML/W3C type string
STRING[n]	mapped on XML/W3C type string

NOTE:

- The name of the variable can be either a direct address or any symbols that appear in the FactoryCast namespace.
- If a Web service function has an optional parameter, Visual Studio .NET creates an additional boolean parameter in the corresponding proxy. Visual Studio .NET adds "Specified" at the end of the name of that parameter.

Read service

The aim of this request is to read symbols and direct addresses. The symbols passed as request arguments are translated using the namespace to get the address of the value to read. Direct addresses are passed as request argument. When all addresses are known, a set of Unity requests is issued by contiguous addresses (i.e., if registers 10, 11, and 13 are requested, two requests are issued: one for 10 and 11, and one for 13) and by type of variable desired (coils, registers). Once all the values are retrieved, the SOAP response is built using the variable name, type and values.

Parameters

Input: list of string ItemName

Output: array of Item ReadResult

Example

The following is a example SOAP request and response:

Request

```
POST /ws/ExtendedSymbolicXmlDa.asmx  HTTP/1
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.schneider-
electric.com/ws/ExtendedSymbolicsXmlDa/Read"
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
      <Read xmlns="http://www.schneider-
electric.com/ws/ExtendedSymbolicsXmlDa/">
        <VariableList>
            <Name>string</Name>
            <Name>string</Name>
        </VariableList>
      </Read>
    </soap:Body>
  </soap:Envelope>
```

Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>

<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">

                 <soap:Body>

    <ReadResponse xmlns="http://www.schneider-
electric.com/ws/ExtendedSymbolicXMLDa/">

    <ReadResult>

    <Item>

     <Name>string</Name>

     <VariableType>string</VariableType>

     <Value />

    </Item>

    <Item>

     <Name>string</Name>

     <VariableType>string</VariableType>

     <Value />

    </Item>

    </ReadResult>

   </ReadResponse>

 </soap:Body>

</soap:Envelope>
```

Write service

The aim of this request is to write symbols or direct addresses. The symbols, their types, and values are passed as request arguments. The names of the variables are translated using the namespace to get the address of the value to read. Direct addresses are passed as request argument. When all addresses are known, a set of Unity requests is issued by contiguous addresses (i.e., if register 10, 11, and 13 are requested, two requests are issued: one for 10 and 11, and one for 13) and by type of variable desired (coils, registers).

Once all the values are written, the SOAP response is built.

Parameters

Input: list of (string ItemName, Value, Type)

Output: none

Example

The following is a example SOAP request and response:

Request

```
POST /maquette/ExtendedSymbolicXmlDa.asmx HTTP/1.1
Host: 139.160.65.83:8080
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
```

```
SOAPAction: "http://www.schneider-
electric.com/ws/ExtendedSymbolicsXmlDa/Write"
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
 <soap:Body>
     <Write xmlns="http://www.schneider-
electric.com/ws/ExtendedSymbolicXmlDa/">
       <ItemList>
         <Item>
           <Name>string</Name>
           <VariableType>string</VariableType>
           <Value />
         </Item>
         <Item>
           <Name>string</Name>=
           <VariableType>string</VariableType>
           <Value />
         </Item>
      </ItemList>
     </Write>
   </soap:Body>
</soap:Envelope>
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/">
   <soap:Body>
     <WriteResponse xmlns="http://www.schneider-
electric.com/ws/ExtendedSymbolicXmlDa/">
    </WriteResponse>
          </soap:Body>
</soap:Envelope>
```

Browse service

The aim of this request is to read a list of symbolic variables (PLC, unlocated and I/O) present on the namespace of the FactoryCast module.

The response contains all the symbols and their types.

Parameters

Input: none or the optional parameter

Output: list of descriptions (name, address and symbol type)

Example

The following is a example SOAP request and response:

Request

```
POST /maquette/ExtendedSymbolicXmlDa.asmx
HTTP/1.1 Host: 139.160.65.83:8080
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
SOAPAction: xmlns="http://www.schneider-
electric.com/ws/ExtendedSymbolicXmlDa/Browse"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/2003/05/soap-
envelope/">
   <soap:Body>
<Browse xmlns="http://www.schneider-
electric.com/ws/ExtendedSymbolicXmlDa/Browse" />
   </soap:Body>
         </soap:Envelope>
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/">
   <soap:Body>
     <BrowseResponse xmlns="http://www.schneider-
electric.com/ws/ExtendedSymbolicXmlDa/">
        <BrowseResult>
        <Description>
           <Name>string</Name>
           <Address>string</Address>
           <VariableType>string</VariableType>
         </Description>>
         <Description>
           <Name>string</Name>
           <Address>string</Address>
           <VariableType>string</VariableType>
         </Description>
        </BrowseResult>
     </BrowseResponse>
   </soap:Body>
 </soap:Envelope>
```

How to optimize massive symbolic requests

Overview

The symbolic read/write request allows you to perform requests of up to 128 symbols.

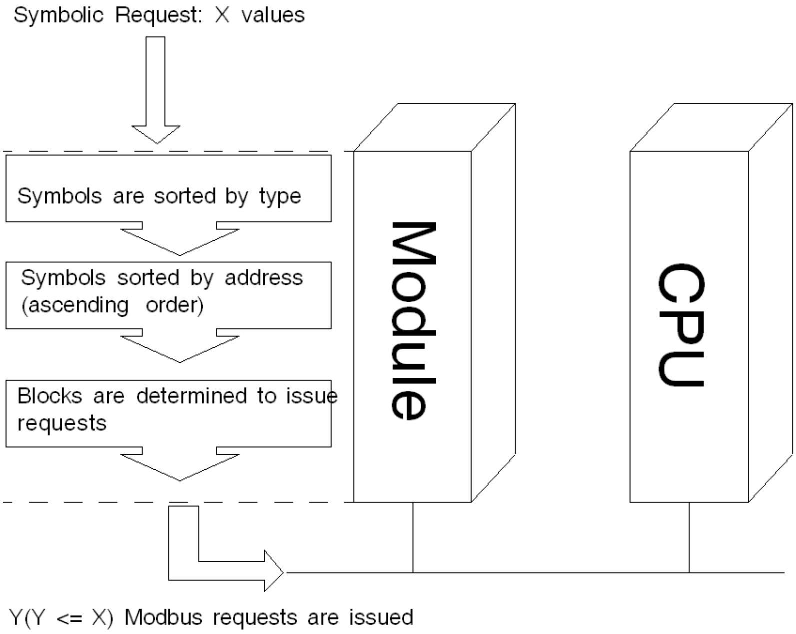
You can experience a lag when using a large amount of symbols. There are several rules you can apply to improve your application's speed, by reducing the number of internal requests.

The purpose of this topic is to help you create a coherent namespace and use symbolic requests efficiently.

Symbolic request process

Symbolic requests enable the user to use symbols instead of addresses to handle PLC data. This request system uses the namespace on the module to resolve the addresses. Once the addresses are known, there is an optimization process to sort the addresses by type (coils, registers) in ascending order.

The sorted addresses should be grouped by coherent block to minimize the number of requests sent to the CPU. The graphic below shows the three steps of the symbolic request optimization:


Symbolic Request: X values
Symbols are sorted by type
Symbols sorted by address (ascending order)
Blocks are determined to issue requests
Module
CPU
Y(Y <= X) Modbus requests are issued

Optimization method

To perform a massive symbolic request efficiently, it is necessary to pay close attention to the topology of the symbols:

Step	Action
1	Group symbols by type: don't mix coils and registers
2	If you can, sort symbols in ascending order
3	Try to determine blocks of symbols (i.e., contiguous symbol addresses). These blocks will be processed in only one low-level request.

Optimization example

Considering a relatively simple namespace file:

```
|STATION:0.0
16
|Bit_mem1|%M0|0|0|1|1|1|false
|Bit_mem2|%M1|0|0|1|1|1|false
|Bit_mem3|%M2|0|0|1|1|1|false
|Bit_mem4|%M4|0|0|1|1|1|false
|Bit_mem5|%M5|0|0|1|1|1|false
|Bit_mem6|%M6|0|0|1|1|1|false
|Engine_Brake_lvl|%MW0|0|0|4|1|1|false
|Engine_Brake_dsk|%MW1|0|0|4|1|1|false
|Engine_gauge_1|%MW2|0|0|4|1|1|false
|Engine_gauge_2|%MW3|0|0|4|1|1|false
|Engine_gauge_3|%MW4|0|0|4|1|1|false
|Engine_gauge_4|%MW5|0|0|4|1|1|false
|Engine2_Brake_lvl|%MW100|0|0|4|1|1|false
|Engine2_Brake_dsk|%MW101|0|0|4|1|1|false
|Engine2_gauge_1|%MW102|0|0|4|1|1|false
|Engine2_gauge_2|%MW103|0|0|4|1|1|false
|Engine2_gauge_3|%MW104|0|0|4|1|1|false
|Engine2_gauge_4|%MW105|0|0|4|1|1|false
0
1
false
```

There are values for two engines (engine and engine2).

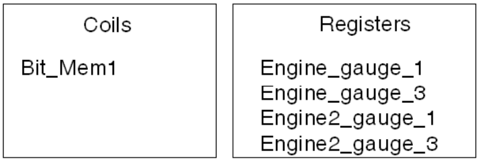
There are also Boolean values (Bit_mem).

If the end user asks for every symbol in their namespace order, they will issue one SOAP request with 16 values. As type and addresses (in ascending order) sort these values, the optimization will only determine blocks of addresses. Here we can see three blocks:

- coils from address 0 to 6,
- registers from address 0 to 5,
- registers from address 100 to 105.

There will be three low-level requests sent to the CPU.

If the end user asks for five symbols in one SOAP request: Engine_gauge_3, Engine_gauge_1, Bit_Mem1, Engine2_gauge_3, Engine2_gauge_1.

Step	Action
1	Since symbols are mixed, the first optimization step is to sort between coils and registers.
2	In this step, symbols of the same type are sorted by address. As the SOAP request is not well ordered, this step has to process the data. At the end of this step we have two arrays: an array of one coil and an array of four registers:  Coils Bit_Mem1 Registers Engine_gauge_1 Engine_gauge_3 Engine2_gauge_1 Engine2_gauge_3
3	This step should find contiguous addresses. Since there are no contiguous addresses, we have five blocks of one address. The final result is that five low-level requests are sent to the CPU.

In brief

When you require intensive use (in number, in time constraints) of symbolic requests, these rules must be followed:

- group the symbols by types,
- sort the symbols in ascending order,
- group the symbols by blocks of contiguous addresses.

Visual Basic Examples

Overview

To help you start writing your application, here is a Visual Basic .NET (2003 or 2005) example of accessing SOAP requests.

Preliminary Steps

The following table describes the actions to carry out before using the example.

Step	Action
1	In Visual Basic .NET, select **Project** → **Add WebReference**.
2	Type `http://Module @IP/ws/ModbusXmlDa`.
3	Select the desired function.
4	Click the **Add Reference** button.

Example

The following example reads ten consecutive registers from register 5.

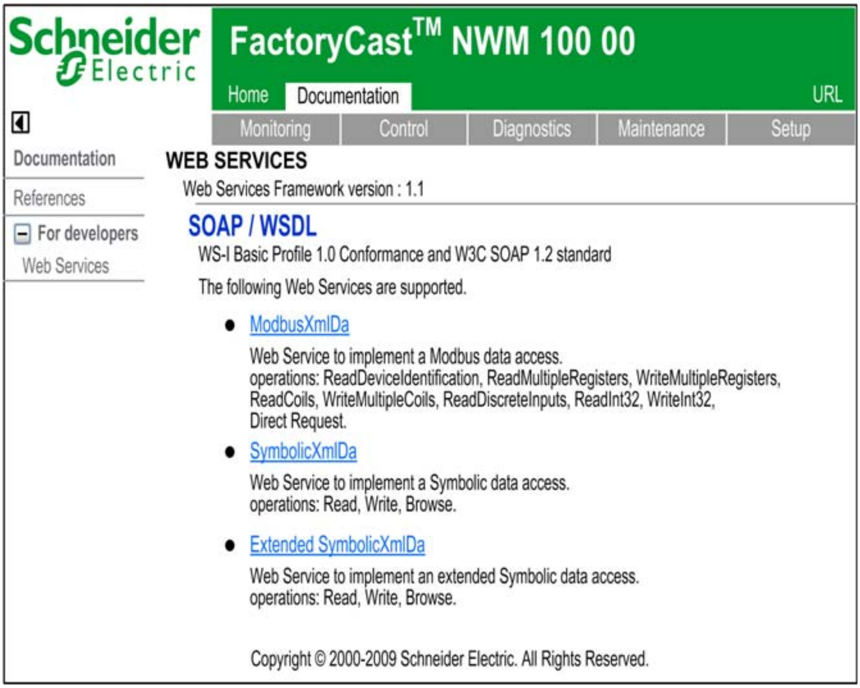
```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click

        Dim ws As New
WindowsApplication26.WebReference.ModbusXmlDa

        ListBox1.DataSource = ws.ReadMultipleRegisters(0, 5,
10)

    End Sub

End Class
```

Online SOAP Documentation

Overview

When you open a FactoryCast HMI Web server page in a browser, a **Documentation** hyperlink is available.

This link takes you to SOAP online documentation, which describes the SOAP Web services available, explains the syntax of SOAP APIs, and provides Visual Basic examples.



WEB SERVICES

Web Services Framework version : 1.1

SOAP / WSDL

WS-I Basic Profile 1.0 Conformance and W3C SOAP 1.2 standard

The following Web Services are supported.

- ModbusXmlDa

  Web Service to implement a Modbus data access.
  operations: ReadDeviceIdentification, ReadMultipleRegisters, WriteMultipleRegisters, ReadCoils, WriteMultipleCoils, ReadDiscreteInputs, ReadInt32, WriteInt32, Direct Request.
- SymbolicXmlDa

  Web Service to implement a Symbolic data access.
  operations: Read, Write, Browse.
- Extended SymbolicXmlDa

  Web Service to implement an extended Symbolic data access.
  operations: Read, Write, Browse.

Copyright © 2000-2009 Schneider Electric. All Rights Reserved.

Flash Recommendation

B

Recommendation on Datalogging Service

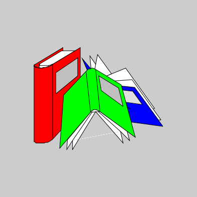
Size of the Log File

The following table shows you an estimation of the log file size in bytes depending on the number of variables logged and the number of logs:

Number of logs	Number of variables						
	1	2	5	10	20	50	100
1	65	110	245	470	920	2270	4520
2	130	220	490	940	1840	4540	9040
5	325	550	1225	2350	4600	11350	22600
10	650	1100	2450	4700	9200	22700	45200
20	1300	2200	4900	9400	18400	45400	90400
50	3250	5500	12250	23500	46000	113500	226000
100	6500	11000	24500	47000	92000	227000	452000

NOTE: To prolong the internal Flash life, rerun the log file at intervals greater than 30 minutes.

Glossary



A

applet

Software component that runs in the context of another program, for example a Web browser.

ASCII

American Standard Code for Information Interchange.

Pronounced "aski". This is an American code (but now an international standard) which allows all alphanumerical characters used in English, punctuation marks, some graphics characters and various commands to be defined with 7 bits.

B

BIT

Contraction of Binary Digit.

This is the binary unit of information content, which can represent two separate values (or states): 0 or 1.

A field of 8 bits constitutes 1 **byte**.

C

configuration

The configuration comprises all the data that defines the device (invariable) and that is necessary to the operation of the module.

CPU

Central Processing Unit.

The microprocessor. This comprises the entire control unit and the arithmetic unit. The purpose of the control unit is to extract the execution instruction from the central memory along with the data needed to execute this instruction, to establish electrical connections in the arithmetic and logic unit and to start the processing of this data in the unit. **ROM** or **RAM** memories are sometimes included on the same chip, and sometimes even I/O interfaces or buffers.

D

DHCP

Dynamic Host Configuration Protocol: Protocol allowing a station connected to the network to obtain its configuration dynamically.

DNS

Domain Name System: It stores and associates many types of information with domain names, but most importantly, it translates domain names (computer hostnames) to IP addresses.

driver

Program which informs the operating system of the presence and characteristics of a peripheral.

F

firewall

Information technology (IT) security device which is configured to permit, deny or proxy data connections set and configured by the organization’s security policy.

Flash memory

Form of non-volatile computer memory that can be electrically erased and reprogrammed.

FTP

File Transfer Protocol: Network file transfer protocol.

H

HTML

HyperText Markup Language: the predominant markup language for the creation of web pages. It provides a means to describe the structure of text-based information in a document and to supplement that text with interactive forms, embedded images, and other objects.

HTTP

HyperText Transfer Protocol: Network transfer protocol for documents written in hypertext (links).

I

IP

Internet Protocol: Data-oriented protocol used for communicating data across a packet-switched internetwork (i.e. the Internet).

M

MIB

Management Information Base: Database used by the SNMP protocol for network management and containing information on data transmission, station or router components, etc.

- MIB II: standard MIB
- Schneider Automation MIB: private MIB

N

NTP

Network Time Protocol: Protocol for synchronizing the clocks of computer systems over packet-switched, variable-latency data networks.

P

PL7

Schneider Automation PLC programming software.

PLC

Programmable Logic Controller: It is a small computer used for automation of industrial processes, such as control of machinery on factory assembly lines.

R

RS485

Serial connection standard operates at +/-5V differential. The connection uses the same wire for transmission and receipt. Their "3-status" outputs allow them to switch to listening mode when transmission is completed.

S

SMTP

Simple Mail Transfer Protocol: Application protocol used to transmit messages via the Internet and direct them to a mailbox.

SNMP

Simple Network Management Protocol: Network management protocol for controlling a network remotely by polling the stations for their status and modifying their configuration, performing security tests and viewing information relating to data transmission. It can also be used to manage software and databases remotely.

SQL

Structured Query Language: Used to query (request data from) a relational database.

T

TCP

Transmission Control Protocol: Virtual circuit protocol that is one of the core protocols of the Internet protocol suite, often simply referred to as TCP/IP.

TCP/IP

The set of communications protocols that implement the protocol stack on which the Internet and most commercial networks run.

U

UDP

User Datagram Protocol: One of the core protocols of the Internet protocol suite. Using UDP, programs on networked computers can send short messages sometimes known as datagrams to one another.

URL

Uniform Resource Locator: The global address of documents and other resources on the World Wide Web.

X

XML

Extensible Markup Language: it is aimed to facilitate the sharing of data across different information system. It is a simplified subset of the SGML and is designed to be relatively human-legible.

Index