Telepace Studio User and Reference Manual

Telepace Studio
User and Reference Manual

Revision 5.0.2
Monday, April 14, 2014

Telepace Studio User and Reference Manual

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.

Licensed Source Code Information
This application contains source code licensed from third parties.

diff.cs
Copyright (c) 2005-2009 by Matthias Hertel, http://www.mathertel.de/
All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
* Neither the name of the copyright owners nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Safety Information

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 or Warning 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.

CAUTION

CAUTION used without the safety alert symbol, indicates a potentially hazardous situation which, if not avoided, **can result in** equipment damage..

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 the installation, and has received safety training to recognize and avoid the hazards involved.

BEFORE YOU BEGIN

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

⚠CAUTION

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.
- Remove tools, meters, and debris from equipment.

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

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and grounds, except those grounds installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove ground from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

OPERATION AND ADJUSTMENTS

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

Technical Support

Support related to any part of this documentation can be directed to one of the following support centers.

Technical Support: The Americas

Available Monday to Friday 8:00am – 6:30pm Eastern Standard Time

Toll free within North America 1-888-226-6876

Direct Worldwide +1 (613) 591-1943

Email TechnicalSupport@controlmicrosystems.com

Technical Support: Europe, Africa, Middle East

Available Monday to Friday 8:30am – 5:30pm Central European Standard Time

Direct Worldwide +31 (71) 597-1655

Email euro-support@controlmicrosystems.com

Technical Support: Asia Pacific

Available Monday to Friday 8:30am – 5:30pm Australian Eastern Standard Time

Direct Worldwide +61 3 9249 9580

Email au-support@controlmicrosystems.com

Whats New Release Notes

This section of the user manual provides a quick snapshot of changes in each version of Telepace Studio. Please review this section for descriptions of new features or improvements to be found in your version of Telepace Studio. Use the quick links below to see what is new in your version of Telepace Studio.

Telepace Studio version 5.0.2

Telepace Studio 5.0.8

Telepace Studio version 5.0.8 is a maintenance release to correct the following items:

- Telepace Studio would occasionally freeze when entering Monitoring Online mode.
- The performance of the ladder canvas would decrease after opening the Search dialog in a project that contains many tags.

Telepace Studio 5.0.7

Telepace Studio version 5.0.7 is a maintenance release to correct the following items:

- Telepace Studio would occasionally lockup after changing a register format to "Floating Point" from the Register Editor popup dialog.
- Telepace Studio would generate a message when validating a "Time and Date" field of a DLGF function block.
- Improved DTD (Document Type Definition) processing and handling of XML files.
- Added diagnostics messages to the Diagnostic window during new file creation.

Telepace Studio 5.0.6

Telepace Studio version 5.0.6 is a maintenance release to correct the following items:

Improvements

- Telepace Studio now requires only a .LAD file when importing a TelePACE3 project. The .CSV and .XML files are no longer required.
- Improved notification messages displayed during a TelePACE3 project import.
- The usability of the Register Editor dialog has been improved. An "Apply" button has been added to allow multiple register modifications without having to close and re-open the dialog.
- The speed of Binary to CSV data conversions has been improved for large datasets.
- Improved the interface display at screen resolutions of 125% and 150% in Windows 7.

Corrected Items

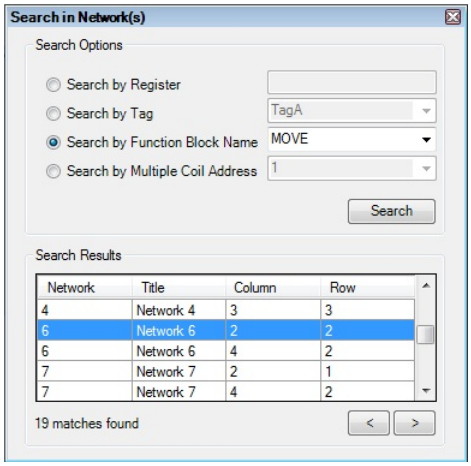
- Communication messages were not cleared without having to restart Telepace Studio.
- Using the "Find Next" search for function block names would jump to unexpected function blocks.
- A *SetImage* exception message was occasionally displayed in Diagnostic window when monitoring online.
- A blank diagnostic message was reported in the Diagnostic window when the Monitor Online button was pressed.
- The Cut and Copy buttons in the Ladder Canvas were enabled with no valid canvas object selection.
- The Program Status displayed zero RAM usage after a program was downloaded.
- Unable to change the Protocol Type in the MSIP function block.
- The TOTL function block did not reserve enough registers.
- The TOTL function block did not list all of the used registers if the number of records was changed.
- Search for Used Registers in the Register Editor did not work correctly.
- Search did not find all registers used by a single function block.
- Search for a Tag Name did not return all instances of the tag.
- The File Management transfer message did not clearly indicate the status of a file transfer from the local file system to the controller.
- The DNP settings in the project file did not account for various regional settings, causing Telepace Studio to reject certain project files.

Telepace Studio 5.0.5

Telepace Studio version 5.0.5 is a maintenance release to add new features and to correct known issues.

New Features

- The Ladder Canvas search functionality is improved and a new Search in Network(s) dialog is added..



Search Options

The **Search by Register** radio button activates the register select window. The user may enter a numeric register address for which to search.

The **Search by Tag** radio button activates the tag select window. Tags used in the ladder logic program are displayed when the drop down menu is selected. The user may begin typing the desired tag name to filter the drop down list items.

The **Search by Function Block** radio button activates the function block select window. Function blocks used in the ladder logic program are displayed when the drop down menu is selected. The user may begin typing the desired function block name to filter the drop down list items.

The **Search by Multiple Coil Address** radio button activates the Multiple Coil address select window. All instances of multiple coils used in the ladder logic program are displayed when the drop down menu is selected. The user may begin typing the desired multiple coil register address to filter the drop down list items.

The **Search** button begins the search of the entire program based on the parameters selected in the Search Options group, and lists the results in the table in the Search Results group.

Search Results

The table in the Search Results group will be populated with the returned search results. The first search result from the current network position will be automatically selected in the ladder canvas. Should no result be found on the current network or later, the search will wrap and the first search result in the program will be selected in the ladder canvas. The subsequent search results can be navigated using the previous (<) and next (>) buttons, or by double clicking a specific search result in the table.

The total number of search results is indicated in the lower left corner, below the search results table. Should the ladder logic or tags change after performing a search, the text indicating the number of found search results will be denoted in red with an asterisk and accompanied by a tooltip indicating that the search results may no longer be accurate and a new search should be performed.

- Microsoft .NET framework 3.5 is now included on the Telepace Studio installation CD. The.NET 3.5 framework is required in order to install and run Telepace Studio
- The Telepace Studio installation now installs the latest released firmware for controllers when Telepace Studio is installed.

Corrected Issues

- In some cases the Monitor Element dialog was not correctly displayed when selected by the user. The complete Monitor Element dialog is now displayed in all cases.
- The Search for Multiple Coils would return results for both multiple coils and contacts. The functionality is corrected to return only multiple coil addresses.

Telepace Studio 5.0.4

Telepace Studio version 5.0.4 is a maintenance release to correct following issues:

- Ladder canvas search did not start at top of logic when search dialog opened and closed. This issue is corrected and all searches now begin at the parameters entered for the Start location in the Search For Networks(s) dialog.
- The CMP (Compare two signed values) function block did not allow a change from a constant value to a register value. This issue is corrected and changes from a constant value to a register value, and vise versa are supported.
- Telepace Studio would generate errors and then close when networks were deleted in the On-Line mode. This issue is corrected in Telepace Studio. All cut network, delete network and modify network commands are properly executed in Telepace Studio.
- One-Shot coil history was reset, causing an energized One-Shot coil to re-energize, when On-Line changes were made to a network containing the One-Shot coil. This issue is corrected and now only On-Line changes to the modified element cell are written to the controller, not the entire network.

Telepace Studio 5.0.3

Telepace Studio version 5.0.3 is a maintenance release to correct following issue:

- When using Telepace Studio on a PC with Windows 7 attempts to print a project would generate numerous diagnostic messages.

This issue is corrected in Telepace Studio version 5.0.3.

Telepace Studio 5.0.2

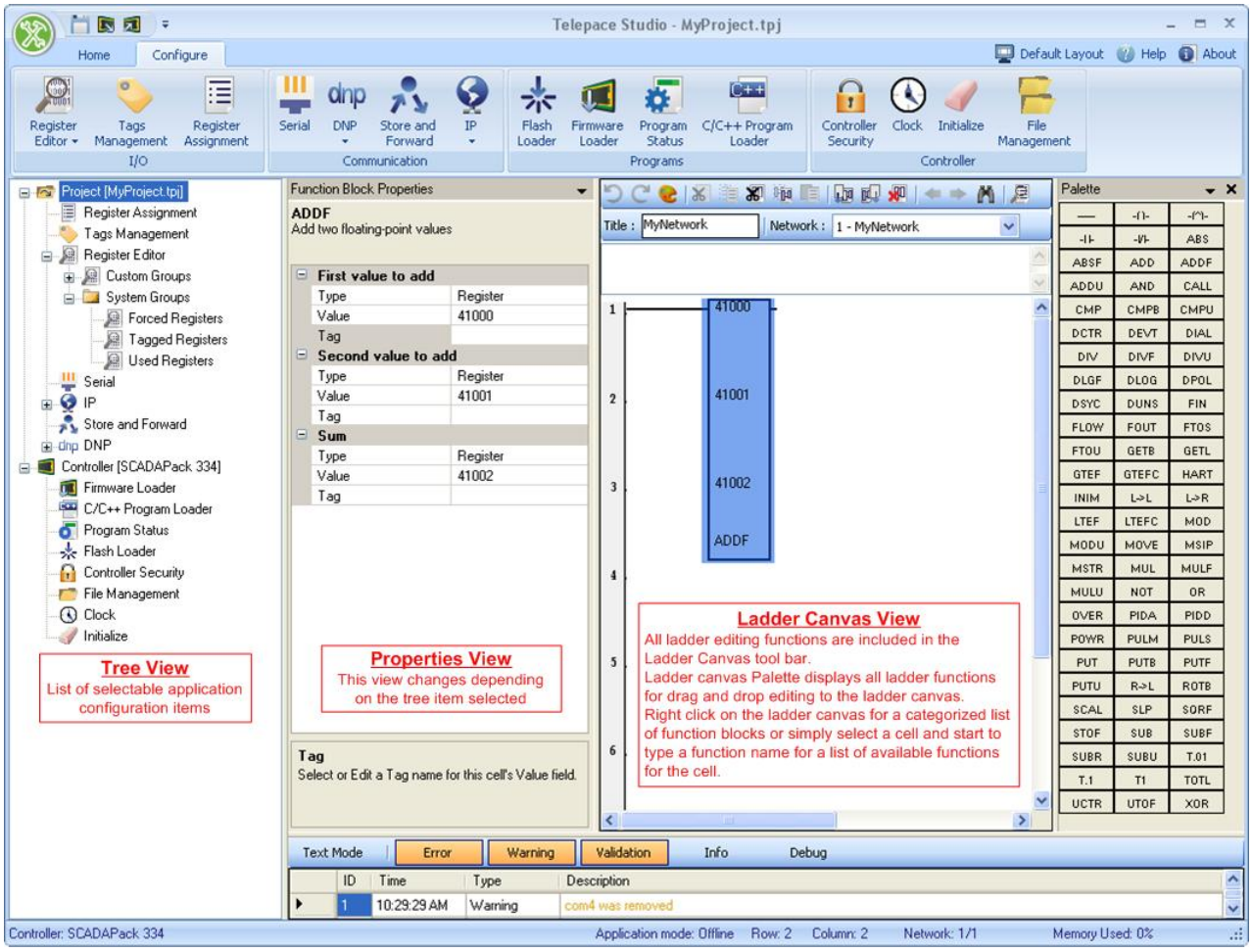
Telepace Studio version 5.0.2 is a major update of Telepace Studio, bringing a new layout, more intuitive programming functionality and simplified application monitoring capabilities. The new Telepace Studio application layout is shown at the bottom of this screen.

Important Differences

Telepace Studio version 5.0.2 files are not backward compatible, i.e., cannot be opened using previous versions of Telepace Studio. Telepace Studio can read the logic application and configuration parameters from controllers that have been programmed using previous versions of Telepace Studio.

New Features

Tree View	Located at the left of the user interface a tree view is added to provide fast access to common application tools. When a tool is selected from the tree, the Properties Panel changes to provide editing functionality for the selected tool. The register editor now includes System Groups in addition to the user created Custom Groups. System groups include: • Forced Registers • Tagged Registers • Used Registers All IP configurations are selectable from the tree view and edited in the properties panel. Separate tabs for each type of configuration are no longer used. All DNP configurations are selectable from the tree view and edited in the properties panel. Separate tabs for each DNP property are no longer used.
Ladder Canvas	The right click selection for Function Blocks is now categorized for easier selection of needed function blocks. The function block palette is a selectable list of all available function blocks. Needed functions can be dragged and dropped onto the ladder canvas. The properties panel changes to a edit view for the selected function block. A new 'auto complete' feature enables power users to open a list of function blocks available by simply starting to type the name of a function block in a selected empty cell on the ladder canvas. The list will contain only those function blocks that fit into the cell space selected. Ladder canvas actions are moved from ribbon to the canvas with a ladder canvas tool ribbon, network title edit window and network search. Normally open or normally closed contacts can now be changed form one type to the other by selecting the contact in the ladder canvas and using the '/' key.
Function Block Tool Box	The function block palette and function block properties panel replace the function block toolbox. These are used for all function block configuration and editing. Tag names can now be added, and edited, in the function block properties edit panel.
Properties Panel	The Properties panel is now the focus for any controller or programming parameter editing. When a tool is selected from the tree, the Properties Panel changes to provide editing functionality for the selected tool. Individual property tabs for parameter editing have been removed.
Register Editor	The register editor now includes System Groups in addition to the user created Custom Groups. The system groups includes Forced, Tagged and Used register groups. The register editor listing, as seen in the properties panel, no longer contains the format column. To see the format of a register in the register listing simply hover the mouse over the register row. The register format is displayed in the hover text. Adding and editing registers is now done using popup windows. Multiple registers can have their configurations changed between Available Always and Available Online.
Shortcut Keys	Ctrl+T gives the Tree view keyboard focus. Ctrl+F6 navigates between the views in the application. Ctrl+E gives focus to the function block properties panel when the ladder canvas has focus.
Register Assignment	Individual register assignment modules can be added to the register editor using the mouse right click and then selecting Monitor Module from the menu displayed. Register assignment modules may then be added to an existing or new Register Editor group.
File Saving	Telepace Studio projects are now saved to a single file rather than the ten separate files in previous versions.



Installing Telepace Studio

To use Telepace Studio, you need to install the program on your system. The automated installation is straight forward and only takes a few minutes.

Some virus checking software may interfere with Setup. If you experience difficulties with Setup, disable your virus checker and run Setup again.

System Requirements

Telepace Studio is supported on these platforms.

- Windows XP Professional, on 32-bit platforms.
- Windows Vista Ultimate, on 32-bit and 64-bit platforms.
- Windows Vista Enterprise, on 32-bit and 64-bit platforms.
- Windows 7 Professional on 32-bit and 64-bit platforms.
- Windows 7 Ultimate on 32-bit and 64-bit platforms.

In addition to the recommended configuration for Windows on the platform, Telepace Studio requires the following resources.

- 250 MB of free disk space
- 256 MB RAM
- Minimum supported monitor resolution is 1024x768. Recommended monitor resolution is 1280x1024.
- .NET 3.5 framework

Pre-installation Checklist

1. The Telepace Studio CD is required.

Telepace Studio Installation Process

To install the Telepace Studio:

1. Insert the Telepace Studio CD in to your CD drive.
2. The install wizard will auto run.
3. Follow the setup directions on the screen.

Telepace Studio Registration Process

Telepace Studio encourages users to register their copy of Telepace Studio. Telepace Studio achieves this by prompting users to register on each start up until the registration process is initiated. When Telepace Studio is started for the first time the Product Registration dialog is opened as shown below.

- Clicking the **Register Later** button closes the Product Registration dialog and Telepace Studio starts normally. This option may be selected each time Telepace Studio is started.
- Clicking the **Register Now** button starts the default web browser directing the user to the Telepace Studio Registration page.

To complete the registration process complete the online form and click **Submit**. To reset the entered fields to default click the **Rest** button.

The Serial Number is printed on the Telepace Studio CD case. Contact Control Microsystems if the CD case is not available when the online registration is being completed.

Start Telepace Studio

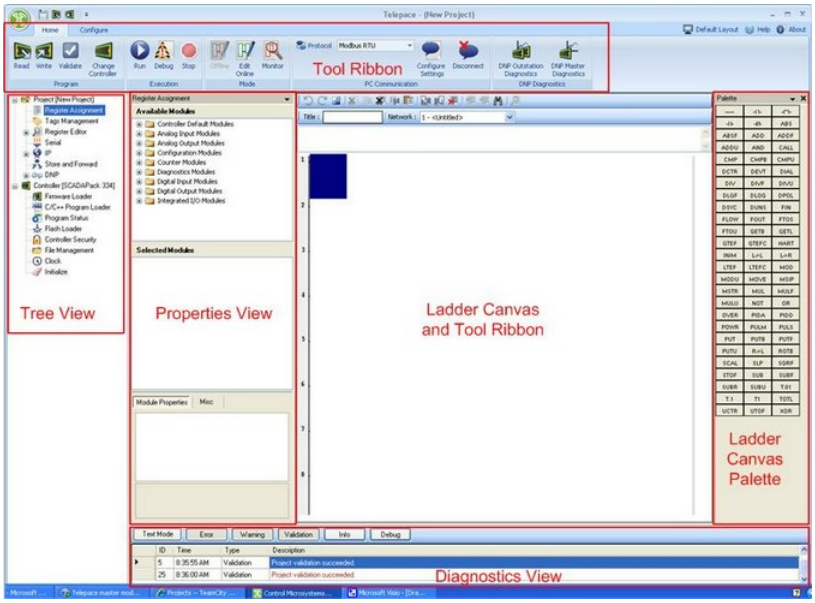
To run the ladder logic editor:

- Click **Start**, then select the Telepace Studio icon grouping, and select Telepace Studio.

Telepace Studio Environment

Telepace Studio is a powerful tool for developing, debugging, monitoring and documenting ladder logic programs. SCADAPack controllers executes ladder logic and C-application programs simultaneously, providing you with maximum flexibility in implementing your control strategy. This manual provides documentation on the Telepace Studio program including the Ladder Editor and the ladder logic programming language. We strongly encourage you to read it, and to notify us if you identify any items that you feel should be clarified, or included in future documentation releases.

Telepace Studio provides the tools needed for developing debugging, monitoring and documenting ladder logic programs.



The Tool Ribbon contains two tabs each containing a number of program groups. These groups contain commands for configuring parameters used in a ladder logic program.

The Tree View provides a listing of the configuration parameters used in ladder logic applications. These parameters are the same as those selectable from the Tool Ribbon.

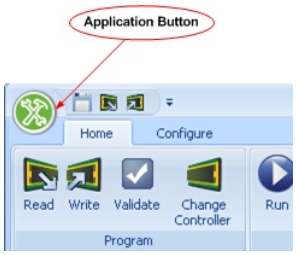
The Properties View is used to edit the parameters for selected items via the Tool Ribbon, Tree View commands or ladder logic function blocks. This view changes depending on the Ribbon or Tree View command selected.

The Ladder Canvas and Tool Ribbon contains commands used in creating and editing a ladder logic program.

The Ladder Canvas Palette contains the function blocks available for a ladder logic program.

Telepace Studio Application Button

The Telepace Studio application button opens a menu containing commands for saving and opening application projects, printing applications and setting the Telepace Studio user options. The Telepace Studio application button located in the upper left of the Telepace Studio user interface.



Once you click on the Telepace button, the following menu appears:

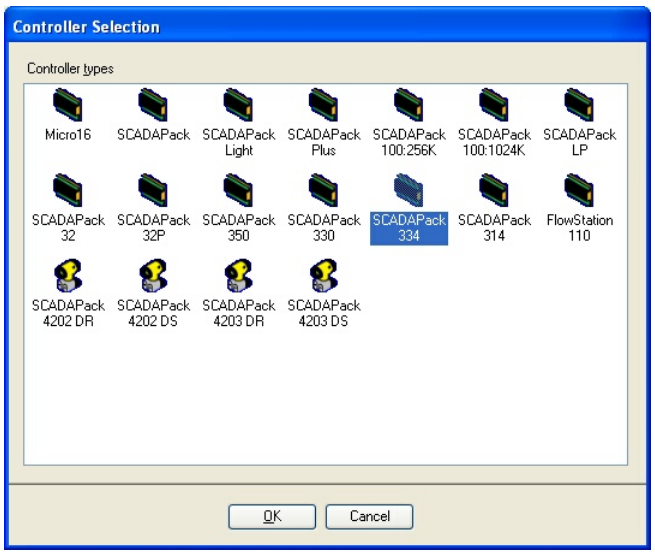
The menu associated with the Telepace Studio button is similar to the File menu in previous versions of Telepace. The Telepace Studio button provides access to the file management commands, print command and user options settings. The menu commands are displayed at the left, the Recent Projects list is displayed at the right and the Telepace

- **New:** start a new Telepace Studio project file.
- **Open:** open an existing Telepace Studio project file.
- **Save:** save a Telepace Studio project.
- **Save As:** save a Telepace Studio project with a different name or location.
- **Print:** open the print dialog to set the print format and print the Telepace project.
- **Recent Projects:** displays the last eight recently used Telepace projects.
- **Telepace Studio Options:** opens the Telepace Studio Options dialog to set the floating point format and Delete Network confirmation options.
- **Exit:** closes the Telepace Studio application.

New

The **New** command is used to start a new Telepace Studio project. This command is selected from the Telepace Studio Projects group or from the Telepace Studio button.

When selected, the **New** command, opens the **Controller Selection** dialog.



Telepace Studio Studio projects are closely integrated with the controller type. When starting a new Telepace Studio project the controller type needs to be selected.

To select the controller type for the new project:

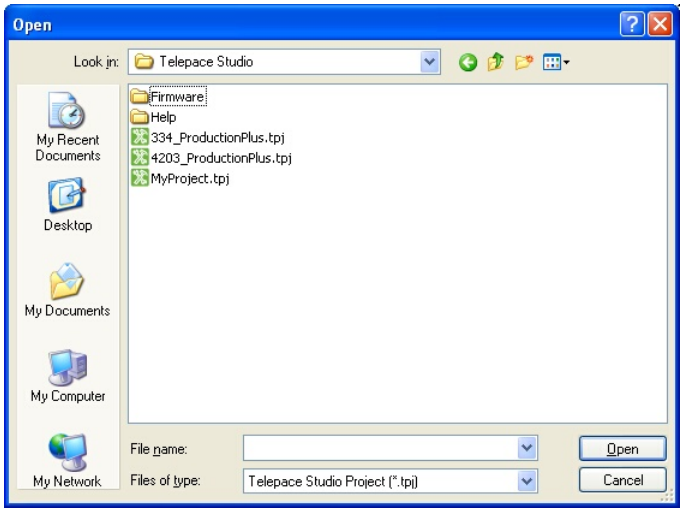
- Click on the **controller type** icon to highlight it.
- Click the **OK** button to select the controller type and close the dialog.

To close the dialog without selecting a new controller type click the **Cancel** button.

Open

The **Open** command is used to open an existing Telepace project. This command is selected from the Telepace Studio Projects group or from the Telepace Studio button.

When the **Open** command is selected, the Open dialog opened as shown below.



Use the **Look in:** drop down menu to navigate to the folder containing the Telepace Studio Project file or the Old File Format (Telepace 3.3x) file to be opened. Once the folder is selected files of the **Files of type:** selection are displayed.

Telepace Studio can open Telepace Ladder Logic files created with Telepace version 3.3x. When these files are opened they are converted to Telepace Studio project files.

The **Files of type:** selection is used to determine the types of files displayed in the Existing Projects window. The files that can be opened using this dialog are either Telepace Studio project files (.TPJ) or Telepace Ladder Logic files (.LAD).

- Use the **Telepace Studio Project (.TPJ)** selection to display only Telepace Studio projects that have been created with Telepace Studio.
- Use the **Old File Format (.LAD)** selection to display Telepace ladder logic files that were created using Telepace version 3.3x.

Opening a Project File

When a file is selected the file name and folder location are displayed in the **File name**: window.

- Click the **Open** button to open the file and close the Open dialog.
- Click the **Cancel** button to close the Open dialog without selecting a file.

Opening an Old File Format (.LAD) File

When a file is selected the file name and folder location are displayed in the **File name**: window.

- Click the **Open** button to open the file and close the Open dialog.
- Click the **Cancel** button to close the Open dialog without selecting a file.

Save

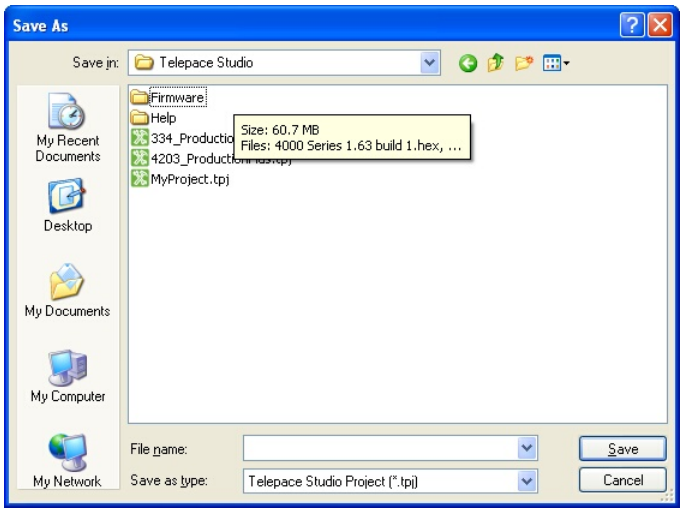
The **Save** command is used to save a Telepace Studio project. This command is selected from the Telepace Studio Application Button.

When the **Save** command is selected, the current Telepace Studio project is saved to the folder location selected when the Open command was used to open the project. When the New command was used to start the project the Save As dialog is opened and used to save the project.

Save As

The **Save As** command is used to save a Telepace Studio project using a different project file name or project file location. This command is selected from the Telepace Studio Projects group or from the Telepace Studio button.

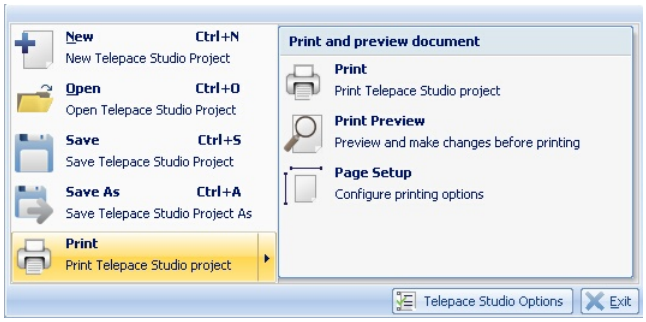
When the **Save As** command is selected, the Save As dialog is opened.



- The **Save in**: menu selection is used to select the folder location to save the project file.
- New folders can be created using the **Create New Folder** icon.
- The **File name:** combo box allows the direct typing of a file name or the selection of previously saved project files.
- The **Save** button save the Telepace Studio project file to the selected folder location and closes the dialog.
- The **Cancel** button closes the dialog without saving the Telepace Studio project file.
- Telepace Studio projects consist of a number of number of XML files for each project that is created. These files are updated with any new information when the project is saved.

Print

When the **Print** command is selected, a print menu opens to the right, on top of the Recent Projects list.



The following commands are selectable from the print menu. Click on the links below to get further information on these print menu commands.

- Print
- Print Preview
- Page Setup

Printing Differences from Telepace 3.3x

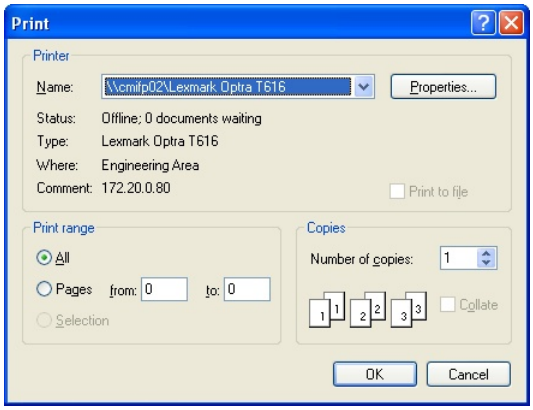
Some Telepace 3.3x print features are implemented differently in Telepace Studio.

- Telepace Studio measurement units are determined by the Windows system settings.
- Telepace Studio prints the date, time, and page header, and does not provide page start number option.
- Telepace Studio uses Arial font and sizes it automatically.
- The Ladder image on the cover page is not printed.
- The file name is not printed in the header section of each page.

See Print Topics for a complete list and description of the topic that are printed.

Print

The Print command opens a standard Windows Print dialog as shown below.



The **Name:** drop down menu is used to select the printer.

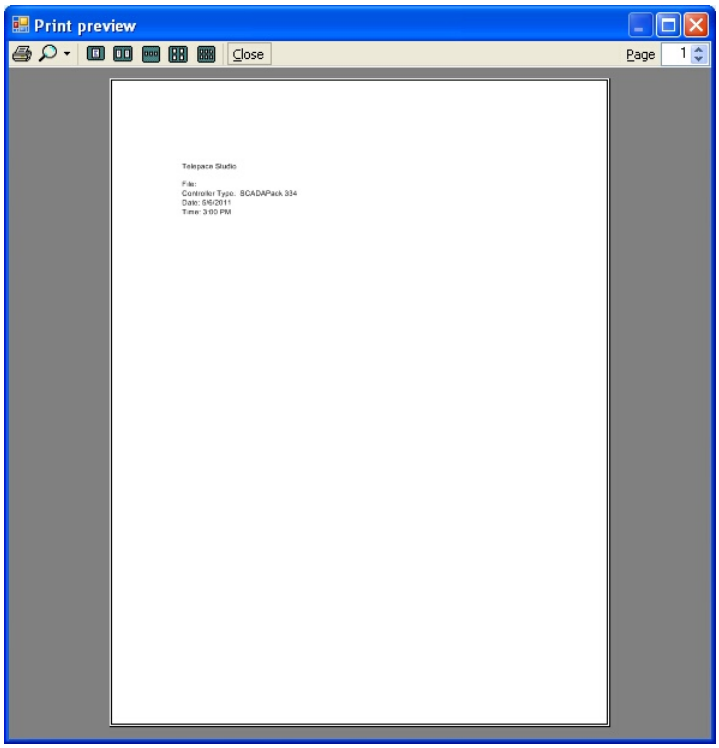
The **Properties** button opens a dialog to change the printer properties. The contents of this dialog are determined by the printer driver on the PC.

The **Print range** option selects the range of pages printed. The All and Pages options are available and Selection option is disabled.

The **Copies** option selects the number of copies printed, and whether the pages are collated.

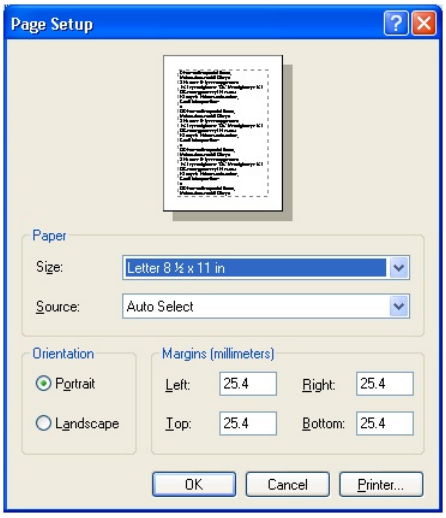
Print Preview

The **Print Preview** command opens a standard Windows Print Preview dialog. The Preview window provides a preview of the print topics before printing to the selected printer.



Page Setup

The Page Setup command opens the standard Windows page setup dialog. The page setup dialog provides options to change paper size, paper source, orientation and margin spacing.



Print Topics

The topics listed below are printed when the Telepace Studio project is printed. Telepace does not provide a print items selection as did Telepace version 3.3x.

To see the print items included in each topic click your mouse on the topic below. Use the + and - controls at the side of each topic to open and close the print items description for each topic.

- Cover Page
  The cover page lists the application name, project file name, controller type, date, and time.
- Address Tags sorted by Tag
  This topic lists Tag references for Addresses sorted by Tag in ascending order. The list is displayed with Tag and Address columns.
- Address Tags sorted by Address
  This topic lists Tag references for Addresses sorted by Address in ascending order. The list is displayed with Address and Tag columns.
- Constant Tags sorted by Tag
  This topic lists Tag references for Constants sorted by Tag in ascending order. The list is displayed with Tag and Constant columns.
- Network Titles
  This topic lists the network titles of each ladder logic network in the project.
- Ladder Logic Network
  Each Ladder Logic Network is printed starting at Network number 1. Networks are printed one to a page and include Network Comments and Element Configuration information for logic elements that use Element Configuration.
- Coils In Use
  This topic lists coils used in networks are displayed with Address and Tag columns.
- Register Used
  This topic lists register addresses used and their locations in networks. The list is printed with Address and Location (Network:Row:Column) columns.
- Register Assignment
  This topic lists the Register Assignment with these columns:
  Module
  Address
  Start Register
  End Register
  Registers
  Parameters are printed if the register assignment module has configuration parameters.
  I/O Module Error Indication is printed.
- Outputs On Stop Options
  This topic lists the Outputs on Stop Options. Outputs on Stop settings determine the state of the controller outputs when the ladder logic program is not executing. Digital outputs can be held at their last value or set to the OFF state.Analog outputs can be held at their last level or set to zero scale.
- Serial Port Settings
  This topic lists the controller serial port settings for serial ports on the controller. Each serial port has following sub items:
  Port
  Protocol
  Addressing
  Station
  Duplex
  Baud Rate
  Data Bits
  Parity
  Stop Bits
  Rx Flow
  Tx Flow
  Port Type
  Store and Forward (Routing for DNP protocol)
  Enron Modbus
  Enron Station
- TCP/IP Settings
  This topic lists the TCP/IP Configuration Settings for the controller, if TCP/IP is supported by the controller.
  IP Address
  Subnet Mask
  Gateway (LAN or PPP)
- Modbus IP Settings
  Modbus IP Settings lists the following items:

  Modbus Common
  Addressing
  Station
  Store and Forward

Enron Modbus
Enron Station

Modbus TCP Protocol
Server
Master Idle Timeout
Server Idle Timeout
TCP Port

Modbus RTU over UDP Protocol
Server
UDP Port

Modbus ASCII over UDP Protocol
Server
UDP Port

DNP in TCP Protocol
Server
Master Idle Timeout
Server Idle Timeout
TCP Port

DNP in UDP Protocol
Server
UDP Port

PPP Configuration

If the controller type supports PPP, then PPP Configuration lists the following items for each serial port:
Port
Auto Answer (PPP Server)
Local IP address - listed only If PPP server enabled
Local Subnet Mask - listed only If PPP server enabled
Remote IP address - listed only If PPP server enabled
Remote Specified IP - listed only If PPP server enabled
Authentication - listed only If PPP server enabled
Inactivity Timeout - listed only If PPP server enabled

Friendly IP List

Friendly IP List lists the following items:
Friendly IP List enabled
for each entry in the list
Start address
End address

Store and Forward Settings

If Store and Forward is supported on the controller then the Store and Forward Settings lists the following items:
Slave Interface
Slave Station
Forward Interface
Forward Station
Forward IP Address
Timeout

DNP – Node Identification

This topic lists the DNP Addressing for the controller. The RTU Address and Master Addresse(s) are listed. When DNP is not used the topic lists the Telepace default values.

DNP – Event Management

This topic lists the DNP Event Management settings. When DNP is not used the topic lists the Telepace default values.The topic lists the event management items:

Event Storage Objects
Binary Input
16-Bits Analog Input
32-Bits Analog Input
Short Floating Point Analog Input
16-Bits Counter Input
32-Bits Counter Input

For each object, the sub items are:
Event Reporting Method
Event Buffer Size

Event Transmission
Enable Unsolicited Events for Class 1, Class 2, and Class 3.

Event Initialization
Enable Unsolicited Events on Start UP
Send Initial Unsolicited Event on Start Up
Enable Resend Unreported Events Wait Time
Resend Unreported Events Wait Time if it is enabled.

DNP – Communication

This topic lists the DNP Routing, DNP Protocol and DNP Dial UP. When DNP is not used the topic lists the Telepace default values. The following settings are printed:
DNP Routing
DNP Routing entries are printed and include the following information.
Remote Station Address
Port to Remote
Data Link Retries
Data Link Timeout
IP Address – if the DNP in TCP/UDP is selected.
Primary Phone – if serial port is selected.
Secondary Phone - if serial port is selected.

Protocol Settings
Application Layer
Confirm
Retries
Timeout
Maximum Fragment Length

Data Link Layer
Confirm
Retries
Timeout

Time Synchronization
Time Synchronization Type

Time Synchronization Interval – if interval type is selected.

Select Before Operation
Operate Timeout

Interoperability
Report only Level 2 Objects in Class Poll
Limit Maximum Events in Response

Dial Up
Modem Initialization
Attempts
Dial Type
Connect Timeout
Inactivity Timeout
Pause Time

DNP – Input/Output

This topic lists the Allow Duplicate Modbus Address state and DNP Input/Output objects settings. For each object type, the group settings and the entries for the object are printout.

Allow Duplicated Module Address
Binary Input
Start DNP Point Number
Number of Points
Binary Input entries are listed with following columns:
DNP Point Number
Modbus Address
Event Class
Debounce
16-Bits Analog Input
Start DNP Point Number
Number of Points
16-Bits Analog Input entries are listed with following columns:
DNP Point Number
Modbus Address
Event Class
Deadband
32-Bits Analog Input
Start DNP Point Number
Number of Points
32-Bits Analog Input entries are listed with following columns:
DNP Point Number
Modbus Address 1
Modbus Address 2
Event Class
Deadband
Short Float Analog Input
Start DNP Point Number
Number of Points
Word Order
Short Float Analog Input entries are listed with following columns:
DNP Point Number
Modbus Address 1
Modbus Address 2
Event Class
Deadband
16-Bits Counter Input
Start DNP Point Number
Number of Points
16-Bits Counter Input entries are listed with following columns:
DNP Point Number
Modbus Address
Event Class
Deadband
32-Bits Counter Input
Start DNP Point Number
Number of Points
Word Order
32-Bits Counter Input entries are listed with following columns:
DNP Point Number
Modbus Address 1
Modbus Address 2
Event Class
Deadband
Binary Output
Start DNP Point Number
Number of Points
Binary Output entries are listed with following columns:
DNP Point Number
Modbus Address 1
Modbus Address 2
Control Type
16-Bits Analog Output
Start DNP Point Number
Number of Points
16-Bits Analog Output entries are listed with following columns:
DNP Point Number
Modbus Address
32-Bits Analog Output
Start DNP Point Number
Number of Points
32-Bits Analog Output entries are listed with following columns:
DNP Point Number
Modbus Address 1
Modbus Address 2
Short Float Analog Output
Start DNP Point Number
Number of Points
Short Float Analog Output entries are listed with following columns:
DNP Point Number
Modbus Address 1
Modbus Address 2

DNP – Master

This topic lists the DNP Master Layer, Master Poll, Master Address Mapping settings, and Map Change Events.

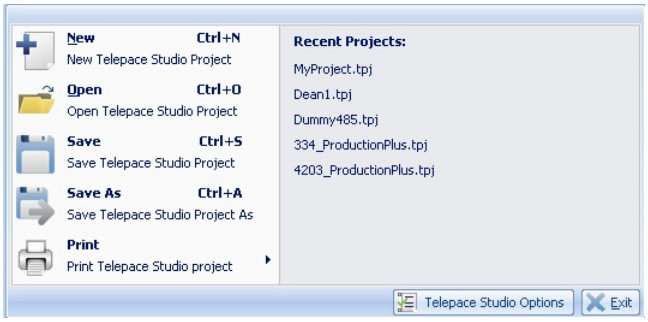
Master Layer
Enable Mimic Mode
Base Poll Interval
Master Poll Schedules
The Master Poll Schedules are printed. Each entry has the following columns:
Station
IIN Flags Modbus Register
Class 0 Poll Rate
Class 0 Poll Offset – if it is enabled.
Class 1 Poll Rate
Class 1 Poll Offset – if it is enabled.
Class 1 Limit Maximum Events – if it is enabled.
Class 2 Poll Rate
Class 2 Poll Offset – if it is enabled.
Class 2 Limit Maximum Events – if it is enabled.
Class 3 Poll Rate
Class 3 Poll Offset – if it is enabled.
Class 3 Limit Maximum Events – if it is enabled.
Time Synchronization Poll Rate
Remote Class 1 Event Acceptance
Remote Class 2 Event Acceptance
Remote Class 3 Event Acceptance
Address Mapping
Map Change Events
All address mapping entries are printout with columns:
Station
Object Type
Remote First Point
Number of Points
Local First Register

- Register Editor Groups
  This topic lists the contents of Register Groups in the Register Editor.

  These columns are printed:
  Register
  Tag
  Value for registers available Always
  Format
  Force for registers available Always
  Available
  Configuration
- Index
  The index lists the start page number for each print topic.

Recent Projects List

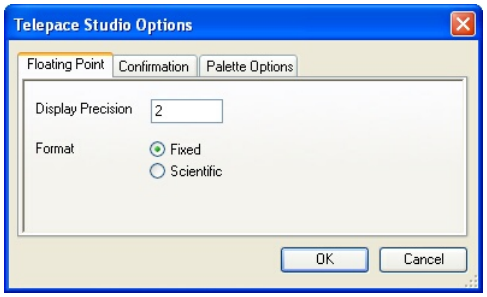
The **Recent Projects** list provides an updated list of the last eight accessed Telepace projects.



To select and open a project from the Recent Projects, list click on the project in the list. The selected file is opened and the Application Button menu is closed.

Telepace Studio Options

The Telepace Studio Options command opens the **Telepace Studio Options** dialog as shown below. There are three tab selections for setting the Telepace Studio options for Floating Point format, Confirmation prompts and Function Block palette location.



Floating Point Options

The Floating Point options dialog is opened by clicking the **Floating Point Options** tab. The Floating point options have two configuration items, **Display Precision** and **Format**.

The **Display Precision** entry defines the number of digits following the decimal point. The range of valid entries is 1 through 7 with a default value of 2.

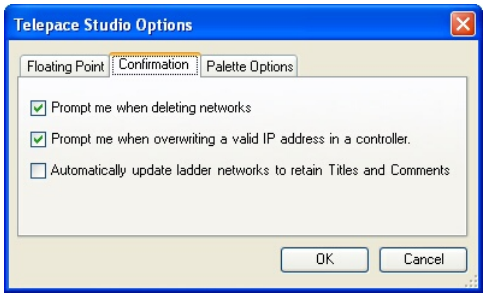
The **Format** selection defines how floating point numbers are displayed in the Telepace Studio project.

- The **Fixed** option displays numbers in the format ####.##.
- The Scientific option displays numbers in the format #.### E###.

As multiple views are used in Telepace Studio views which contain floating point values are updated when the floating point format is changed.

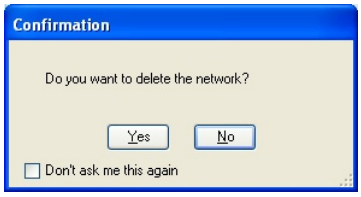
Confirmation Option

The Confirmation dialog is opened by clicking the **Confirmation** tab. The confirmation option is used to select whether a prompt is presented when the Delete Network command is selected.



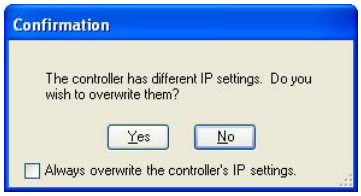
Click the **Prompt me when deleting networks** check box to be presented with a confirmation window whenever the Delete Network command is selected. When unselected the option does not provide a prompt when the delete network command is selected. The option is checked by default.

When the **Prompt me when deleting networks** option is selected the following confirmation dialog is displayed when the delete network command is selected.



- Select **Yes** to delete the network and close the dialog.
- Select No to close the dialog without deleting the network.
- Check the Don't ask me **this again** check box to disable the Prompt. When the option is disabled the confirmation dialog is no longer displayed.

Click the **Prompt me when overwriting a valid IP address in a controller** check box to be presented with a confirmation window whenever the IP address for a controller is changed. When unselected the option does not provide a prompt when the IP address is changed. The option is checked by default.

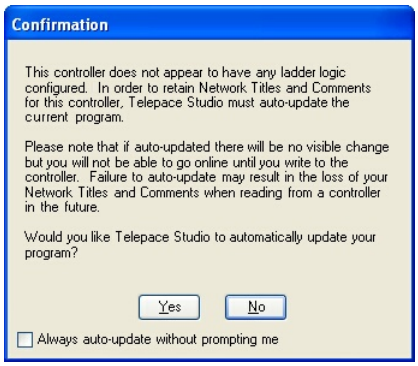


- Select **Yes** to overwrite the IP settings and close the dialog.
- Select **No** to close the dialog without overwriting the IP settings.
- Check the **Don't ask me this again** check box to disable the Prompt. When the option is disabled the confirmation dialog is no longer displayed.

Click the **Automatically update ladder networks to retain Titles and Comments** check box to be presented with a confirmation window whenever any of the following occurs:

- When a Telepace 3.x or 4.x file, containing logic, is opened using Telepace Studio 5.x.
- When Telepace 4.x file, containing logic, is written to a controller and then read back using Telepace Studio 5.x.
- When a controller is cold booted and then read from before anything is written to the controller.

When unselected the option does not provide a prompt. The option is unchecked by default.



- Select **Yes** to automatically update the logic program.

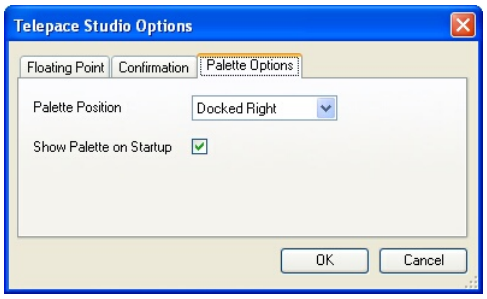
- Select **No** to close the dialog without updating the logic program.
- Check the **Always auto-update without prompting me** check box to disable the Prompt. When the option is disabled the confirmation dialog is no longer displayed.

Palette Options

The Palette Options dialog is opened by clicking the **Palette Options** tab. The palette options set the location of the Function Block palette. The palette may be docked to the left or right of the Ladder Canvas or floating.

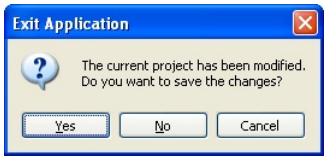
The **Palette Position** selection defines where the function block palette is displayed in the Telepace Studio project.

- The **Docked Left** option displays the function block palette to the left of the ladder canvas.
- The **Docked Right** option displays the function block palette to the right of the ladder canvas.
- The **Floating** option displays the function block palette as a floating view.
- The **Show Palette on Startup** check box option selects displaying of the palette each time Telepace Studio is started.



Exit

The Exit button closes Telepace Studio. If a Telepace project has been edited, the Exit Application dialog is displayed prompting a save of the project.



Click the **Yes** button to save the current project or to open the File Save As dialog if the current project is a new project.

Click the **No** button to exit Telepace Studio without saving any changes.

Click the **Cancel** button to close the dialog and return to the current Telepace project.

Quick Access Toolbar

The **Quick Access Toolbar** is the small area to the upper left of the Ribbon bar. It contains commands you use frequently.

Telepace Studio has three default quick access buttons. These are Save, Read and Write.

You can set up the **Quick Access Toolbar** so that a command you use often, for example, debug ladder logic, is on the **Quick Access Toolbar**.

- Right-click on the command icon you want to add to the **Quick Access Toolbar**.
- Click **Add to Quick Access Toolbar.**

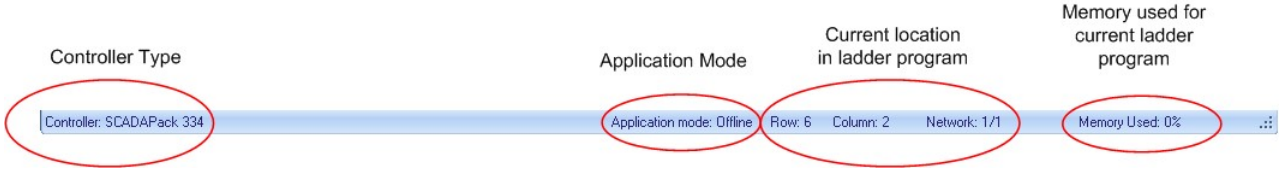
Clicking the drop down menu at the right of the Quick Access Toolbar opens a menu selection.

- Select the **Show Below the Ribbon** command to move the Quick Access Toolbar under the Ribbon. Once the Quick Access Toolbar is located under the Ribbon it can be moved above the Ribbon by clicking the drop down menu and selecting **Show Above the Ribbon**.
- Select the **Minimize the Ribbon** command to minimize the ribbon and display only the Ribbon Tabs. The Ribbon is opened when a Tab is selected and closed once a command is selected or you navigate away from the Ribbon. To restore the Ribbon, uncheck the **Minimize the Ribbon** command.

When changes are made to the **Quick Access Toolbar,** they are saved and used each time you open Telepace.

Status Bar

The Status Bar is located at the bottom of the Telepace Studio main screen. The Status Bar provides information for the Ladder Logic application.

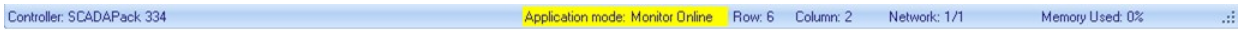


The Controller Type section indicates the Controller Type selected for this application.

The Application Mode section indicates the current status of ladder logic application. This status is one of Offline, Edit Online and Monitor Online modes.

The Current Location in Ladder Program section indicates the cursor location in the Ladder Canvas.

The Memory Used section provides a visual indication of the amount of logic memory used in the application.



Default Layout

The **Default Layout** button returns Telepace Studio to the default layout.

To restore the Telepace default layout:

- Click the **Default Layout** button. Telepace Studio will rearrange itself to the default layout. No project configurations are lost when the Default Layout button is used.

Help

The **Help** button opens the Telepace Studio User and Reference Manual help manual. This help manual is in Compiled Help file format (.CHM). The help file is fully keyword searchable and provides enhanced navigation of help topics.

About

The About button opens the About Telepace dialog as shown below. The About dialog provides information on the version of Telepace and contact information for Technical Support.

Telepace Studio Studio File Locations

The default file locations for files used by Telepace are listed below. The default file locations differ depending on the Window OS being used.

Telepace Studio Saved Ladder Logic Files

The Telepace Studio application files are saved to the following default file location.

Windows 7:

C:\Users\username\Documents\Telepace Studio Projects

Windows Vista:

C:\Users\username\Documents\Telepace Studio Projects

Windows XP:

C:\Documents and Settings\username\My Documents\Telepace Studio Projects

Telepace Studio Firmware Files

This folder contains controller firmware files.

C:\Program Files\Control Microsystems\Telepace Studio\Firmware

Hot Keys and Shortcuts

Hot keys and shortcuts provide keyboard access to some common Telepace commands. The shortcut keys are **Alt** keys, **Ctrl** keys and **Function** keys.

A complete display of **Alt** key shortcuts are shown when the Alt key is pressed. Telepace commands that support Alt key shortcuts will display a letter or number under the command.

The following **Ctrl** key and **Function** key shortcuts are available:

Ctrl Keys

For **Ctrl** key shortcuts hold the Ctrl key down and press the letter key for the shortcut desired. The letter key needs to be lower case.

Ctrl - E When the Ladder Canvas has focus this gives the function block Properties view keyboard focus.

Ctrl - F Open Search Dialog

Ctrl - N Open New File

Ctrl - O Open File

Ctrl - S Save File

Ctrl - C Copy Selected

Ctrl - T Give the Tree view keyboard focus.

Ctrl - V Paste Selected

Ctrl - Z Undo

Ctrl - F6 Cycles keyboard focus between Tree view, Properties Panel view, Ladder Canvas view and Diagnostics view.

Function Keys

For **Function** key shortcuts press the function key for the shortcut desired.

F1 Open Help File

F2 Navigate to Previous Search Result (applies only when Search In Network(s) dialog is open)

F3 Navigate to Next Search Result (applies only when Search In Network(s) dialog is open)

F5 Toggle Vertical Shunt

F6 Insert Horizontal Shunt

F7 Go To Previous Network

F8 Go To Next Network

F10 Toggle Alt Key display

Other Hot Keys

/ When the Ladder Canvas has focus and the cursor is on a contact this toggles between NO (normally open) and NC (normally closed) contacts.

Telepace 3.3x Commands in Telepace Studio

This section provides a list of the commands with which Telepace 3.3x users are familiar. The new user interface replaces menus, toolbars, and many of the task panes that were used in Telepace 3.3x. The new user interface is designed to make it easier to find the features needed to perform different tasks when working with Telepace. This section is for experienced Telepace 3.30 users who want to find familiar commands quickly in Telepace Studio.

The order of the subtopics in this section matches the default order for Telepace version 3.4x. The subtopic headings match the menu and toolbar names in Telepace 3.3x.

File Menu

Telepace 3.4x Location	Telepace Studio
New	Telepace Application button | New
Open	Telepace Application button | Open
Save	Telepace Application button | Save
Save As	Telepace Application button | Save As
Page Setup	Telepace Application button | Print menu
Print	Telepace Application button | Print menu
Select Print Items	Telepace prints the open Ladder Logic project. You no longer need to select the items to print.
Exit	Telepace Application button | Exit button

Edit Menu

Telepace 3.4x Location	Telepace Studio
Undo	Ladder Canvas Toolbar | Undo
Cut Selected	Ladder Canvas Toolbar | Cut
Copy Selected	Ladder Canvas Toolbar | Copy

Cut Networks	Ladder Canvas Toolbar | Cut Networks
Copy Networks	Ladder Canvas Toolbar | Copy Networks
Paste	Ladder Canvas Toolbar | Paste
insert vertical shunt	Right click on the Ladder Canvas and select Toggle Vertical Shunt from the menu.
insert element	Right click on a cell in the Ladder Canvas. Drag a function block from the palette. Click on a cell and start typing the name of a function block.
insert column	not supported in Telepace Studio
insert row	not supported in Telepace Studio
insert network before	Ladder Canvas Toolbar | Insert Before
insert network after	Ladder Canvas Toolbar | Insert After
delete vertical shunt	Select the cell on the Ladder Canvas and click Delete Right click the cell and select Toggle Vertical Shunt
delete elements	Select the function block and click Delete. Right click the function block and select Delete from the menu.
delete column	not supported in Telepace Studio
delete row	not supported in Telepace Studio
delete network	Ladder Canvas Toolbar | Delete Network
delete all networks	Ladder Canvas Toolbar | Cut Networks
Toggle Vertical Shunt	Right click on the Ladder Canvas and select Toggle Vertical Shunt from the menu.
Tag Names	Configure Ribbon | I/O Group | Tags Management Select Tags Management for the Tree.
Erase All Tags	Configure Ribbon | I/O Group | Tags Management Select Tags Management from the Tree.
Export Tag Names	Configure Ribbon | I/O Group | Tags Management Select Tags Management from the Tree.
Import Tag Names	Configure Ribbon | I/O Group | Tags Management Select Tags Management from the Tree.
Network Title	Ladder Canvas Toolbar | Title
Element Configuration	Function Block Properties
Registers	Configure Ribbon | I/O Group | Register Editor Select Register Editor from the Tree.

Search Menu

Telepace 3.4x Location	Telepace Studio
Next Network	Ladder Canvas Toolbar | Next
Previous Network	Ladder Canvas Toolbar | Previous
Go Network	Ladder Canvas Toolbar | Network
Find Address	Ladder Canvas Toolbar | Search
Find Device	Ladder Canvas Toolbar | Search
Find Tag Name	Configure Ribbon | I/O Group | Tags Management Select Tags Management from the Tree. Configure Ribbon | I/O Group | Register Editor | Tagged Registers
Repeat Last Find	Ladder Canvas Toolbar | Search
Replace Address	not supported in Telepace Studio
Multiple Coils	Ladder Canvas Toolbar | Search

Controller Menu

Telepace 3.4x Location	Telepace Studio
Type	Status Bar | Controller:
Serial Ports	Configure Ribbon | Communication Group | Serial Select Serial from the Tree.
IP Configuration	Configure Ribbon | Communication Group | IP
Register Assignment	Configure Ribbon | I/O Group | Register Assignment Select Register Assignment from the Tree.
Outputs on Stop	Configure Ribbon | I/O Group | Register Assignment. Select Register Assignment from the Tree.
Store and Forward	Configure Ribbon | Communication Group | Store and Forward Select Store and Forward from the Tree.
DNP Configuration	Configure Ribbon | Communication Group | DNP Configuration Select DNP from the Tree.
DNP Status	Home Ribbon | DNP Diagnostics | DNP Outstation Diagnostics
DNP Master Status	Home Ribbon | DNP Diagnostics | DNP Master Diagnostics
Initialize	Configure Ribbon | Controller Group | Initialize
Real Time Clock	Configure Ribbon | Controller Group | Clock Select Clock from the Tree.
Monitor Element	Ladder Canvas | Right Click | Monitor Element Ladder Canvas Toolbar | Monitor Element
List Forced Registers	Configure Ribbon | I/O Group | Register Editor | Forced

	Registers Select Register Editor | Forced Registers from the Tree.
Remove All Forces	Configure Ribbon | I/O Group | Register Editor | Forced Registers Select Register Editor | Forced Registers from the Tree.
Lock Controller	Configure Ribbon | Controller Group | Controller Security Select Register Editor | Forced Registers from the Tree.
Unlock Controller	Configure Ribbon | Controller Group | Controller Security Select Controller Security from the Tree
Override Controller Lock	Configure Ribbon | Controller Group | Controller Security Select Controller Security from the Tree
Show Locked Status	Configure Ribbon | Controller Group | Controller Security Select Controller Security from the Tree
C/C++ Program Loader	Configure Ribbon | Programs Group | C/C++ Program Loader Select C/C++ Program Loader from the Tree
Flash Loader	Configure Ribbon | Programs Group | Flash Loader Select Flash Loader from the Tree
Program Status	Configure Ribbon | Programs Group | Program Status Select Program Status from the Tree

Communications Menu

Telepace 3.4x Location	**Telepace Studio**
Read from Controller	Home Ribbon | Program Group | Read
Write to Controller	Home Ribbon | Program Group | Read
PC Communications Settings	Home Ribbon | PC Communication Group
Connect to Controller	Home Ribbon | PC Communication Group
Disconnect from Controller	Home Ribbon | PC Communication Group

Activity Menu

Telepace 3.4x Location	**Telepace Studio**
Edit Off Line	Home Ribbon | Mode Group | Edit Off Line
Edit On Line	Home Ribbon | Mode Group | Edit On Line
Monitor	Home Ribbon | Mode Group | Edit On Line

Operation Menu

Telepace 3.4x Location	**Telepace Studio**
Stop	Home Ribbon | Execution Group | Stop
Debug	Home Ribbon | Execution Group | Debug
Run	Home Ribbon | Execution Group | Run

Options Menu

Telepace 3.4x Location	**Telepace Studio**
Screen Font	Telepace Studio uses the Window system font.
Colors	not supported in Telepace Studio
Floating Point Settings	Application Button | Telepace Studio Options
Tool Bar	not used in Telepace Studio
Title Bar	Title Bar
Status Bar	Status Bar
Single Tag Names	Tag name and Address are displayed in Telepace Studio.
Double Tag Names	Tag name and Address are displayed in Telepace Studio.
Tag and Address	Tag name and Address are displayed in Telepace Studio.
Numeric Address	Tag name and Address are displayed in Telepace Studio.
Allow Multiple Coils	Supported in Telepace Studio
Messages	Application Button | Telepace Options

Help Menu

Telepace 3.4x Location	**Telepace Studio**
Contents	Telepace Menu Bar | Help
About Program	Telepace Menu Bar | About

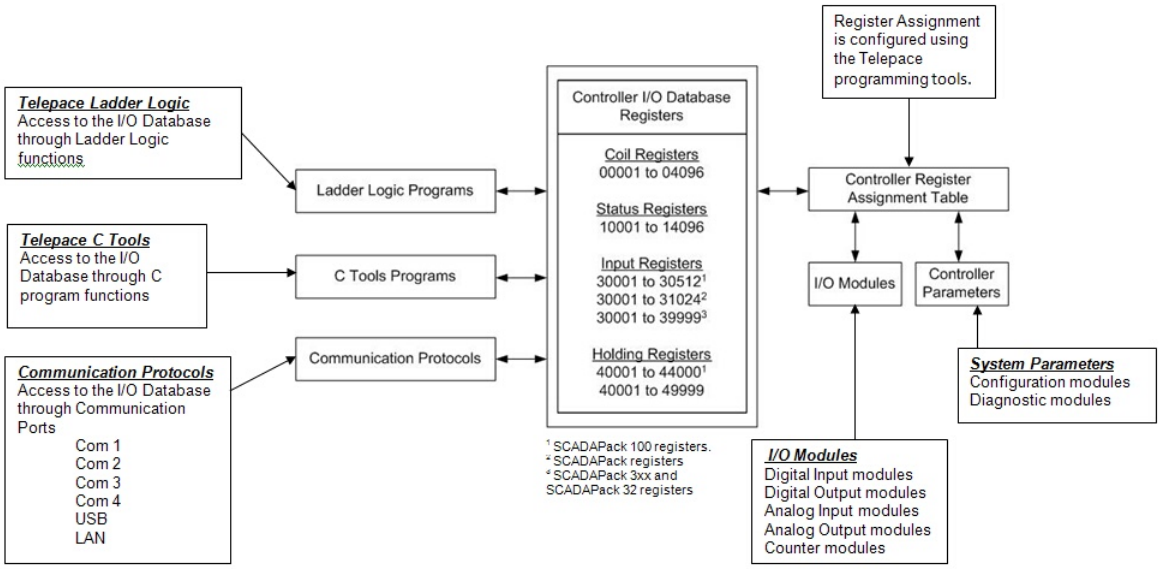
Telepace Ladder Logic Fundamentals

Telepace Studio is the programming environment for developing Ladder Logic programs for SCADAPack controllers. At the heart of of all SCADAPack controllers is the I/O Database (Input / Output Database). The I/O Database is simply a set of Modbus registers that conform to Modbus register naming conventions.

- **Coil Registers**, or digital output registers, are single bit registers. These registers use the five digit naming convention **0xxxx**. For example digital output register 100 is named 00100. These registers are read and write registers. This means the register value can be changed by writing to it or the value may be read using Telepace Studio.
- **Status Registers**, or digital input registers, are single bit registers. These registers use the five digit naming convention **1xxxx**. For example digital input register 200 is named 10200. These registers are read only registers. This means the register value can only be read using Telepace Studio.
- **Input Registers**, or analog input registers, are sixteen bit registers. These registers use the five digit naming convention **3xxxx**. For example analog input register 300 is named 30300.These registers are read only registers. This means the register value can only be read using Telepace Studio.
- **Holding Registers**, or analog output registers, are single bit registers. These registers use the five digit naming convention **4xxxx**. For example analog output register 400 is named 40400. These registers are read and write registers. This means the register value can be changed by writing to it or the value may be read using Telepace Studio.

Using the I/O Database

A simplified overview of the controller I/O data base is shown in the following diagram.



A Telepace Studio program typically needs to provide the following functions:

- **Control** the local controller I/O (Inputs and Outputs) using Function Blocks and Register Assignment.
- **Communicate** with other controllers and/or with a SCADA Host using available communication ports.
- **Interact** with C/C++ programs such as Realflo or custom user C/C++ applications through the I/O Database.
- **Diagnostic** information for control and communication operations using Register Assignment.

Function Blocks

In a fundamental sense function blocks used in Telepace Studio control (read and/or write) the data in the I/O database. For example a SCAL function block is used to read a value (4xxxx or 3xxxx registers) and provide a scaled output value (4xxxx registers). The registers used in the function block need to meet the requirements of the function, that is the input value is read (3xxxx and 4xxxx registers can be read) and the output value is written (4xxxx registers can be written).

Register Assignment

SCADAPack controllers have an internal real time operating system (RTOS) that controls the physical operation of the controller. This physical operation includes digital and analog inputs, digital and analog outputs, communication port handling and task management control for internal operations. Register Assignment provides a means to map physical data to the appropriate I/O Database registers.

- Physical data that is read only, such as a digital input wired to an On/Off switch or a transmitter input wired to a tank level meter, needs to be mapped to 1xxxx or 3xxxx type registers.
- Physical output data, such as digital out put controlling a beacon or an analog output that is controlling a valve positioner, needs to be mapped to 0xxxx or 4xxxx type registers.
- Diagnostic data such as protocol status (i.e. protocol messages sent and received), Ladder Logic program status (i.e. running or stopped) or the number of current IP connections are some examples of diagnostic information that may be useful in applications. As this data is read from the RTOS it needs to be mapped to 1xxxx or 3xxxx type registers.
- Configuration data such as serial port baud rate, controller IP address or protocol station number allows parameters to be changed in the controller through the I/O database. As this data is written to the RTOS it needs to be mapped to 0xxxx or 4xxxx type registers.

Communication Protocols

SCADAPack controllers typically communicate with other controllers and SCADA Host applications using a communications protocol such as Modbus, DNP3 or DF1. Protocol messages contain data that is read from or written to an external device (i.e SCADA Host). This data is written to or read from the I/O database automatically by the RTOS. This powerful feature provides a number of benefits to the Telepace Studio programmer.

- Data received from another controller or SCADA Host is automatically placed into the I/O database and available for Ladder Logic programs. The function blocks used in a Ladder Logic program access this data for use in the function blocks. For example a PIDA function may have the setpoint (or any other PID parameter) updated by a SCADA Host without any programming or user interaction.
- Local physical I/O may be controlled from another controller or SCADA Host when the physical I/O is assigned to the I/O Database using the Register Assignment modules.
- Data that needs to be sent to or read from another controller uses communication function blocks such as the MSTR or MSIP. These function blocks use I/O database registers that have been written to by function blocks or register assignment modules.

Interaction with C/C++ Programs

SCADAPack controllers may run one or more C/C++ programs concurrently with the Ladder logic program. An example of this would be a controller that is running Realflo and a Ladder Logic program. Realflo data such as instantaneous flow rate can be put into the I/O Database and the Ladder Logic program could use this data to perform a control function.

Diagnostic Information

The major components of a ladder program consist of ladder networks, ladder function blocks, and associated network comments. It is important to fully understand each of these major elements and how they are executed as a program within the controller.

The Ladder Logic I/O Database is an integral part of any logic program. Each function block uses I/O Database registers and may pass or receive information from other function blocks using the database.

Ladder Logic Function Blocks provide the programming functions for a logic program.

Ladder Logic Networks are a collection of one or more function blocks performing a programming function.

Telepace I/O Database Registers

The I/O database is different on the 16 and 32-bit controllers.

The 16-bit controllers comprise of SCADAPack (Light and Plus), SCADAPack 100, SCADAPack LP, SCADAPack 4202 DR and SCADAPack 4202 DS. The SCADAPack 100: 1024K controller differs from the SCADAPack 100: 256 K controller in that it has the same I/O database as other SCADAPack controllers. The SCADAPack 100: 1024K controller has firmware version 1.80 or newer and a controller ID that is greater than or equal to A182922.

The 32-bit controllers comprise SCADAPack 314, SCADAPack 33x, SCADAPack 35x, SCADAPack 32, SCADAPack 4203 DR and SCADAPack 4203 DS.

The I/O database contains general purpose and user-assigned registers. Ladder Logic and C application programs to store processed information and to receive information from a remote device may use general-purpose registers. Initially every register in the I/O database are general-purpose registers.

User-assigned registers are mapped directly from the I/O database to physical I/O hardware, or to controller system configuration and diagnostic parameters. The Register Assignment performs the mapping of registers from the I/O database to physical I/O hardware and system parameters.

User-assigned registers are initialized to the default hardware state or system parameter when the controller is reset. Assigned output register values are not maintained their values during power outages. Assigned output registers do retain their values during application program loading.

General-purpose registers retain their values during power outages and application program loading. The values change only when written by an application program or a communication protocol.

The TeleBUS communication protocols provide a standard communication interface to the controller. The TeleBUS protocols are compatible with the widely used Modbus RTU and ASCII protocols. They provide access to the I/O database in the controller.

The I/O database is divided into four types of I/O registers. Each of these types is initially configured as general purpose registers by the controller.

Coil Registers

Coil registers are single bit registers located in the digital output section of the I/O database. Coil, or digital output, database registers may be assigned to 5000 I/O digital output modules or SCADAPack I/O modules through the Register Assignment. Coil registers may also be assigned to controller on board digital outputs and to system configuration modules.

There are 4096 coil registers numbered 00001 to 04096. Ladder logic programs, C language programs, and the TeleBUS protocols can read from and write to these registers.

Status Registers

Status registers are single bit registers located in the digital input section of the I/O database. Status, or digital input, database registers may be assigned to SCADAPack I/O digital input modules or SCADAPack I/O modules through the Register Assignment. Status registers may also be assigned to controller on board digital inputs and to system diagnostic modules.

There are 4096 status registers numbered 10001 to 14096. Ladder logic programs and the TeleBUS protocols can only read from these registers. C language programs can read data from and write data to these registers.

Input Registers

Input registers are 16 bit registers located in the analog input section of the I/O database. Input database registers may be assigned to SCADAPack I/O analog input modules or SCADAPack I/O modules through the Register Assignment. Input registers may also be assigned to controller internal analog inputs and to system diagnostic modules.

There are 1024 input registers numbered 30001 to 39999. Ladder logic programs and the TeleBUS protocols can only read from these registers. C language programs can read data from and write data to these registers.

Holding Registers

Holding registers are 16 bit registers located in the analog output section of the I/O database. Holding, or analog output, database registers may be assigned to SCADAPack I/O analog output modules or SCADAPack analog output modules through the Register Assignment. Holding registers may also be assigned to system diagnostic and configuration modules.

There are 9999 holding registers numbered 40001 to 49999. Ladder logic programs, C language programs, and the TeleBUS protocols can read from and write to these registers.

Reserved Modbus Registers

The FlowStation and SCADAPack 4202/4203 controllers have Modbus register ranges mapped to specific controller functions. These registers cannot be used with register assignment and for element configuration. They can be used for any other uses.

Controller	Start	End
FlowStation 110	2049	4096
	12049	14096
	35000	39999
	45000	49999
SCADAPack 4202	40001	40499
SCADAPack 4203	40001	40499

Ladder Logic Memory Usage

Memory usage in a Ladder Logic application program is based on the number of networks and number of elements used by a program.

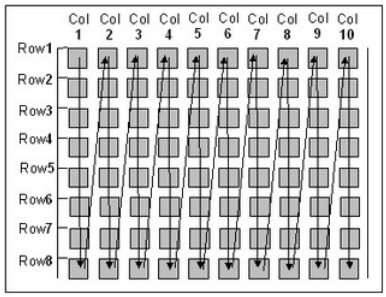
Networks	Each network in an application requires one word of memory, whether the network is used or not.
Columns	Each column that is occupied in a network requires one word of memory.
Single Elements	Each single element requires one word of memory.
Double Elements	Each double element requires two words of memory.
Triple Elements	Each triple element requires three words of memory.

Program Execution Order

The controller evaluates each element, or function block, in the network in a sequence that starts at the top left hand corner; or ROW 1, COLUMN 1.

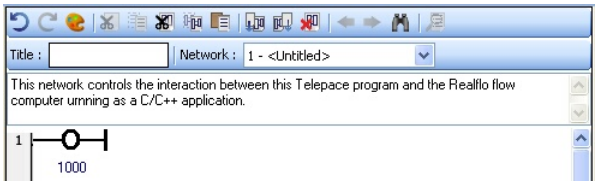
The ladder evaluation moves down column 1 until it reaches row 8. The evaluation then continues at the top of column 2 and moves down to row 8 again.

This process continues until the entire network has been evaluated. If there is more than one network, evaluation continues to the next sequential network in the program until the entire program has been evaluated.



Network Comments

Each network in the Telepace project can have a comment attached that describes the functionality of the network logic. The Network Comments area of the Ladder Canvas is above the network logic.



Subroutines

Subroutines permit conditional execution of parts of a Ladder Logic program. Subroutines can be used to reduce the scan time of a program by only scanning code when it is absolutely needed. Reduced scan time will increase the frequency of program execution.

Subroutines can also reduce the size of the program by placing frequently used or repeated code into subroutines that are called from various locations in the main program.

The Ladder Logic program consists of the main program followed by a number of subroutines.

The main program is defined as the logic networks up to the start of the first subroutine, or until the end of the program if no subroutines exist.

A subroutine is defined as the logic networks from a subroutine element until the next subroutine element, or the end of the program if there are no more subroutines.

The program is executed from the start of the main program to the end of the main program. If a subroutine call function block is encountered, execution transfers to the start of the subroutine and continues until the end of the subroutine. Execution then returns to the element after the subroutine call element.

Subroutine execution can be nested. This allows subroutines to call other subroutines.

Subroutine execution cannot be recursive. This avoids potential infinite loops in the ladder logic program.

Two elements control the definition and execution of subroutines. The **SUBR** element defines the start of a subroutine. The **CALL** element executes a subroutine.

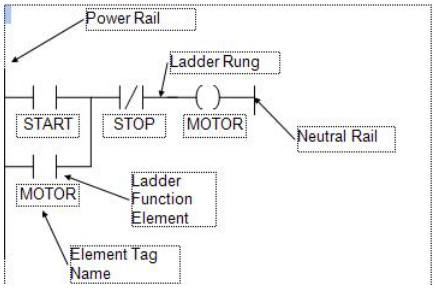
Network Elements

Network elements are contacts, coils, and function blocks with Shunts being used to interconnect elements. Coils represent either physical outputs in the controller or internal (memory only) outputs in the I/O Database Registers database. Contacts represent physical status inputs from the controller or internal (memory only) status inputs in the I/O Database Registers . Function blocks are used to perform specific functions, such as moving data, manipulating data or communicating data.

Networks

A network is a diagrammatic representation of control logic similar to a wiring schematic, showing interconnection of relays, timers, contacts and other control elements. As networks are created they can be given Titles, moved to new locations in the project using the Cut Networks and Paste Networks commands and copied to the Windows clipboard using the Copy Networks command.

A simplified network, describing the main parts of a network is shown below.



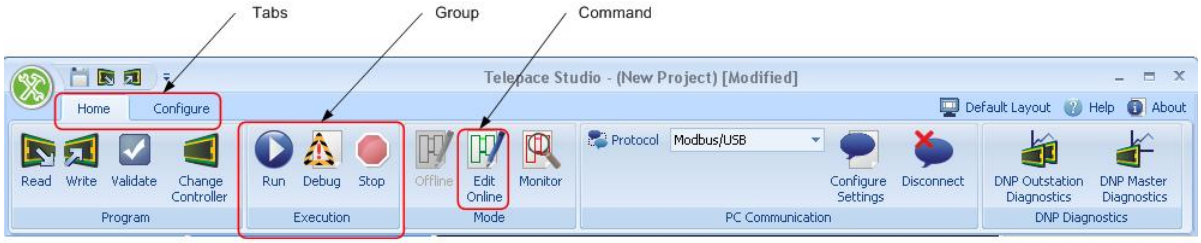
Networks in the Telepace Ladder Editor contain up to eight rungs. Each rung can contain a maximum of 10 logic elements.

The highlighted cursor in the Ladder Canvas occupies one logic element position.

The number of networks in a program is only limited by the Ladder Logic Memory available.

Telepace Studio Program Reference

The Telepace Studio Ribbon bar groups the tools and commands by task. The tabs on the Ribbon bar are task-oriented. Groups within each tab break tasks into subtasks. The command buttons in each group carry out a command.



There are three basic parts of the Ribbon bar:

Tabs The tabs on the Ribbon bar are task oriented. There are two tabs across the top of the Ribbon Bar, Home and Configure. Each tab represents an activity area.

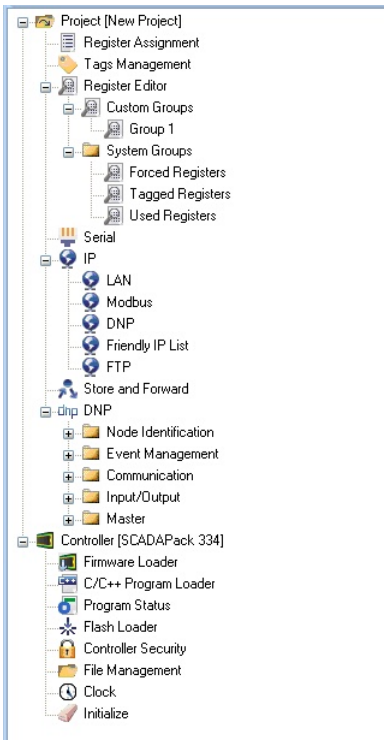
Groups Each tab has several groups that contain related commands. For example, the **I/O** group contains the commands for opening the Register Editor view, the Tags Management view and the Register Assignment view.

Commands Each Group contains one or more commands. A command is selected by clicking the command button.

The Ribbon Bar makes everything in Telepace Studio centralized and easy to find. If you want to increase your workspace, you can temporarily hide the Ribbon bar by double-clicking on the active tab. To restore the Ribbon bar, double-click on the active tab to reveal the groups again.

Tree View

Commands available in the Ribbon bar are also made available with the Tree View located at the left of the Telepace Studio application window. An expanded tree view is shown below.



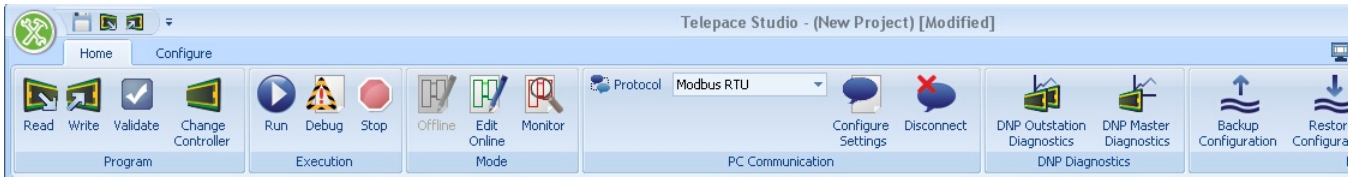
Related Topics
Quick Access Toolbar

Home Ribbon

The Home ribbon provides commands for programming the controller and debugging Telepace Studio programs.

For controller types except FlowStation the ribbon is displayed as shown below.

For **FlowStation** controllers the ribbon is displayed as shown below.



The Program Group contains commands to read and write Projects to and from the target controller and to validate the projects.

The Execution Group commands control the execution of the ladder logic program in the controller.

The Mode Group commands select the mode of operation for the Telepace Studio program.

The PC Communication commands select the communication protocols used between Telepace Studio and the target controller.

The DNP Diagnostics Group commands are used to monitor DNP diagnostics for controllers configured as DNP Outstations and DNP Master stations.

The FlowStation Group commands are used to manage FlowStation configurations when the controller type is set for FlowStation. This group is not displayed unless the controller type is FlowStation.

Program Group

The **Program** group contains the commands to Read the Telepace Studio project from the target controller into the Telepace Studio program; to Write the Telepace Studio project to the target controller; to Validate the Telepace Studio project and to Change Controller.

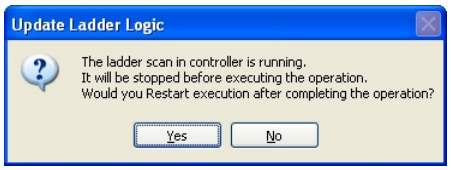
Read

The Read command is used to read the ladder logic program, register assignment, serial port settings and Ethernet port settings from a controller into the Telepace program. The read program replaces the current program in the ladder editor. The new program has a [Modified] attached to the file name in the workspace header. The communication progress dialog box displays information about the read in progress.

Write

The Write command writes the ladder logic program, register assignment, serial port settings and Ethernet settings in the Telepace program to the controller. The program replaces the current program in the controller. The communication progress dialog box displays information about the write progress, and allows you to Start or Cancel the write. Click Cancel to stop a write in progress.

If the program in the controller is executing, a dialog box appears to request the next action.



- Press **Yes** to continue execution of the new program after the write is complete. Telepace Studio stops the program in the controller before writing the new program to the controller.
- Press **No** to stop execution of the ladder logic program when the write is complete.

Validate

The **Validate** command is used to verify that a Telepace Studio project does not contain any configuration that would cause the project to not execute correctly in the target controller type. A Telepace Studio project may be validated, using the Validate button, at any time during during project development.

In addition to validating a project by selecting the validate command a validation of a Telepace Studio project file is project occurs when:

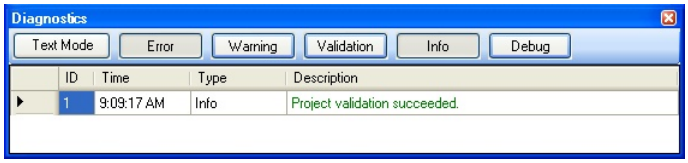
- a Telepace Studio project file is opened.
- a Telepace Studio project file is read from a controller.
- the controller type is changed.
- before a Telepace Studio project file is written to a controller.

The following project configurations are validated.

DNP Configuration	• Validation checks that the communication ports defined in the DNP Routing table support DNP protocol.
IP Configuration	• Validation checks the PPP Settings, Gateway Port Settings and DHCP settings. The settings validated depend on the controller type, not all settings are supported by all controller types. • Validation checks the FTP Username and Password data. This setting is validated depending on the controller type, FTP is not supported by all controller types.
Ladder Logic Configuration *	• Validation checks ladder logic elements based on element type, register range, network location and network position. This validation does not check the ladder logic programming (this is the responsibility of the programmer) but does check that the logic elements used in application will operate correctly.
Register Assignment Configuration *	• Validation checks that communication ports used in Register Assignment modules are valid for the controller type. Not all communication ports are support by all controller types. • Validation checks IO Module properties to ensure module is supported by the controller type. Not all Register Assignment modules are support by all controller types.
Store and Forward Configuration *	• Validation checks that the slave and forward communication interfaces are supported by the controller type. Not all communication interfaces are support by all controller types. • Validation checks that there are no blank entries in the Store and Forward table.

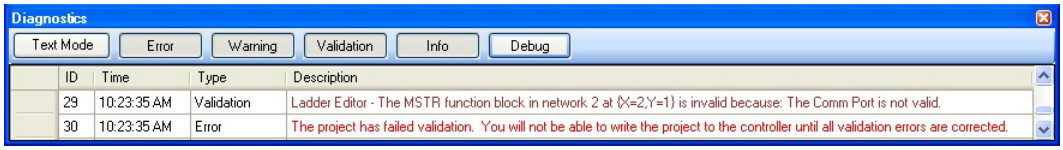
Telepace Studio checks only those configurations marked with an asterisk when the controller type is changed for a project.

When the Validate command detects no errors with the Telepace Studio project a message is added to the Diagnostic display as shown below.



When the Validate command detects invalid configuration with the Telepace Studio project a message is displayed as shown below.

The Diagnostics window will display the nature of the Validation error as shown below. Telepace Studio will not Write the project to the target controller until the Validation errors are corrected.

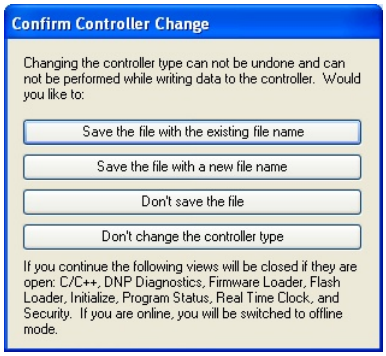


Change Controller

The **Change Controller Type** command is used to change the controller type for a Telepace Studio project.

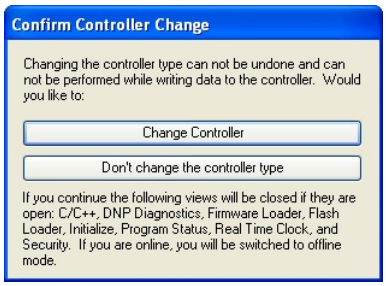
In the current Telepace Studio project select the **Change Controller** command in the Controller group of the Configure Ribbon.

If there are any unsaved changes in the current Telepace project the following dialog is displayed to save the current project.



- Click the **Save the file with the existing file name** button to File Save the project with the existing file name.
- Click the **Save the file with a new file name** button to open the File Save As dialog.
- Click the **Don't Save the file** button to change the controller type without saving the current Telepace project.
- Click the **Don't change the controller type** button to cancel the Change Controller command.

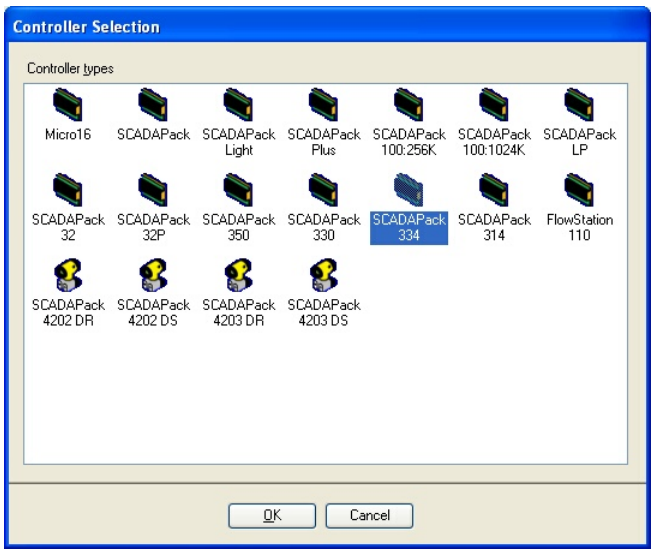
Once the file has been saved the following dialog is displayed to confirm that the controller type is to be changed.



When changing controller type Telepace Studio cannot not be actively communicating with a controller. Any views which require communication with a target controller are closed and the programming mode is switched to **Off Line**.

- Click the **Change Controller** button to open the Controller Selection dialog and change the controller type.
- Click the **Don't change the controller type** button to cancel the Change Controller command.

When Change Controller is selected the Controller Selection dialog is opened.



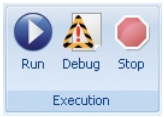
The Controller Selection dialog displays all the controller types supported by Telepace.

- Click on the **controller types** icon for the controller type you want to change to.
- Click the **OK** button to confirm the controller type selection, close the dialog and start the validation process.
- Click the **Cancel** button to cancel the Change Controller command.

When the **OK** button is clicked Telepace performs a **validation** of the project. SCADAPack controllers differ in the number of serial ports, Ethernet ports and USB ports. As well the Function Blocks, Register Assignment and Modbus addresses supported differ between controller types. In light of these differences Telepace needs to validate the current project when the controller type is changed.

Execution Group

The **Execution Group** commands control the execution of the ladder logic program in the controller.



Stop

The **Stop** command stops execution of the ladder logic program in the controller. When the stop command is used the Diagnostics view will display a message indicating the program has been stopped.

Debug

The **Debug** command start's the execution of the ladder logic program in debug mode. In debug mode the Monitor On Line feature of the editor can be used to monitor the operation of the program. The program runs slower in debug mode than in run mode.

Run

The **Run** command starts execution of the ladder logic program in run mode. The Monitor On Line feature of the editor cannot be used to monitor execution in this mode.

Mode Group

The **Mode Group** is located in the Home ribbon and is shown below.

The Mode Group commands select the mode of operation for the Telepace Studio program. The Telepace Studio program can operate in three different programming modes, Edit Off Line, Edit On Line and Monitor On Line. When a mode is selected the icon for the mode is greyed out.

Offline

The Offline mode enables editing of a ladder logic program without a connection to a controller. In this mode the editing commands affect only the program in use Telepace Studio. Forcing commands cannot be used in this mode. Any other Telepace Studio commands can be used in this mode.

The Application Mode on the Status Bar is shown in the graphic following.

Edit Online

The Edit On Line selection is used to edit a ladder logic program that loaded into a target controller. Editing commands affect the program in the controller and the Telepace Studio file.

On-line editing maintains a list of changes in the PC memory. Changes (not the entire program) are written to the controller when the Write command is selected. Switching modes will prompt you to write your changes so you can't forget. Cut and paste functionality is supported in on-line editing.

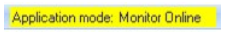
The Application Mode on the Status Bar changes to a red background when in the Edit Online mode.

Monitor

The Monitor selection enables the monitoring of a program executing in a controller. The editor shows the power flow through the ladder network in the Network Canvas. No changes can be made to the program in monitor mode. Editing commands are disabled on the menus. Forcing commands can be used in this mode.

Switch to Offline mode before performing a Cold Boot on the target controller. The ladder logic application in the Telepace Network Ladder Editor may be altered by the Cold Boot process.

The Application Mode on the Status Bar changes to a yellow background when in the Edit Online mode.



PC Communication Group

The **PC Communication Group** is located in the Home ribbon and is shown below. The PC Communication defines the communication protocol and communication link used for communication between a personal computer (PC) running Telepace Studio and a target controller.

When a new Telepace project is started the default PC Communication settings are used. When a project is saved the active communication settings are saved with the project.

- Select a communication protocol using the Protocol drop down list.
- Select the Configure command to configure the communication properties for the Protocol.
- Select the Disconnect command to disconnect the communication interface to allow other applications to use the PC resource.

To see how various communication protocols are configured click on a Protocol link in the table below. The link will take you to a full explanation of the protocol and provide all configuration details for the protocol.

Protocol	Description
ClearSCADA	The ClearSCADA protocol driver is used for communicating with a local or remote ClearSCADA server.

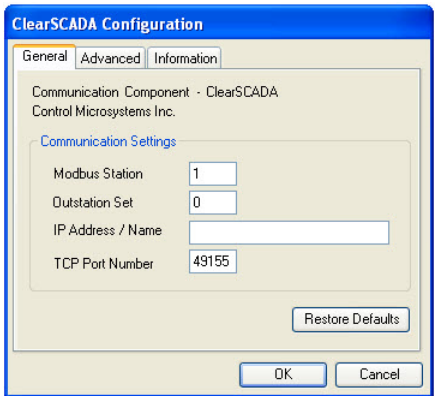
DNP	The DNP protocol driver is used to communicate over a serial DNP network.
DNP/TCP	The DNP/TCP protocol driver is used to communicate over an Ethernet DNP network.
DNP/UDP	The DNP/UDP protocol driver is used to communicate over an Ethernet DNP network.
Modbus ASCII	The Modbus ASCII protocol driver is used to communicate over a serial network, using Modbus ASCII framing.
Modbus ASCII in TCP	The Modbus ASCII in TCP protocol driver is used to communicate over an Ethernet network. The Modbus ASCII message is embedded in a TCP message.
Modbus ASCII in UDP	The Modbus ASCII in UDP protocol driver is used to communicate over an Ethernet network. The Modbus ASCII message is embedded in a UDP message.
Modbus RTU	The Modbus RTU protocol driver is used to communicate over a serial network, using Modbus RTU framing,
Modbus RTU in TCP	The Modbus RTU in TCP protocol driver is used to communicate over an Ethernet network. The Modbus RTU message is embedded in a TCP message.
Modbus RTU in UDP	The Modbus RTU in UDP protocol driver is used to communicate over an Ethernet network. The Modbus UDP message is embedded in a UDP message.
Modbus TCP	The Modbus/TCP protocol driver is an extension of serial Modbus, which defines how Modbus messages are encoded within and transported over TCP/IP-based networks. The Modbus/TCP protocol uses a custom Modbus protocol layer on top of the TCP protocol. to verify accurate delivery of the packet.
Modbus UDP	The Modbus/UDP protocol driver is similar to Modbus/TCP communication mode. It has the same message format with the Modbus/TCP but uses UDP.
Modbus USB	The Modbus USB protocol driver provides the means to communicate with SCADAPack controllers equipped with a Universal Serial Bus (USB) peripheral port using Modbus/USB messaging.
SCADAServer	The SCADAServer protocol driver specifies a SCADAServer Host connection. Applications will act as an OPC client and route all programming commands through the SCADAServer Host.

ClearSCADA

The ClearSCADA protocol driver is used for communicating with a local or remote ClearSCADA server. The ClearSCADA server will then, in turn, communicate with devices as per its configuration. The ClearSCADA protocol driver communicates with the ClearSCADA server using a TCP connection.

General Tab

When ClearSCADA protocol is selected for configuration the ClearSCADA Configuration dialog is opened with the General tab selected as shown below.



The General tab component information section contains the name of Communication Component and the author, Control Microsystems.

The **Communications Settings** grouping contains the details necessary to establish communication to a device through a local or remote ClearSCADA installation.

The **Modbus Station** entry specifies the station address of the target device. Valid values are 1 to 65534.

The **Outstation Set** entry specifies the ClearSCADA outstation set to which the target device is attached. The valid range is 0 to 65535. The default value is 0.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **TCP Port Number** entry specifies the TCP port on the **ClearSCADA** server. The valid range is 0 to 65535. The default value is 49155

- Click **Restore Defaults** to restore default values to fields on this page, except for the **IP Address / Name** field. The contents of this field will remain unchanged.

Advanced Tab

See Advanced Tab for details on the fields associated with this tab.

Information Tab

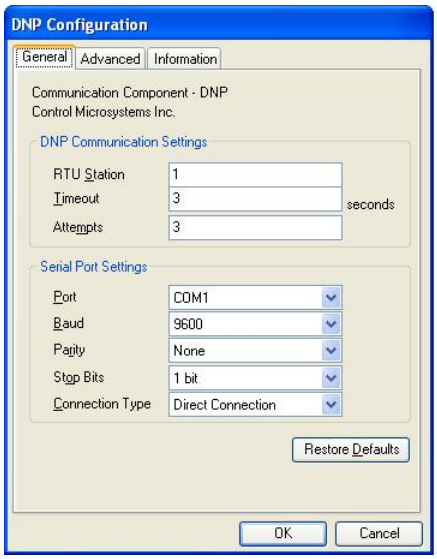
See Information Tab for the fields associated with this tab.

DNP

The DNP driver is used to communicate over a serial DNP network to SCADAPack controllers configured for DNP communication.

General Tab

When DNP is selected for configuration the DNP Configuration dialog is opened with the General tab selected as shown below.



The **DNP Communication Settings** grouping contains DNP specific communication settings including the station address, the timeout interval as well as the number of attempts.

The **RTU Station** parameter specifies the DNP station number of the target device. The valid range is 0 to 65519. The default is station 1.

The **Timeout** parameter specifies the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid values are 1 to 255. The default value is 3 seconds. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter specifies the number of times to send a command to the controller before giving up and reporting this to the host application. Valid values are 1 to 20. The default value is 3 attempts.

This **Serial Port Settings** grouping contains details directly related to the PC's communication port including the port number, the baud rate, parity and stop bit settings.

The **Port** parameter specifies the PC serial port to use. The DNP driver determines what serial ports are available on the PC and presents these in the drop-down menu list. The available serial ports list will include any USB to serial converters used on the PC. The default value is the first existing port found by the driver.

The **Baud** parameter specifies the baud rate to use for communication. The menu list displays selections for 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, and 57600. The default value is 9600.

The **Parity** parameter specifies the type of parity to use for communication. The menu list displays selections for none, odd and even parity. The default value is None.

The **Data Bits** parameter specifies the number of data bits contained in the character frame. Valid values are for this field is 7 and 8 bits. The default value is 8 bits.

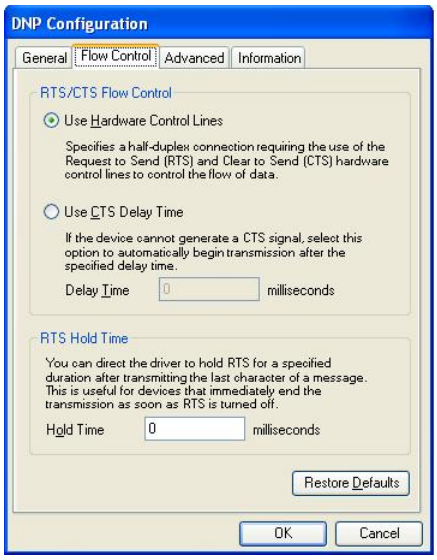
The **Stop Bits** parameter specifies the number of stop bits to use for communication. The menu list displays selections for 1 and 2 stop bits. The default value is 1 bit.

The **Connection Type** parameter specifies the serial connection type. The Modbus ASCII driver supports direct serial connection with no flow control, Request-to-send (RTS) and clear-to-send (CTS) flow control and PSTN dial-up connections. The menu list displays selections for Direct Connection, RTS/CTS Flow Control and Dial Up Connection. The default selection is Direct Connection.

- Select **Direct Connection** for RS-232 for RS-485 connections not requiring the hardware control lines on the serial ports.
- Select **RTS/CTS Flow Control** to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking. Selecting RTS/CTS Flow Control adds a new tab, Flow Control, to the Modbus ASCII Configuration dialog. Refer to the Flow Control Parameters section below for configuration details.
- Select **Dial Up Connection** to communication over dial up modems. Selecting Dial Up Connection adds a new tab, Dial Up, to the Modbus ASCII Configuration dialog. Refer to the Dial Up Parameters section below for configuration details.
- Click **Restore Defaults** to restore default values to all fields on this page.

Configuration (Flow Control)

Flow Control parameters are used to configure how RTS and CTS control is used. When RTS/CTS Flow Control is selected for Connection Type the Flow Control tab is added to the DNP Configuration dialog. When the Flow Control tab heading is clicked the Flow Control dialog is opened as shown below.



The **RTS/CTS Flow Control** grouping contains two mutually exclusive options, Use Hardware Control Lines and Use CTS Delay Time. These options enable the driver to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking.

The **Use Hardware Control Lines** option specifies a half-duplex connection requiring the use of the Request to Send (RTS) and Clear to Send (CTS) hardware control lines to control the flow of data. This selection is used with radios and dedicated telephone line modems. The driver turns on the RTS signal when it wants to transmit data. The modem or other device then turns on CTS when it is ready to transmit. The driver transmits the data, and then turns off the RTS signal. This selection is mutually exclusive of the Use CTS Delay Time selection described below. This is the default selection.

The **Use CTS Delay Time** option is selected if the device cannot generate a CTS signal. The driver will assert RTS then wait the specified Delay Time, in milliseconds, before proceeding. This option is mutually exclusive with the Use Hardware Control Lines selection described above.

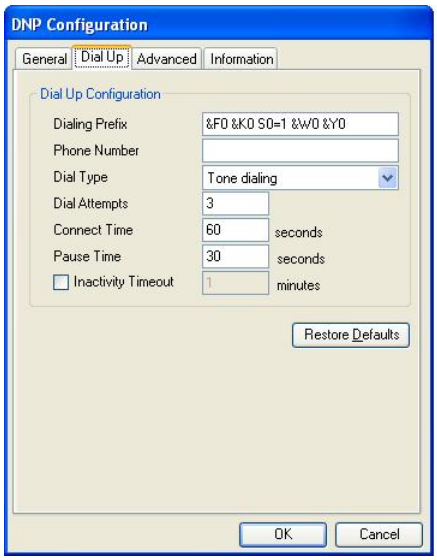
The **Delay Time** parameter sets the time in milliseconds that the driver will wait after asserting RTS before proceeding. The value of this field needs to be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

The **Hold Time** parameter specifies the time, in milliseconds, that the driver will hold RTS after the last character is transmitted. This is useful for devices that immediately end transmission when RTS is turned off. The value of this field needs to be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

- Click **Restore Defaults** to restore default values to the fields on this page.

DNP Configuration (Dial Up)

Dial-up parameters are used to configure a dial up connection. When Dial Up is selected for Connection Type the Dial-up tab is added to the DNP Configuration dialog. When the Dial-up tab heading is clicked the Dial-up dialog is opened as shown below.



The **Dialing Prefix** parameter specifies the commands sent to the modem before dialing. A maximum of 32 characters can be entered. All characters are valid. The default value is “&F0 &K0 S0=1 &W0 &Y0”.

The **Phone Number** parameter specifies the telephone number of the remote controller. A maximum of 32 characters can be entered. All characters are valid. This field’s default value is blank.

The **Dial Type** parameter specifies the dialing type. Valid values are Pulse and Tone. The default value is Tone.

The **Dial Attempts** parameter specifies how many dialing attempts will be made. Valid values are 1 to 10. The default value is 1.

The **Connect Time** parameter specifies the amount of time in seconds the modem will wait for a connection. Valid values are 6 to 300. The default value is 60.

The **Pause Time** parameter specifies the time in seconds between dialing attempts. Valid values are 6 to 600. The default value is 30.

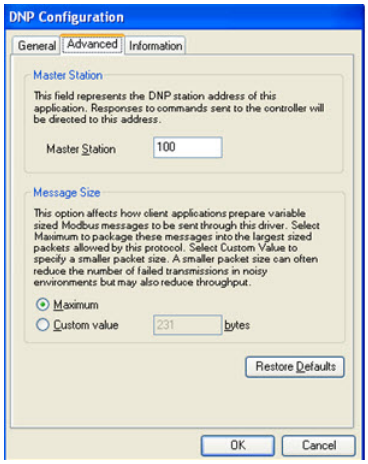
Check the **Inactivity Timeout** check box to automatically terminate the dialup connection after a period of inactivity. The Inactivity Time edit box is enabled only if this option is checked. The default state is checked.

Enter the inactivity period, in minutes, in the **Inactivity Timeout** box. The dialup connection will be terminated automatically after the specified number of minutes of inactivity has lapsed. This option is only active if the Inactivity Timeout box is checked. Valid values are from 1 to 30 minutes. The default value is 1.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the Phone Number field. The content of this field will remain unchanged.

Advanced Tab

DNP Configuration Advanced parameters set the DNP master station address and message size control. When the Advanced tab heading is clicked the **Advanced** dialog is opened as shown below.



The **Master Station** parameter is the DNP station address assumed by this communication component. When this driver sends out commands, responses from the controller will be directed to this address. The default value is 100.

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of missed transmissions. A smaller value can reduce the number of missed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 231. The default value is 231.

- Click **Restore Defaults** to restore default values to the fields on this page.

Information Tab

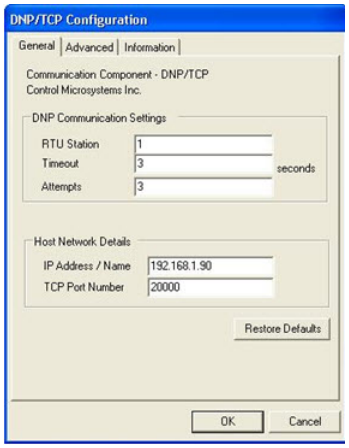
See Information Tab for the fields associated with this tab.

DNP/TCP

The DNP/TCP protocol driver is used to communicate over an Ethernet DNP network to SCADAPack controllers configured for DNP/TCP communication.

General Tab

When DNP/TCP protocol is selected for configuration the DNP/TCP Configuration dialog is opened with the General tab selected as shown below.



The **DNP Communication Settings** grouping contains DNP specific communication settings including the DNP Station address, the timeout interval as well as the number of attempts.

The **RTU Station** parameter specifies the DNP station number of the target device. The valid range is 0 to 65519. The default is station 1.

The **Timeout** parameter specifies the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid values are 1 to 255. The default value is 3 seconds. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter specifies the number of times to send a command to the controller before giving up and reporting this to the host application. Valid values are 1 to 20. The default value is 3 attempts. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Host Network Details** grouping contains information about the IP network including the target's IP address or name, and the TCP port number on which it is

listening. More details on these below.

The **IP Address / Name** parameter specifies the Ethernet IP address of the target RTU, or a DNS name that can be resolved to an IP address. The default value is blank. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **TCP Port No.** field specifies the TCP port of the remote device. Valid values are 0 to 65535. The default value is 20000.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

Advanced Tab

DNP/TCP Configuration Advanced parameters set the DNP master station address and message size control. When the Advanced tab heading is clicked the **Advanced** dialog is opened as shown below.

The **Master Station** parameter is the DNP station address assumed by this communication component. When this driver sends out commands, responses from the controller will be directed to this address. The default value is 100.

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of missed transmissions. A smaller value can reduce the number of missed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 231. The default value is 231.

- Click **Restore Defaults** to restore default values to all fields on this page.

Information Tab

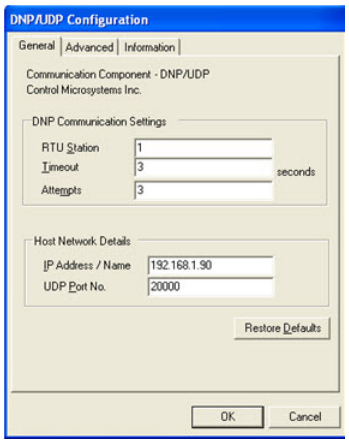
See Information Tab for the fields associated with this tab.

DNP/UDP

The DNP/UDP protocol driver is used to communicate over an Ethernet DNP network to SCADAPack controllers configured for DNP/UDP communication.

General Tab

When DNP/UDP protocol is selected for configuration the DNP/UDP Configuration dialog is opened with the General tab selected as shown below.



The **DNP Communication Settings** grouping contains DNP specific communication settings including the DNP Station address, the timeout interval as well as the number of attempts.

The **RTU Station** parameter specifies the DNP station number of the target device. The valid range is 0 to 65519. The default is station 1.

The **Timeout** parameter specifies the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid values are 1 to 255. The default value is 3 seconds. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter specifies the number of times to send a command to the controller before giving up and reporting this to the host application. Valid values are 1

to 20. The default value is 3 attempts.

The **Host Network Details** grouping contains information about the IP network including the target's IP address or name, and the UDP port number on which it is listening. More details on these below.

The **IP Address / Name** parameter specifies the Ethernet IP address of the target RTU, or a DNS name that can be resolved to an IP address. The default value is blank. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **UDP Port No.** field specifies the UDP port of the remote device. Valid values are 0 to 65534. The default value is 20000.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

Advanced Tab

DNP/UDP Configuration Advanced parameters set the DNP master station address and message size control. When the Advanced tab heading is clicked the **Advanced** dialog is opened as shown below.

The **Master Station** parameter is the DNP station address assumed by this communication component. When this driver sends out commands, responses from the controller will be directed to this address. The default value is 100.

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of missed transmissions. A smaller value can reduce the number of missed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 231. The default value is 231.

- Click **Restore Defaults** to restore default values to all fields on this page.

Information Tab

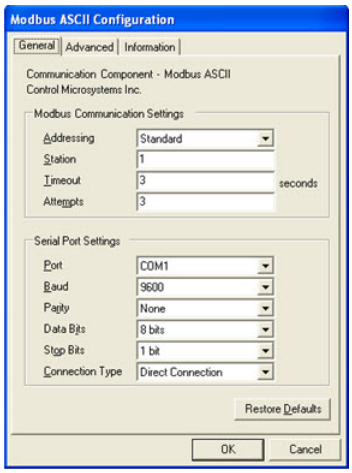
See Information Tab for the fields associated with this tab.

Modbus ASCII

The Modbus ASCII protocol driver is used to communicate over a serial network, using Modbus ASCII framing, to SCADAPack controllers configured for Modbus ASCII protocol.

General Tab

When Modbus ASCII is selected for configuration the Modbus ASCII Configuration dialog is opened with the General tab selected as shown below.



The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid entries are 1 to 255. The default is 3. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this to the host application. Valid entries are 1 to 20. The default is 3.

This **Serial Port Settings** grouping contains details directly related to the PC's communication port including the port number, the baud rate, parity and stop bit settings.

The **Port** parameter specifies the PC serial port to use. The DNP driver determines what serial ports are available on the PC and presents these in the drop-down menu list. The available serial ports list will include any USB to serial converters used on the PC. The default value is the first existing port found by the driver.

The **Baud** parameter specifies the baud rate to use for communication. The menu list displays selections for 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, and 57600. The default value is 9600.

The **Parity** parameter specifies the type of parity to use for communication. The menu list displays selections for none, odd and even parity. The default value is None.

The **Data Bits** parameter specifies the number of data bits contained in the character frame. Valid values are for this field is 7 and 8 bits. The default value is 8 bits.

The **Stop Bits** parameter specifies the number of stop bits to use for communication. The menu list displays selections for 1 and 2 stop bits. The default value is 1 bit.

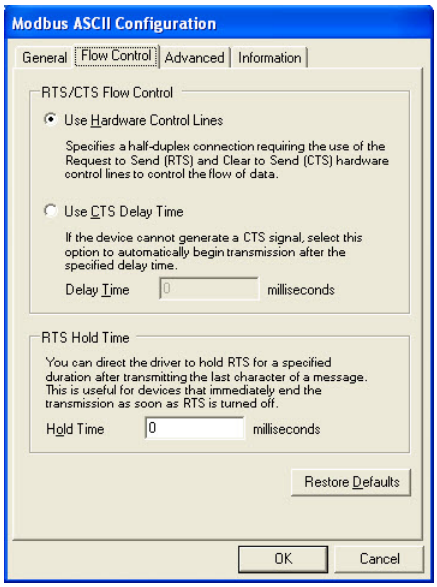
The **Connection Type** parameter specifies the serial connection type. The Modbus ASCII driver supports direct serial connection with no flow control, Request-to-send (RTS) and clear-to-send (CTS) flow control and PSTN dial-up connections. The menu list displays selections for Direct Connection, RTS/CTS Flow Control and Dial Up

Connection. The default selection is Direct Connection.

- Select **Direct Connection** for RS-232 for RS-485 connections not requiring the hardware control lines on the serial ports.
- Select **RTS/CTS Flow Control** to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking. Selecting RTS/CTS Flow Control adds a new tab, Flow Control, to the Modbus ASCII Configuration dialog. Refer to the Flow Control Parameters section below for configuration details.
- Select **Dial Up Connection** to communication over dial up modems. Selecting Dial Up Connection adds a new tab, Dial Up, to the Modbus ASCII Configuration dialog. Refer to the Dial Up Parameters section below for configuration details.
- Click **Restore Defaults** to restore default values to all fields on this page.

Modbus ASCII Configuration (Flow Control)

Flow Control parameters are used to configure how RTS and CTS control is used. When RTS/CTS Flow Control is selected for Connection Type the Flow Control tab is added to the Modbus ASCII Configuration dialog. When the Flow Control tab heading is clicked the Flow Control dialog is opened as shown below.



The **RTS/CTS Flow Control** grouping contains two mutually exclusive options, Use Hardware Control Lines and Use CTS Delay Time. These options enable the driver to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking.

The **Use Hardware Control Lines** option specifies a half-duplex connection requiring the use of the Request to Send (RTS) and Clear to Send (CTS) hardware control lines to control the flow of data. This selection is used with radios and dedicated telephone line modems. The driver turns on the RTS signal when it wants to transmit data. The modem or other device then turns on CTS when it is ready to transmit. The driver transmits the data, and then turns off the RTS signal. This selection is mutually exclusive of the Use CTS Delay Time selection described below. This is the default selection.

The **Use CTS Delay Time** option is selected if the device cannot generate a CTS signal. The driver will assert RTS then wait the specified Delay Time, in milliseconds, before proceeding. This option is mutually exclusive with the Use Hardware Control Lines selection described above.

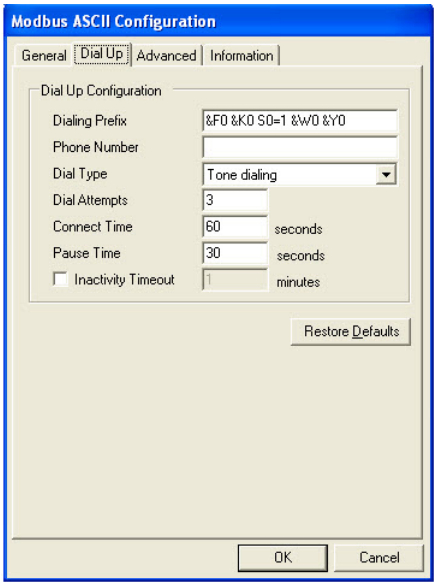
The **Delay Time** parameter sets the time in milliseconds that the driver will wait after asserting RTS before proceeding. The value of this field needs to be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

The **Hold Time** parameter specifies the time, in milliseconds, that the driver will hold RTS after the last character is transmitted. This is useful for devices that immediately end transmission when RTS is turned off. The value of this field needs to be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

- Click **Restore Defaults** to restore default values to all fields on this page.

Modbus ASCII Configuration (Dial up)

Dial Up parameters are used to configure a dial up connection. When Dial Up is selected for Connection Type the Dial Up tab is added to the Modbus ASCII Configuration dialog. When the Dial Up tab heading is clicked the Dial Up dialog is opened as shown below.



The **Dialing Prefix** parameter specifies the commands sent to the modem before dialing. A maximum of 32 characters can be entered. All characters are valid. The default value is “&F0 &K0 S0=1 &W0 &Y0”.

The **Phone Number** parameter specifies the telephone number of the remote controller. A maximum of 32 characters can be entered. All characters are valid. This field’s default value is blank.

The **Dial Type** parameter specifies the dialing type. Valid values are Pulse and Tone. The default value is Tone.

The **Dial Attempts** parameter specifies how many dialing attempts will be made. Valid values are 1 to 10. The default value is 1.

The **Connect Time** parameter specifies the amount of time in seconds the modem will wait for a connection. Valid values are 6 to 300. The default value is 60.

The **Pause Time** parameter specifies the time in seconds between dialing attempts. Valid values are 6 to 600. The default value is 30.

Check the **Inactivity Timeout** check box to automatically terminate the dialup connection after a period of inactivity. The Inactivity Time edit box is enabled only if this option is checked. The default state is checked.

Enter the inactivity period, in minutes, in the **Inactivity Timeout** box. The dialup connection will be terminated automatically after the specified number of minutes of inactivity has lapsed. This option is only active if the Inactivity Timeout box is checked. Valid values are from 1 to 30 minutes. The default value is 1.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the Phone Number field. The content of this field will remain unchanged.

Advanced Tab

See Advanced Tab for details on the fields associated with this tab.

Information Tab

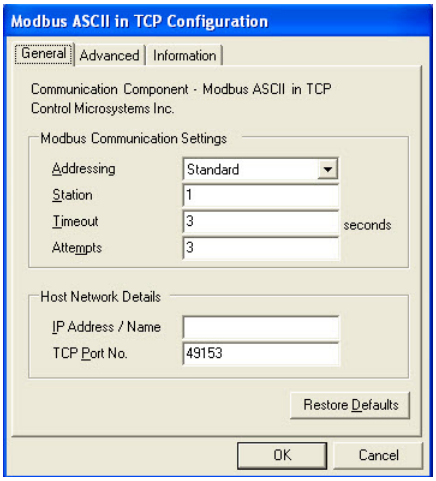
See Information Tab for the fields associated with this tab.

Modbus ASCII in TCP

Modbus ASCII in TCP message format is exactly same as that of the Modbus ASCII protocol. The main difference is that Modbus ASCII in TCP protocol communicates with a SCADAPack controller through the Internet and Modbus ASCII through the serial port. The Modbus ASCII in TCP protocol does not include a six-byte header prefix, as with the Modbus\TCP, but does include the Modbus ‘CRC-16’ or ‘LRC’ check fields.

General Tab

When Modbus ASCII in TCP is selected for configuration the Modbus ASCII in TCP Configuration dialog is opened with the General tab selected as shown below.



The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid entries are 1 to 255. The default is 3. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **TCP Port No.** field specifies the TCP port of the remote device. Valid values are 0 to 65535. The default value is 49153.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

Advanced Tab

See Advanced Tab for details on the fields associated with this tab.

Information Tab

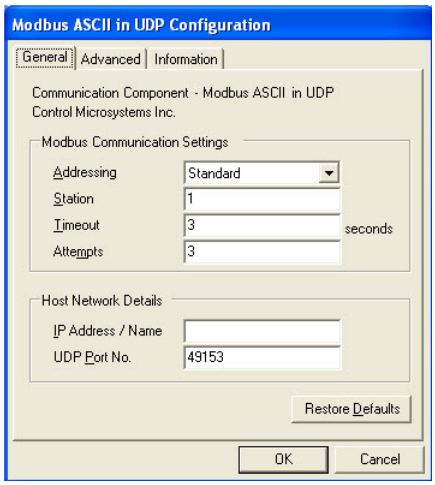
See Information Tab for the fields associated with this tab.

Modbus ASCII in UDP

Modbus ASCII in UDP protocol is similar to Modbus ASCII in TCP protocol. It has the same message format as the Modbus ASCII in TCP. The only difference between them is one uses TCP protocol and another uses UDP protocol.

General Tab

When Modbus ASCII in UDP is selected for configuration the Modbus ASCII in UDP Configuration dialog is opened with the General tab selected as shown below.



The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid entries are 1 to 255. The default is 3. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **UDP Port No.** field specifies the UDP port of the remote device. Valid values are 0 to 65535. The default value is 49153.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

Advanced Tab

See Advanced Tab for details on the fields associated with this tab.

Information Tab

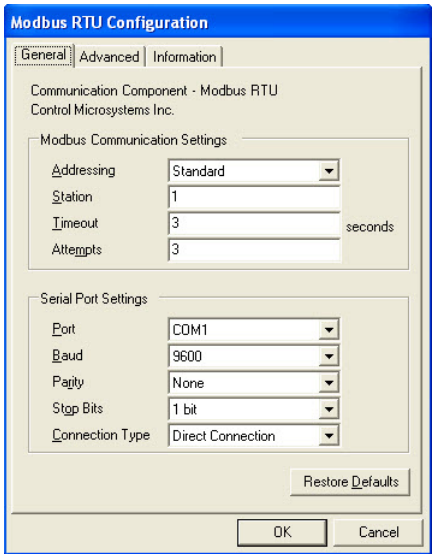
See Information Tab for the fields associated with this tab.

Modbus RTU

The Modbus RTU protocol driver is used to communicate over a serial network, using Modbus RTU framing, to SCADAPack controllers configured for Modbus RTU protocol.

General Tab

When Modbus RTU is selected for configuration the Modbus RTU Configuration dialog is opened with the General tab selected as shown below.



The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid entries are 1 to 255. The default is 3. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this to the host application. Valid entries are 1 to 20. The default is 3.

This **Serial Port Settings** grouping contains details directly related to the PC's communication port including the port number, the baud rate, parity and stop bit settings.

The **Port** parameter specifies the PC serial port to use. The DNP driver determines what serial ports are available on the PC and presents these in the drop-down menu list. The available serial ports list will include any USB to serial converters used on the PC. The default value is the first existing port found by the driver.

The **Baud** parameter specifies the baud rate to use for communication. The menu list displays selections for 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, and 57600. The default value is 9600.

The **Parity** parameter specifies the type of parity to use for communication. The menu list displays selections for none, odd and even parity. The default value is None.

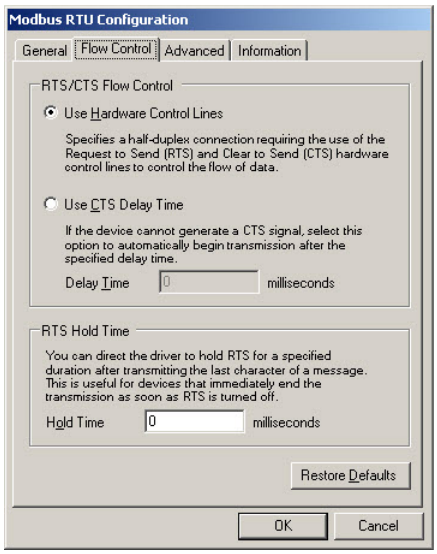
The **Stop Bits** parameter specifies the number of stop bits to use for communication. The menu list displays selections for 1 and 2 stop bits. The default value is 1 bit.

The **Connection Type** parameter specifies the serial connection type. The Modbus RTU driver supports direct serial connection with no flow control, Request-to-send (RTS) and clear-to-send (CTS) flow control and PSTN dial-up connections. The menu list displays selections for Direct Connection, RTS/CTS Flow Control and Dial Up Connection. The default selection is Direct Connection.

- Select **Direct Connection** for RS-232 for RS-485 connections that not requiring the hardware control lines on the serial ports.
- Select **RTS/CTS Flow Control** to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking. Selecting RTS/CTS Flow Control adds a new tab, Flow Control, to the Modbus RTU Configuration dialog. Refer to the Flow Control Parameters section below for configuration details.
- Select **Dial Up Connection** to communication over dial up modems. Selecting Dial Up Connection adds a new tab, Dial Up, to the Modbus RTU Configuration dialog. Refer to the Dial Up Parameters section below for configuration details.
- Click **Restore Defaults** to restore default values to all fields on this page.

Modbus ASCII Configuration (Flow Control)

Flow Control parameters are used to configure how RTS and CTS control is used. When RTS/CTS Flow Control is selected for Connection Type the Flow Control tab is added to the Modbus RTU Configuration dialog. When the Flow Control tab heading is clicked the Flow Control dialog is opened as shown below.



The **RTS/CTS Flow Control** grouping contains two mutually exclusive options, Use Hardware Control Lines and Use CTS Delay Time. These options enable the driver to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking.

The **Use Hardware Control Lines** option specifies a half-duplex connection requiring the use of the Request to Send (RTS) and Clear to Send (CTS) hardware control lines to control the flow of data. This selection is used with radios and dedicated telephone line modems. The driver turns on the RTS signal when it wants to transmit data. The modem or other device then turns on CTS when it is ready to transmit. The driver transmits the data, and then turns off the RTS signal. This selection is mutually exclusive of the Use CTS Delay Time selection described below. This is the default selection.

The **Use CTS Delay Time** option is selected if the device cannot generate a CTS signal. The driver will assert RTS then wait the specified Delay Time, in milliseconds, before proceeding. This option is mutually exclusive with the Use Hardware Control Lines selection described above.

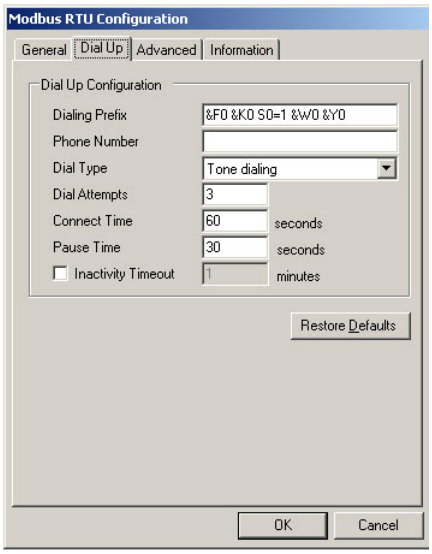
The **Delay Time** parameter sets the time in milliseconds that the driver will wait after asserting RTS before proceeding. The value of this field needs to be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

The **Hold Time** parameter specifies the time, in milliseconds, that the driver will hold RTS after the last character is transmitted. This is useful for devices that immediately end transmission when RTS is turned off. The value of this field needs to be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

- Click **Restore Defaults** to restore default values to all fields on this page.

Modbus ASCII Configuration (Dial up)

Dial Up parameters are used to configure a dial up connection. When Dial Up is selected for Connection Type the Dial Up tab is added to the Modbus RTU Configuration dialog. When the Dial Up tab heading is clicked the Dial Up dialog is opened as shown below.



The **Dialing Prefix** parameter specifies the commands sent to the modem before dialing. A maximum of 32 characters can be entered. All characters are valid. The default value is “&F0 &K0 S0=1 &W0 &Y0”.

The **Phone Number** parameter specifies the telephone number of the remote controller. A maximum of 32 characters can be entered. All characters are valid. This field’s default value is blank.

The **Dial Type** parameter specifies the dialing type. Valid values are Pulse and Tone. The default value is Tone.

The **Dial Attempts** parameter specifies how many dialing attempts will be made. Valid values are 1 to 10. The default value is 1.

The **Connect Time** parameter specifies the amount of time in seconds the modem will wait for a connection. Valid values are 6 to 300. The default value is 60.

The **Pause Time** parameter specifies the time in seconds between dialing attempts. Valid values are 6 to 600. The default value is 30.

Check the **Inactivity Timeout** check box to automatically terminate the dialup connection after a period of inactivity. The Inactivity Time edit box is enabled only if this

option is checked. The default state is checked.

Enter the inactivity period, in minutes, in the **Inactivity Timeout** box. The dialup connection will be terminated automatically after the specified number of minutes of inactivity has lapsed. This option is only active if the Inactivity Timeout box is checked. Valid values are from 1 to 30 minutes. The default value is 1.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the Phone Number field. The content of this field will remain unchanged.

Advanced Tab

See Advanced Tab for details on the fields associated with this tab.

Information Tab

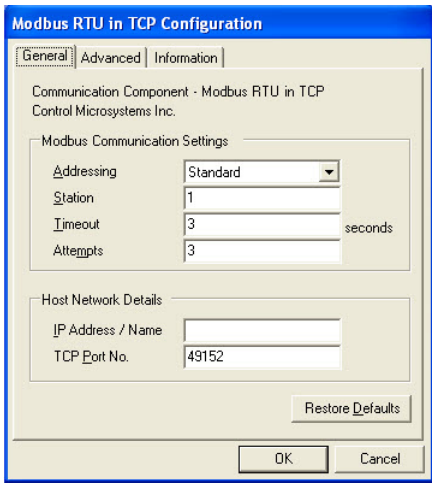
See Information Tab for the fields associated with this tab.

Modbus RTU in TCP

Modbus RTU in TCP message format is exactly same as that of the Modbus RTU protocol. The main difference is that Modbus RTU in TCP protocol communicates with a controller through the Internet and Modbus RTU protocol through the serial port. The Modbus RTU in TCP protocol does not include a six-byte header prefix, as with the Modbus\TCP, but does include the Modbus 'CRC-16' or 'LRC' check fields. The Modbus RTU in TCP message format supports Modbus RTU message format.

General Tab

When Modbus RTU in TCP is selected for configuration the Modbus RTU in TCP Configuration dialog is opened with the General tab selected as shown below.



The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid entries are 1 to 255. The default is 3. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **TCP Port No.** field specifies the TCP port of the remote device. Valid values are 0 to 65535. The default value is 49152.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

Advanced Tab

See Advanced Tab for details on the fields associated with this tab.

Information Tab

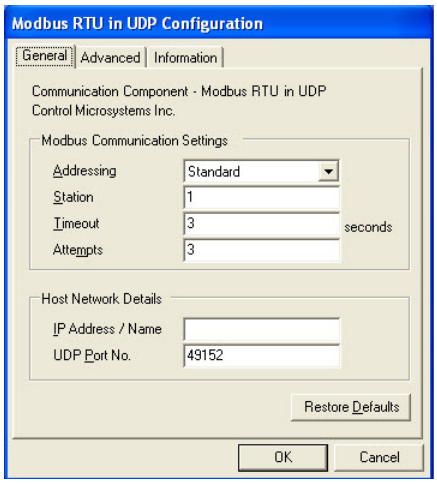
See Information Tab for the fields associated with this tab.

Modbus RTU in UDP

Modbus RTU in UDP protocol is similar to Modbus RTU in TCP protocol. It has the same message format as the RTU in TCP message.

General Tab

When Modbus RTU in UDP is selected for configuration the Modbus RTU in UDP Configuration dialog is opened with the General tab selected as shown below.



The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid entries are 1 to 255. The default is 3. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **UDP Port No.** field specifies the UDP port of the remote device. Valid values are 0 to 65535. The default value is 49152.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

Advanced Tab

See Advanced Tab for details on the fields associated with this tab.

Information Tab

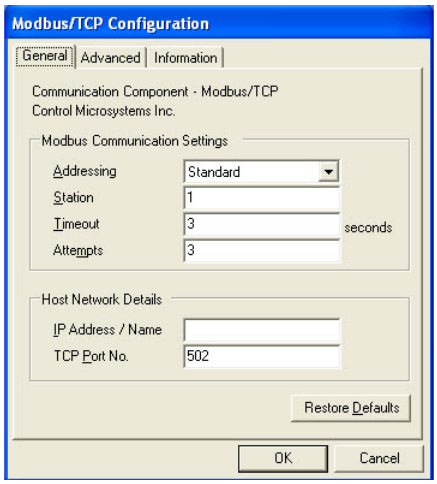
See Information Tab for the fields associated with this tab.

Modbus/TCP

Modbus/TCP is an extension of serial Modbus, which defines how Modbus messages are encoded within and transported over TCP/IP-based networks. The Modbus/TCP protocol uses a custom Modbus protocol layer on top of the TCP protocol. Its request and response messages are prefixed by six bytes. These six bytes consist of three fields: transaction ID field, protocol ID field and length field. The encapsulated Modbus message has exactly the same layout and meaning, from the function code to the end of the data portion, as other Modbus messages. The Modbus 'CRC-16' or 'LRC' check fields are not used in Modbus/TCP. The TCP/IP and link layer (e.g. Ethernet) checksum mechanisms instead are used to verify accurate delivery of the packet.

General Tab

When Modbus/TCP is selected for configuration the Modbus/TCP Configuration dialog is opened with the General tab selected as shown below.



The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid entries are 1 to 255. The default is 3. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **TCP Port No.** field specifies the UDP port of the remote device. Valid values are 0 to 65535. The default value is 502.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

Advanced Tab

See Advanced Tab for details on the fields associated with this tab.

Information Tab

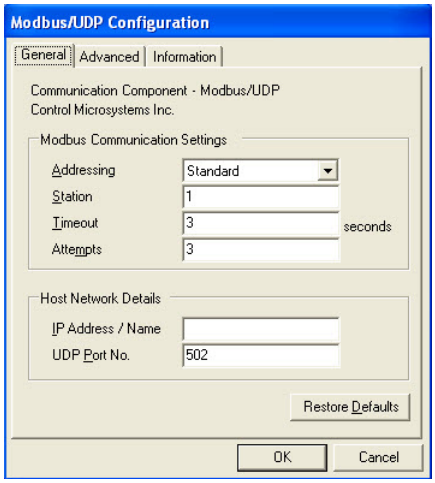
See Information Tab for the fields associated with this tab.

Modbus/UDP

Modbus/UDP communication mode is similar to Modbus/TCP communication mode. It has the same message format with the Modbus/TCP. The only difference between them is one uses TCP protocol and another uses UDP protocol.

General Tab

When Modbus/UDP is selected for configuration the Modbus/ UDP Configuration dialog is opened with the General tab selected as shown below.



The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts). Valid entries are 1 to 255. The default is 3. For some operations, such as File Management, the default timeout value of 3 seconds may not be long enough when lower baud rates are used.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **UDP Port No.** field specifies the UDP port of the remote device. Valid values are 0 to 65535. The default value is 502.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

Advanced Tab

See Advanced Tab for details on the fields associated with this tab.

Information Tab

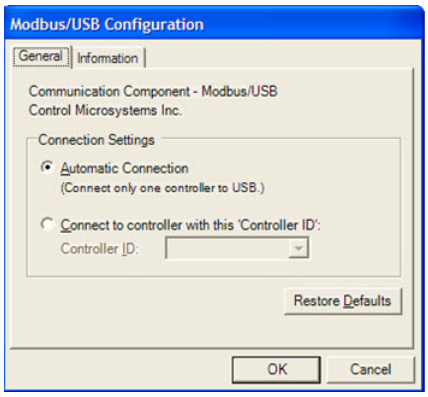
See Information Tab for the fields associated with this tab.

Modbus/USB

This driver provides the means to communicate with SCADAPack controllers equipped with a Universal Serial Bus (USB) peripheral port using Modbus/USB messaging. The driver does not require configuration making it possible to connect and communicate with a SCADAPack controller almost instantaneously.

General Tab

The general page identifies the type of driver and its author. This page also allows a user to specify how the application searches and connects to a USB equipped SCADAPack controller. User input depends on the number of USB equipped controllers connected on the bus.

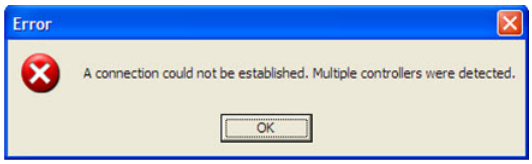


The **Connection Settings** grouping presents two connection options:

- Select **Automatic Connection** when a single controller is present on the bus.

A typical scenario involves a single controller connected directly to a USB port on the host PC.

The driver will reject connection requests by the application if this mode is selected with multiple controllers detected on the bus. In this case, the following message is displayed:

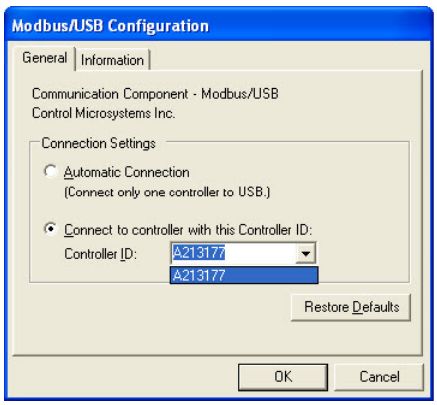


This option is selected by default.

- Select **Connect to controller with this 'Controller ID** when multiple controllers are present on the bus.

A typical scenario involves more than one USB equipped SCADAPack controller connected via a USB hub to the host PC.

If multiple controllers exist on the bus, the controller ID drop down box will display a list of all identifiable devices. The user connects to the controller in question by selecting its Controller ID from the list. The Controller ID takes on the format 'A123456' and can be found printed on the controller casing or the circuit board.



This option can also be used when there is a single controller present on the bus. However, the Controller ID needs to be known and entered in the Controller ID dialog.

The chosen controller does need to be present on the bus at configuration time.

- Click on the **Restore Defaults** button to reset the page contents to its default state.

In the default state, the **Automatic Connection** option is selected and the **Controller ID** text box is disabled. Any text in the Controller ID text box remains but is displayed in grey.

Advanced Tab

See Advanced Tab for details on the fields associated with this tab.

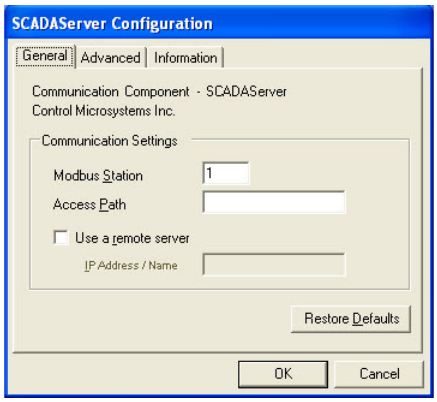
SCADAServer

The SCADAServer protocol specifies a SCADAServer Host connection. Applications will act as an OPC client and route all programming commands through the SCADAServer Host to the SCADAPack controller. The type of connection to the field device: no flow control, hardware flow control or dial-up modem is configured in the

SCADAServer Host itself.

General Tab

When SCADAServer is selected for configuration the SCADAServer Configuration dialog is opened with the General tab selected as shown below.



The **Communication Settings** grouping contains the details necessary to establish communication to a device through a local or remote SCADAServer installation.

The **Modbus Station** parameter specifies the station address of the target device. The valid range is 1 to 65534. The default is station 1.

The **Access Path** parameter specifies the access path to a SCADAServer connection. This parameter is entered as a string with a maximum size of 16 characters. This access path was named when a connection was defined within the SCADAServer installation. If the access path is left blank, the default SCADAServer connection will be used, as defined within the SCADAServer installation. The default for this entry is blank.

The **Use a remote server** check box defines whether the SCADAServer connection uses a SCADAServer installation installed on the same physical PC as the client application or on a remote PC. If the SCADAServer installation is located on a separate machine, check this option and enter the host name or IP address of the remote PC into the “IP Address / Name” edit box. If the SCADAServer installation is located on the same PC as the client application leave this box unchecked. The default state for this check box is unchecked.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

- Click **Restore Defaults** to restore default values to all fields on this page.

Advanced Tab

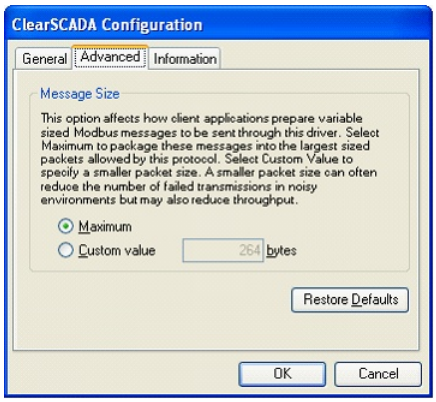
See Advanced Tab for details on the fields associated with this tab.

Information Tab

See Information Tab for the fields associated with this tab.

Advanced Tab

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of missed transmissions. A smaller value can reduce the number of missed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.



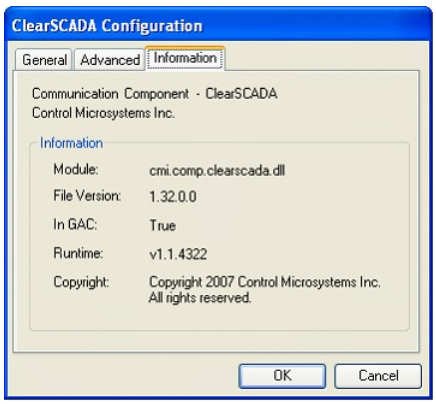
The Maximum selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The Custom Value selection specifies a custom value for the message size. This value will indicate to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 264. The default value is dependent on the protocol communication component.

- Click **Restore Defaults** to restore default values to all fields on this page.

Information Tab

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.



The Communication Component indicates the component where the Information tab was opened. The communication component is is one of:

ClearSCADA
DNP
DNP/TCP
DNP/UDP
Modbus ASCII
Modbus ASCII in TCP
Modbus ASCII in UDP
Modbus RTU
Modbus RTU in TCP
Modbus RTU in UDP
Modbus TCP
Modbus UDP
Modbus USB
SCADAServer

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

DNP Diagnostics Group

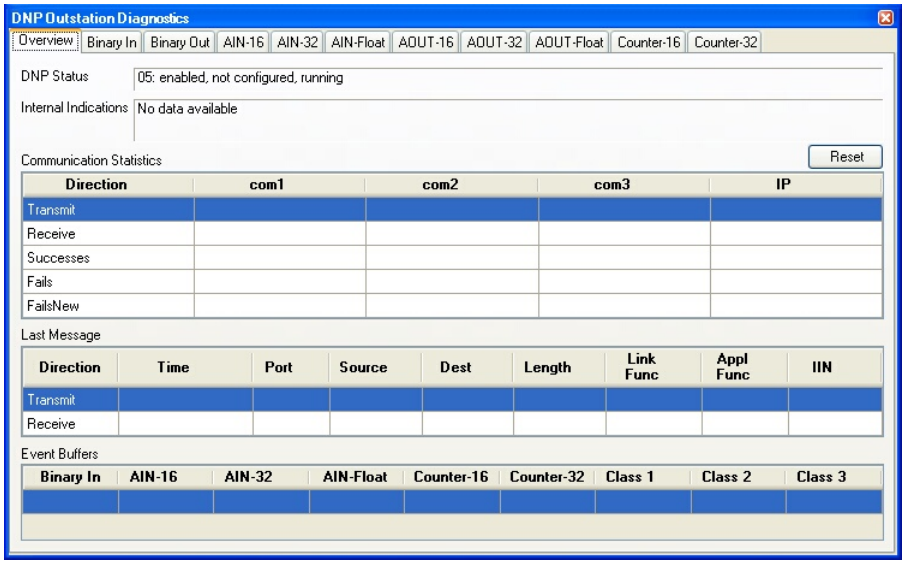
SCADAPack controllers support being configured as a DNP Outstation (Remote) and the SCADAPack 314, 33x, 350 and 4203 controllers support being configured as a DNP master station. The DNP Diagnostics group provides commands to monitor and diagnose the controller DNP communications.

Use DNP Outstation Diagnostics command to open the DNP outstation diagnostics view. This view is used to monitor and diagnose DNP communications when the controller is configured as a DNP Outstation.

Use DNP Master Diagnostics command to open the DNP master diagnostics view. This view is used to monitor and diagnose DNP communications when the controller is configured as an DNP Master station.

DNP Outstation Diagnostics

The **DNP Outstation Diagnostics** view displays the run-time diagnostics for the local DNP outstation. The Outstation Diagnostics view is divided into five areas of diagnostic information: DNP Status, Internal Indications, Communication Statistics, Last Message and Event Buffer. Each of these is explained below.



The **DNP Outstation Diagnostics** window provides information on the status of the DNP protocol running in the controller. Depending on the status the window may contain the following text.

- **Enabled** or **Disabled** indicates whether the controller firmware supports DNP protocol.
- **Configured** or **Not Configured** indicates whether the controller has been configured with DNP protocol on at least one communications port.
- **Running** or **Not Running** indicates whether the DNP tasks are running in the controller.

The **Internal Indications** window displays the current state of the DNP internal indications (IIN) flags in the controller. For a detailed description of the IIN flags see Internal Indications (IIN) Flags. Bits 0 – 7 (the first octet) are displayed on the left, then bits 8 - 15 (second octet) on the right.

The **Communication Statistics** window displays the message statistics for each DNP communication port. The statistics include the total number of messages transmitted and received and the total number of successes, failures, and failures since last success (which will only be updated for messages sent by this controller) for each communication port. The counters increment whenever a new DNP message is sent or received on the port, and roll over after 65535 messages.

- Click the **Reset** button to reset the counters to zero.

The **Last Message** window displays information about the recent DNP message. The information is updated each time a new message is received or transmitted. The Last Message window contains the following information.

- **Direction** displays whether the message was received or transmitted.
- **Time** displays the time at which the message was received or sent.
- **Port** displays which communication port was used for the message.
- **Source** displays the source DNP station address for the message.
- **Dest** displays the destination DNP station address for the message.
- **Length** displays the message length in bytes.
- **Link** Func displays the Link Layer function code.
- **Appl Func** displays the Application Layer function code.
- **IIN** displays the Internal indications received with the last message

The **Event Buffers** window displays the number of events in each type of event buffer and the allocated buffer size. The event buffers displayed are:

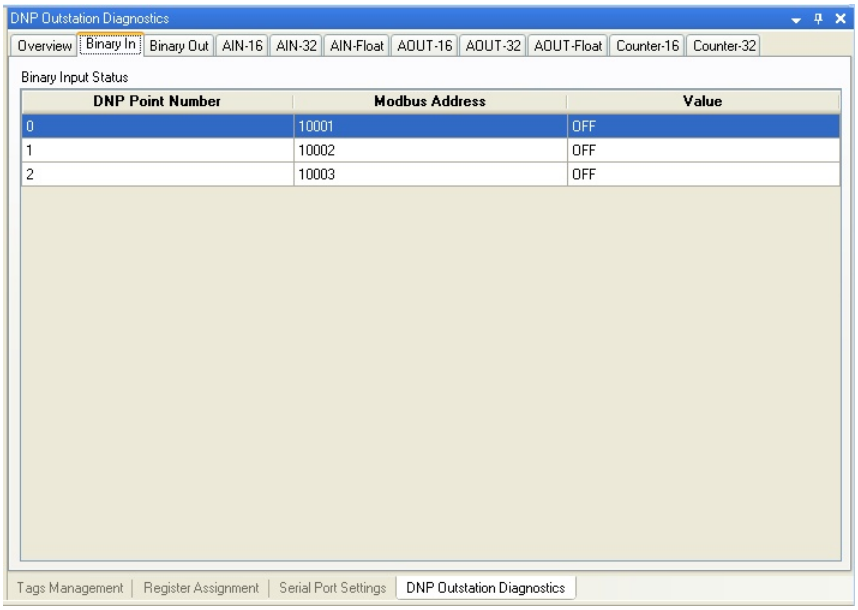
- **Binary In** (binary inputs)
- **AIN-16** (16-bit analog inputs)
- **AIN-32** (32-bit analog inputs)
- **AIN-Float** (floating point analog inputs)
- **Counter-16** (16-bit counter inputs)
- **Counter-32** (32-bit counter inputs)
- **Class 1** (class 1 events)
- **Class 2** (class 2 events)
- **Class 3** (class 3 events)

Point Status Tabs

The point status tabs display the state of each point of the selected type in the controller. The following tabs are displayed.

- **Binary In** (binary inputs information)
- **Binary Out** (binary outputs information)
- **AIN-16** (16-bit analog inputs information)
- **AIN-32** (32-bit analog inputs information)
- **AIN-Float** (short float analog inputs information)
- **AOUT-16** (16-bit analog outputs information)
- **AOUT-32** (32-bit analog outputs information)
- **AOUT-Float** (short float analog outputs information)
- **Counter-16** (16-bit counter inputs information)
- **Counter-32** (32-bit counter inputs information)

Each of the tabs displays information in the same format. The example below shows the appearance of the binary input page.



DNP Master Diagnostics

When the **DNP Master Diagnostics** command is selected the DNP Master Diagnostics view is displayed. This view shows the run-time DNP diagnostics and status of the DNP outstations and current data values for the DNP points in these outstations.

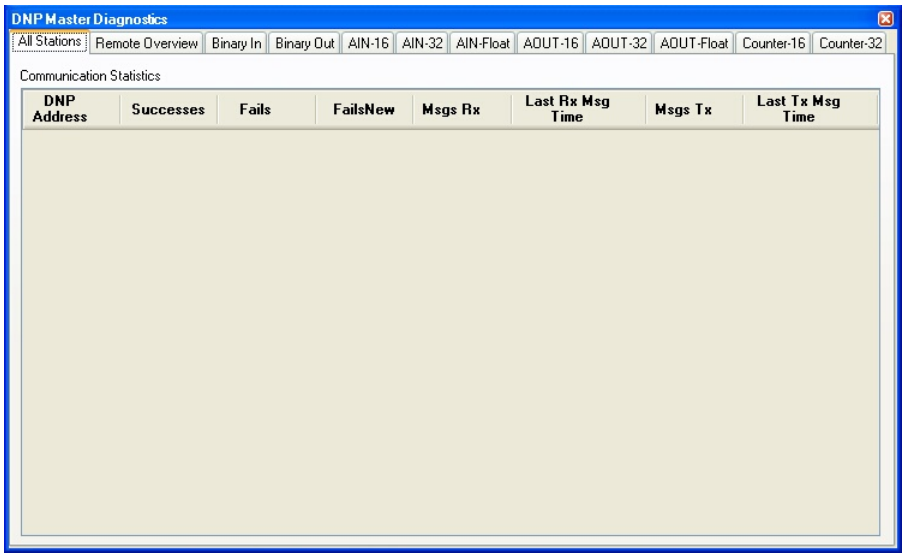
The DNP Master Diagnostics view has a number of selectable tabs and opens with the **All Stations** tab selected. The following tabs are displayed.

- **All** Stations
- **Remote Overview**
- **Binary In** (binary inputs information)
- **Binary Out** (binary outputs information)
- **AIN-16** (16-bit analog inputs information)
- **AIN-32** (32-bit analog inputs information)
- **AIN-Float** (short float analog inputs information)
- **AOUT-16** (16-bit analog outputs information)
- **AOUT-32** (32-bit analog outputs information)

- **AOUT-Float** (short float analog outputs information)
- **Counter-16** (16-bit counter inputs information)
- **Counter-32** (32-bit counter inputs information)

All Stations Tab

The All Stations tab displays the run-time communications diagnostics for all outstations polled by the master or outstations reporting unsolicited data to the master.



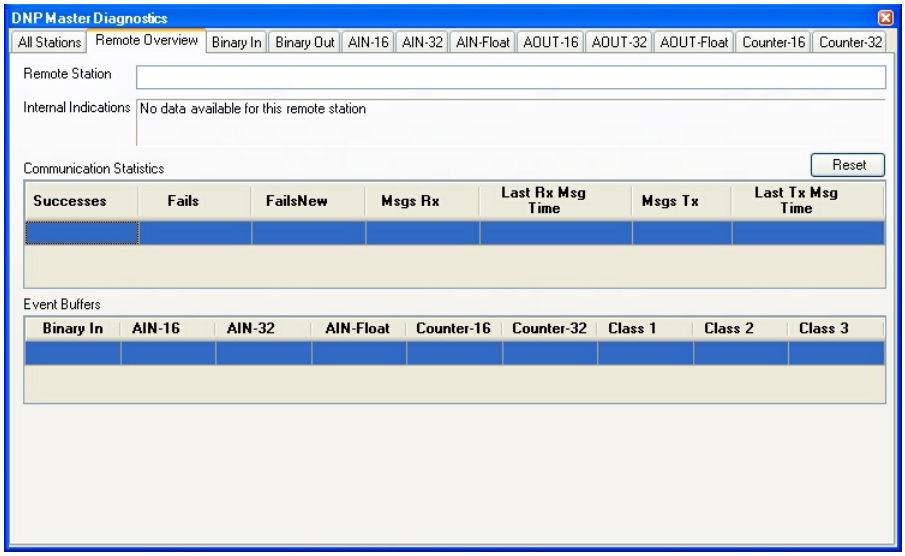
The **Communication Statistics** window displays a list of all outstations and the communication statistics for each station in the list. The statistics counters increment whenever a new DNP message is sent or received, and roll over after 65535 messages. The following statistics are displayed.

- **DNP** Address displays the DNP address of the outstation.
- Successes display the number of successful message transactions between this master and the corresponding remote station. This number includes master polls to the remote station and unsolicited responses from the outstation.
- Fails displays the number of failed message transactions between this master and the corresponding remote station. This counter increments by 1 for a failed message transaction irrespective of the number of application layer retries.
- FailsNew displays the number failed message transactions between this master and the corresponding remote station since the last successful poll.
- Msgs Rx displays the number of DNP packets (frames) received from the outstation station. This number includes frames containing unsolicited responses from the outstation.
- Last Rx Msg Time displays the time the last DNP packet (frame) was received from the outstation.
- Msgs Tx displays the number of DNP packets (frames) sent to the outstation.
- Last Tx **Msg** Time displays the time the last DNP packet (frame) was sent to the outstation.

The Msgs Tx and Msgs Rx counters could be greater than or equal to the Successes and Fails counters.

Remote Overview Tab

The Remote Overview tab displays the run-time diagnostics and current data values for a selected remote station. The data shown is from the image of the data in the master station.



The **Remote Station** window is where the DNP address of the remote station is entered. When the Remote station field is changed all data fields on this tab and the

following I/O tabs are updated with the values for the newly selected Remote Station.

The **Internal Indications** window displays the current state of the DNP internal indications (IIN) flags for the selected remote station. For a detailed description of the IIN flags see Internal Indications IIN Flags.

The **Communication Statistics** window displays communication statistics for the remote station selected. The statistics counters increment whenever a new DNP message is sent or received, and roll over after 65535 messages. The following statistics are displayed.

- **Successes** displays the number of successful messages received in response to master polls sent to the station. This number includes unsolicited responses from the outstation.
- **Fails** displays the number of failed or no responses to master polls sent to the outstation.
- **FailsNew** displays the number failed or no responses to master polls sent to the outstation since the last successful poll.
- **Msgs Rx** displays the number of messages received from the outstation station. This number includes unsolicited responses from the outstation.
- **Last Rx Msg Time** displays the time the last message was received from the outstation.
- **Msgs Tx** displays the number of messages sent to the outstation station.
- **Last Tx Msg Time** displays the time the last message was sent to the outstation.

Click **Reset** to reset the counters to zero.

Event Buffers shows the number of events in each type of event buffer and the allocated buffer size. The buffers shown are for binary inputs, 16-bit analog inputs, 32-bit analog inputs, Floating point analog inputs, 16-bit counter inputs, and 32-bit counter inputs, and Class 1, 2, and 3 events.

The **Event Buffers** window displays the number of events in each type of event buffer and the allocated buffer size for the selected remote station. The event buffers displayed are:

- Binary In (binary inputs)
- AIN-16 (16-bit analog inputs)
- AIN-32 (32-bit analog inputs)
- AIN-Float (floating point analog inputs)
- Counter-16 (16-bit counter inputs)
- Counter-32 (32-bit counter inputs)
- Class 1 (class 1 events)
- Class 2 (class 2 events)
- Class 3 (class 3 events)

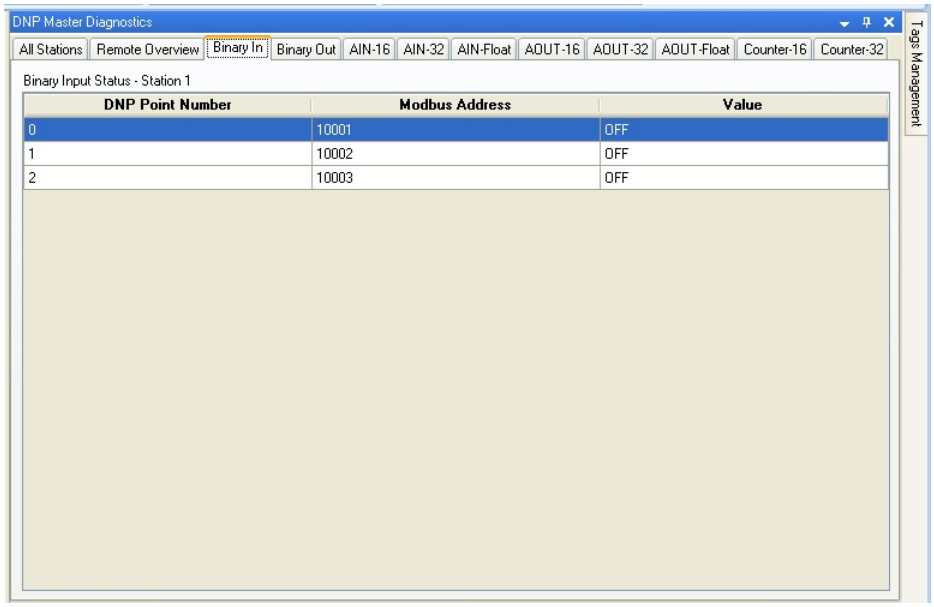
Due to a limitation of the DNP3 protocol, an Unsolicited message from an outstation is not capable of including information stating which data class generated the message. As a result, all Unsolicited events when received by the master will be counted as Class 1 events. Events which are polled by the master, however, do contain class information and will be counted in the Event Buffer for the appropriate class.

Remote Point Status Tabs

The point status tabs show the state of each point of the selected type in the remote station selected on the Remote Overview tab. The values shown are from the image of the remote station in the master station.

Class 0 polling of an outstation needs to be enabled in the master in order to allow that outstation's DNP points to be listed on these tabs. This is the only way for the master to retrieve a complete list of all points in an outstation.

The example below shows the appearance of the Binary In tab.



The **DNP Point Number** column shows the DNP address of the point.

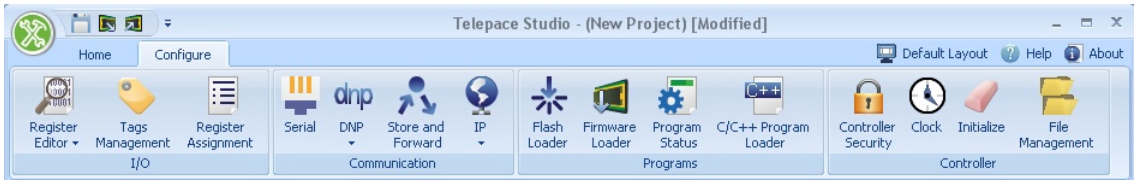
The **Modbus Address** column shows the Modbus register address of the point. This is only relevant for points that have an address mapping in the master station. For points that have an address mapping, this will show the Modbus register address of the point. For points not having an address mapping, this will show '---'.

The **Value** column shows the value of the point. Binary points are shows as OFF or ON. Numeric points show the numeric value of the point.

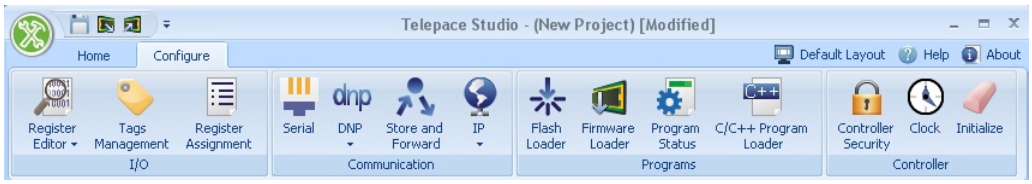
Configure Ribbon

The **Configure Ribbon** provides commands for loading and checking the status of programs, installing new firmware and setting the controller security and controller type. The ribbon is displayed differently depending on the controller type.

For controller types except FlowStation the ribbon is displayed as shown below.



For **FlowStation** controllers the ribbon is displayed as shown below.



The I/O Group contains commands for managing the I/O for the selected target controller.

The Communications Group contains commands for configuring the communication ports on the selected target controller.

The Programs Group contains commands to load programs into the controller flash memory, load new firmware, load C/C++ applications and to check the status of programs running in the controller.

The Controller Group commands are used to change the controller type, set the controller security, Real Time Clock and initialize the memory in the controller. File management functions are available for non-FlowStation controllers.

I/O Group

The **I/O Group** is located in the Configure Ribbon.

The I/O group contains the following commands:

- The Register Editor command opens the Register Editor view. This view allows the online monitoring and control and the offline editing of registers used in a Telepace Studio application.
- The Tags Management command opens the Tag Management view. The Tags Management view is used to view and create tag names for registers used in the Telepace Studio application.
- The Register Assignment command opens the Register Assignment view. The Register Assignment view is used to add and remove I/O modules from the register assignment.

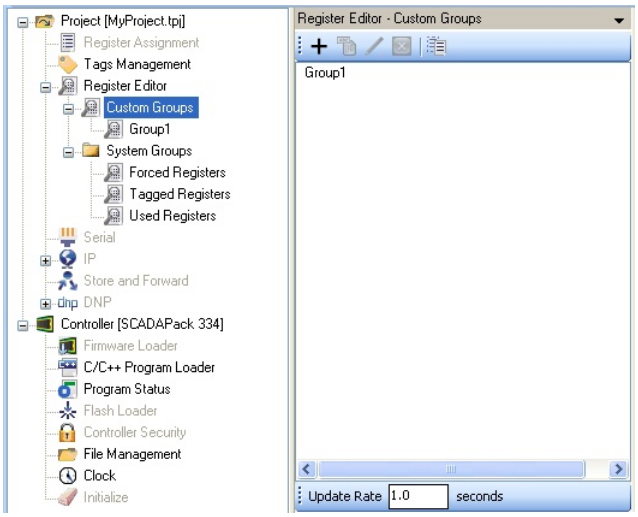
Register Editor

The **Register Editor** view allows the online monitoring and control and the offline editing of registers used in a Telepace Studio project. These registers may be added in a single group, or in multiple groups for monitoring. Individual registers may be selected for on-line and off-line editing of their data.

The Register Editor can be selected from the navigation tree or from the I/O group in the Configure Tool Ribbon. The Register Editor may be selected in the Offline, Edit Online and Monitor Online modes.

Navigation Tree Location

Tool Ribbon Location



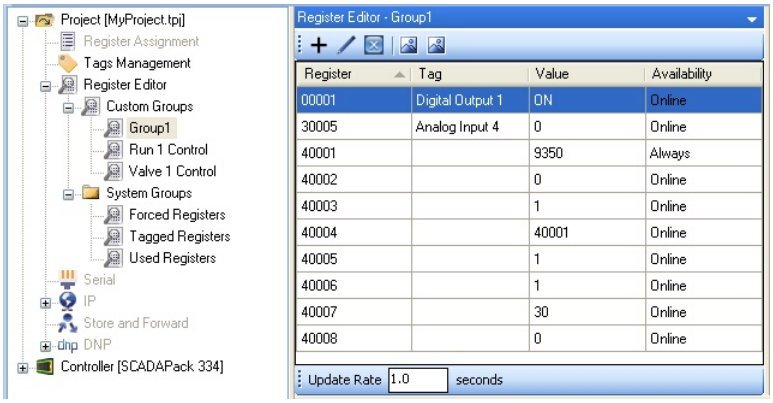
When a group or register list is selected from the Register Editor the property view changes to a view of the group or register list.

- The **Custom** Groups selection opens a list of user created register groups. Groups are named as they are created. Telepace Studio creates the group Group1 by default.
- The **Forced Registers** selection displays a list of registers that have been forced.
- The **Tagged Registers** selection displays a list of registers that have been assigned a tag name.
- The **Used Registers** selection displays a list of all registers that are used in the application.

The **Update Rate** field determines how often the register group is updated when in the on-line mode. Valid values are 0.1 to 1000 seconds. The default value is 1.0 seconds. This is a direct entry field and the value entered is used when the keyboard enter key is clicked or the mouse is navigated to another area.

Register Editor View

The Register Editor view displays the registers in the selected register group. See the Editing Custom Groups topic for information on adding, deleting and editing registers in a custom group.

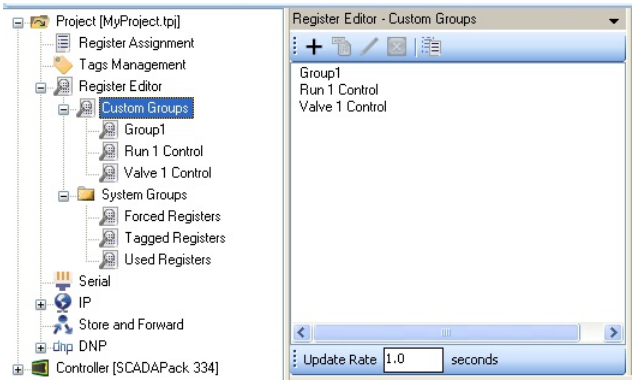


The Register Editor view displays the Register, the assigned Tag name for the register, the Value of the register and the register Availability in a list of all registers in the group.

The columns in the Register Viewer may be sorted in ascending or descending order by clicking the column heading. Columns are sorted in alphabetical order except the Value column. The Value column is sorted by digit value and alphabetical value.

Custom Groups

Custom register groups allows users to manage how registers are viewed. Registers that are used in various parts of an application can be grouped together to provide an easy to view, and edit, list of registers. When in the Online mode, only the registers in the current selected group are updated at the Update Rate. This improves response time as large groups of registers can take a long time to update. The screen shot below shows the Custom groups list containing two groups that have been added as well as the original Group1 group.



The toolbar at the top of the Custom Groups list provides easy access to the commands used to manage groups. Hover your mouse over the toolbar to see the tool tip for each icon.

Add a Group

Groups are added to the list using the Add Group command. Click the first icon in the toolbar to select this command or right click the mouse on the Custom Groups item in the navigation tree. When the Add Group command is selected a new group is displayed in an edit window in the list. Type in a name for the group or leave the default name for the group. The group name may be edited at any time.

- The custom group name may contain letters, numbers, special characters and spaces. The group name is 1 to 32 characters long and are case sensitive.
- Group Names need to be unique. A message will be displayed if a group name is entered that already exists in the Telepace Studio project.

Rename a Group

Groups may be renamed using the Rename Group command. Highlight the group to be renamed in the list and click the second icon in the toolbar to select this command or right click the mouse on the group in the list and select Rename Group from the command list displayed. When the Rename group command is selected the highlighted group will change to an edit window. Type in the new name for the group in the edit window.

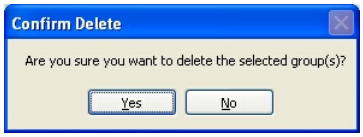
- The custom group name may contain letters, numbers, special characters and spaces. The group name is 1 to 32 characters long and are case sensitive.
- Group Names need to be unique. A message will be displayed if a group name is entered that already exists in the Telepace Studio project.

Edit a Group

Each custom group typically contains registers that have been grouped for a specific purpose. Registers may be added, deleted or edited in a custom group. See the topic Editing Custom Groups for information on managing the registers in a custom group.

Deleting a Group

Groups may be deleted using the Delete Group command. Highlight the group to be deleted in the list and click the fourth icon in the toolbar to select this command or right click the mouse on the group in the list and select Delete Group from the command list displayed. When the Delete group command is selected the Confirm Delete dialog is displayed.



- Click the **Yes** button to delete the selected group.

Importing Groups

Custom Groups may be imported from Telepace Studio versions 4.x and 5.x projects using the Import Groups command. Clicking on the fifth icon, Import Groups, opens the Import Groups from File dialog. Telepace Studio version 4.x uses an XML file to store register groups. These files will be listed in the dialog along with Telepace Studio version 5.0.2 project files, with the TPJ file extension.

- Use the **Look in:** navigation list to locate the folder containing the Telepace Studio project to import the groups from.
- Highlight the file to import from in the Import Groups form File dialog and click the **Open** button. The Register Groups from the selected file are imported to the current Telepace Studio project.

If the import group has the same name as that in the current project, then the entries in the import group will be added to the current group in the project.

If imported group's name does not exist in the current project, then the group will be created in the current project.

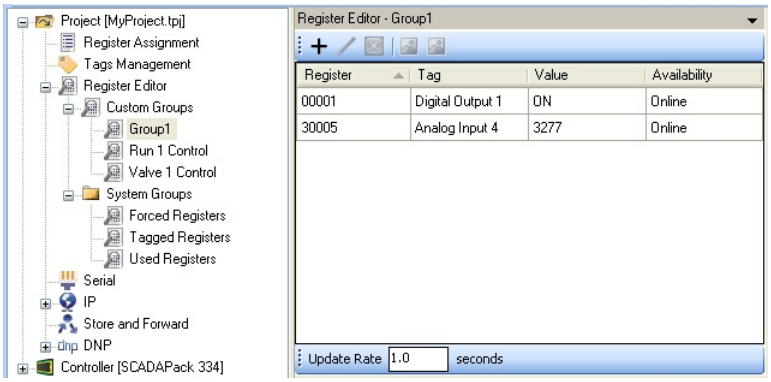
If some entries in the file conflict with the current entries, then the current entries will be overwritten by the entries of imported group. However, if the register in the imported group is used by the element configuration of the current ladder program, the settings of the register will obey the current ladder program settings.

Tags are not imported with the import group. Any tags associated with registers in the current group are retained.

Editing Custom Groups

Editing custom register groups involves adding, deleting and modifying registers in the custom group. To edit the register contents of a group highlight the group to edit by clicking on the group name in the navigation tree.

The screen capture below displays the Register Editor window for the selected group, Group1. Two registers have been added to Group1. The register address, assigned tag name, the current value and the availability type for the registers are displayed.



The toolbar at the top of the Custom Groups list provides easy access to the commands used to manage groups. Hover your mouse over the toolbar to see the tool tip for each icon.

Add Register

Registers are added to the group using the Add Register command. Click the first icon in the toolbar to select this command or right click in the Register Editor display list and select the Add Register command from the displayed command window. When the command is selected the **Add Registers to *group name*** dialog is opened. The dialog will display the group name in the header. In the dialog below registers are to be added to Group1.

The **Available Registers** section of the dialog lists the registers available for the controller type. See the Telepace I/O Database Registers topic for the registers available for your controller type.

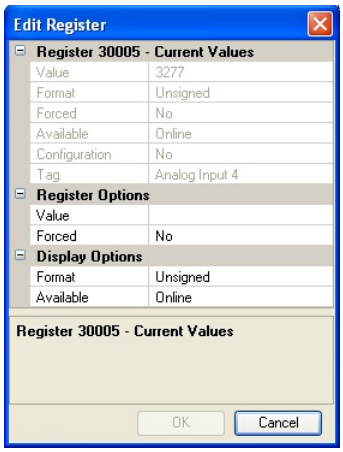
The **Registers to Add** section of the dialog defines the registers to be added and information about how the registers contents are displayed.

- The **Start Register** window defines the start address of a single register or a block of registers. The start register and the end register need to be of the same type, i.e. coil, status, input or holding registers.
- The **End Register** window defines the end address of a single register or a block of registers. The end register and the start register need to be of the same type, i.e. coil, status, input or holding registers.
- The **Format** selection defines how the registers will be displayed in the value column. See the Register Formats topic for the register format types available.
- The **Availability** selection defines when register values can be viewed and edited. Valid values are **Always** and **Online**. The Always selection makes the register values available in both online and offline editing modes. The Online selection makes the register values available only in the online editing modes. The default is **Always** if Telepace Studio is in the offline mode and **Online** if Telepace Studio is in the edit offline or monitor mode.

Once the registers to add have been defined in the Add Registers dialog click the Add button to add the registers to Register Editor group list. If a register has a tag name assigned the tag name will be displayed with register address.

Edit Register

Registers are edited using the **Edit Register** command. Click the second icon in the toolbar to select this command or right click on the register in the Register Editor display list and select the Edit Register command from the displayed command window. When the command is selected the **Edit Register** dialog is opened as shown below.



The **Register xxxxx - Current Values** section of the dialog displays the current information for the selected register.

- The **Value** window contains the current value of the register.
- The **Format** window displays the current format of the register. This parameter was set when the register was added to the group and can be changed in the Display Options section of this dialog.
- The **Forced** window displays the forced status of the register. If the register is currently forced the status will be Yes and if not forced the status will be No.
- The **Available** window displays the current availability for the register, either Online or Offline. This parameter was set when the register was added to the group and can be changed in the Display Options section of this dialog.
- The **Configuration** window displays whether the register is used in an element configuration for a function block or not. Some function blocks such as the PIDA, PIDD and MSTR use multiple registers in their configuration. The values of these registers determine how the function block will operate. These types of function blocks use Element Configuration to populate the registers with the correct information needed for the function block to operate as programmed. If the register is currently used in an element configuration the status will be **Yes** and if not the status will be **No**.
- The **Tag** window displays the assigned tag name if one exists. The window is blank if the register has no tag name assigned.

The **Register Options** section of the dialog allows the changing of the register value and the forced status.

- The **Value** entry window accepts any valid value to be entered for the selected register. This feature allows you to change a register value and the Telepace Studio program will then use the new value. This is useful in testing programs. For example, if a counter function is set to count to a certain value and when the value is reached the program performs some other function, it would be beneficial to set the counter register to a value near the limit to confirm the program work as expected.

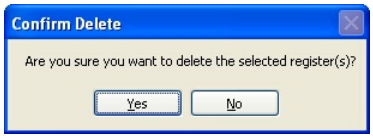
  When a register is assigned to physical I/O using a Register Assignment the controller will automatically update the register with data from the physical I/O. For example if a register is assigned to a physical analog input point the register will be updated with the actual value from the input point. In this case the value entry will accept a new value but the value will immediately be over written with the data from the analog point. Registers that have been assigned to physical I/O need to be set to forced. When a register is forced any valid value may be entered for the selected register.
- The **Forced** selection sets the forced status for a register. When a register is forced it is not updated by the I/O system on the controller (register assignment) or by the ladder Logic or C/C++ programs. When the status is set to forced the register is set to the Value entered.

The **Display Options** section of the dialog allows the changing of the register format and availability.

- The **Format** selection defines how the registers will be displayed in the value column. See the Register Formats topic for the register format types available.
- The **Available** selection box is initialized to the current value. The **Always** selection makes the register values available in both online and offline editing modes. The Online selection makes the register values available only in the online editing modes. The default is **Always** if Telepace Studio is off-line, and **Online** if Telepace Studio is on-line.

Delete Register

Registers are deleted from the Register Editor using the Delete Register(s) command. Click the third icon in the toolbar to select this command or right click on the register in the Register Editor display list and select the Delete Register(s) command from the displayed command window. When the command is selected the Confirm Delete dialog is opened as shown below.



- Click **Yes** to delete the selected register from the Register Editor.

Multiple registers in the Register Editor list may be selected using the keyboard **<Shift>** and **<Ctrl>** buttons.

- To select a sequential block of registers first click your mouse on the first register, hold down the **<Shift>** key and then click on the last register. The block of register will be highlighted and may be deleted.
- To select multiple registers first click your mouse on a register, hold down the **<Ctrl>** key and then click on any other registers that are to be deleted. The registers will be highlighted and may be deleted.

Available Always

When a register is set for Available Always the register Value is displayed in both the Offline and Online modes. The difference between this setting and the Available Online setting is that the register value is displayed and can be changed in the Offline mode. When the program is written to the target controller the values in registers set for Always Available are written. This feature allows you to preset registers when a program is written tot he controllers.

Highlight the registers in the Register Editor list that are to be set for Available Always and then click the fourth icon in the toolbar. The availability for the register(s) will change to Always in the Availability column.

Available Online

When a register is set for Available Online the register Value is displayed in the Online mode only. When in the Offline mode the Register Editor does not display the register value.

Highlight the registers in the Register Editor list that are to be set for Available Online and then click the fifth icon in the toolbar. The availability for the register(s) will change to Online in the Availability column.

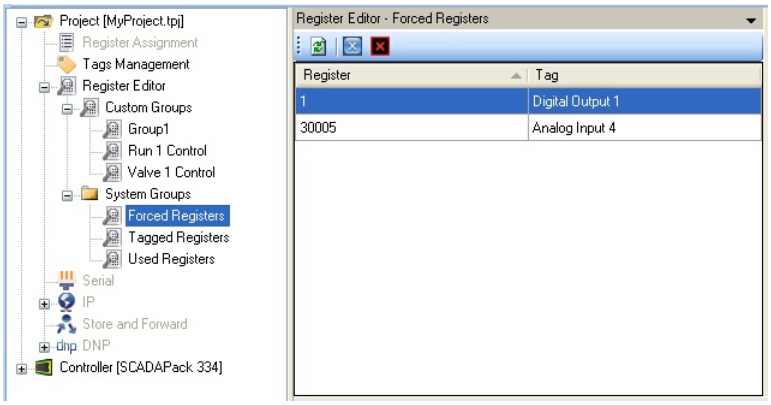
System Groups

Telepace Studio provides a system group folder containing three system groups.

- The Forced Registers group is automatically updated with any registers that have been forced in the target controller.
- The Tagged Registers group is automatically updated with any registers that have been assigned tag names.
- The Used Registers group is automatically updated with registers that are currently used in the Telepace Studio application.

Forced Registers

The Forced Registers group is a list of registers that have been forced in the target controller. The list displays the Register and the assigned Tag for the register.



The toolbar at the top of the **Custom Groups** list provides easy access to the commands used to manage groups. Hover your mouse over the toolbar to see the tool tip for each icon.

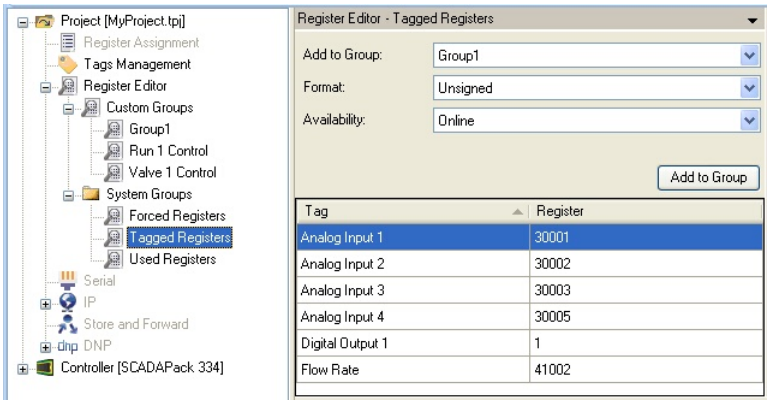
The first icon, **Refresh**, updates the list.

The second icon, **Unforce Selected Registers**, changes the force status for the selected register(s) to unforced. Multiple registers may be selected for unforcing. Use the **<Shift> or <Ctrl>** keys to select multiple registers.

The third icon, **Unforce All Registers**, changes the force status for all registers in the list to unforced.

Tagged Registers

The Tagged Registers group is a list of every register that has been assigned a tag name in the program. From this system group registers can be added to any custom group. The Tagged Register group contains editing functions to add the register to any custom group(s), change the format for the register and change the availability of the register. The list displays the Register and the assigned Tag for the register.



The **Add to Group** window displays the current group the register is assigned or the default group, Group1, if the register has not been assigned to a custom group. The drop down selection contains a list of all custom groups.

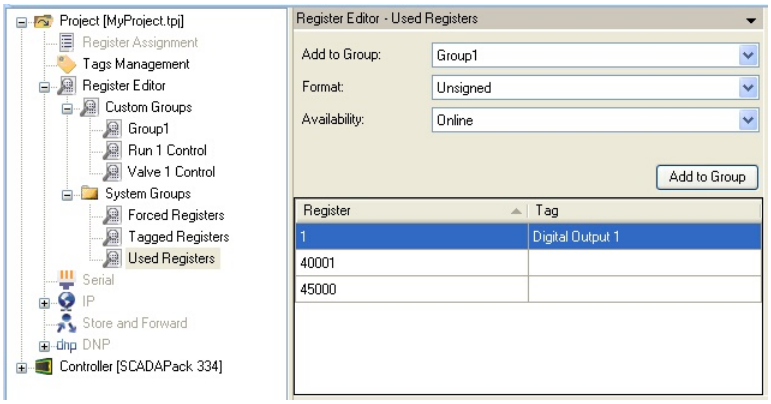
- To add a tagged register to one or more custom groups first highlight the tagged register to be added then select the group to added the register to and click the **Add to Group** button. Use the **<Shift>** or **<Ctrl>** keys to select multiple registers for adding.
- Repeat this procedure to add a register or multiple registers to another group.

The **Format** selection defines how the registers will be displayed in the value column. See the Register Formats topic for the register format types available.

The **Availability** selection defines when the register value is displayed in the custom group.When a register is set for Available Always the register Value is displayed in both the Offline and Online modes. When a register is set for Available Online the register Value is displayed in the Online mode only. See the Editing Custom Groups topic for further details on register availability.

Used Registers

The Used Registers group is a list of registers that are used in the ladder logic program. The list does not include registers used in Register Assignment. The list displays the Register and the assigned Tag for the register.



The **Add to Group** window displays the current group the register is assigned or the default group, Group1, if the register has not been assigned to a custom group. The drop down selection contains a list of all custom groups.

- To add a tagged register to one or more custom groups first highlight the tagged register to be added then select the group to added the register to and click the **Add to Group** button. Use the **<Shift>** or **<Ctrl>** keys to select multiple registers for adding.
- Repeat this procedure to add a register or multiple registers to another group.

The **Format** selection defines how the registers will be displayed in the value column. See the Register Formats topic for the register format types available.

The **Availability** selection defines when the register value is displayed in the custom group.When a register is set for Available Always the register Value is displayed in both the Offline and Online modes. When a register is set for Available Online the register Value is displayed in the Online mode only. See the Editing Custom Groups topic for further details on register availability.

Register Formats

Register formats define how the register value is displayed in the Register Editor. The register formats available are listed in the following table.

Format	Description
Unsigned	Displays a single register as an unsigned integer. Unsigned integer data are in the range 0 to 65535. This data requires a single 16-bit register, 3xxxx or 4xxxx.
Signed	Displays a single register as a signed integer. Signed integer data are in the range –32768 to 32767. This data requires a single 16-bit register, 3xxxx or 4xxxx.
Unsigned Double	Displays two consecutive registers as an unsigned double integer. Unsigned long integer data are in the range 0 to 4,294,967,295. This data requires two 16-bit registers, 3xxxx or 4xxxx. The lower numbered register contains the lower order word.
Signed Double	Displays two consecutive registers as a signed double integer. Signed double integer data are in the range –2,147,483,648 to 2,147,483,647. This data requires two 16-bit registers, 3xxxx or 4xxxx. The lower numbered register contains the lower order word.
Floating Point	Displays two consecutive registers as a floating point number. Floating-point data are in the IEEE single precision floating-point format. This data requires two 16-bit registers, 3xxxx or 4xxxx. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.
Hexadecimal	Displays a single register as a hexadecimal number.
Binary	Displays a single register as a binary number.
ASCII	Displays a single register as two ASCII characters. The first character is in the low order byte, the second in the high order byte.
Boolean	Displays coil and status registers which have no format.

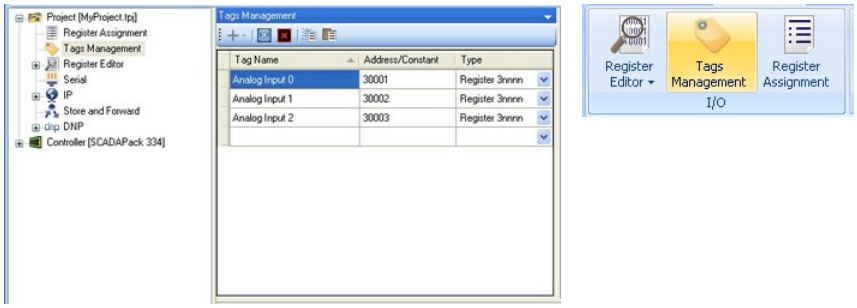
Tags Management

Tags are used to provide useful descriptions of registers used in a Telepace project. The **Tags Management** view is used to create, delete and view tags.

The Tag Management Editor can be selected from the navigation tree or from the I/O group in the Configure Tool Ribbon. The Tag Management Editor may be selected in the Offline, Edit Online and Monitor Online modes.

Tag names are saved as part of the Telepace Studio program and are not stored in the target controller. When a program is read from a target controller into a new Telepace Studio project there are no tag names assigned to any registers in the new project.

Navigation Tree Location **Tool Ribbon Location**



The toolbar at the top of the Tags Management list provides easy access to the commands used to manage tags. Hover your mouse over the toolbar to see the tool tip for each icon.

The **Tags Management** view displays a grid containing three columns and initially a blank tag entry row. When tags are entered the blank row will be the last row on the grid.

- The **Tag Name** field contains the tag name for the tag entry. Tag names are a maximum of 32 alphanumeric characters and are case sensitive.
- The **Address/Constant** field contains the Modbus register address if the tag is a register type or a numeric value if the tag is a constant type.
- The **Type** field is a register or constant type selected from the drop down selector at the right of the field. Valid selections are:
  - Constant
  - Register 0nnnn – Coil register type.
  - Register 1nnnn – Status register type.
  - Register 3nnnn – Input register type.
  - Register 4nnnn – Holding register type.

The toolbar at the top of the Tags Management view provides easy access to the commands used to manage groups. Hover your mouse over the toolbar to see the tool tip for each icon.

Adding a Single Tag

Single tags are added by entering a complete Tag entry row using the Tag Name, Address/Constant and Type fields. There is no limit to the number of tag names that can be added.

The fields may be entered in any order but all fields need to have an entry. If a field is left blank Telepace Studio will display a message prompting for an entry.

There are a number of methods that can be used to navigate between fields:

- Use your mouse and click in any field.
- Use the keyboard **Tab** key to move to the next field to the right.
- Use the keyboard **Arrow** keys to move between fields.

When a field is entered a blank entry row is created in the grid under the tag being entered.

Tags are saved when the field entries have been completed and a field in the blank entry row navigated to using one of the methods listed above.

Telepace Studio checks to see if the newly created tag matches an existing tag in the current project and displays a message if the tag name matches an existing tag name or if a Modbus address of a tag matches the Modbus address of an existing tag.

Adding Multiple Tags

Telepace Studio provides the functionality to automatically add multiple copies of a tag name to the Tags Management grid. There is no limit to the number of tag names that can be added. When multiple tags are created each new tag retains the copied tags Tag Name and adds a post script containing an underscore and a sequential number starting at 2 to the Tag Name.

When the new tags created are of Address type the **Address/Constant** field will contain sequential register addresses starting at the next sequential address. There needs to be enough sequential registers for the number of multiple tags selected or Telepace Studio will present a message indicating there are not enough registers..

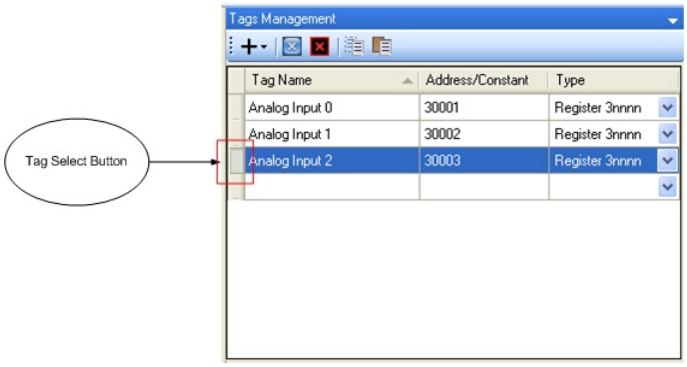
When the new tags created are of Constant type the **Address/Constant** field will contain a value of 1 for each new tag name.

To create multiple copies of a selected tag:

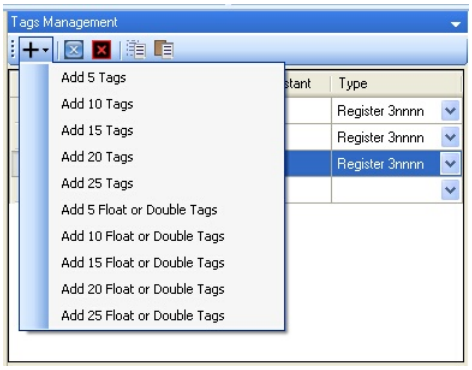
- Left click the mouse on the **Tag Select** button at the left edge of the tag entry. See below for the location of the tag select button.
- Or click the **Add Tags** icon, the first icon in the toolbar.
- Hover on the **Add Tag** names command in the command list displayed and select the number of tags to add.

The following screen shots show the creating of multiple tags created from the Analog Input 2 tag.

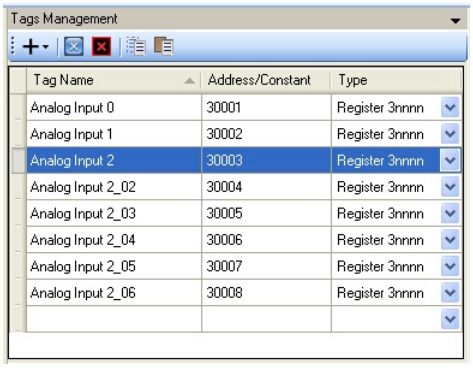
Highlight the tag by clicking the **tag select** button.



Click the **Add Tags** button and select the number of sequential tags to add. The Float of Double Tags use two registers.



The Add 5 Tags selection produces the following tag list.



Delete Selected Tags

Once the tag is selected, it may be removed using one of the following methods:

- Press the keyboard **Delete** key.
- Right click the mouse and select **Delete Selected Tags** from the command menu.
- Click the **Delete Selected Tags** button.

Delete All Tags

The **Delete All Tags** command is used to erase all tag names in a Telepace Studio project. This command can be used before importing tags from another project or a CSV file so that there are no conflicts between existing tag names and imported tag names.

To Delete All Tags in a Telepace Studio project:

- Right click the mouse anywhere in the **Tags Management** grid and select **Delete All Tags** from the command menu.
- Or click the **Delete All Tags** icon in the toolbar.

Import Tags

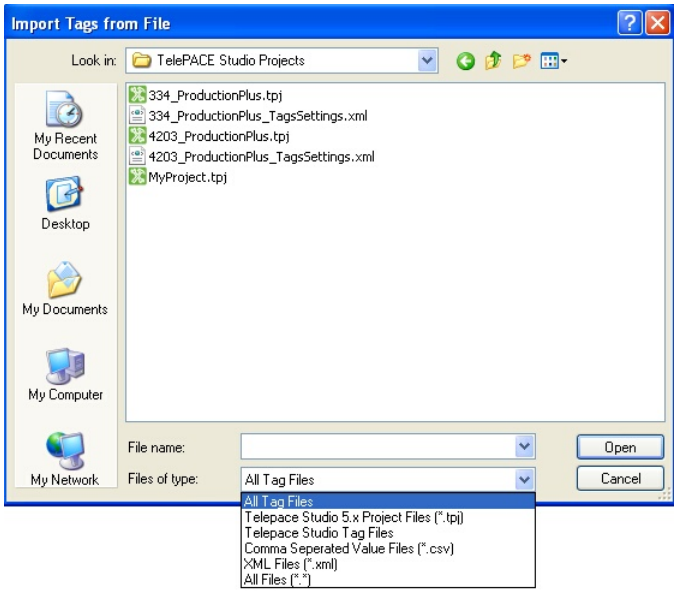
Importing tags is a useful means of adding tag names to a project without using the Tags Management view to add single or multiple tags. Tags may be imported from the following file types:

- Telepace Studio 5.x Project Files (*tpj)
- Telepace Studio Tag Files
- Comma Separated Value Files (*.csv)
- XML Files (*.xml)

To Import Tags to a Telepace Studio project:

- Right click the mouse anywhere in the **Tags Management** grid and select **Import Tags** from the command menu.
- Or click the **Import Tags** icon in the toolbar.

The Import Tags from File dialog is opened. Select the file to import tags from and click **Open**.



When the tags are imported Telepace Studio displays an information dialog to confirm the tags were added or to list any conflicts in the import.

Export Tags

Telepace Studio exports tags in the current project to tags to a comma separated variable (**CSV**) tag list file. The format of the CSV file that is export is shown in the following example:

```
Type,Value,Tag
Constant,101,BaroPress
Register 0nnnn,1,Stop
Register 1nnnn,10001,Start
Register 3nnnn,30001,Run1 DP (raw)
Register 4nnnn,42100,Flow Rate
```

The CSV file contains a title line followed by multiple data lines containing the **Type**, **Value,** and **Tag** fields. Commas separate each of these fields.

The **Type** field indicates the type of data the tag contains. The **Type** field is one of the following:

Constant

Register 0nnnn

Register 1nnnn

Register 3nnnn

Register 4nnnn

The **Value** field contains the Modbus register address for the tag or the value of Constant type.

The **Tag** field contains the tag name.

Register Assignment

Register Assignment is used to assign I/O database registers to user-assigned registers using I/O modules. An I/O Module can refer to an actual I/O hardware module such as a *5401 Digital Input Module* or it may refer to a set of controller parameters, such as serial port settings.

- I/O modules used by the controller needs to be assigned to I/O database registers in order for the I/O points to be used by the ladder program.
- The configuration and diagnostic I/O modules used by the controller needs to be assigned to I/O database registers in order for the I/O points to be used by the ladder program.

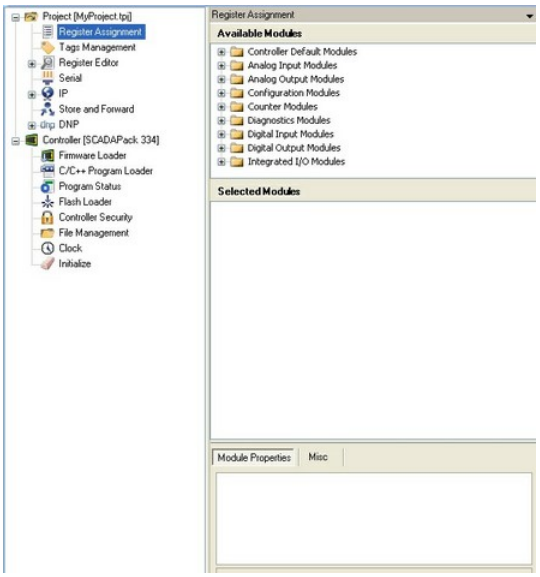
Ladder logic programs may read data from, or write data to the physical I/O only through user-assigned registers in the I/O database. Ladder logic programs may read data from, or write data to, the I/O hardware through user-assigned registers in the I/O database.

The **Register Assignment** command opens the Register Assignment view enabling you to assign controller I/O modules, configuration modules and diagnostic modules to database registers that are used in your Telepace Studio program.

The Register Assignment command is selected from the navigation tree or from the I/O group in the Configure Tool Ribbon. Register Assignment may be selected in the Offline, Edit Online and Monitor Online modes.

Navigation Tree Location

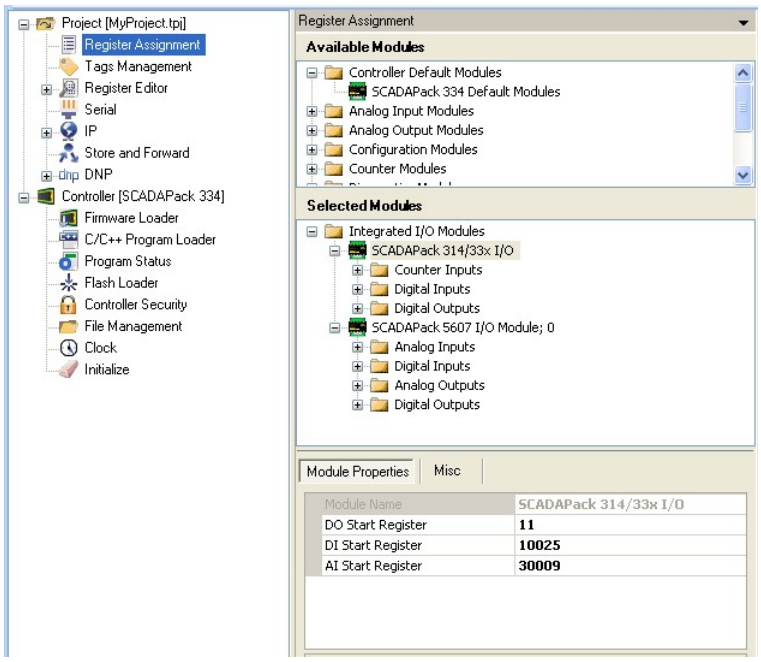
Tool Ribbon Location



Register Assignment View

The **Register Assignment** view is launched from the **Register Assignment** command located on the Edit ribbon.

The Register Assignment workspace is presented as a single view divided into three sections, Available Modules, Selected Modules and Module Properties / Misc as shown in the figure below.



The **Available Modules** section presents a list of the I/O modules supported by the selected controller type. A module that has to be added to the project is selected from this view.

This view incorporates a tree view structure, which allows all I/O modules supported by the controller to be presented in a categorized manner. The I/O modules have been categorized into top level folders based on the module type. Child nodes representing each available module type are placed under the top level category. The presentation of I/O modules within this view cannot be changed.

The Available Modules view will display modules supported by the target controller. For a complete list of Register Assignment modules see Register Assignment Reference.

Modules to be added to the project are selected from this view. See Adding a Module for information on adding a Register Assignment module.

The **Selected Modules** is similar to the Available Modules view, modules assigned to this project, and thus targeted to the controller, are displayed in a categorized manner in a tree structure.

Underneath each main category node in this tree view, is a selected I/O module. If applicable, the module address that uniquely identifies this module on the I^2C bus is displayed besides the module name. In other cases, the communication interface associated with this module is displayed besides the module name, as displayed in the figure below.

Displayed beneath each Selected I/O module, as child nodes, are the I/O channels associated with the parent module. Each channel node displays an associated index that specifies this particular channel type and offset from a one-base index, the corresponding Modbus register, and tag name attached to this channel.

The module property section also displays certain channel properties such as the Modbus register and the tag name associated with a particular channel, as shown below.

The **Module Properties** section presents a view of all module properties, besides those presented as read-only fields, can be edited from the **Module Properties** data grid only. When a module is added to the project, default module properties required for a proper configuration are assigned automatically and displayed in the **Module**

Properties data table. You can change these properties if needed.

For example:

- The module address, if supported, is automatically set to the next available module address.

The starting Modbus register is set automatically to a register that is not already used by an I/O module currently in the project.

  - **Digital Output Starting Registers** are assigned within the range **00001** to **04096.** Beginning zeros of this register type are not displayed.
  - **Digital Input Starting Registers** are assigned within the range **10001** to **14096.**
  - **Analog Input Starting Registers** are assigned within one of the following ranges as determined by controller type:
    - **30001** to **33051**
    - **30001** to **21024**
    - **30001** to **39999**
  - **Analog Output Starting Modbus Registers** are assigned within one of the following ranges determined by the controller type:
    - **40001** to **49999**
    - **40001** to **44999**
  - **Analog Output Starting Register** for SCADAPack 4202/4203 controllers is 40500.

Editing Module Properties

Modules that can have multiple instances supported on the 1^2C bus, present a **Module Address** field.

To change the module address from one automatically assigned:

- Click on the drop-down arrow at the end of the text box beside the **Module Address** field.
- Select an address from those available from the drop-down list.

Addresses that have already been used are not displayed.

Editing the Module Communication Interface

Modules that support communication interfaces have a **Comm Interface** field, in place of the **Module Address** field. To change the selected communication interface, click on the drop-down arrow at the end of the field and select the desired interface. This list is populated with communication interfaces that have not been used, and which are supported by this controller and I/O modules on the bus.

Editing the Starting Modbus Register

A module will have at least one starting Modbus register. As stated above, default non-conflicting Modbus registers are assigned automatically when a module is added to the project. Register conflicts are checked only against other register assignment modules, and not against registers used within the Network canvass.

To change a module's starting **Modbus address**, select the module from the *Selected Modules* view, and edit the entry of the desired Modbus register type.

Invalid register types, as well as conflicting entries, will not be allowed as indicated by the message below. The register assignment workspace does not only complain about an invalid entry, it also presents a suggestion as to how to fix it.

For instance, if a user attempts to assign a module to a Modbus register already used, the change will be rejected, but the next available register range, with the register assignment scope, will be presented, as shown below.

If the wrong type of register is assigned to the module, a similar message will be displayed.

If the wrong type of register is assigned to the module, a similar message will be displayed.

The **Misc I/O Properties** tab page of the register assignment editor, presents check boxes to change properties not associated with a particular controller.

I/O Module Error Indication

This controller property determines if the controller displays I/O module communication errors on the Status LED. If enabled, the controller will blink the Status LED if there is an I/O error. By default, the field is enabled.

Digital Outputs on Stop

This selection controls the state of the controller digital outputs when the ladder logic program is stopped.

The state of the digital outputs may be set to Hold its last value or to turn Off when the ladder program is stopped.

The **Digital Outputs on Stop** drop-down list box specifies the state of the digital outputs. The two options are Hold and Off.

Analog Outputs on Stop

This selection controls the state of the controller analog outputs when the ladder logic program is stopped.

The state of the analog outputs may be set to Hold its last value or to go to Zero when the ladder program is stopped.

The **Analog Outputs on Stop** drop-down list box specifies the state of the analog outputs. The two options are Hold and Zero.

Communication Group

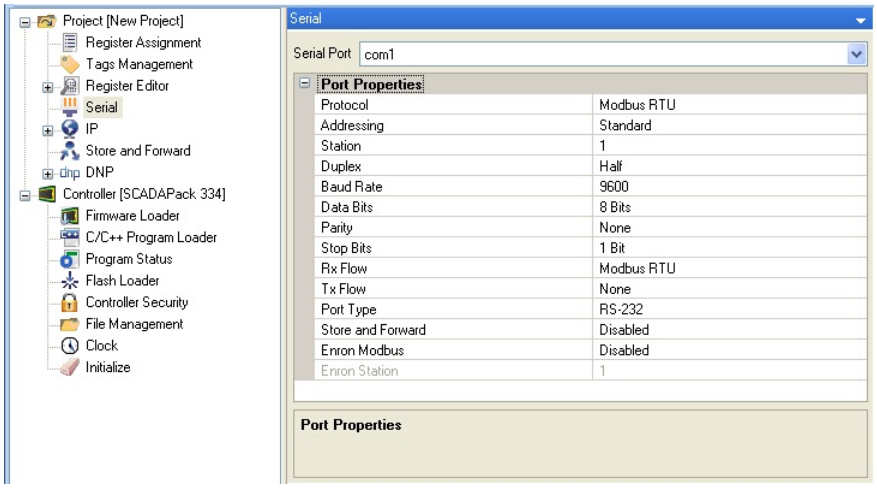
The **Communication Group** is located on the Edit Ribbon.

The Communication group contains the following commands:

- The **Serial** command opens the Serial Port Settings view. This view is used to configure the serial ports on the project controller.
- The **DNP** command opens the DNP Configuration view to provide entries to fully configure the project controller as a DNP slave or, depending on the controller type, a DNP master.
- The **Store and Forward** command is used to configure store and forward messaging.
- The **IP** command is used to configure the IP Communication settings for the project controller.

Serial Port Settings

Serial Port Settings view is opened from the Communications group on the Configure ribbon or from the Tree view. This view is used to configure the serial ports on the controller.



The **Serial Port** drop-down menu selects the target controller serial port to configure. The valid serial ports available for configuration depend on the controller type. Click here, Supported Serial Ports to see the serial ports supported for each controller type.
The SCADAPack 4202 and 4203 controllers support Sensor protocol only on com1. The serial port settings for com1 cannot be edited for these controllers.

Port Properties

The drop-down menus for each parameter let you configure the serial ports for your target controller and Telepace project.

Protocol Parameters

Selects the communication protocol type. The valid protocols depend on the controller type. Click here, Supported_Protocols to see the protocols ports supported for each controller type.

Addressing Method

The Modbus RTU and ASCII protocols support two type of Modbus station addressing. The **Addressing** entry selects between the two addressing modes.

- The **Standard** addressing method selection allows a Modbus address of between 1 and 255 and is compatible with standard Modbus devices.
- The Extended addressing method allows a maximum of 65534 stations. Extended Modbus addressing is fully compatible with standard Modbus addressing for addresses between 0 and 254. This selection is only available for Modbus RTU and ASCII protocols.

The default Addressing Method is **Standard.**

Station Address

The **Station** entry sets the station address for the selected controller serial port. Valid addresses depend on the protocol and addressing mode selected, as shown below.

None	1 to 255
Modbus RTU	Standard: 1 to 255, Extended: 1 to 65534
Modbus ASCII	Standard: 1 to 255, Extended: 1 to 65534
DF1 Full Duplex BCC	0 to 254
DF1 Full Duplex CRC	0 to 254
DF1 Half Duplex BCC	0 to 254
DF1 Half Duplex CRC	0 to 254
DNP	Valid DNP Address.

Duplex

The **Duplex** drop down menu selects full or half-duplex operation for the selected Port. Valid and default duplex settings depend on the serial port and controller type, as shown in the table below. Selects full or half-duplex operation for the selected Port. Valid and default duplex settings depend on the serial port and controller type. Click here, Supported_Duplex to see the duplex settings supported for each controller type.
The duplex is forced to half if an MVT transmitter is configured on the port.

Baud Rate

The **BaudRate** drop-down menu selects the communication speed for the selected serial port. Valid baud rates depend on the serial port and controller type. The default value is 9600 baud except for com1 on SCADAPack controllers. Com2 is fixed at 4800 baud for com1 on SCADAPack controllers. Click here, Supported_BaudRate to see the baud rates supported for each controller type.

Data Bits

The **DataBits** drop-down menu selects the number of data bits. Valid selections are 7 and 8 bits. This parameter is forced to 8 bits when the protocol type is Modbus RTU, PPP or any DF1 protocol. The default selection is 8 bits.

Parity

The **Parity** drop down menu selects the parity for the selected port. Valid selections depend on the serial port, controller type and data bits. The default selection is none. Click here, Supported Parity to see the supported parity settings supported for each controller type.

Stop Bits

The **StopBits** drop down menu selects the number of stop bits for the selected serial port. The valid selections displayed, 1 or 2 stop bits, depend on the controller type and the serial port selected. The default selection is 1.

Receive Flow Control

The **RxFlow** drop down menu selects the receiver flow control for the selected port. Valid selections depend on the protocol, controller type, and serial port. If there is only one valid value the control is disabled. If there is more than one possible value, the default selection is none. Click here, Supported_RxFlow to see the Receive Flow control supported for each controller type.

Transmit Flow Control

The **TxFlow** drop-down menu selects the transmitter flow control for the selected port. Valid selections depend on the protocol, controller type, and serial port. The default selection is none. Click here, Supported TxFlow to see the Transmit Flow control protocols ports supported for each controller type.

Port Type

The **PortType** drop down menu selects the type of serial port. Valid selections depend on the serial port and controller type as shown in the table below. The default selection is RS-232. The options are as follows:

- RS-232: for a regular RS-232 connection.
- RS-232 Dial-up modem: If an external dial-up modem is used on the RS-232 connection.
- RS-232 Collision Avoidance: RS-232 connection with collision avoidance based on the CD signal is available only when the DNP protocol type is selected on the serial port, and the serial port supports handshaking.

  When this flow control is enabled, the protocol uses the Carrier Detect (CD) signal provided by the serial port to detect if the communication medium is in use. If it is, it waits until the medium is free before transmitting.

  Prior to transmitting each Data Link (DL) frame, the controller will test the CD line. If it is active, a countdown equal to the DL timeout will be set and CD will be monitored every 100 ms throughout this countdown period. If the Data Link timeout is set to the minimum of 100 ms, the CD line will be tested once.

  If the CD line reports inactive (line not in use), a frame will be transmitted immediately, and a new DL timeout is started as normal. On the other hand, if CD remains active during the DL timeout, the transmission attempt will not work. If a non-zero retry is configured in the Data Link layer, the test will be repeated until the number of retries has been exhausted.

  RS-232 Collision Avoidance is supported only on serial ports that support handshaking and whose protocol type is set for DNP.
- RS-485: for a regular RS-485 connection.

Store and Forward

The **Store and Forward** drop down menu selects whether store and forward messaging, or DNP Routing is enabled for the port. Valid selections are enabled and disabled. If this option is enabled, messages will be forwarded according to the settings in the store and forward routing table. The default selection is disabled.

This control is disabled when PPP protocol is selected for a serial port, or if any of the DF1 protocols are selected and for com 1 on the SCADAPack 4202 or 4203 of controllers.

The **Store and Forward** menu selection changes to **Routing** menu selection when DNP selected for a serial port. Valid selections are enabled and disabled. Routing needs to be enabled on a serial port to enable routing of DNP messages.

Enron Modbus

The **EnronModbus** drop down menu selects whether Enron Modbus is Enabled or Disabled for the port. If this option is enabled, the controller, in addition to regular Modbus messages, will handle Enron Modbus messages.

The Enable selection is only available if the serial port protocol is set to Modbus RTU or Modbus ASCII. The Enable selection is not available for any other protocols. Selects whether Enron Modbus is enabled for the port. If this option is enabled, the controller, in addition to regular Modbus messages, will handle Enron Modbus messages. Valid selections depend on the protocol. This control is disabled when PPP protocol is selected for a serial port and for com1 on the 4202 controllers.

Enron Station

The **EnronStation** entry selects the Enron Modbus station address for the serial port. Valid entries depend on the protocol. The Enron station needs to be different from the Modbus station set in the **Station** control. This allows Enron Modbus and Modbus communication can occur on the same port. This entry is greyed out if Enron Modbus is not enabled.

- When the Address selection is set to Standard the valid addresses are 1 to 255. The default value is 2.
- When the Address selection is set to Extended the valid addresses are 1 to 65534. The default value is 2.

Selects the Enron Modbus station address for the serial port. Valid entries depend on the protocol. The Enron station must be different from the Modbus station set in the Station control. This ensures Enron Modbus and Modbus communication can occur on the same port. This entry is greyed out if Enron Modbus is not enabled.

Reset Port To Default

The **Reset Port to Default** command is used to reset the serial port configuration to the default settings.

To Reset Selected Port:

- Right click the mouse anywhere on the **Port Properties** grid and then click on the Reset Selected Port command.

Supported Serial Ports

Com1 is selectable for the following controllers:

Micro16	SCADAPack	SCADAPack Plus
SCADAPack Light	SCADAPack LP	SCADAPack 100
SCADAPack 32	SCADAPack 32P	SCADAPack 314
SCADAPack 330	SCADAPack 334	SCADAPack 350
SCADAPack 4202 DR	SCADAPack 4202 DS	SCADAPack 4203 DR
SCADAPack 4203 DS	FlowStation 110	

Com2 is selectable for the following controllers:

Micro16	SCADAPack	SCADAPack Plus
SCADAPack Light	SCADAPack LP	SCADAPack 100
SCADAPack 32	SCADAPack 32P	SCADAPack 314
SCADAPack 330	SCADAPack 334	SCADAPack 350

SCADAPack 4202 DR	SCADAPack 4202 DS	SCADAPack 4203 DR
SCADAPack 4203 DS	FlowStation 110	

Com3 is selectable for the following controllers:

SCADAPack	SCADAPack Plus	SCADAPack LP
SCADAPack 32	SCADAPack 330	SCADAPack 334
SCADAPack 350	SCADAPack 4202 DR	SCADAPack 4202 DS
SCADAPack 4203 DR	SCADAPack 4203 DS	

Com4 is selectable for the following controllers:

SCADAPack Plus	SCADAPack Light	SCADAPack 32
SCADAPack 32P		

Supported Baud Rates

Controller	Communication Port Supported Baud Rates
Micro16 **Com1, Com2**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400
SCADAPack **Com1, Com2** **Com3**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack Plus **Com1, Com2** **Com3, Com4**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack Light **Com1, Com2** **Com4**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack LP **Com1, Com2** **Com3**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack 100 **Com1, Com2**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400
SCADAPack 32 **Com1, Com2, Com4** **Com3**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack 32P **Com1, Com2, Com4**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack 4202 DR **Com1** **Com2** **Com3**	4800 300, 600, 1200, 2400, 4800, 9600, 19200, 38400 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack 4202 DS **Com1** **Com2** **Com3**	4800 300, 600, 1200, 2400, 4800, 9600, 19200, 38400 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack 4203 DR **Com1** **Com2** **Com3**	4800 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack 4203 DS **Com1** **Com2** **Com3**	4800 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack 314 **Com1 and Com2**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack 330 **Com1, Com2, Com3**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack 334 **Com1, Com2, Com3**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
SCADAPack 350 **Com1, Com2, Com3**	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200

FlowStation 110 Com1 and Com2	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200

Supported Duplex

The **Duplex** setting selects full or half-duplex operation for the selected Port. Valid and default duplex settings depend on the serial port and controller type as shown in the table below.

Full duplex systems allow data to be sent and received from both ends of the system simultaneously.

Half duplex systems allow communication between a receiver and a transmitter in both directions, but in only one direction at a time. Half duplex systems need to use a method of signalling between each other that ensures the flow of data is being sent from one end only and that the other end is ready to receive the data. This signaling is referred to as flow control. Flow control may be software flow control such as XON/XOFF or hardware flow control.

When a SCADAPack controller is communicating with a half duplex device, such as a 5902 modem, hardware handshaking is used between the two devices. Hardware handshaking is a term used in describing serial communication between two serial devices using hardware data and flow control. These lines are generally defined as GND (ground), Rx (received data), Tx (transmit data), CTS (clear to send) and RTS (request to send). Each end of the communication path asserts its RTS line when it needs to send data and its CTS line when it is ready to receive data. For SCADAPack controllers the serial port needs to be set for half-duplex operation to communicate with half duplex devices.

Controller	Com1	Com2	Com3	Com4
Micro16	Full or Half	Full or Half	N/A	N/A
SCADAPack	Full or Half	Full or Half	Half	N/A
SCADAPack Plus	Full or Half	Full or Half	Half	Half
SCADAPack Light	Full or Half	Full or Half	N/A	Half
SCADAPack LP	Half	Full or Half	Half	N/A
SCADAPack 100	Full or Half	Full or Half	Half	N/A
SCADAPack 32	Full or Half	Full or Half	Half	Full or Half
SCADAPack 32P	Full or Half	Full or Half	N/A	Full or Half
SCADAPack 4202 DR	Full	Half	Half	N/A
SCADAPack 4202 DS	Full	Half	Half	N/A
SCADAPack 4203 DR	Half	Full or Half	Full or Half	N/A
SCADAPack 4203 DS	Half	Full or Half	Full or Half	N/A
SCADAPack 314	Full or Half	Full or Half	N/A	N/A
SCADAPack 330	Full or Half	Full or Half	Full or Half	N/A
SCADAPack 334	Full or Half	Full or Half	Full or Half	N/A
SCADAPack 350	Half	Full or Half	Full or Half	N/A
FlowStation 110	Full or Half	Full or Half	N/A	N/A

The duplex is forced to half if an MVT transmitter is configured on the port.

Supported Parity

The **Parity** drop down menu selects the parity for the selected port. Valid selections depend on the serial port, controller type and data bits, as shown in the table below.

Controller	Parity Selections
Micro16	Com1: none, even, odd Com2: none, even, odd
SCADAPack	Com1: none, even, odd Com2: none, even, odd Com3: (7-bits): even, odd, space, mark Com3: (8-bits): none, even, odd, mark
SCADAPack Plus	Com1: none, even, odd Com2: none, even, odd Com3: (7-bits): even, odd, space, mark Com3: (8-bits): none, even, odd, mark Com4: (7-bits): even, odd, space, mark Com4: (8-bits): none, even, odd, mark
SCADAPack Light	Com1: none, even, odd Com2: none, even, odd Com4: (7-bits): even, odd, space, mark Com4: (8-bits): none, even, odd, mark
SCADAPack LP	Com1: none, even, odd Com2: none, even, odd Com3: (7-bits): even, odd, space, mark Com3: (8-bits): none, even, odd, mark
SCADAPack 100	Com1: none, even, odd Com2: none, even, odd
SCADAPack 32	Com1: none, even, odd Com2: none, even, odd

	Com3: (7-bits): even, odd, space, mark Com3: (8-bits): none, even, odd, mark Com4: none, even, odd
SCADAPack 32P	Com1: none, even, odd Com2: none, even, odd Com4: none, even, odd
SCADAPack 4202 DR	Com1: none Com2: none, even, odd Com3: none, even, odd, mark
SCADAPack 4202 DS	Com1: none Com2: none, even, odd Com3: none, even, odd, mark
SCADAPack 4203 DR	Com1: none Com2: none, even, odd Com3: none, even, odd, mark
SCADAPack 4203 DS	Com1: none Com2: none, even, odd Com3: none, even, odd, mark
SCADAPack 314	Com1: none, even, odd Com2: none, even, odd
SCADAPack 330	Com1: none, even, odd Com2: none, even, odd Com3: none, even, odd
SCADAPack 334	Com1: none, even, odd Com2: none, even, odd Com3: none, even, odd
SCADAPack 350	Com1: none, even, odd Com2: none, even, odd Com3: none, even, odd
FlowStation 110	Com1: none, even, odd Com2: none, even, odd

The default selection is none.

Supported PortType

The **PortType** drop down menu selects the type of serial port. Valid selections depend on the serial port and controller type as shown in the table below. The default selection is RS-232.

Controller		**Port Type Selections**
Micro16		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance RS-485
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
SCADAPack		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance RS-485
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com3	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
SCADAPack Plus		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance RS-485
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com3	RS-232 RS-232 Dial Up Modem
	Com4	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance

SCADAPack Light		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance RS-485
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com4	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
SCADAPack LP		
	Com1	RS-485
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com3	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
SCADAPack 100		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 port type applies for RS-485 or RS-232 operation.
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
SCADAPack 32		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance RS-232 port type applies for RS-485. Use Jumper J9 on controller to select 485 operation.
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com3	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com4	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
SCADAPack 32P		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance RS-232 port type applies for RS-485. Use Jumper J9 on controller to select 485 operation.
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com4	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
SCADAPack 4202 DR		
	Com2	RS-232 RS-232 Dial Up Modem
	Com3	RS-232
SCADAPack 4202 DS		
	Com2	RS-232 RS-232 Dial Up Modem
	Com3	RS-232
SCADAPack 4203 DR		
	Com2	RS-232 RS-232 Dial Up Modem

	Com3	RS-232
SCADAPack 4203 DS		
	Com2	RS-232 RS-232 Dial Up Modem
	Com3	RS-232
SCADAPack 314		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
SCADAPack 330		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com3	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
SCADAPack 334		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com3	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
SCADAPack 350		
	Com1	RS-485
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com3	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
FlowStation 110		
	Com1	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance
	Com2	RS-232 RS-232 Dial Up Modem RS-232 Collision Avoidance

Supported Protocols

All controllers support the following serial protocols.

Protocol	Description
None	No protocol is used to send and receive unformatted strings through the serial port. This is typically used with C/C++ applications.
Modbus RTU	The Modbus RTU protocol sends each 8-bit byte contains two 4-bit Hexadecimal characters. The advantage of Modbus RTU is better data throughput.
Modbus ASCII	The Modbus ASCII protocol sends each 8-bit byte as two ASCII characters. The advantage of Modbus ASCII is that time intervals of up to one second are allowed without causing an error.
DF1 Full Duplex BCC	The DF1 Full-Duplex BCC protocol is compatible with the DF1 Full-Duplex data link layer protocol with block check character (BCC) error checking.
DF1 Full Duplex CRC	The TeleBUS DF1 Full-Duplex CRC protocol is compatible with the DF1 Full-Duplex data link layer protocol with 16-bit cyclic redundancy check (CRC-16) error checking. The duplex is forced to half if an MVT transmitter is configured on the port.
DF1 Half Duplex BBC	The TeleBUS DF1 Half-Duplex BCC protocol is compatible with the DF1 Half-Duplex data link layer protocol with block check character (BCC) error checking.
DF1 Half Duplex CRC	The TeleBUS DF1 Half-Duplex CRC protocol is compatible with the DF1 Half-Duplex data link layer protocol with 16-bit cyclic redundancy check (CRC-16) error checking.
DNP	The Distributed Network Protocol (DNP) is compatible with DNP-3 serial communication.

Notes

1. For SCADAPack 100 controllers with firmware older than version 1.80 DF1 protocols are not supported.
2. The SCADAPack 4202 and 4203 controllers support Sensor protocol only on com1.
3. The default protocol selection for all serial ports is Modbus RTU with the exception of SCADAPack controllers as noted above.

Supported Protocols by Controller Type

Controller Type	Valid Protocols	Default Protocol
Micro16 **SCADAPack** **SCADAPack Plus** **SCADAPack Light** **SCADAPack LP** **SCADAPack 100**	None Modbus RTU Modbus ASCII DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC DNP For SCADAPack 100 controllers with firmware older than version 1.80 DF1 protocols are not supported.	Modbus RTU
SCADAPack 32 **SCADAPack 32P**	None Modbus RTU Modbus ASCII DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC DNP PPP	Modbus RTU
SCADAPack family of programmable controllers (4202 DR, 3202 DS, 4203 DR and 4203 DS)	Com 1 fixed as Sensor. Com 2 and Com 3: None Modbus RTU Modbus ASCII DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC DNP	Com1 fixed as Sensor. Com2 and Com3 default is Modbus RTU.
SCADAPack 314/330/334 **SCADAPack 350**	None Modbus RTU Modbus ASCII DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC	Modbus RTU
FlowStation 110	Modbus RTU	Modbus RTU

Supported RxFlow

The **RxFlow** drop down menu selects the receiver flow control for the selected port. Valid selections depend on the protocol, controller type, and serial port, as shown in the tables below. If there is more than one possible value, the default selection is none.

Receive flow control is performed at the receiver end of a communication system to prevent the receive buffer from over filling. An over flowed receive buffer will result in all data not being processed by the receiver and thus the communication will not work as expected. Depending on the type of SCADAPack controller the selections for flow control are XON/XOFF, ModbusRTU, IgnoreCTS or None.

XON / XOFF

XON / XOFF flow control is often referred to as software flow control.This method uses three wire serial cables (RX, TX and GND lines) and the ASCII control characters XON and XOFF. The receiver sends an XON character when it is ready to receive data and sends an XOFF character when needs the transmitter to stop sending data, such as when the receive buffer is getting full.

ModbusRTU

ModbusRTU flow control is another type of software flow control and is used with the Modbus RTU protocol. This method uses the quiet time (no incoming data) at the end of a Modbus RTU message to detect the end of a message. This quiet time is 3.5 character times and thus changes depending on the baud rate used for communications. When the receiver detects the end of a message it

IgnoreCTS

Hardware flow control uses the RTS (Request to Send) and the CTS (Clear to Send) control lines on the serial cable.Each end of the communication path assert the RTS line when it needs to send data and the CTS line when it is ready to receive data. IgnoreCTS flow control is used with hardware flow control devices that do not support CTS (clear to send). Data is sent regardless of the status of the CTS line.

None

The None selection results in need for the application running on the SCADAPack controller or the protocol used to provide flow control. This setting is typically used with most protocols when they send a message and then wait for a response.

Protocol Type: None

The table below shows the valid selections when you select None as your protocol type.

Controller	Receive Flow Control Selections
Micro16	Com1: None, Xon/Xoff Com2: None, Xon/Xoff
SCADAPack	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: none
SCADAPack Plus	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None Com4: None
SCADAPack Light	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None
SCADAPack LP	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None
SCADAPack 100	Com1: None, Xon/Xoff Com2: None, Xon/Xoff
SCADAPack 32	Com1: None, Modbus RTU Com2: None, Modbus RTU Com3: None, Modbus RTU Com4: None, Modbus RTU
SCADAPack 32P	Com1: None, Modbus RTU Com2: None, Modbus RTU Com4: None, Modbus RTU
SCADAPack 4202 DR	Com1: None Com2: None Com3: None, Modbus RTU
SCADAPack 4202 DS	Com1: None Com2: None Com3: None, Modbus RTU
SCADAPack 4203 DR	Com1: None Com2: None Com3: None
SCADAPack 4203 DS	Com1: None Com2: None Com3: None
SCADAPack 314	Com1: None Com2: None
SCADAPack 330	Com1: None Com2: None Com3: None
SCADAPack 334	Com1: None Com2: None Com3: None
SCADAPack 350	Com1: None Com2: None Com3: None

Protocol Type: Modbus RTU

The table below shows the valid selections when you select Modbus RTU as your protocol type.

Controller	Receive Flow Control Selections
Micro16	Com1: None Com2: None
SCADAPack	Com1: None Com2: None Com3: Modbus RTU
SCADAPack Plus	Com1: None Com2: None Com3: Modbus RTU Com4: Modbus RTU
SCADAPack Light	Com1: None Com2: None Com4: Modbus RTU
SCADAPack LP	Com1: None Com2: None Com3: Modbus RTU
SCADAPack 100	Com1: None Com2: None
SCADAPack 32	Com1: Modbus RTU Com2: Modbus RTU Com3: Modbus RTU

	Com4: Modbus RTU
SCADAPack 32P	Com1: Modbus RTU Com2: Modbus RTU Com4: Modbus RTU
SCADAPack 4202 DR	Com1: None Com2: None Com3: Modbus RTU
SCADAPack 4202 DS	Com1: None Com2: None Com3: Modbus RTU
SCADAPack 4203 DR	Com1: None Com2: Modbus RTU Com3: Modbus RTU
SCADAPack 4203 DS	Com1: None Com2: Modbus RTU Com3: Modbus RTU
SCADAPack 314	Com1: Modbus RTU Com2: Modbus RTU
SCADAPack 330	Com1: Modbus RTU Com2: Modbus RTU Com3: Modbus RTU
SCADAPack 334	Com1: Modbus RTU Com2: Modbus RTU Com3: Modbus RTU
SCADAPack 350	Com1: Modbus RTU Com2: Modbus RTU Com3: Modbus RTU
FlowStation 110	Com1: Modbus RTU Com2: Modbus RTU

Protocol Type: Modbus ASCII

The table below shows the valid selections when you select Modbus ASCII as your protocol type.

Controller	Receive Flow Control Selections
Micro16	Com1: None, Xon/Xoff Com2: None, Xon/Xoff
SCADAPack	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: none
SCADAPack Plus	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None Com4: None
SCADAPack Light	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com4: None
SCADAPack LP	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None
SCADAPack 100	Com1: None, Xon/Xoff Com2: None, Xon/Xoff
SCADAPack 32	Com1: None, Modbus RTU Com2: None, Modbus RTU Com3: None, Modbus RTU Com4: None, Modbus RTU
SCADAPack 32P	Com1: None, Modbus RTU Com2: None, Modbus RTU Com4: None, Modbus RTU
SCADAPack 4202 DR	Com1: None Com2: None Com3: None, Modbus RTU
SCADAPack 4202 DS	Com1: None Com2: None Com3: None, Modbus RTU
SCADAPack 4203 DR	Com1: None Com2: None Com3: None
SCADAPack 4203 DS	Com1: None Com2: None Com3: None
SCADAPack 314	Com1: None Com2: None

SCADAPack 330	Com1: None Com2: None Com3: None
SCADAPack 334	Com1: None Com2: None Com3: None
SCADAPack 350	Com1: None Com2: None Com3: None

DF1 Protocols

The table below shows the valid selections when you select DF1 protocols as your protocol type.

Controller	Receive Flow Control Selections
Micro16	Com1: None Com2: None
SCADAPack	Com1: None Com2: None Com3: none
SCADAPack Plus	Com1: None Com2: None Com3: None Com4: None
SCADAPack Light	Com1: None Com2: None Com4: None
SCADAPack LP	Com1: None Com2: None Com3: None
SCADAPack 100	Com1: None Com2: None
SCADAPack 32	Com1: None Com2: None Com3: None Com4: None
SCADAPack 32P	Com1: None Com2: None Com4: None
SCADAPack 4202 DR	Com1: None Com2: None Com3: None
SCADAPack 4202 DS	Com1: None Com2: None Com3: None, Ignore CTS
SCADAPack 4203 DR	Com1: None Com2: None Com3: None
SCADAPack 4203 DS	Com1: None Com2: None Com3: None
SCADAPack 314	Com1: None Com2: None
SCADAPack 330	Com1: None Com2: None Com3: None
SCADAPack 334	Com1: None Com2: None Com3: None
SCADAPack 350	Com1: None Com2: None Com3: None

Protocol Type: DNP

The table below shows the valid selections when you select DNP as your protocol type.

Controller	Receive Flow Control Selections
Micro16	Com1: None Com2: None
SCADAPack	Com1: None Com2: None Com3: none
SCADAPack Plus	Com1: None Com2: None Com3: None

	Com4: None
SCADAPack Light	Com1: None Com2: None Com4: None
SCADAPack LP	Com1: None Com2: None Com3: None
SCADAPack 100	Com1: None Com2: None
SCADAPack 32	Com1: None Com2: None Com3: None Com4: None
SCADAPack 32P	Com1: None Com2: None Com4: None
SCADAPack 4202 DR	Com1: None Com2: None Com3: None
SCADAPack 4202 DS	Com1: None Com2: None Com3: None, Ignore CTS
SCADAPack 4203 DR	Com1: None Com2: None Com3: None
SCADAPack 4203 DS	Com1: None Com2: None Com3: None
SCADAPack 330	Com1: None Com2: None
SCADAPack 330	Com1: None Com2: None Com3: None
SCADAPack 334	Com1: None Com2: None Com3: None
SCADAPack 350	Com1: None Com2: None Com3: None

Supported TxFlow

The **TxFlow** drop-down menu selects the transmitter flow control for the selected port. Valid selections depend on the protocol, controller type, and serial port, as shown in the table below.

Transmit flow control is performed at the transmitter end of a communication system. to prevent the receive buffer from over filling. an over flowed receive buffer will result in all data not being processed by the receiver.

XON/XOFF

XON / XOFF flow control is often referred to as software flow control.This method uses the ASCII control characters XON and XOFF. The receiver sends an XON character when it is ready to receive data and sends an XOFF character when needs the transmitter to stop sending data, such as when the receive buffer is getting full.

IgnoreCTS

Hardware flow control uses the RTS (Request to Send) and the CTS (Clear to Send) control lines on the serial cable.Each end of the communication path assert the RTS line when it needs to send data and the CTS line when it is ready to receive data. IgnoreCTS flow control is used with hardware flow control devices that do not support CTS (clear to send). Data is sent regardless of the status of the CTS line.

None

The None selection results in need for the application running on the SCADAPack controller or the protocol used to provide flow control. This setting is typically used with most protocols when they send a message and then wait for a response.

Protocol Type: None

The table below shows the valid selections when you select None as your protocol type.

Controller	Receive Flow Control Selections
Micro16	Com1: None, Xon/Xoff Com2: None, Xon/Xoff
SCADAPack	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None, Ignore CTS
SCADAPack Plus	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None, Ignore CTS Com4: None, Ignore CTS
SCADAPack Light	Com1: None, Xon/Xoff

	Com2: None, Xon/Xoff Com4: None, Ignore CTS
SCADAPack LP	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None, Ignore CTS
SCADAPack 100	Com1: None, Xon/Xoff Com2: None, Xon/Xoff
SCADAPack 32	Com1: None, Ignore CTS Com2: None, Ignore CTS Com3: None, Ignore CTS Com4: None, Ignore CTS
SCADAPack 32P	Com1: None, Ignore CTS Com2: None, Ignore CTS Com4: None, Ignore CTS
SCADAPack 4202 DR	Com1: None Com2: None Com3: None, Ignore CTS
SCADAPack 4202 DS	Com1: None Com2: None Com3: None, Ignore CTS
SCADAPack 4203 DR	Com1: None Com2: None Com3: None
SCADAPack 4203 DS	Com1: None Com2: None Com3: None
SCADAPack 314	Com1: None Com2: None
SCADAPack 330	Com1: None Com2: None Com3: None
SCADAPack 334	Com1: None Com2: None Com3: None
SCADAPack 350	Com1: None Com2: None Com3: None

Protocol Type: Modbus RTU

The table below shows the valid selections when you select Modbus RTU as your protocol type.

Controller	Receive Flow Control Selections
Micro16	Com1: None Com2: None
SCADAPack	Com1: None Com2: None Com3: Ignore CTS
SCADAPack Plus	Com1: None Com2: None Com3: Ignore CTS Com4: Ignore CTS
SCADAPack Light	Com1: None Com2: None Com4: Ignore CTS
SCADAPack LP	Com1: None Com2: None Com3: Ignore CTS
SCADAPack 100	Com1: None Com2: None
SCADAPack 32	Com1: Ignore CTS Com2: Ignore CTS Com3: Ignore CTS Com4: Ignore CTS
SCADAPack 32P	Com1: Ignore CTS Com2: Ignore CTS Com4: Ignore CTS
SCADAPack 4202 DR	Com1: None Com2: None Com3: Ignore CTS
SCADAPack 4202 DS	Com1: None Com2: None Com3: Ignore CTS
SCADAPack 4203 DR	Com1: None

	Com2: None Com3: None
SCADAPack 4203 DS	Com1: None Com2: None Com3: None
SCADAPack 314	Com1: None Com2: None
SCADAPack 330	Com1: None Com2: None Com3: None
SCADAPack 334	Com1: None Com2: None Com3: None
SCADAPack 350	Com1: None Com2: None Com3: None
FlowStation 110	Com1: None Com2: None

Protocol Type: Modbus ASCII

The table below shows the valid selections when you select Modbus ASCII as your protocol type.

Controller	Receive Flow Control Selections
Micro16	Com1: None, Xon/Xoff Com2: None, Xon/Xoff
SCADAPack	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None, Ignore CTS
SCADAPack Plus	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None, Ignore CTS Com4: None, Ignore CTS
SCADAPack Light	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com4: None, Ignore CTS
SCADAPack LP	Com1: None, Xon/Xoff Com2: None, Xon/Xoff Com3: None, Ignore CTS
SCADAPack 100	Com1: None, Xon/Xoff Com2: None, Xon/Xoff
SCADAPack 32	Com1: None, Ignore CTS Com2: None, Ignore CTS Com3: None, Ignore CTS Com4: None, Ignore CTS
SCADAPack 32P	Com1: None, Ignore CTS Com2: None, Ignore CTS Com4: None, Ignore CTS
SCADAPack 4202 DR	Com1: None Com2: None Com3: None, Ignore CTS
SCADAPack 4202 DS	Com1: None Com2: None Com3: None, Ignore CTS
SCADAPack 4203 DR	Com1: None Com2: None Com3: None
SCADAPack 4203 DS	Com1: None Com2: None Com3: None
SCADAPack 314	Com1: None Com2: None
SCADAPack 330	Com1: None Com2: None Com3: None
SCADAPack 334	Com1: None Com2: None Com3: None
SCADAPack 350	Com1: None Com2: None Com3: None

Protocol Type: DF1 Protocols

The table below shows the valid selections when you select DF1 Protocols as your protocol type.

Controller	Receive Flow Control Selections
Micro16	Com1: None Com2: None
SCADAPack	Com1: None Com2: None Com3: Ignore CTS
SCADAPack Plus	Com1: None Com2: None Com3: Ignore CTS Com4: Ignore CTS
SCADAPack Light	Com1: None Com2: None Com4: Ignore CTS
SCADAPack LP	Com1: None Com2: None Com3: Ignore CTS
SCADAPack 100	Com1: None Com2: None
SCADAPack 32	Com1: Ignore CTS Com2: Ignore CTS Com3: Ignore CTS Com4: Ignore CTS
SCADAPack 32P	Com1: Ignore CTS Com2: Ignore CTS Com4: Ignore CTS
SCADAPack 4202 DR	Com1: None Com2: None Com3: Ignore CTS
SCADAPack 4202 DS	Com1: None Com2: None Com3: Ignore CTS
SCADAPack 4203 DR	Com1: None Com2: None Com3: None
SCADAPack 4203 DS	Com1: None Com2: None Com3: None
SCADAPack 314	Com1: None Com2: None
SCADAPack 330	Com1: None Com2: None Com3: None
SCADAPack 334	Com1: None Com2: None Com3: None
SCADAPack 350	Com1: None Com2: None Com3: None

Protocol Type: DNP

The table below shows the valid selections when you select DNP as your protocol type.

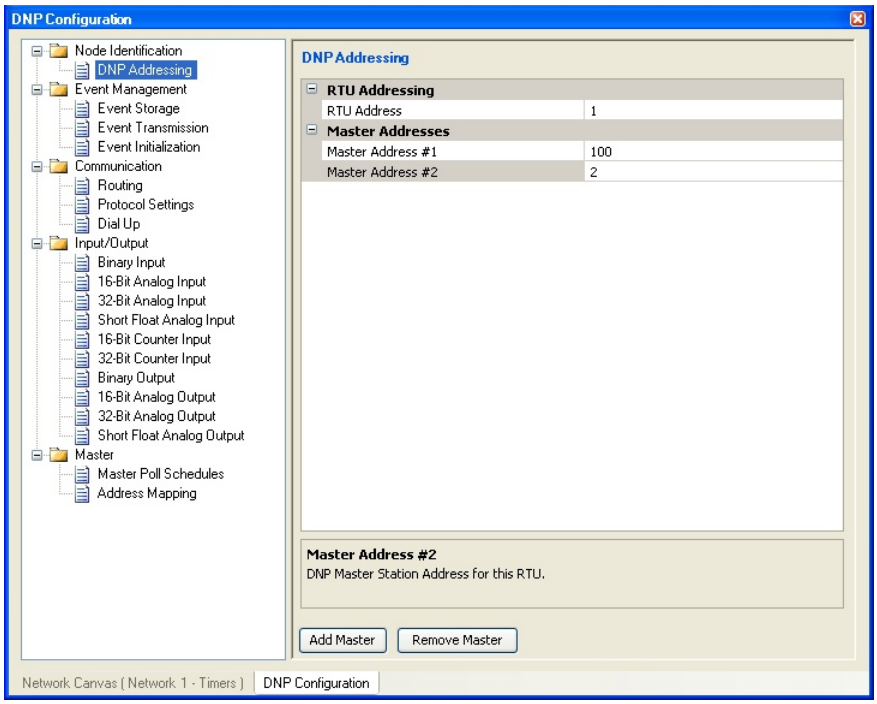
Controller	Receive Flow Control Selections
Micro16	Com1: None Com2: None
SCADAPack	Com1: None Com2: None Com3: Ignore CTS
SCADAPack Plus	Com1: None Com2: None Com3: Ignore CTS Com4: Ignore CTS
SCADAPack Light	Com1: None Com2: None Com4: Ignore CTS
SCADAPack LP	Com1: None Com2: None Com3: Ignore CTS
SCADAPack 100	Com1: None Com2: None
SCADAPack 32	Com1: Ignore CTS Com2: Ignore CTS Com3: Ignore CTS

	Com4: Ignore CTS
SCADAPack 32P	Com1: Ignore CTS Com2: Ignore CTS Com4: Ignore CTS
SCADAPack 4202 DR	Com1: None Com2: None Com3: Ignore CTS
SCADAPack 4202 DS	Com1: None Com2: None Com3: Ignore CTS
SCADAPack 4203 DR	Com1: None Com2: None Com3: None
SCADAPack 4203 DS	Com1: None Com2: None Com3: None
SCADAPack 314	Com1: None Com2: None
SCADAPack 330	Com1: None Com2: None Com3: None
SCADAPack 334	Com1: None Com2: None Com3: None
SCADAPack 350	Com1: None Com2: None Com3: None

DNP3

The **DNP Configuration** view is launched from the DNP Configuration command located in the Communication Group of the Configure Ribbon. The DNP Configuration command opens the DNP Configuration view. This view provides entries to fully configure the project controller as a DNP slave or, depending on the controller type, a DNP master.

For further explanation of DNP architecture and terminology used refer to the DNP3 Reference section of this manual.



The DNP Configuration view consists of three general panes when first opened.

- The **selection pane** is a tree view that allows for the selection of DNP configuration properties for a node. The tree view is divided into top level folders that contain related configuration properties. Top level folders are Node Identification, Event Management, Communication, Input/Output and Master. The folders may be collapsed and expanded using the tree control at the left of each top level folder.
- The configuration pane displays and allows the editing of the selected DNP configuration properties. The properties are displayed in a grid format.
- The information pane provides helpful information for the DNP configuration properties.

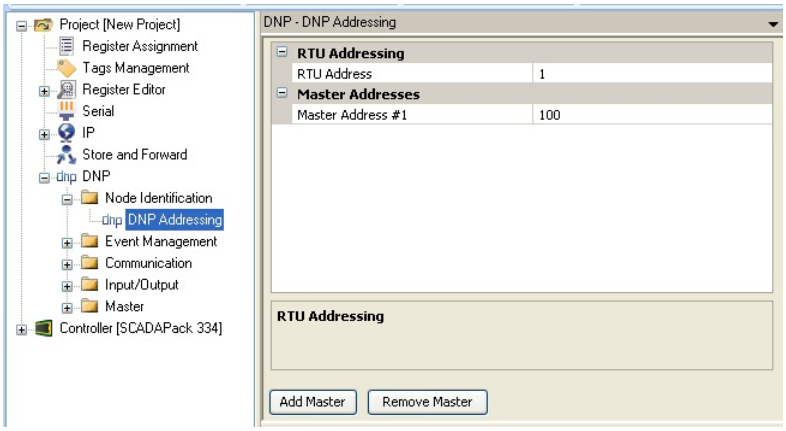
The view changes depending on the DNP configuration property selected in the selection pane.

Node Identification

The Node Identification folder contains the DNP Addressing leaf. When the DNP Addressing leaf is selected the DNP Addressing view is opened in the configuration pane.

DNP Addressing

The DNP Addressing View is used to specify the controller Node DNP address and the host DNP addresses that the controller Node will communicate with.



The **RTU Addressing** parameter specifies the address of this RTU. It is the source address used by this DNP driver when communicating with a master station. Each DNP station in a network needs to have a unique address, including the Master station. Valid entries for RTU Station Address are 0 to 65519.

The **Master Addresses** grid list contains a list of Master station addresses that the SCADAPack controller will respond to. The default list contains one master address of 100. This address may be edited, or changed, and up to 8 master stations may be added to the list. Valid entries for Master Station Addresses are 0 to 65519.

- When a master station polls for event data, the controller will respond with any events that have not yet been reported to that master station.
- When an unsolicited response becomes due, it will be sent to each configured master station in turn. A complete unsolicited response message transaction, including retries, will be sent to the first configured master station. When this transaction has finished, a complete unsolicited response message transaction including retries will be sent to the next configured master station, and so on for all the configured master stations.
- Change events will be retained in the event buffer until they have been successfully reported to all configured master stations.

Select the **Add Master** button to enter a new address to the Master Station Address list. The new master address is a default value and can be edited by clicking in the master address grid and typing a new master address. Up to 8 entries can be added to the table. An error message is displayed if the table is full.

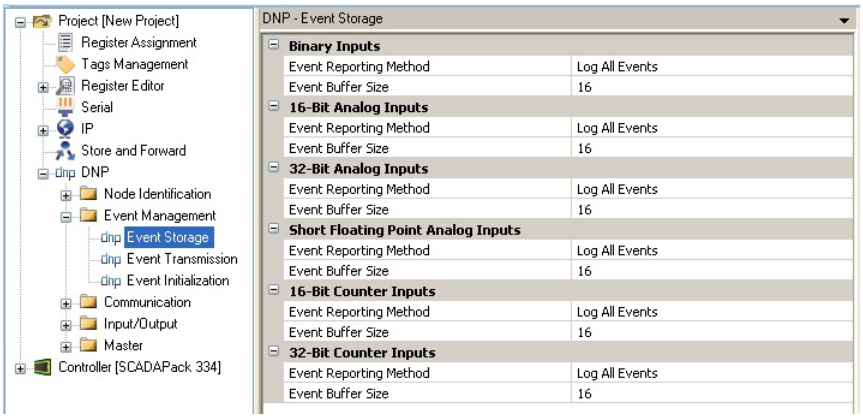
Select the **Remove Master** button to remove any selected Master Addresses from the grid list.

Event Management

The Event Management folder contains the Event Storage, Event Transmission and Event Initialization leaves.

Event Storage

The Event Storage selection specifies how input events are stored in the controller. When the Event Storage leaf is selected the Event Storage view is opened in the configuration pane.The memory available for DNP configuration in the controller varies according to other features and C++ applications in the controller. It cannot be predicted in advance. The configuration needs to be written to the controller which will determine if the configuration fits.



Binary Inputs

The **Event Reporting Method** selection specifies how binary input events are stored. The selections are:

- A **Change of State** event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data.
- A Log All Events is an event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. This is the default setting.

The **Event Buffer Size** parameter specifies the maximum number of binary input change events buffered by the RTU. The buffer holds all binary input change events,

regardless of the class to which they are assigned.

If the buffer is completely full the RTU will overwrite the oldest events with the newest; the **Event Buffer Overflowed** IIN flag will also be set to indicate that the buffer has overflowed.

The Event Buffer size should be at least equivalent to the number of binary inputs defined as Change of State type. This will allow all binary inputs to change simultaneously without losing any events. The value of this parameter depends on how often binary input change events occur and the rate at which the events are reported to the master station.

The valid values for this parameter are 0 - 65535. Default value is 16.

16-Bit Analog Inputs

The **Event Reporting Method** selection specifies how 16-Bit Analog input events are reported. The selections are:

- A **Change of State** event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data.
- A Log All Events is an event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. This is the default setting.

The **Event Buffer Size** parameter specifies the maximum number of 16-Bit analog input change events buffered by the RTU. The buffer holds all 16-Bit analog input change events, regardless of the class to which they are assigned.

If the buffer is completely full the RTU will overwrite the oldest events with the newest; the **Event Buffer Overflowed** IIN flag will also be set to indicate that the buffer has overflowed.

The Event Buffer size should be at least equivalent to the number of 16-Bit analog inputs defined as Change of State type. This will allow all 16-Bit analog inputs to change simultaneously without losing any events. The value of this parameter depends on how often 16-Bit analog input change events occur and the rate at which the events are reported to the master station.

The valid values for this parameter are 0 - 65535. Default value is 16.

32-Bit Analog Inputs

The **Event Reporting Method** selection specifies how 32-Bit Analog input events are reported. The selections are:

- A Change of State event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data.
- A Log **All Events** is an event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. This is the default setting.

The **Event Buffer Size** parameter specifies the maximum number of 32-Bit analog input change events buffered by the RTU. The buffer holds all 32-Bit analog input change events, regardless of the class to which they are assigned.

If the buffer is completely full the RTU will overwrite the oldest events with the newest; the **Event Buffer Overflowed** IIN flag will also be set to indicate that the buffer has overflowed.

The Event Buffer size should be at least equivalent to the number of 32-Bit analog inputs defined as Change of State type. This will allow all 32-Bit analog inputs to change simultaneously without losing any events. The value of this parameter depends on how often 32-Bit analog input change events occur and the rate at which the events are reported to the master station.

The valid values for this parameter are 0 - 65535. Default value is 16.

Short Floating Point Analog Inputs

The **Event Reporting Method** selection specifies how Short Floating Point Analog input events are reported. The selections are:

- A **Change of State** event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data.
- A Log All Events is an event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. This is the default setting.

The **Event Buffer Size** parameter specifies the maximum number of short floating point analog input change events buffered by the RTU. The buffer holds all short floating Point analog input change events, regardless of the class to which they are assigned.

If the buffer is completely full the RTU will overwrite the oldest events with the newest; the **Event Buffer Overflowed** IIN flag will also be set to indicate that the buffer has overflowed.

The Event Buffer size should be at least equivalent to the number of short floating Point analog inputs defined as Change of State type. This will allow all short floating Point analog inputs to change simultaneously without losing any events. The value of this parameter depends on how often analog input change events occur and the rate at which the events are reported to the master station.

The valid values for this parameter are 0 - 65535. Default value is 16.

16-Bit Counter Inputs

The **Event Reporting Method** selection specifies how 16-Bit Counter input events are reported. The selections are:

- A **Change of State** event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data.
- A Log All Events is an event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. This is the default setting.

The **Event Buffer Size** parameter specifies the maximum number of 16-Bit counter input change events buffered by the RTU. The buffer holds all 16-Bit counter input change events, regardless of the class to which they are assigned.

If the buffer is completely full the RTU will overwrite the oldest events with the newest; the **Event Buffer Overflowed** IIN flag will also be set to indicate that the buffer has overflowed.

The Event Buffer size should be at least equivalent to the number of short floating Point analog inputs defined as Change of State type. This will allow all 16-Bit counter input inputs to change simultaneously without losing any events. The value of this parameter depends on how often 16-Bit counter input change events occur and the rate at which the events are reported to the master station.

The valid values for this parameter are 0 - 65535. Default value is 16.

32-Bit Counter Inputs

The Event Reporting Method selection specifies how 32-Bit Counter input events are reported. The selections are:

- A **Change of State** event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data.
- A **Log All Events** is an event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. This is the default setting.

The **Event Buffer Size** parameter specifies the maximum number of 16-Bit counter input change events buffered by the RTU. The buffer holds all 16-Bit counter input change events, regardless of the class to which they are assigned.

If the buffer is completely full the RTU will overwrite the oldest events with the newest; the **Event Buffer Overflowed** IIN flag will also be set to indicate that the buffer has overflowed.

The Event Buffer size should be at least equivalent to the number of short floating Point analog inputs defined as Change of State type. This will allow all 32-Bit counter input inputs to change simultaneously without losing any events. The value of this parameter depends on how often 32-Bit counter input change events occur and the rate at which the events are reported to the master station.

The valid values for this parameter are 0 - 65535. Default value is 16.

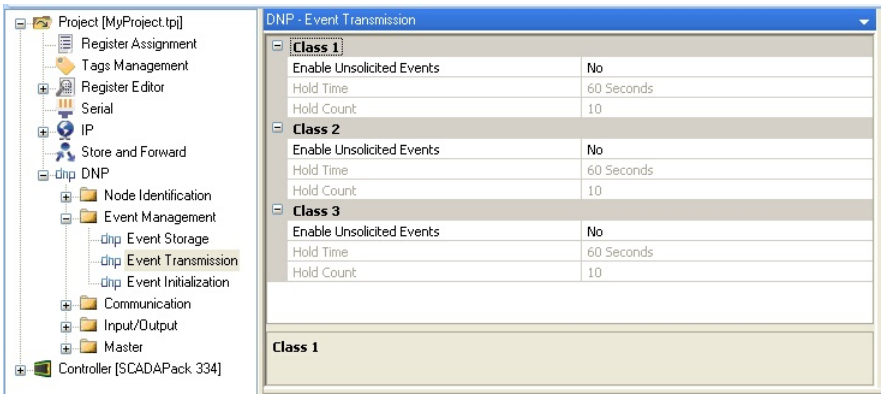
Event Processing

For SCADAPack 32 and SCADAPack 32P controllers input events are processed by the DNP driver at a rate of 100 events every 100 ms. If more than 100 counter input events need to be processed they are processed sequentially in blocks of 100 until all events are processed. This allows the processing of 1000 counter input events per second.

SCADAPack 314, SCADAPack 330/334, SCADAPack 350/357 and SCADAPack controllers attempt to process all DNP points every 100ms. If there are more DNP points than the controller can scan in 100ms the controller will respond by slowing the DNP scan rate. The DNP scan rate is backed off in multiples of 100ms until there is sufficient time to execute non-DNP functions (logic programs, C programs and communication). In the event the overload is transitory the DNP scan rate will return to the original scan rate of 100ms.

Event Transmission

The Event Transmission selection specifies how input events are reported to the DNP master station(s). When the Event Transmission leaf is selected the Event Transmission view is opened in the configuration pane.



The **Event Transmission** view contains parameters for Class 1, Class 2 and Class 3 events. The parameters are identical for each event class and are explained below.

The **Enable Unsolicited Events** parameter controls the generation of unsolicited events. If unsolicited responses are disabled for a Class the controller will not generate unsolicited events for that Class. If unsolicited responses are enabled the controller generates unsolicited events for that Class if its value or state exceeds a defined threshold. Valid selections are:

- Select **No** to disable unsolicited events.
- Select Yes to enable unsolicited events.

The default value is No.

The controller does not transmit collected unsolicited messages (or responses) to a master, even after the Hold Time or Hold Count conditions have been met, unless its 'Enable Unsolicited Responses on Start Up' field is set to 'Yes' or the master enables the trigger of unsolicited messages.

To configure a master to control unsolicited message transmission from a remote, see the Master Poll Schedules configuration pane.

The **Hold Time** parameter is used only when unsolicited responses are enabled for a Class. This parameter defines the maximum period (in seconds) the RTU will be allowed to hold its events before reporting them to the DNP master station. When the hold time has elapsed since the first event occurred, the RTU will report to the DNP master station all events accumulated up to then. This parameter is used in conjunction with the Hold Count parameter in customizing the unsolicited event reporting characteristics. The value used for the Hold Time depends on the frequency of event generation, topology and performance characteristics of the system. To send an unsolicited response as soon as an event occurs, set the Hold Time parameter to 0.

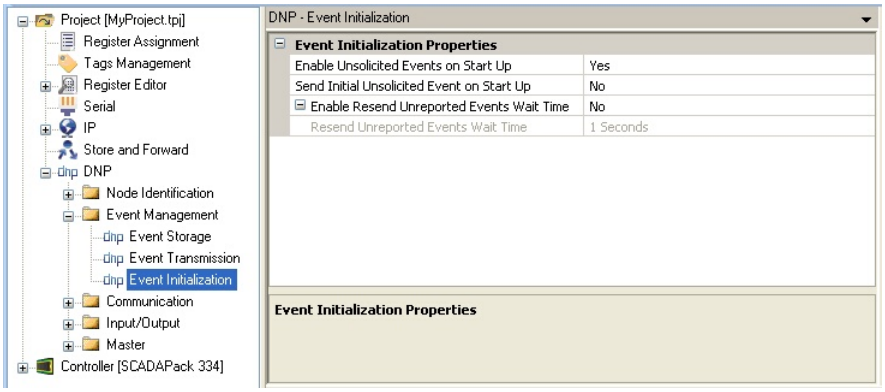
The valid values for this parameter are 0 - 65535. The default value is 60 seconds.

The **Hold Count** parameter is used only when unsolicited responses are enabled for a Class. This parameter defines the maximum number of events the RTU will be allowed to hold before reporting them to the DNP master station. When the hold count threshold is reached, the RTU will report to the master, all events accumulated up to that point. This parameter is used in conjunction with the Hold Time in customizing the unsolicited event reporting characteristics. To guarantee an unsolicited response is sent as soon as an event occurs, set the Hold Count parameter to 1.

The valid values for this parameter are 1 - 65535. The default value is 10.

Event Initialization

The Event Initialization selection specifies how unsolicited events are reported to the DNP master station(s). When the Event Initialization leaf is selected the Event Initialization view is opened in the configuration pane.



The **Enable Unsolicited Responses on Start Up** parameter enables or disables unsolicited responses on startup. This affects the default controller behaviour after a start-up or restart. Some hosts require devices to start up with unsolicited responses enabled. It should be noted this is non-conforming behaviour according to the DNP standard. Valid selections are:

- Yes
- No

The default selection is Yes.

The **Send Initial Unsolicited Response on Startup** parameter enables or disables Send Initial unsolicited responses on startup. This parameter controls whether an initial unsolicited response with null data is sent after a start-up or restart. Valid selections are:

- Yes
- No

The default selection is No.

The **Enable Resend Unreported Events Wait Time** parameter enables or disables the retransmission of events after all attempts to report the events have been unsuccessful. Many communications networks experience occasional communication loss. In such networks, even when message retries are used, there is a chance that some messages will be unsuccessful meaning there is a chance some unsolicited messages and change events will not be reported to the master station. The events remain in the outstation buffers until polled or additional events are generated.

To address this, and ensures delivery of high priority events, the **Enable Resend Unreported Events Wait Time** parameter is added to the DNP configuration. This parameter controls a timer for retrying the transmission of unsolicited messages.

- Select **No** to disable the Resend Unreported Events Wait Time. This is the default selection.
- Select **Yes** to enable the Resend Unreported Events Wait Time parameter.

The **Resend Unreported Events Wait Time** is the number of seconds to wait before resending unreported events. Whenever a DNP unsolicited message is unsuccessful, including all its retries, instead of just retiring the message and reporting it as a missed message, an unsolicited resend timer is initiated. After the configured time delay has passed, the unsolicited message will be sent again, including the configured retries. This process will be repeated continuously until the unsolicited message is successfully sent and acknowledged. In the case of multiple masters the unsolicited resend timer is uninitiated after the retries are expired for the last master in the polling list.

Valid values are 1 to 65535 seconds. The default value is 1 seconds. The control is unselected by default.

SCADAPack firmware 2.44 (and later), SCADAPack 32 Firmware 1.92 (and later), SCADAPack 350 or SCADAPack firmware 1.25 (and later) and SOLARPack 410 firmware 1.32 (and Later) support this feature.

If **Enable Resend Unreported Events Wait Time** is not selected, the controller will not resend unreported events after all attempts have been unsuccessful, until polled or until additional events are generated and their reporting threshold is reached.

The **Enable Resend Unreported Events Wait Time** control can be selected even when no classes are enabled. This allows the feature to be used in a mimic controller that is being used to pass outstation events to a host.

Communication

The communication folder contains the Routing, Protocol Settings and Dial Up leaves.

Routing

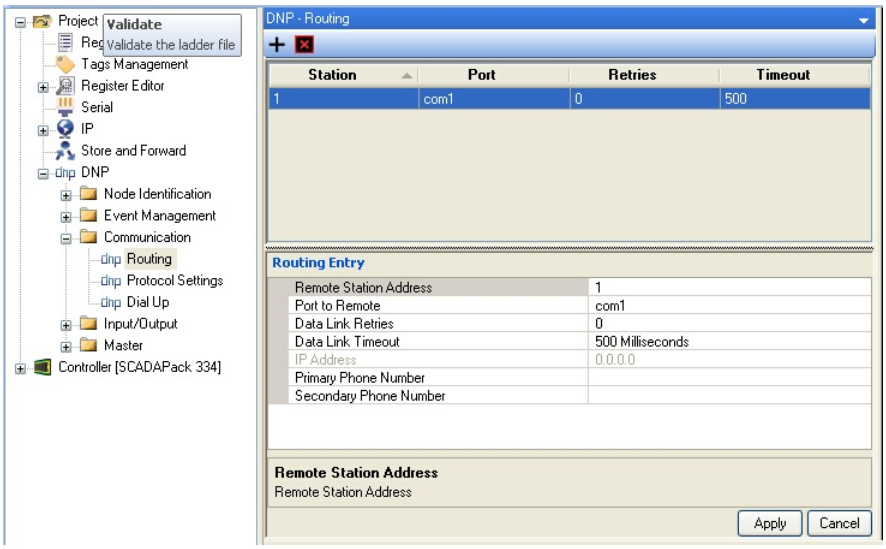
In a typical application the SCADAPack controller, configured for DNP, will act as a DNP slave station in a network. The SCADA system will communicate directly with all the DNP slave stations in the SCADA system.

DNP routing is a method for routing, or forwarding, of messages received from the SCADA system, through the SCADAPack controller, to a remote DNP slave station. The SCADAPack DNP slave station will respond to all messages sent to it from the SCADA system, as well as broadcast messages. When it receives a message that is not sent to it the message is sent on the serial port defined in the routing table.

The advantage of this routing ability is that the SCADA system can communicate directly with the SCADAPack controller and the SCADAPack controller can handle the communication to remote DNP slave stations.

The DNP Routing table displays each routing translation as a row, with column headings, in the table. The table will hold a maximum of 128 entries.

When the **Add** button is clicked the grid list for DNP Routing Entry is displayed with default values.



DNP Routing Entry

The DNP Routing Entry is displayed with default values when the **Add** button is clicked. Edit the default entries to meet the requirements of your application.

The **Remote Station Address** parameter is the address of the remote DNP station. Valid values for this field are from 0 to 65519. The default value is 1 when the view is first opened. The default value is incremented by 1 on each click of the Add button.

The **Port to Remote** parameter is the communications port, which should be used to communicate with the remote DNP station. This parameter is a drop down list containing the valid communications ports for the controller. For SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 and SCADAPack 32P controllers the list will contain DNP in TCP and DNP in UDP in addition to the available serial ports for the controller type. The default value is com1.

The **Data Link Retries** parameter is the maximum number of Data Link retries which should be used for this DNP station in the case of communication errors. This field overrides the Data Link Retries field set in the Protocol Settings view configuration. Valid values for this field are 0 to 255. The default value is 0.

The **Data Link Timeout** edit control displays the maximum time (in milliseconds) to wait for a Data Link response before retrying the message. This field overrides the Data Link Timeout field in the global DNP parameters in the Data Link Layer configuration. Valid values for this field are 100 to 60000, in multiples of 100. The default value is 500 Milliseconds.

The **IP Address** parameter is only enabled if the controller type is a SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P. Enter the IP address of the remote DNP station. The default value is 0.0.0.0.

The phone number parameters allow automatic dialing for stations that use dial-up ports. The Phone Number parameters are enabled only when the Port selected is a serial port.

The **Primary Phone Number** is the dialing string that will be used for the primary connection to the station. The controller will make 1 or more attempts, as configured in the Dial Up view to connect using this number. If this connection is unsuccessful then the Secondary Phone Number will be dialed, if it is entered.

Valid values are any ASCII string. The maximum length is 32 characters. Leave this blank if you are not using a dial-up connection. The default value is blank. The serial port type needs to be set to RS-232 Modem for dial-up operation.

The **Secondary Phone Number** is the dialing string that will be used for the secondary connection to the station. The controller will make 1 or more attempts, as configured in the Application layer, to connect using this number. This number is used after the primary connection is unsuccessful on all attempts.

Valid values are any ASCII string. The maximum length is 32 characters. Leave this blank if you are not using a dial-up connection. The default value is blank. The serial port type must be set to RS-232 Modem for dial-up operation.

- Click the **Apply** button to save the DNP Routing Entry in the DNP Configuration.
- Click the **Cancel** button to reapply the default values to the DNP Routing Entry grid.

Routing Table

The **Station** column displays the address of the remote DNP station.

The **Port** column displays the serial communications port, which should be used to communicate with this DNP station.

The **Retries** column displays the maximum number of Data Link retries, which should be used for this DNP station in the case of communication errors.

The **Timeout** column displays the maximum time (in milliseconds) to wait for a Data Link response before retrying the message.

The **IP Address** column displays the IP address of the remote DNP station.

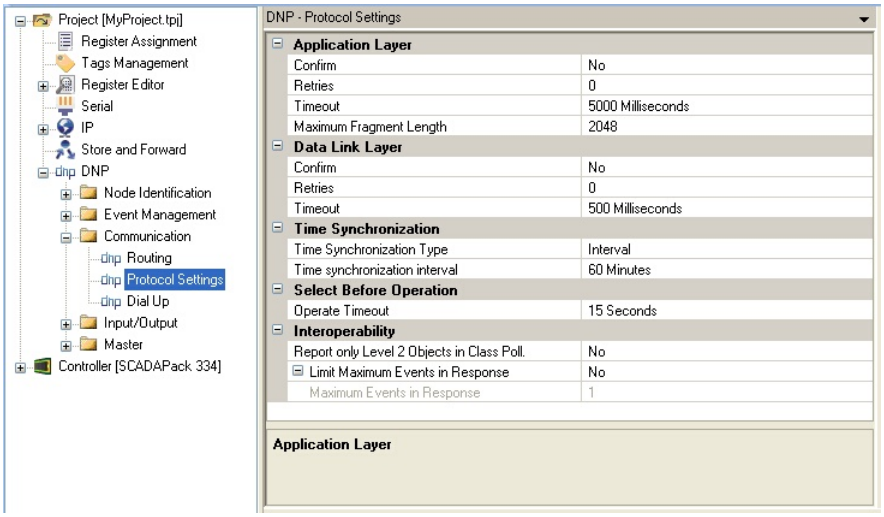
The **Primary Phone Number** is the dialling string that will be used for the primary connection to the station.

The **Secondary Phone Number** is the dialling string that will be used for the secondary connection to the station.

- Click the **Delete** button to delete a selected entry in the DNP Routing Entry table.

Protocol Settings

Protocol settings define how this node will communicate with other DNP devices.



Application Layer

The **Confirm** parameter specifies whether the SCADAPack controller requests a confirmation from the master station for any data transmitted. When it is disabled, the controller does not request a confirmation from the master station and assumes that the master receives the data it sends successfully. However if the data includes event data (including unsolicited messages), the controller requests a confirmation from the master regardless of whether this feature is enabled or disabled. Valid selections for this parameter are:

- Select **No** to have the controller not request a confirmation from the master station and assume that the master receives the data it sends successfully. This is the default setting.
- Select **Yes** to have the controller request a confirmation from the master station for any data transmitted.

The **Retries** parameter specifies the maximum number of times the application layer will retry sending a response or an unsolicited response to the master station. This does not include any retries performed by the data link layer. Valid values are between 0 and 255. The default value is 0.

The **Timeout** is the expected time duration (in milliseconds) that the master station's application layer requires to process and respond to a response from the SCADAPack controller. This SCADAPack controller uses this value in setting its timeout interval for master station responses. This value should be large enough to prevent response timeouts. The value needs to be kept small enough so as not to slow system throughput. The value of this element is dependent on the master station. Valid values are between 100 and 60000 milliseconds. The default value is 5000 Milliseconds.

The **Maximum Fragment Length** is maximum size of a single response fragment that the RTU will send. If the complete response message is too large to fit within a single fragment, then the SCADAPack controller will send the response in multiple fragments. Valid values are between 100 and 2048 bytes. The default value is 2048.

This parameter is adjustable to allow for interoperability with simple DNP3 devices that require smaller application layer fragments. Devices with limited memory may restrict the application layer fragment size to as low as 249 bytes.

The Maximum Fragment Length parameter applies to responses from read commands only. It does not affect unsolicited responses.

Using application layer Confirmation and Retries is inherently more efficient than using data link layer Confirmation and Retries. Each fragment sent by the Application layer may require as many as 10 data link layer frames to be sent, each with its own confirmation message. The application layer is typically preferred for message confirmation for this reason.

Data Link Layer

The **Data Link Confirmation** parameter specifies whether or not the RTU requests the underlying data link transmitting its response to use a high quality service, which generally means that the data link requires the receiving data link to confirm receipt of all messages.

- Select **No** to disable Data Link Confirmation.
- Select **Yes** to enable Data Link Confirmation.

The **Retries** parameter specifies the maximum number of times the data link layer will retry sending a message to the master station. This parameter is only used when responding to a request from a Master station, when there is no corresponding entry in the Routing dialog for that station. This is independent of the application layer retries. The valid values for this parameter are 0 - 255. Setting the value to 0 disables sending retries. The default value is 0.

Using data link layer Confirmation and Retries is inherently less efficient than application layer Confirmation and Retries. Each fragment sent by the Application layer may require as many as 10 data link layer frames to be sent, each with its own confirmation message. The data link layer is typically not used for message confirmation for this reason.

The **Timeout** parameter specifies the expected time duration that the master station's data link layer requires to process and respond to a message from the RTUs data link layer. It is used by the RTU in setting its timeout interval for master station responses. This value should be large enough to prevent response timeouts. The value needs to be kept small enough so as not to slow system throughput. The value of this element is dependent on the master station. It is expressed in milliseconds. Valid values are 10 to 60000 milliseconds. The default value is 500 milliseconds.

Time Synchronization

The **Time Synchronization Type** parameter defines when and how often the SCADAPack outstation prompts the master station to synchronize the SCADAPack controller time. Messages need to be sent between the Master and Remote stations for Time Synchronization to work. Valid selections for this parameter are:

- The **None** selection will cause the SCADAPack controller to not request Time Synchronization.
- The **At Start Up Only** selection will cause the SCADAPack controller to request Time Synchronization at startup only.
- The **Interval** selection will cause the SCADAPack controller to request Time Synchronization at startup and then every Time Synchronization Interval minutes after receiving a time synchronization from the master. Valid entries for Interval are between 1 and 32767 minutes. The default value is 60 minutes.

Time Synchronization may instead be initiated by the Master for each Outstation. This may be selected in the Add/Edit Master Poll dialog. It is not required to enable

Time Synchronization at both the Master and the Outstation.

Select Before Operate

The **Operate Timeout** parameter specifies the timeout interval between a Select and Operate request from the Master. If after receiving a valid Select control output request from the master, the RTU does not receive the corresponding Operate request within this timeout interval, the control output request is unsuccessful. The value of this parameter, expressed in seconds, is dependent on the master station, the data link and physical layer. Valid values are 1 to 6500 seconds. The default value is 15 seconds. The Master needs to have the Select/Operate functionally in order to use this feature.

Interoperability

The **Report only Level 2 Compliant Objects in Class Polls** parameter affects how Short Float Analog Input, Short Float Analog Output, and 32-bit Analog Output objects are reported. These objects are converted to 32-bit Analog Input and 16-bit Analog Output objects when this parameter is selected. Valid selections are:

- Select **Yes** to convert Short Float Analog Input, Short Float Analog Output, and 32-bit Analog Output objects to 32-bit Analog Input and 16-bit Analog Output objects.
- Select **No** to not convert Short Float Analog Input, Short Float Analog Output, and 32-bit Analog Output objects to 32-bit Analog Input and 16-bit Analog Output objects. This is the default selection.

The **Limit Maximum Events in Read Response** parameter allows limiting the number of events in a read response.

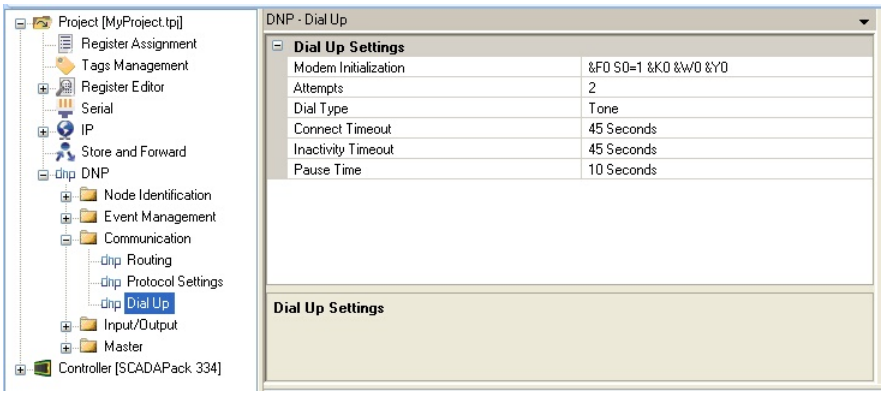
- Select **No** to not limit the number of events. This is the default setting.
- Select **Yes** to limit the number of events in a read response. Enter the number of events in the Maximum Events in Response parameter.

The **Limit Maximum Events in Response** parameter is the limit for the number of events in a read response. Valid values are 1 to 65535. The default value is disabled.

The Maximum Events parameter applies to responses from read commands only. It does not affect unsolicited responses.

Dial Up

The Dial Up section of the dialog defines modem parameters used when a dial up modem is used to communicate with stations that use dial up communication. The phone numbers for the stations are defined in the Routing configuration.



The **Modem Initialization** is the string that will be sent to the modem prior to each call. This is an ASCII null-terminated string. The maximum length of the string is 64 characters, including the null terminator. The default string is &F0 S0=1 &K0 &W0 &Y0.

The **Attempts** controls the maximum number of dial attempts that will be made to establish a Dial Up connection. The valid values for this parameter are 1 – 10. The default value is 2.

The **Dial Type** parameter controls whether tone or pulse dialing will be used for the call. Valid values are Tone dialing or Pulse dialing. The default value is Tone dialing.

The **Connect Timeout** controls the maximum time (in seconds) after initiating a dial sequence that the firmware will wait for a carrier signal before hanging up. The valid values for this parameter are 1 – 65535. The default value is 45.

The **Inactivity Timeout** controls the maximum time after message activity that a connection will be left open before hanging up. The valid values for this parameter are 1 – 65535 seconds. The default value is 45 seconds.

The **Pause Time** controls the delay time (in seconds) between dial events, to allow time for incoming calls. The valid values for this parameter are 1 – 65535. The default value is 10.

Input / Output

The Input/Output folder contains leaves for configuring the DNP data in this node. When a leaf is selected the configuration pane displays the configuration properties for the selected DNP data type.

The configuration pane for each data type has a number of buttons across the top of the pane. These buttons operate in a similar fashion for all DNP data types.

The left most button is the **Add** button. This adds a configured point to the list for the data type.

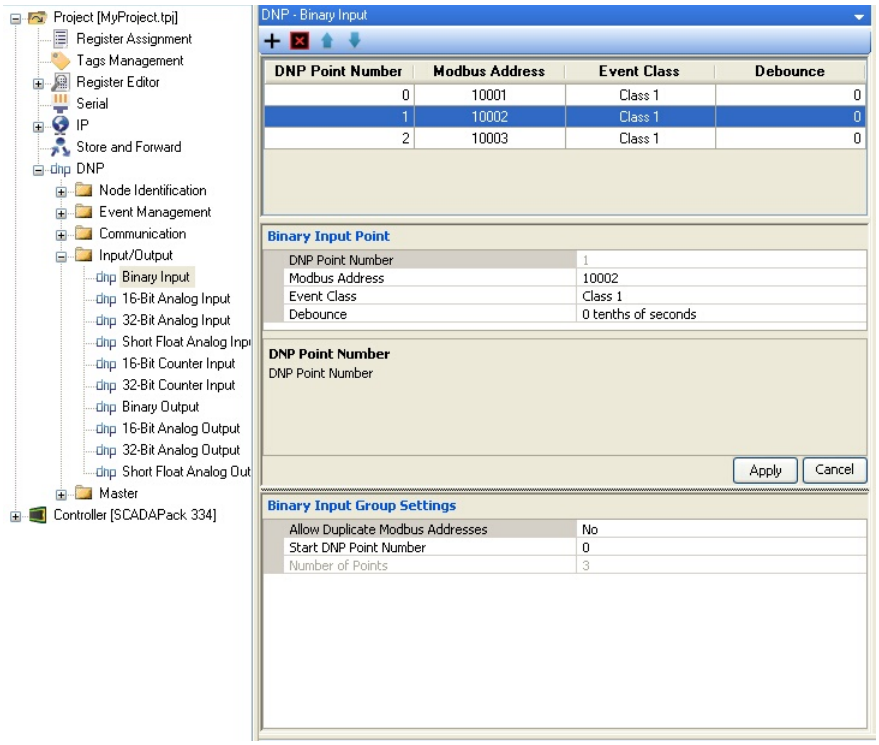
The next button is the **Delete** button. This deletes a selected point in the table for the data type.

The next button is the **Move Up** button. This moves the selected entry up in the list for the data type.

The next button is the **Move Down** button. This moves the selected entry down in the list for the data type.

Binary Input

The Binary Input leaf opens the Binary Point configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



Binary Input List

As Binary Inputs are defined they are added to the Binary Input grid when the Add button is clicked.

The grid list displays the DNP Point Number, Modbus Address, Event Class and Debounce value for each point added.

To move a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Move Up** or **Move Down** buttons to move the point.

To delete a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Delete** button to delete the point.

Binary Input Point

The **DNP Point Number** window displays the DNP Binary Input address of the point. Each Binary Input is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the Binary Input Group Settings grid and increments by one with each added Binary Input point.

The **Modbus Address** parameter specifies the Modbus address of the Binary Input assigned to the DNP Address. The SCADAPack and Micro16 controllers use Modbus addressing for all digital inputs. See the section Valid Modbus Addresses for DNP Points for valid Modbus addresses for each controller type.

The **Class of Event Object** parameter specifies the event object class the Binary Input is assigned. The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Debounce** parameter sets the frequency of change events. The input needs to remain in the same state for the debounce time for a change of state to be detected. The input is sampled every 0.1s. Changes shorter than the sample time cannot be detected. Valid values are 0 to 255 tenths of seconds. The value 0 means no debounce. The default value is 0.

To add the configured point to the Binary Input grid list:

- Click the **Add** button at the top of the DNP Configuration view.

Binary Input Group Settings

The **Allow Duplicate Modbus Addresses** parameter determines if the Modbus I/O database addresses assigned to the DNP data points need to be unique.

- Select **No** if you want to allow only one point to use the Modbus address.
- Select **Yes** if you want to allow more than one point to use the same Modbus address.

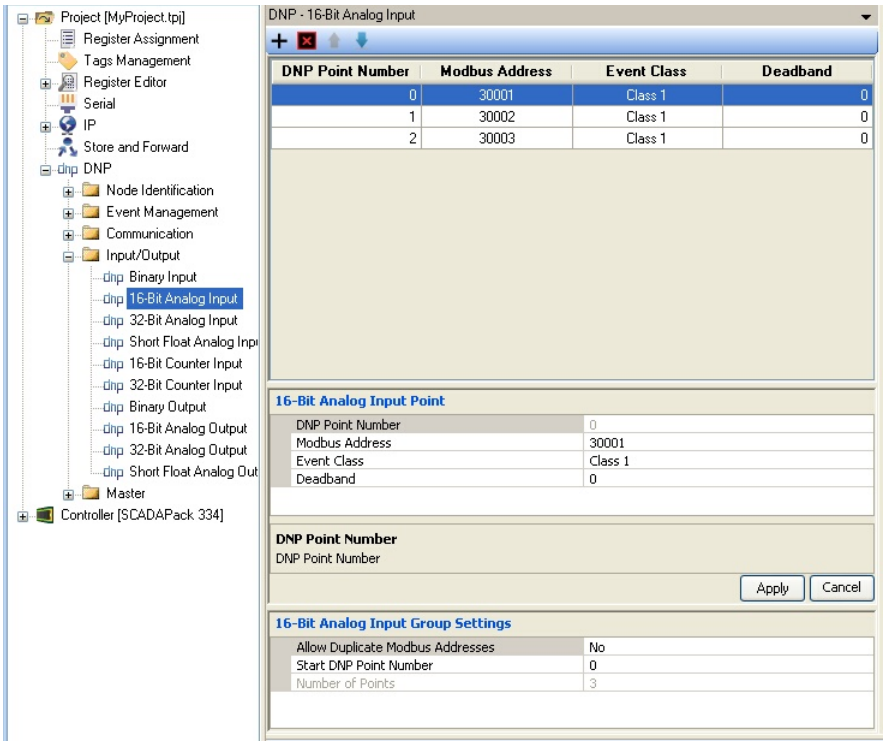
The default selection is No.

The **Start DNP Point Number** parameter specifies the starting DNP address of the first Binary Input point. Valid values are 0 – 65535.

The **Number of Points** displays number of binary inputs reported by this RTU. This value will increment with the addition of each configured Binary Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

16-Bit Analog Input

The 16-Bit Analog Input leaf opens the 16-Bit Analog Input configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



16-Bit Analog Input Point

The **DNP Point Number** window displays the DNP 16-Bit Analog Input address of the point. Each 16-Bit Analog Input is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the 16-Bit Analog Input Group Settings grid and increments by one with each added Input.

The **Modbus Address** parameter specifies the Modbus address of the 16-Bit Analog Input assigned to the DNP Address. The SCADAPack and Micro16 controllers use Modbus addressing for all analog inputs. See the section Valid Modbus Addresses for DNP Points for valid Modbus addresses for each controller type.

The **Class of Event Object** parameter specifies the event object class the 16-Bit analog Input is assigned. The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Deadband** parameter specifies whether the RTU generates events. The value entered is the minimum number of counts that the 16-Bit Analog Input needs to change in order to generate an event. Valid deadband values are 0 to 65535. A deadband value of 0 will cause any change to create an event.

To add the configured point to the 16-Bit Analog Input Point grid list:

- Click the **Add** button at the top of the DNP Configuration view.

16-Bit Analog Input Grid List

As 16-Bit Analog Inputs are defined they are added to the 16-Bit Analog Input grid when the **Add** button is clicked.

The grid list displays the DNP Point Number, Modbus Address, Event Class and Deadband value for each point added.

To move a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Move Up** or **Move Down** buttons to move the point.

To delete a point in the grid list:

1. Select the point by clicking the mouse anywhere in the point row and click the **Delete** button to delete the point.

16-Bit Analog Input Group Settings

The **Allow Duplicate Modbus Addresses** parameter determines if the Modbus I/O database addresses assigned to the DNP data points need to be unique.

- Select **No** if you want to allow only one point to use the same Modbus address.
- Select **Yes** if you want to allow more than one point to use the same Modbus address.

The default selection is No.

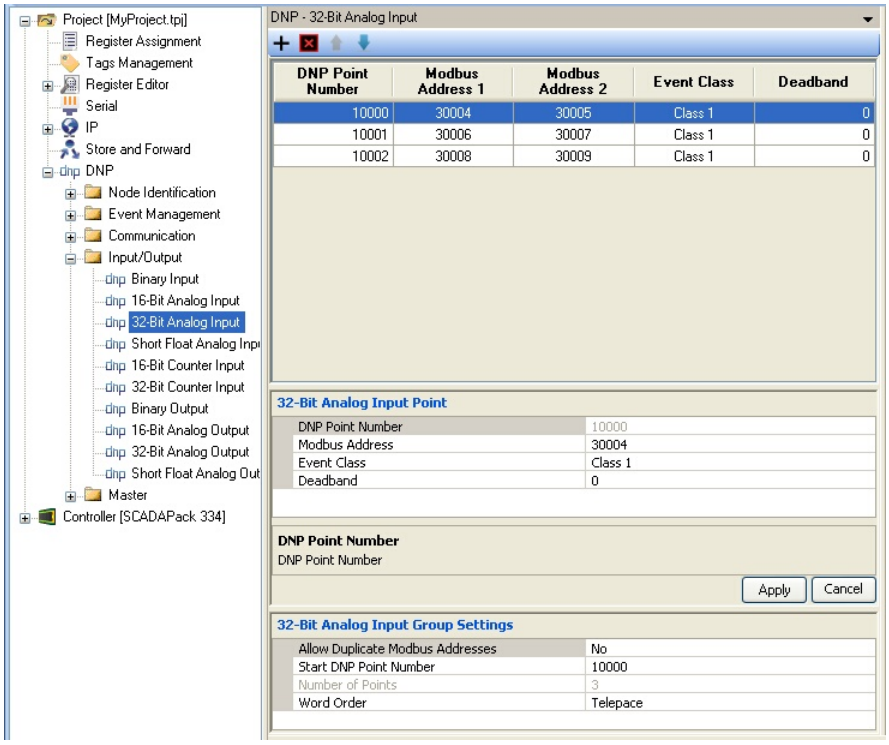
The **Start DNP Point Number** parameter specifies the starting DNP address of the first Binary Input point. Valid values are 0 – 65535.

The **Number of Points** displays number of 16-Bit analog inputs reported by this RTU. This value will increment with the addition of each configured 16-Bit Analog Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

32-Bit Analog Input

The 32-Bit Analog Input leaf opens the 32-Bit Analog Input configuration grid in the configuration pane. If no points have been defined the configuration grid is not

shown. Click the **Add** button to open the configuration grid.



32-Bit Analog Input Point

The **Modbus Address** parameter specifies the Modbus addresses of the 32-Bit Analog Input assigned to the DNP Address. 32-Bit Analog Inputs use two consecutive Modbus registers for each assigned DNP Address, the address that is entered and the next consecutive Modbus register. See the section Valid Modbus Address for DNP Points for valid Modbus addresses for each controller type.

The **Class of Event Object** parameter specifies the event object class the 32-Bit analog Input is assigned. The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Deadband** parameter specifies whether the RTU generates events. The value entered is the minimum number of counts that the 32-Bit Analog Input needs to change in order to generate an event. Valid deadband values are 0 to 4,294,967,295. A deadband value of 0 will cause any change to create an event.

To add the configured point to the 32-Bit Analog Input Point grid list:

- Click the **Add** button at the top of the DNP Configuration view.

32-Bit Analog Input Grid List

As 32-Bit Analog Inputs are defined they are added to the 32-Bit Analog Input Point grid when the Add button is clicked.

The grid list displays the DNP Point Number, Modbus Address 1, Modbus Address 2, Event Class and Deadband value for each point added.

To move a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Move Up** or **Move Down** buttons to move the point.

To delete a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Delete** button to delete the point.

32-Bit Analog Input Group Settings

The **Allow Duplicate Modbus Addresses** parameter determines if the Modbus I/O database addresses assigned to the DNP data points need to be unique.

- Select **No** if you want to allow only one point to use the same Modbus address.
- Select **Yes** if you want to allow more than one point to use the same Modbus address.

The default selection is No.

The **Start DNP Point Number** parameter specifies the starting DNP address of the first Binary Input point. Valid values are 0 – 65535.

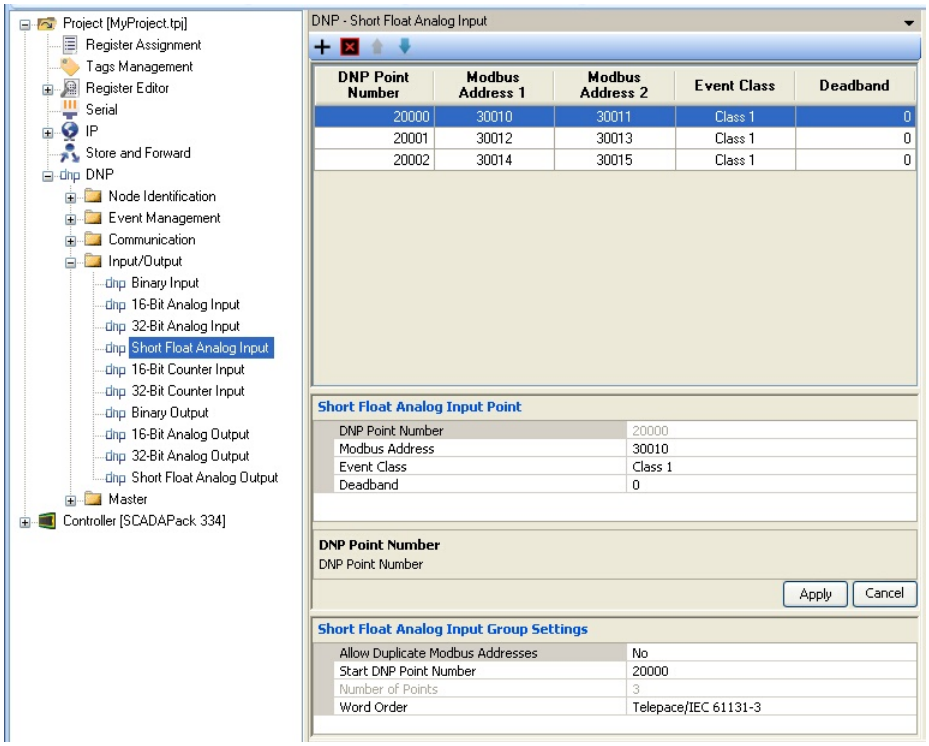
The **Number of Points** displays number of 32-Bit analog inputs reported by this RTU. This value will increment with the addition of each configured 32-Bit Analog Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Word Order** selection specifies the word order of the 32-bit value. The selections are:

- **Telepace** Least Significant Word in first register.
- **IEC 61131-3** Most Significant Word in first register.

Short Float Analog Input

The Short Float Analog Input leaf opens the Short Float Analog Input configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



Short Float Analog Input Point

The **DNP Point Number** window displays the DNP Short Float Analog Input address of the point. Each Short Float Analog Input is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the Short Float Analog Input Group Settings grid and increments by one with each added Input.

The **Modbus Address** parameter specifies the Modbus addresses of the Short Float Analog Input assigned to the DNP Address. Short Float Analog Inputs use two consecutive Modbus registers for each assigned DNP Address, the address that is entered and the next consecutive Modbus register. See the section Valid Modbus Address for DNP Points for valid Modbus addresses for each controller type.

The **Class of Event Object** parameter specifies the event object class the short float analog Input is assigned. The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Deadband** parameter specifies whether the RTU generates events. The value entered is the minimum number of counts that the Short Float Analog Input needs to change in order to generate an event. A deadband value of 0 will cause any change to create an event.

To add the configured point to the Short Float Analog Input Point grid list:

- Click the **Add** button at the top of the DNP Configuration view.

Short Float Analog Input Grid List

As Short Float Analog Inputs are defined they are added to the Short Float Analog Input grid when the Add button is clicked.

The grid list displays the DNP Point Number, Modbus Address 1, Modbus Address 2, Event Class and Deadband value for each point added.

To move a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Move Up** or **Move Down** buttons to move the point.

To delete a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Delete** button to delete the point.

Short Float Analog Input Group Settings

The Allow Duplicate Modbus Addresses parameter determines if the Modbus I/O database addresses assigned to the DNP data points need to be unique.

- Select **No** if you want to allow only one point to use the same Modbus address.
- Select **Yes** if you want to allow more than one point to use the same Modbus address.

The default selection is No.

The **Start DNP Point Number** parameter specifies the starting DNP address of the first Binary Input point. Valid values are 0 – 65535.

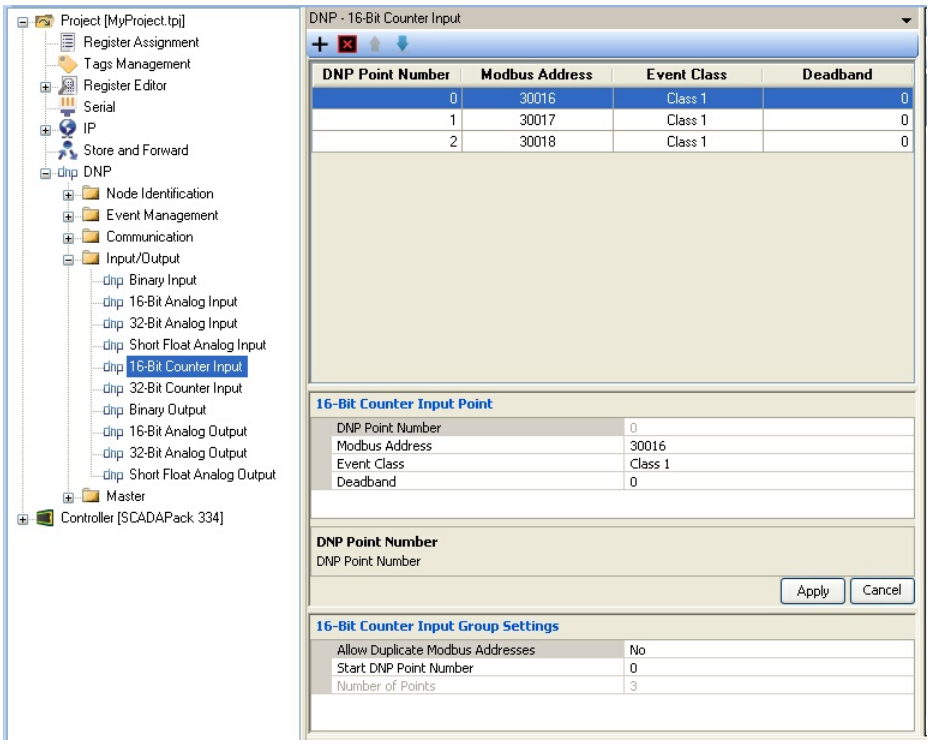
The **Number of Points** displays number of Short Float analog inputs reported by this RTU. This value will increment with the addition of each configured Short Float Analog Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Word Order** selection specifies the word order of the 32-bit value. The selections are:

- **Telepace / IEC 61131-3 (MSW First)** Most Significant Word in first register.
- **Reversed (LSW First)** Least Significant Word in first register.

16-Bit Counter Input

The 16-Bit Counter Input leaf opens the 16-Bit Counter Input configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



16-Bit Counter Input Point

The **DNP Point Number** window displays the DNP 16-Bit Counter Input address of the point. Each 16-Bit Counter Input is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the 16-Bit Counter Input Group Settings grid and increments by one with each added Input.

The **Modbus Address** parameter specifies the Modbus address of the 16-Bit Counter Input assigned to the DNP Address. The SCADAPack and Micro16 controllers use Modbus addressing for all analog inputs. See the section Valid Modbus Address for DNP Points for valid Modbus addresses for each controller type.

The **Class of Event Object** parameter specifies the event object class the 16-Bit Counter Input is assigned. The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Deadband** parameter specifies whether the RTU generates events. The value entered is the minimum number of counts that the 16-Bit Counter Input needs to change in order to generate an event. Valid deadband values are 0 to 65535. A deadband value of 0 will cause any change to create an event.

To add the configured point to the 16-Bit Counter Input Point grid list:

- Click the **Add** button at the top of the DNP Configuration view.

16-Bit Counter Input Grid List

As 16-Bit Analog Inputs are defined they are added to the 16-Bit Counter Input grid when the Add button is clicked.

The grid list displays the DNP Point Number, Modbus Address, Event Class and Deadband value for each point added.

To move a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Move Up** or **Move Down** buttons to move the point.

To delete a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Delete** button to delete the point.

16-Bit Counter Input Group Settings

The **Allow Duplicate Modbus Addresses** parameter determines if the Modbus I/O database addresses assigned to the DNP data points need to be unique.

- Select **No** if you want to allow only one point to use the same Modbus address.
- Select **Yes** if you want to allow more than one point to use the same Modbus address.

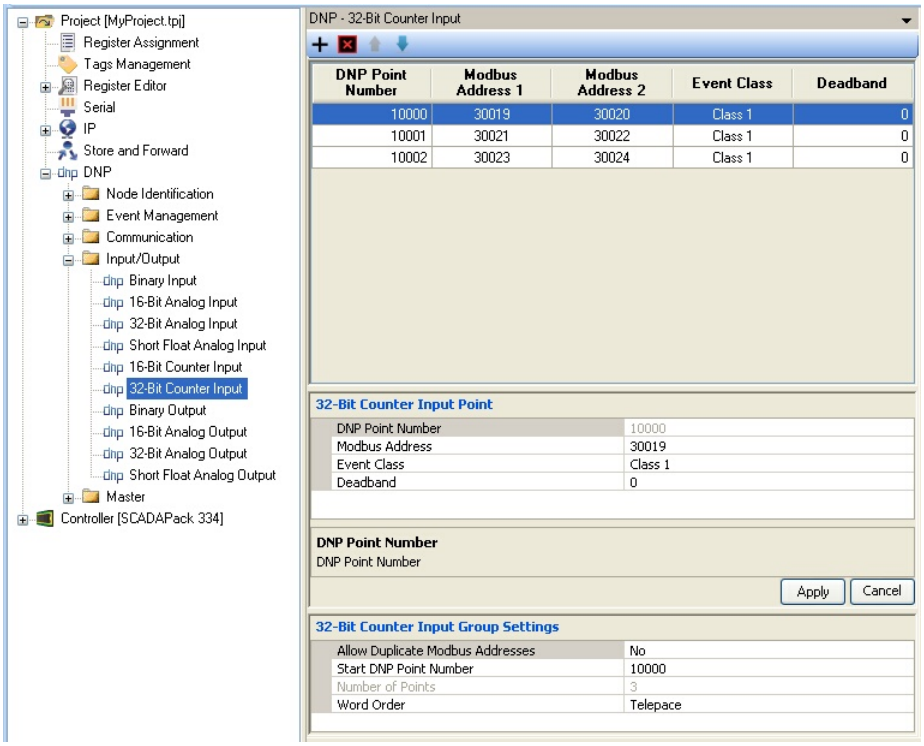
The default selection is No.

The **Start DNP Point Number** parameter specifies the starting DNP address of the first Counter Input point. Valid values are 0 – 65535.

The **Number of Points** displays number of 16-Bit Counter inputs reported by this RTU. This value will increment with the addition of each configured 16-Bit Counter Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

32-Bit Counter Input

The 32-Bit Counter Input leaf opens the 32-Bit Counter Input configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



32-Bit Counter Input Point

The **DNP Point Number** window displays the DNP 32-Bit Counter Input address of the point. Each 32-Bit Counter Input is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the 32-Bit Counter Input Group Settings grid and increments by one with each added Input.

The **Modbus Address** parameter specifies the Modbus addresses of the 32-Bit Counter Input assigned to the DNP Address. 32-Bit Counter Inputs use two consecutive Modbus registers for each assigned DNP Address, the address that is entered and the next consecutive Modbus register. See the section Valid Modbus Address for DNP Points for valid Modbus addresses for each controller type.

The **Class of Event Object** parameter specifies the event object class the 32-Bit Counter Input is assigned. The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Deadband** parameter specifies whether the RTU generates events. The value entered is the minimum number of counts that the 32-Bit Counter Input needs to change in order to generate an event. Valid deadband values are 0 to 4,294,967,295. A deadband value of 0 will cause any change to create an event.

To add the configured point to the 32-Bit Counter Input Point grid list:

- Click the **Add** button at the top of the DNP Configuration view.

32-Bit Analog Input Grid List

As 32-Bit Analog Inputs are defined they are added to the 32-Bit Counter Input grid when the Add button is clicked.

The grid list displays the DNP Point Number, Modbus Address 1, Modbus Address 2, Event Class and Deadband value for each point added.

To move a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Move Up** or **Move Down** buttons to move the point.

To delete a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Delete** button to delete the point.

32-Bit Counter Input Group Settings

The **Allow Duplicate Modbus Addresses** parameter determines if the Modbus I/O database addresses assigned to the DNP data points need to be unique.

- Select **No** if you want to allow only one point to use the same Modbus address.
- Select **Yes** if you want to allow more than one point to use the same Modbus address.

The default selection is No.

The **Start DNP Point Number** parameter specifies the starting DNP address of the first Binary Input point. Valid values are 0 – 65535.

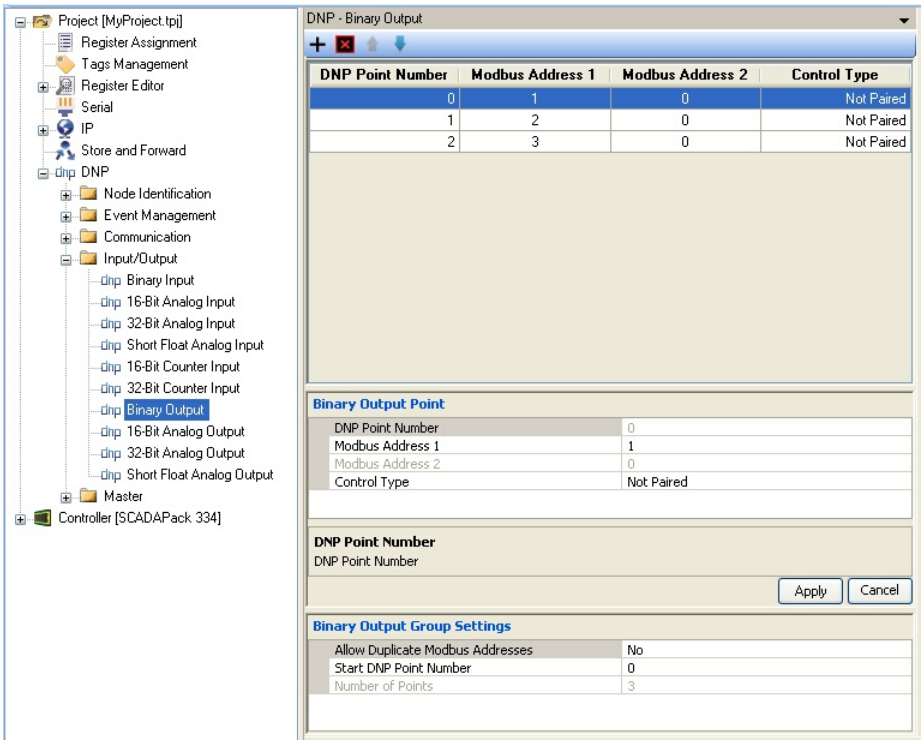
The **Number of Points** displays number of 32-Bit Counter inputs reported by this RTU. This value will increment with the addition of each configured 32-Bit Counter Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Word Order** selection specifies the word order of the 32-bit value. The selections are:

- **Telepace** Most Significant Word in first register.
- **IEC 61131-3** Least Significant Word in first register.

Binary Output

The Binary Output leaf opens the Binary Output configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



Binary Output Point

The **DNP Point Number** window displays the DNP Binary Output address of the point. Each Binary Output is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the Binary Outputs Group Settings and increments by one with each added Output.

The **Modbus Address 1** parameter specifies the Modbus address of the Binary Output assigned to the DNP Address. The SCADAPack and Micro16 controllers use Modbus addressing for all digital outputs. See the section Valid Modbus Address for DNP Points for valid Modbus addresses for each controller type.

The **Modbus Address 2** parameter specifies the second Modbus address of the second Binary Output assigned to the DNP Address when the Paired control type is selected. This selection is not active when the control type is Not Paired. See the section Valid Modbus Address for DNP Points for valid Modbus addresses for each controller type.

The **Control Type** parameter specifies whether the Binary Output is a paired control or not. If it is a paired control, i.e. trip/close output type, this means that the DNP address is associated to two physical control outputs and requires two Modbus addresses per DNP address. Control type selections are:

- Paired
- Not Paired

To add the configured point to the 32-Bit Analog Output Point grid list:

- Click the **Add** button at the top of the DNP Configuration view.

Binary Output Grid List

As Binary Outputs are defined they are added to the Binary Outputs grid when the Add button is clicked.

The grid list displays the DNP Point Number, Modbus Address 1, Modbus Address 2 and Control Type value for each point added.

To move a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Move Up** or **Move Down** buttons to move the point.

To delete a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Delete** button to delete the point.

Binary Output Group Settings

The **Allow Duplicate Modbus Addresses** parameter determines if the Modbus I/O database addresses assigned to the DNP data points need to be unique.

- Select **No** if you want to allow only one point to use the same Modbus address.
- Select **Yes** if you want to allow more than one point to use the same Modbus address.

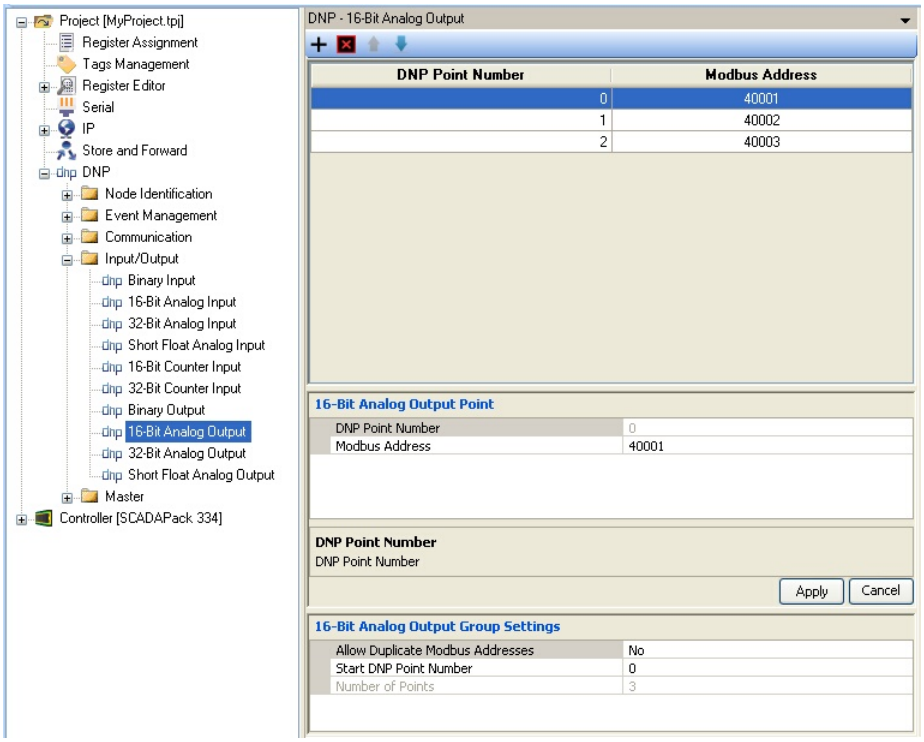
The default selection is No.

The **Start DNP Point Number** parameter specifies the starting DNP address of the first Binary Input point. Valid values are 0 – 65535.

The **Number of Points** displays number of Binary Outputs reported by this RTU. This value will increment with the addition of each configured 32-Bit Counter Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

16-Bit Analog Output

The 16-Bit Analog Output leaf opens the 16-Bit Analog Output configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



16-Bit Analog Output Point

The **DNP Point Number** window displays the DNP 16-Bit Analog Output address of the point. Each 16-Bit Analog Output is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the 16-Bit Analog Outputs Group Settings and increments by one with each added Output.

The **Modbus Address** parameter specifies the Modbus address of the 16-Bit Analog Output assigned to the DNP Address. The SCADAPack and Micro16 controllers use Modbus addressing for all digital outputs. See the section Valid Modbus Address for DNP Points for valid Modbus addresses for each controller type.

To add the configured point to the 32-Bit Analog Output Point grid list:

- Click the **Add** button at the top of the DNP Configuration view.

16-Bit Analog Output Grid List

As 16-Bit Analog Outputs are defined they are added to the Binary Outputs grid when the Add button is clicked.

The grid list displays the DNP Point Number and the Modbus Address values for each point added.

To move a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Move Up** or **Move Down** buttons to move the point.

To delete a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Delete** button to delete the point.

16-Bit Analog Output Group Settings

The **Allow Duplicate Modbus Addresses** parameter determines if the Modbus I/O database addresses assigned to the DNP data points need to be unique.

- Select **No** if you want to allow only one point to use the same Modbus address.

- Select **Yes** if you want to allow more than one point to use the same Modbus address.

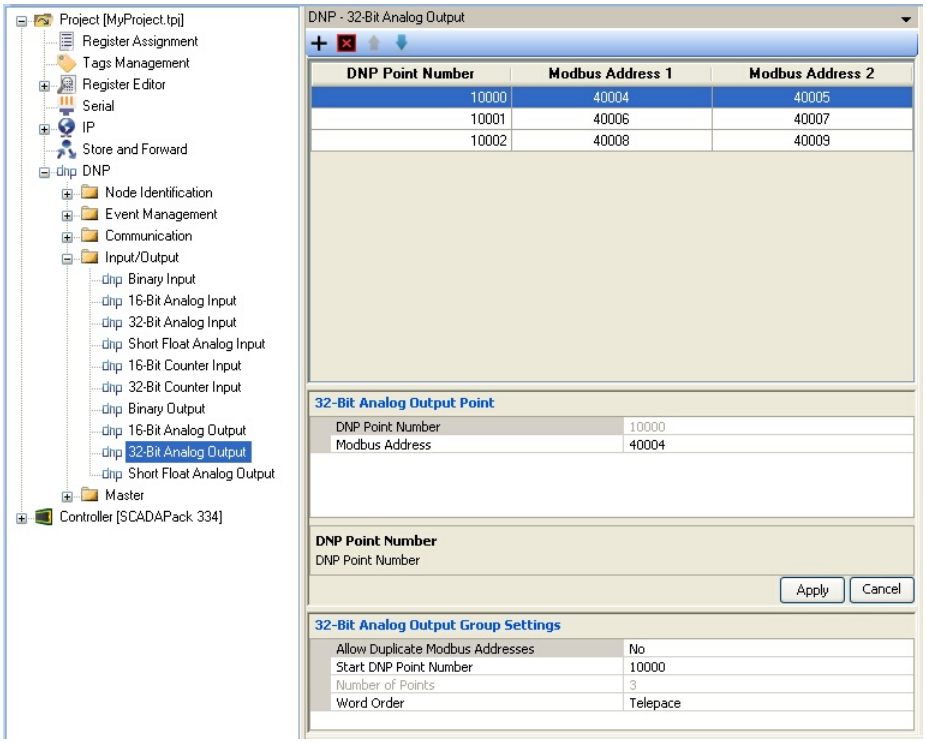
The default selection is No.

The **Start DNP Point Number** parameter specifies the starting DNP address of the first Binary Input point. Valid values are 0 – 65535.

The **Number of Points** displays number of 16-Bit Analog Outputs reported by this RTU. This value will increment with the addition of each configured 16-Bit Analog Output point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

32-Bit Analog Output

The 32-Bit Analog Output leaf opens the 32-Bit Analog Output configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



32-Bit Analog Output Point

The **DNP Point Number** window displays the DNP 32-Bit Analog Output address of the point. Each 32-Bit Analog Input is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the 32-Bit Analog Output Group Settings grid and increments by one with each added Input.

The **Modbus Address** parameter specifies the Modbus addresses of the 32-Bit Analog Output assigned to the DNP Address. 32-Bit Analog Output use two consecutive Modbus registers for each assigned DNP Address, the address that is entered and the next consecutive Modbus register. See the section Valid Modbus Address for DNP Points for valid Modbus addresses for each controller type.

To add the configured point to the 32-Bit Analog Output Point grid list:

- Click the **Add** button at the top of the DNP Configuration view.

32-Bit Analog Input Grid List

As 32-Bit Analog Output are defined they are added to the 32-Bit Analog Output grid when the Add button is clicked.

The grid list displays the DNP Point Number, Modbus Address 1 and Modbus Address 2 value for each point added.

To move a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Move Up** or **Move Down** buttons to move the point.

To delete a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Delete** button to delete the point.

32-Bit Analog Output Group Settings

The **Allow Duplicate Modbus Addresses** parameter determines if the Modbus I/O database addresses assigned to the DNP data points need to be unique.

- Select **No** if you want to allow only one point to use the same Modbus address.
- Select **Yes** if you want to allow more than one point to use the same Modbus address.

The default selection is No.

The **Start DNP Point Number** parameter specifies the starting DNP address of the first Analog Output point. Valid values are 0 – 65535.

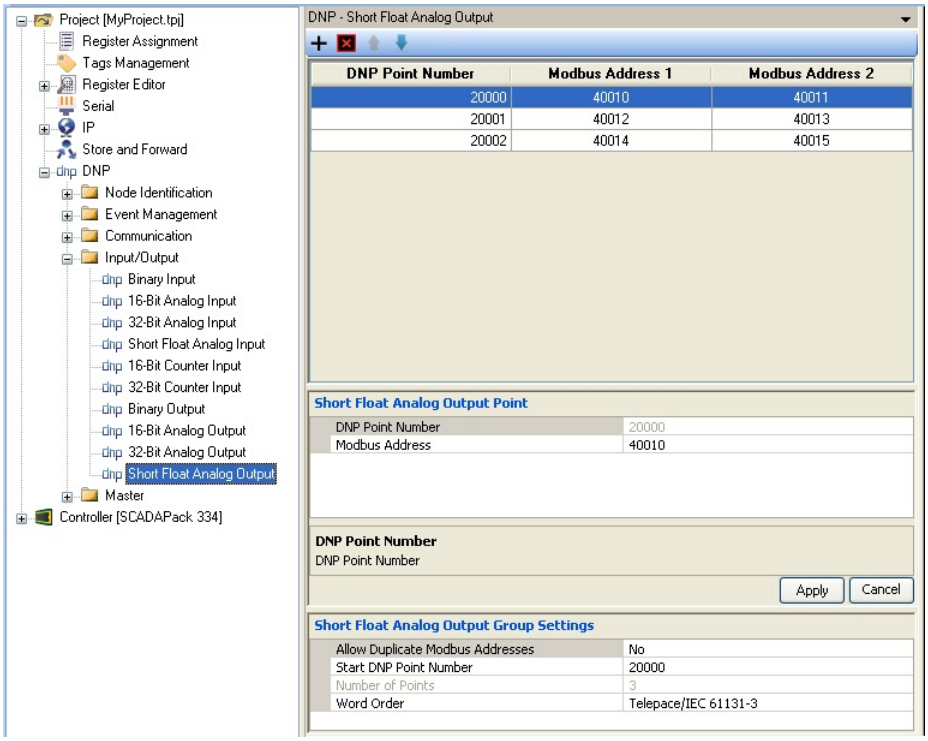
The **Number of Points** displays number of 32-Bit Output points reported by this RTU. This value will increment with the addition of each configured 32-Bit Analog Output point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Word Order** selection specifies the word order of the 32-bit value. The selections are:

- **Telepace** Most Significant Word in first register.
- **IEC 61131-3** Least Significant Word in first register.

Short Float Analog Output

The Short Float Analog Output leaf opens the Short Float Analog Output configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



Short Float Analog Output Point

The **DNP Point Number** window displays the DNP Short Float Analog Output address of the point. Each Short Float Analog Output is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the Short Float Analog Output Group Settings grid and increments by one with each added Input.

The **Modbus Address** parameter specifies the Modbus addresses of the Short Float Analog Output assigned to the DNP Address. Short Float Analog Output use two consecutive Modbus registers for each assigned DNP Address, the address that is entered and the next consecutive Modbus register. See the section Valid Modbus Address for DNP Points for valid Modbus addresses for each controller type.

To add the configured point to the Short Float Analog Input Point grid list:

- Click the **Add** button at the top of the DNP Configuration view.

Short Float Analog Output Grid List

As Short Float Analog Output are defined they are added to the Short Float Analog Output grid when the Add button is clicked.

The grid list displays the DNP Point Number, Modbus Address 1 and Modbus Address 2, value for each point added.

To move a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Move Up** or **Move Down** buttons to move the point.

To delete a point in the grid list:

- Select the point by clicking the mouse anywhere in the point row and click the **Delete** button to delete the point.

Short Float Analog Input Group Settings

The **Allow Duplicate Modbus Addresses** parameter determines if the Modbus I/O database addresses assigned to the DNP data points need to be unique.

- Select **No** if you want to allow only one point to use the same Modbus address.
- Select **Yes** if you want to allow more than one point to use the same Modbus address.

The default selection is No.

The **Start DNP Point Number** parameter specifies the starting DNP address of the first Short Float Analog Output point. Valid values are 0 – 65535.

The **Number of Points** displays number of Short Float analog output reported by this RTU. This value will increment with the addition of each configured Short Float Analog Output point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Word Order** selection specifies the word order of the 32-bit value. The selections are:

- **Telepace / IEC 61131-3 (MSW First)** Most Significant Word in first register.
- **Reversed (LSW First)** Least Significant Word in first register.

Master

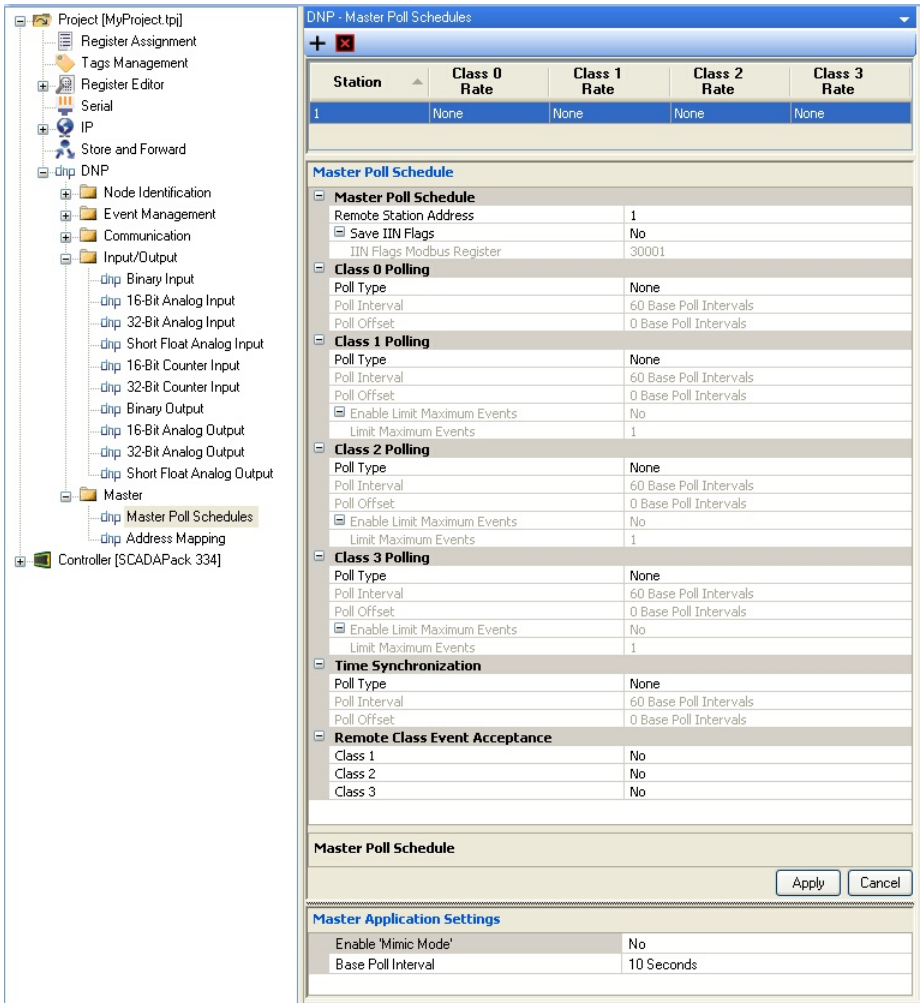
The Master folder defines the properties for configuring a DNP master node. This folder is available only when the controller type is SCADAPack 314, SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P. These controllers support DNP Master.

The Master folder contains the Master Poll Schedules and Address Mapping leaves.

Master Poll Schedules

The Master Poll Schedules property grid defines how the master station will communicate with its remote DNP devices.

The Master Poll Schedules leaf opens the Master Poll Schedule configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



The following buttons are found at the top of the Master Poll Schedule configuration grid.

- The **Add** button is used to add a configured poll schedule to the Master Poll Schedules Grid list.
- The **Delete** button is used to delete configured poll schedules from the Master Poll Schedules Grid list.
- The **Apply** button adds the configured poll schedule to the DNP Configuration for the node.
- The **Cancel** button is used to restore the default settings for the Master Poll Schedule in the Master Poll Schedule configuration grid.

Master Poll Schedule

The **Remote Station Address** parameter is the address of the remote DNP slave device to be polled. Valid values are 0 to 65519.

The **Save IIN Flags** parameter enables or disables storing the IIN (Internal Indications) flags from the remote slave station in a Modbus database register.

- Select Yes to save the IIN flags to a Modbus register. When Yes is selected the IIN Flags Modbus Register parameter is enabled. Valid entries are Modbus register addresses 30001 to 39999 and 40001 to 49999. The default value is 30001.
- Select No to not save the IIN flags to a Modbus register. This is the default value.

The IIN flags are set by the slave to indicate the events in the following table. The events are bit mapped to the Modbus register. All bits except Device Restarted and Time Synchronization required are cleared when the slave station receives any poll or read data command. The master will write to bits 5 and 11 depending on the local

conditions in the master.

Bit	Description
0	last received message was a broadcast message
1	Class 1 data available
2	Class 2 data available
3	Class 3 data available
4	Time Synchronization required
5	not used (returns 0)
6	Device trouble Indicates memory allocation error in the slave, or For master in mimic mode indicates communication failure with the slave device.
7	Device restarted (set on a power cycle)
8	Function Code not implemented
9	Requested object unknown or there were errors in the application data
10	Parameters out of range
11	Event buffer overflowed Indicates event buffer overflow in the slave or master. The slave will set this bit if the event buffer in the slave is overflowed. The master will set this bit if the event buffer in the master has overflowed with events read from the slave. Ensure the event buffer size, in the master and slave, is set to a value that will ensure the buffer does not overflow and events are lost.
12	not used (returns 0)
13	not used (returns 0)
14	not used (returns 0)
15	not used (returns 0)

Class 0 Polling

The Class 0 Polling section of the Master Poll Schedules view specifies the type and rate of polling for Class 0 data.

The **Poll Type** parameter selects the type of polling the master will do for Class 0 data in the remote station.

- The **None** selection disables class 0 polling for the slave station. This is the default selection.
- The **At Start Up** Only selection will cause the master to poll the slave station at startup only.
- The **Interval** selection will cause the master to poll the slave station at startup and then every Poll Interval. The Poll Interval parameter is enabled when Interval poll type is selected.

The **Poll Interval** parameter sets the number of poll intervals to wait between master polls the remote station. Poll intervals are a combination of the Base Poll Interval and the poll interval. For example if the Base Poll interval is 10 seconds and the Poll Interval parameter is set to 60 then the master will poll the slave station every 600 seconds (10 minutes). Valid values are 1 to 32767. The default value is 60. The Base Poll Interval is set in the Master Application Settings grid.

The **Poll Offset** parameter is used to distribute the load on the communication network. The Poll Offset is entered in multiples of the Base Poll interval. Valid values for this parameter are 0 to the Poll Interval value minus 1. Any non-zero value delays the start of polling for the specified objects by that amount. The default value is 0. This control is disabled when None is selected for Poll Type, and enabled otherwise. For an example of using the Poll Offset parameter see the Poll Offset Example section.

Class 1 Polling

The Class 1 Polling section of the dialog specifies the type and rate of polling for Class 1 data.

The **Poll Type** parameter selects the type of polling the master will do for Class 1 data in the remote station.

- The **None** selection disables class 0 polling for the slave station. This is the default selection.
- The **At Start Up Only** selection will cause the master to poll the slave station at startup only.
- The **Interval** selection will cause the master to poll the slave station at startup and then every Poll Interval. The Poll Interval parameter is enabled when Interval poll type is selected.

The **Poll Interval** parameter sets the number of poll intervals to wait between master polls the remote station. Poll intervals are a combination of the Base Poll Interval and the poll interval. For example if the Base Poll interval is 10 seconds and the Poll Interval parameter is set to 60 then the master will poll the slave station every 600 seconds (10 minutes). Valid values are 1 to 32767. The default value is 60. The Base Poll Interval is set in the Master Application Settings grid.

The **Poll Offset** parameter is used to distribute the load on the communication network. The Poll Offset is entered in multiples of the Base Poll interval. Valid values for this parameter are 0 to the Poll Interval value minus 1. Any non-zero value delays the start of polling for the specified objects by that amount. The default value is 0. This control is disabled when None is selected for Poll Type, and enabled otherwise. For an example of using the Poll Offset parameter see the Poll Offset Example section.

The **Enable Limit Maximum Events** parameter enables the limiting of the number of events in poll responses for Class 1/2/3 data.

- The default value is **No** meaning there is no limit on the number of events.
- Select **Yes** to specify a limit in the Limit Maximum Events parameter. The valid values for this parameter are 1 to 65535. The default value is 65535. This control is disabled when None is selected for the Poll Type, and enabled otherwise.

The **Limit Maximum Events** parameter can be used to manage communication load on a system. Consider the example of a master polling some data logging remotes, and the case where one of the remotes has been offline for a long time. The remote will have built up a large number of buffered events. If the master polled it for all events, the reply might take a long time, and cause an unwanted delay in the master's polling cycle. However if the master limits the number of events returned, the reply message duration will be more deterministic and the master can ensure its poll loop timing is maintained. In this case, the event retrieval from the data logger will be distributed over a number of poll cycles.

Class 2 Polling

The Class 2 Polling section of the dialog specifies the type and rate of polling for Class 2 data.

The **Poll Type** parameter selects the type of polling the master will do for Class 2 data in the remote station.

- The **None** selection disables class 0 polling for the slave station. This is the default selection.
- The **At Start Up Only** selection will cause the master to poll the slave station at startup only.
- The **Interval** selection will cause the master to poll the slave station at startup and then every Poll Interval. The Poll Interval parameter is enabled when Interval poll type is selected.

The **Poll Interval** parameter sets the number of poll intervals to wait between master polls the remote station. Poll intervals are a combination of the Base Poll Interval and the poll interval. For example if the Base Poll interval is 10 seconds and the Poll Interval parameter is set to 60 then the master will poll the slave station every 600 seconds (10 minutes). Valid values are 1 to 32767. The default value is 60. The Base Poll Interval is set in the Master Application Settings grid.

The **Poll Offset** parameter is used to distribute the load on the communication network. The Poll Offset is entered in multiples of the Base Poll interval. Valid values for this parameter are 0 to the Poll Interval value minus 1. Any non-zero value delays the start of polling for the specified objects by that amount. The default value is 0. This control is disabled when None is selected for Poll Type, and enabled otherwise. For an example of using the Poll Offset parameter see the Poll Offset Example section.

The **Enable Limit Maximum Events** parameter enables the limiting of the number of events in poll responses for Class 1/2/3 data.

- The default value is **No** meaning there is no limit on the number of events.
- Select **Yes** to specify a limit in the Limit Maximum Events parameter. The valid values for this parameter are 1 to 65535. The default value is 65535. This control is disabled when None is selected for the Poll Type, and enabled otherwise.

The **Limit Maximum Events** parameter can be used to manage communication load on a system. Consider the example of a master polling some data logging remotes, and the case where one of the remotes has been offline for a long time. The remote will have built up a large number of buffered events. If the master polled it for all events, the reply might take a long time, and cause an unwanted delay in the master's polling cycle. However if the master limits the number of events returned, the reply message duration will be more deterministic and the master can ensure its poll loop timing is maintained. In this case, the event retrieval from the data logger will be distributed over a number of poll cycles.

Class 3 Polling

The Class 3 Polling section of the dialog specifies the type and rate of polling for Class 3 data.

The **Poll Type** parameter selects the type of polling the master will do for Class 3 data in the remote station.

- The **None** selection disables class 0 polling for the slave station. This is the default selection.
- The **At Start Up Only** selection will cause the master to poll the slave station at startup only.
- The **Interval** selection will cause the master to poll the slave station at startup and then every Poll Interval. The Poll Interval parameter is enabled when Interval poll type is selected.

The **Poll Interval** parameter sets the number of poll intervals to wait between master polls the remote station. Poll intervals are a combination of the Base Poll Interval and the poll interval. For example if the Base Poll interval is 10 seconds and the Poll Interval parameter is set to 60 then the master will poll the slave station every 600 seconds (10 minutes). Valid values are 1 to 32767. The default value is 60. The Base Poll Interval is set in the Master Application Settings grid.

The **Poll Offset** parameter is used to distribute the load on the communication network. The Poll Offset is entered in multiples of the Base Poll interval. Valid values for this parameter are 0 to the Poll Interval value minus 1. Any non-zero value delays the start of polling for the specified objects by that amount. The default value is 0. This control is disabled when None is selected for Poll Type, and enabled otherwise. For an example of using the Poll Offset parameter see the Poll Offset Example section.

The **Enable Limit Maximum Events** parameter enables the limiting of the number of events in poll responses for Class 1/2/3 data.

- The default value is **No** meaning there is no limit on the number of events.
- Select **Yes** to specify a limit in the Limit Maximum Events parameter. The valid values for this parameter are 1 to 65535. The default value is 65535. This control is disabled when None is selected for the Poll Type, and enabled otherwise.

The **Limit Maximum Events** parameter can be used to manage communication load on a system. Consider the example of a master polling some data logging remotes, and the case where one of the remotes has been offline for a long time. The remote will have built up a large number of buffered events. If the master polled it for all events, the reply might take a long time, and cause an unwanted delay in the master's polling cycle. However if the master limits the number of events returned, the reply message duration will be more deterministic and the master can ensure its poll loop timing is maintained. In this case, the event retrieval from the data logger will be distributed over a number of poll cycles.

Time Synchronization

The Time Synchronization parameter specifies the rate of at which a time synchronization is sent to the remote device, as a multiple of the base poll interval.

The **Poll Type** parameter selects the type of polling the master will do for Time Synchronization with the remote station.

- The **None** selection disables issuing a time synchronization to the remote device. This is the default selection.
- The **At Start Up Only** selection will cause issuing a time synchronization at startup only.
- The **Interval** selection will cause a time synchronization to be sent to the remote device at startup and then every Poll Interval of the base poll interval seconds.

The **Poll Interval** parameter sets the number of poll intervals to wait between Time Synchronization polls to the remote station. Poll intervals are a combination of the Base Poll Interval and the poll interval. For example if the Base Poll interval is 10 seconds and the Poll Interval parameter is set to 60 then the master will poll the slave station every 600 seconds (10 minutes). Valid values are 1 to 32767. The default value is 60. The Base Poll Interval is set in the Master Application Settings grid.

The **Poll Offset** parameter is used to distribute the load on the communication network. The Poll Offset is entered in multiples of the Base Poll interval. Valid values for this parameter are 0 to the Poll Interval value minus 1. Any non-zero value delays the start of polling for the specified objects by that amount. The default value is 0. This control is disabled when None is selected for Poll Type, and enabled otherwise. For an example of using the Poll Offset parameter see the Poll Offset Example at the end of this section.

Remote Class Event Acceptance

The Remote Class Event Acceptance parameters are used in conjunction with the Enable Unsolicited Responses on Start Up parameter on the Event Initialization view.

Certain non-SCADAPack remote devices are designed to start with their Enable Unsolicited Responses on Start Up parameter set to No.

With SCADAPack remote devices the Enable Unsolicited Responses on Start Up parameter may be set to Yes, and the Remote Class Acceptance parameters may be set to No.

The **Class 1** selection displays the status of unsolicited responses from the remote device for Class 1 events.

- Select **No** to disable acceptance of unsolicited responses from the remote device. This is the default selection.
- Select **Yes** causes the master to (after it detects the slave come online) send a command allowing the slave to begin sending Unsolicited Responses of Class 2events.

The **Class 2** selection displays the status of unsolicited responses from the remote device for Class 2 events.

- Select **No** to disable acceptance of unsolicited responses from the remote device. This is the default selection.
- Select **Yes** causes the master to (after it detects the slave come online) send a command allowing the slave to begin sending Unsolicited Responses of Class 1events.

The **Class 3** selection displays the status of unsolicited responses from the remote device for Class 3 events.

- Select **No** to disable acceptance of unsolicited responses from the remote device. This is the default selection.
- Select **Yes** causes the master to (after it detects the slave come online) send a command allowing the slave to begin sending Unsolicited Responses of Class 3 events.

Master Poll Schedules Grid List

As Remote stations are defined they are added to the Master Poll Schedules grid when the Add button is clicked.

The grid list displays the Station, Class 0 Rate, Class 1 Rate, Class 2 Rate and Class 3 Rate.

Master Application Settings

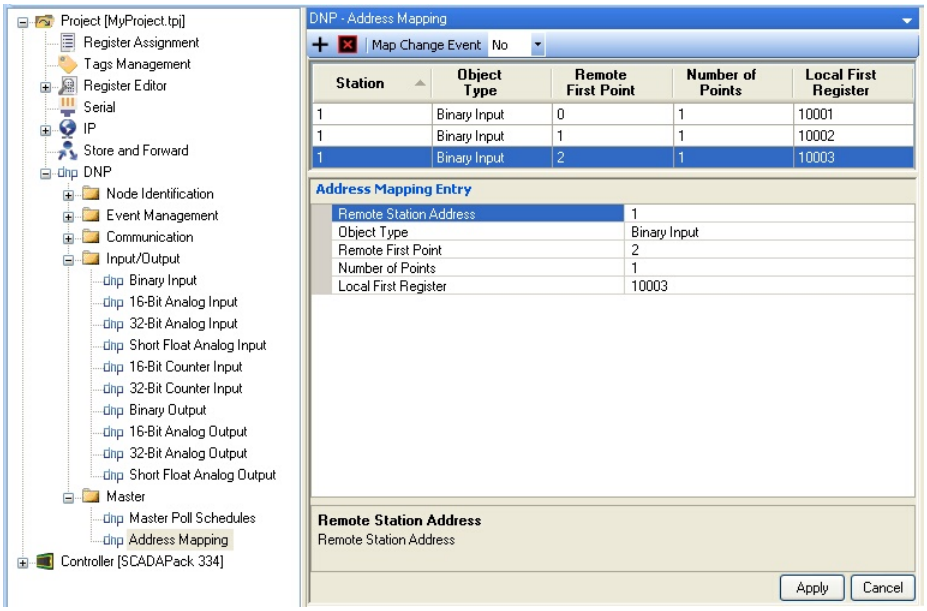
The **Enable Mimic Mode** parameter specifies if DNP Mimic Mode is enabled. The valid selections are Yes or No. When DNP Mimic Mode is enabled the controller will intercept DNP messages destined for a remote DNP station address, and will respond directly, as though the controller were the designated target. For read commands, the controller will respond with data from its Remote DNP Objects corresponding with the target address. For write commands, the controller will write data into its Remote DNP Objects, and issue a direct response to acknowledge the command. It will then issue a new command to write the data to the designated target.

The **Base Poll Interval** parameter is the base interval (in seconds) for polling slave devices. The poll rates and issuing time synchronization will be configured in multiples of the base poll interval. The slave devices with the same poll rates will be polled in the order they appear in the poll table. The valid values for this parameter are 1 to 65535. The default value is 10 seconds.

Address Mapping

The Address Mapping property grid defines how remote device DNP points are mapped to local Modbus registers in the master station.

The Address Mapping leaf opens the Address Mapping configuration grid in the configuration pane. If no points have been defined the configuration grid is not shown. Click the **Add** button to open the configuration grid.



The following buttons are found at the top of the Master Poll Schedule configuration grid.

- The **Add** button is used to add a configured Address Mapping Entry to the Address Mapping Table list.
- The Delete button is used to delete configured Address Mapping Entries from the Address Mapping Table list.
- The Apply button adds the Address Mapping Entries to the DNP Configuration for the node.
- The Cancel button is used to restore the default settings for the Address Mapping Entry in the Address Mapping Entry configuration grid.

Address Mapping Entry

The **Remote Station** parameter displays the address of the remote DNP station. Valid values for this field are from 0 to 65519. The default setting is station 1.

The **Object Type** parameter defines the DNP data Object Type. The list of available types includes:

- Binary Input
- 16-bit Analog Input
- 32-bit Analog Input
- Short Floating Point Analog Input
- 16-bit Counter Input
- 32-bit Counter Input

- Binary Output
- 16-bit Analog Output
- 32-bit Analog Output
- Short Floating Point Analog Output

The Default selection is Binary Input.

The **Remote First Point** parameter defines the starting address of the remote DNP data points. Valid values are from 0 to 65519. The default value is 0.

The **Number of Points** parameter defines the number of remote points to be mapped. Valid values for this field are from 1 to 9999. The default value is 1.

The **Local First Register** edit control displays the starting address of local Modbus register where the remote data points are to be mapped. Valid values depend on the selection of DNP Object Type; refer to section Valid Modbus Address for DNP Points for valid Modbus addresses for each controller type.

The **Add** button checks the data for this Address Mapping entry. If the data entered is invalid, an error message is displayed. The Address Mapping entry is invalid if any of the parameters are out of range. The data is also invalid if it conflicts with another Address Mapping entry. Such conflict occurs when the combination of station number, object type, and object address is not unique.

Address Mapping Table

As Address Mapping entries are defined they are added to the Address Mapping Table grid when the Add button is clicked.

The grid list displays the Station, Object Type, Remote First Point, Number of Points and Local First Register.

Map Change Events

The Map Change Events parameter selects if change events from a remote device are mapped into the local DNP change event buffer.

Mapped change events may trigger an Unsolicited Response message to be sent, after the Hold Count or Hold Time is reached. It may be desired instead to map only static (live) values into local Modbus registers.

- Select **No** to map only static (live) values into the local Modbus registers. The default value is No.
- Select **Yes** to map change events from the remote device into the local DNP change event buffer.

Device Profile Documents

Device Profile Interoperability documents detail the level of implementation for outstation (slave) and master DNP SCADAPack devices. These documents when coupled with other third party device profile documents provide a set of commonly supported DNP functionality.

DNP Slave Device Profile Document

V3.00
DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT
This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	**Var**	**Description**	**Func Codes (dec)**	**Qual Codes (hex)**	**Func Codes**	**Qual Codes (hex)**
1	0	Binary Input - All Variations	1	06		
1	1	Binary Input			129, 130	00, 01
1	2	Binary Input with Status			129, 130	00, 01
2	0	Binary Input Change - All Variations	1	06,07,08		
2	1	Binary Input Change without Time	1	06,07,08	129, 130	17, 28
2	2	Binary Input Change with Time	1	06,07,08	129, 130	17, 28
2	3	Binary Input Change with Relative Time	1	06,07,08	129, 130	17, 28
10	0	Binary Output - All Variations	1	06		
10	1	Binary Output				
10	2	Binary Output Status			129, 130	00, 01
12	0	Control Block - All Variations				
12	1	Control Relay Output Block	3, 4, 5, 6	17, 28	129	echo of request
12	2	Pattern Control Block				

12	3	Pattern Mask				
20	0	Binary Counter - All Variations	1, 7, 8, 9, 10	06		
20	1	32-Bit Binary Counter			129, 130	00, 01
20	2	16-Bit Binary Counter			129, 130	00, 01
20	3	32-Bit Delta Counter				
20	4	16-Bit Delta Counter				
20	5	32-Bit Binary Counter without Flag			129, 130	00, 01
20	6	16-Bit Binary Counter without Flag			129, 130	00, 01
20	7	32-Bit Delta Counter without Flag				
20	8	16-Bit Delta Counter without Flag				
21	0	Frozen Counter - All Variations	1	06		
21	1	32-Bit Frozen Counter			129, 130	00, 01
21	2	16-Bit Frozen Counter			129, 130	00, 01
21	3	32-Bit Frozen Delta Counter				
21	4	16-Bit Frozen Delta Counter				
21	5	32-Bit Frozen Counter with Time of Freeze				
21	6	16-Bit Frozen Counter with Time of Freeze				
21	7	32-Bit Frozen Delta Counter with Time of Freeze				
21	8	16-Bit Frozen Delta Counter with Time of Freeze				
21	9	32-Bit Frozen Counter without Flag			129, 130	00, 01
21	10	16-Bit Frozen Counter without Flag			129, 130	00, 01
21	11	32-Bit Frozen Delta Counter without Flag				
21	12	16-Bit Frozen Delta Counter without Flag				
22	0	Counter Change Event - All Variations	1	06,07,08		
22	1	32-Bit Counter Change Event without Time			129, 130	17, 28
22	2	16-Bit Counter Change Event without Time			129, 130	17, 28
22	3	32-Bit Delta Counter Change Event without Time				
22	4	16-Bit Delta Counter Change Event without Time				
22	5	32-Bit Counter Change Event with Time				
22	6	16-Bit Counter Change Event with Time				
22	7	32-Bit Delta Counter Change Event with Time				
22	8	16-Bit Delta Counter Change Event with Time				
23	0	Frozen Counter Event - All Variations				
23	1	32-Bit Frozen Counter Event without Time				
23	2	16-Bit Frozen Counter Event without Time				
23	3	32-Bit Frozen Delta Counter Event without Time				
23	4	16-Bit Frozen Delta Counter Event without Time				
23	5	32-Bit Frozen Counter Event with Time				
23	6	16-Bit Frozen Counter Event with Time				

23	7	32-Bit Frozen Delta Counter Event with Time				
23	8	16-Bit Frozen Delta Counter Event with Time				
30	0	Analog Input - All Variations	1	06		
30	1	32-Bit Analog Input			129, 130	00, 01
30	2	16-Bit Analog Input			129, 130	00, 01
30	3	32-Bit Analog Input without Flag			129, 130	00, 01
30	4	16-Bit Analog Input without Flag			129, 130	00, 01
30	5	Short Floating Point Analog Input			129, 130	00, 01
31	0	Frozen Analog Input - All Variations				
31	1	32-Bit Frozen Analog Input				
31	2	16-Bit Frozen Analog Input				
31	3	32-Bit Frozen Analog Input with Time of Freeze				
31	4	16-Bit Frozen Analog Input with Time of Freeze				
31	5	32-Bit Frozen Analog Input without Flag				
31	6	16-Bit Frozen Analog Input without Flag				
32	0	Analog Change Event - All Variations	1	06,07,08		
32	1	32-Bit Analog Change Event without Time			129,130	17,28
32	2	16-Bit Analog Change Event without Time			129,130	17,28
32	3	32-Bit Analog Change Event with Time			129,130	17,28
32	4	16-Bit Analog Change Event with Time			129,130	17,28
32	5	Short Floating Point Analog Change Event without Time			129,130	17,28
33	0	Frozen Analog Event - All Variations				
33	1	32-Bit Frozen Analog Event without Time				
33	2	16-Bit Frozen Analog Event without Time				
33	3	32-Bit Frozen Analog Event with Time				
33	4	16-Bit Frozen Analog Event with Time				
40	0	Analog Output Status - All Variations	1	06		
40	1	32-Bit Analog Output Status			129, 130	00, 01
40	2	16-Bit Analog Output Status			129, 130	00, 01
40	3	Short Floating Point Analog Output Status			129, 130	00, 01
41	0	Analog Output Block - All Variations				
41	1	32-Bit Analog Output Block	3, 4, 5, 6	17, 28	129	echo of request
41	2	16-Bit Analog Output Block	3, 4, 5, 6	17, 28	129	echo of request
41	3	Short Floating Point Analog Output Block	3, 4, 5, 6	17, 28	129	echo of request
50	0	Time and Date - All Variations				
50	1	Time and Date	2 (see 4.14)	07 where quantity = 1		
50	2	Time and Date with Interval				
51	0	Time and Date CTO - All Variations				

51	1	Time and Date CTO			129, 130	07, quantity=1
51	2	Unsynchronized Time and Date CTO			129, 130	07, quantity=1
52	0	Time Delay - All Variations				
52	1	Time Delay Coarse			129	07, quantity=1
52	2	Time Delay Fine			129	07, quantity=1
60	0					
60	1	Class 0 Data	1	06		
60	2	Class 1 Data	1 20,21	06,07,08 06		
60	3	Class 2 Data	1 20,21	06,07,08 06		
60	4	Class 3 Data	1 20,21	06,07,08 06		
70	1	File Identifier				
80	1	Internal Indications	2	00 index=7		
81	1	Storage Object				
82	1	Device Profile				
83	1	Private Registration Object				
83	2	Private Registration Object Descriptor				
90	1	Application Identifier				
100	1	Short Floating Point				
100	2	Long Floating Point				
100	3	Extended Floating Point				
101	1	Small Packed Binary-Coded Decimal				
101	2	Medium Packed Binary-Coded Decimal				
101	3	Large Packed Binary-Coded Decimal				
		No Object	13			
		No Object	14			
		No Object	23 (see 4.14)			

DNP3 Master Device Profile Document

DNP v3.00
DEVICE PROFILE DOCUMENT

Vendor Name: Control Microsystems Inc.

Device Name: SCADAPack controllers

Highest DNP Level Supported:	Device Function:
For Requests 2 For Responses 2	■ Master ☐ Slave

Notable objects, functions, and/or qualifiers supported in addition to the Highest DNP Levels Supported (the complete list is described in the attached table):

- Function code 14 (warm restart)
- Function code 20 (Enable Unsolicited Messages) for class 1, 2, 3 objects only.
- Function code 21 (Disable Unsolicited Messages) for class 1, 2, 3 objects only.
- Object 41, variation 1 (32-bit analog output block)

Maximum Data Link Frame Size (octets):	Maximum Application Fragment Size (octets):
Transmitted 292 Received (must be 292)	Transmitted 2048 Received 2048

Maximum Data Link Re-tries:	Maximum Application Layer Re-tries:
☐ None ☐ Fixed at ■ Configurable, range 0 to 255	☐ None ■ Configurable, range 0 to 255

Requires Data Link Layer Confirmation:

☐ Never
☐ Always
☐ Sometimes If 'Sometimes', when?

■ Configurable for Always or Never

Requires Application Layer Confirmation:

☐ Never
☐ Always (not recommended)
☐ When reporting Event Data (Slave devices only)
☐ When sending multi-fragment responses (Slave devices only)

☐ Sometimes If 'Sometimes', when?

■ Configurable for always or only when Reporting Event Data and Unsolicited Messages

Timeouts while waiting for:

Data Link Confirm	☐ None	☐ Fixed at _____	☐ Variable	■ Configurable
Complete Appl. Fragment	☐ None	☐ Fixed at _____	☐ Variable	■ Configurable
Application Confirm	☐ None	☐ Fixed at _____	☐ Variable	■ Configurable
Complete Appl. Response	☐ None	☐ Fixed at _____	☐ Variable	■ Configurable

Others _____

Sends/Executes Control Operations:

WRITE Binary Outputs	☐ Never	☐ Always	☐ Sometimes	■ Configurable
SELECT/OPERATE	☐ Never	☐ Always	☐ Sometimes	■ Configurable
DIRECT OPERATE	☐ Never	☐ Always	☐ Sometimes	■ Configurable
DIRECT OPERATE - NO ACK	☐ Never	☐ Always	☐ Sometimes	■ Configurable
Count > 1	☐ Never	☐ Always	☐ Sometimes	■ Configurable

Pulse On	☐ Never	☐ Always	☐ Sometimes	■ Configurable
Pulse Off	☐ Never	☐ Always	☐ Sometimes	■ Configurable
Latch On	☐ Never	☐ Always	☐ Sometimes	■ Configurable
Latch Off	☐ Never	☐ Always	☐ Sometimes	■ Configurable
Queue	■ Never	☐ Always	☐ Sometimes	☐ Configurable
Clear Queue	■ Never	☐ Always	☐ Sometimes	☐ Configurable

FILL OUT THE FOLLOWING ITEM FOR MASTER DEVICES ONLY:

Expects Binary Input Change Events:

■ Either time-tagged or non-time-tagged for a single event
☐ Both time-tagged and non-time-tagged for a single event
☐ Configurable (attach explanation)

FILL OUT THE FOLLOWING ITEMS FOR SLAVE DEVICES ONLY:

Reports Binary Input Change Events when no specific variation requested: ☐ Never ☐ Only time-tagged ☐ Only non-time-tagged ☐ Configurable to send both, one or the other (attach explanation)	Reports time-tagged Binary Input Change Events when no specific variation requested: ☐ Never ☐ Binary Input Change With Time ☐ Binary Input Change With Relative Time ☐ Configurable (attach explanation)
Sends Unsolicited Responses: ☐ Never ☐ Configurable by class ☐ Only certain objects ☐ Sometimes (attach explanation) ■ ENABLE/DISABLE UNSOLICITED	Sends Static Data in Unsolicited Responses: ☐ Never ☐ When Device Restarts ☐ When Status Flags Change No other options are permitted.
Default Counter Object/Variation: ☐ No Counters Reported ☐ Configurable (attach explanation) ☐ Default Object 20 Default Variation 05	Counters Roll Over at: ☐ No Counters Reported ☐ Configurable (attach explanation) ☐ 16 Bits ☐ 32 Bits
☐ Point-by-point list attached	☐ 16 Bits for 16-bit counters 32 Bits for 32-bit counters ☐ Point-by-point list attached

Sends Multi-Fragment Responses: ■ Yes ☐ No

DNP Configuration Information

Poll Offset Example

The Poll Offset parameter enhances the control over timing of master poll messages, by allowing master poll messages to be staggered.

For example, a master station may have 10 slaves to poll, and needs to poll them every hour. If these are included in the poll table without any poll offset, they will all be polled in quick succession on the hour – resulting in a large burst of communication activity once per hour. On some types of communications networks (particularly radio) it is desirable to distribute communication load more evenly, to minimize the chance of collisions and to avoid the possibility of consuming bandwidth continuously for an extended period of time.

The poll offset parameter enables you to distribute the communication load evenly. In the above example, it is possible to stagger the master polls so slave stations are polled at 6-minute intervals. To do this, set the base poll interval to 10 seconds, and for each slave station set the poll rate and poll offset parameters as follows:

Base Poll (seconds)	Poll Rate (seconds)	Poll Offset (seconds)
10	360	0
10	360	36
10	360	72
10	360	108
10	360	144
10	360	180
10	360	216
10	360	252
10	360	288
10	360	324

Valid Modbus Addresses for DNP Points

The Modbus Address parameter specifies the Modbus address of data points assigned to DNP Addresses. The SCADAPack and Micro16 controllers use Modbus addressing for all data points. The valid Modbus address depends on the controller type as seen in the table below.

Controller Type	Digital Input	Digital Output	Analog Input	Analog Output
SCADAPack 100:256K	10001 to 14096	00001 to 04096	30001 to 30512	40001 to 44000
Micro16	10001 to 14096	00001 to 04096	30001 to 31024	40001 to 49999

SCADAPack	10001 to 14096	00001 to 04096	30001 to 31024	40001 to 49999
SCADAPack Light	10001 to 14096	00001 to 04096	30001 to 31024	40001 to 49999
SCADAPack Plus	10001 to 14096	00001 to 04096	30001 to 31024	40001 to 49999
SCADAPack LP	10001 to 14096	00001 to 04096	30001 to 31024	40001 to 49999
SCADAPack 32	10001 to 14096	00001 to 04096	30001 to 39999	40001 to 49999
SCADAPack 32P	10001 to 14096	00001 to 04096	30001 to 39999	40001 to 49999
SCADAPack 314	10001 to 14096	00001 to 04096	30001 to 39999	40001 to 49999
SCADAPack 330/334	10001 to 14096	00001 to 04096	30001 to 39999	40001 to 49999
SCADAPack 350/357	10001 to 14096	00001 to 04096	30001 to 39999	40001 to 49999
SCADAPack 4202	10001 to 14096	00001 to 04096	30001 to 31024	40001 to 49999
SCADAPack 4203	10001 to 14096	00001 to 04096	30001 to 39999	40001 to 49999

Store and Forward

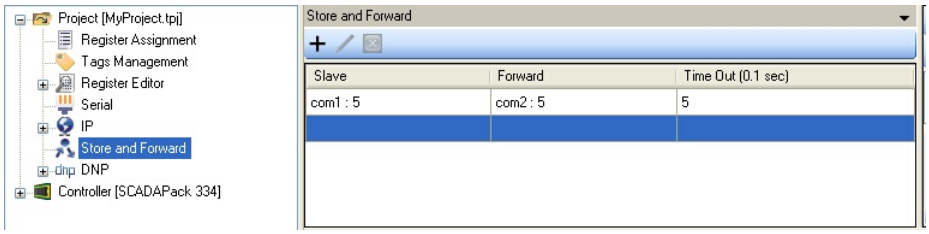
The **Store and Forward** command is used to configure store and forward messaging. A controller configured for store and forward operation receives messages destined for a remote Slave Station on the Slave Interface. The controller forwards the message on the Forward Interface to the Forward Station. Each entry in the Store and Forward view grid is called a Store and Forward translation.

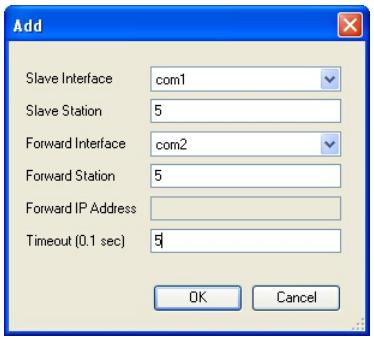
The Store and Forward translation is bi-directional. This means that responses received from the remote slave station are processed in the reverse order in the translation table.

The Slave Interface and the Forward Interface need to have Store and Forward enabled.

This command is available only when the controller type is set to SCADAPack 314, SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32, SCADAPack 32P or SCADAPack 4203.

When the Store and Forward command is selected the Store and Forward grid view is displayed.





The Store and Forward view displays a grid containing six columns and initially a blank Store and Forward entry row. When Store and Forward translations are entered a new blank row is created and is always the last row on the grid. The Store and Forward grid maybe sorted by each column heading in the grid.

The Store and Forward grid may have up to 128 translation entries. A vertical scroll bar is used if the list exceeds the window size.

The **Slave Interface** field contains the slave interface the message is received from for this translation. Messages received on the Slave Interface will be forwarded out the Forward Interface. The drop down list allows the following selections:

- com1
- com2
- com3
- com4
- LAN/PPP

The selections displayed will depend on the controller type used in the project.

The **Slave Station** field contains the Modbus station address of the message received on the Slave Interface. This station address needs to be different from the Interface station address. Valid range for Slave Station is:

- 1 to 255 when standard addressing is selected for the interface.
- 1 to 65534 when extended addressing is selected for the interface.

The **Forward Interface** field contains the interface the message is forwarded from. The Forward Interface may be the same as the Slave Interface. When forwarding to a Modbus TCP or UDP network, the protocol type is selected for the Forward Interface.

The drop down list allows the following selection:

- com1
- com2
- com3
- com4
- Modbus/TCP
- Modbus RTU in UDP
- Modbus ASCII in UDP

The selections displayed will depend on the controller type used in the project.

The **Forward Station** field contains the Modbus station address of the forwarded message. This station address needs to be different from the Forward Interface station address but may be the same as the Slave Interface station address. Valid range for Forward Station is:

- 1 to 255 when standard addressing is selected for the interface.
- 1 to 65534 when extended addressing is selected for the interface.

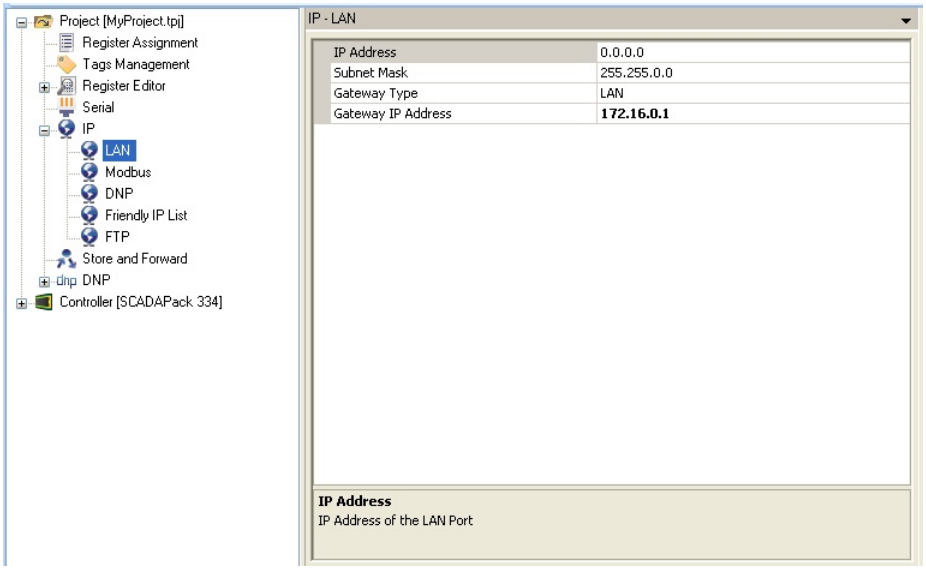
The **Forward IP Address** field contains the IP address of the Forward Station. This field is set to 000.000.000.000 unless a TCP or UDP network is selected for Forward Interface. Valid entries are 1 to 255 for the first byte in the IP address and 0 to 255 for the remaining three bytes.

The **Time Out** field displays the maximum time (in tenths of seconds) the forwarding task waits for a valid response from the Forward Station. The time out should be equal to or less than the time out set for the master message received on the Slave Interface. Valid entries are 0 to 65535.

IP Settings

The IP Settings include settings for LAN, Modbus, DNP, Friendly IP List and FTP parameters. This view is opened from the Communications group on the Configure ribbon or from the Tree view. This view is used to configure the IP port on the controller.

This module is available only when the controller type is set to SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P.



The LAN Tab allows configuration of the Ethernet interface of a SCADAPack controller

The Modbus Tab allows configuration of all Modbus related parameters of the IP stack.

The DNP Tab allows a user to configured DNP TCP and DNP UDP protocols.

The PPP Interface Tab allows configuration of the PPP protocol on controller serial ports. This Tab is only visible if the controller type is SCADAPack 32.

The PPP Accounts Tab is used to configure the username and password list for PPP login authentication. This Tab is only visible if the controller type is SCADAPack 32.

The Friendly IP List Tab is used to create a list of friendly IP addresses that he controller will respond to.

The DHCP Tab is used to configure the DHCP (Dynamic Host Configuration Protocol) server. This tab is only visible when the controller type is FlowStation 110.

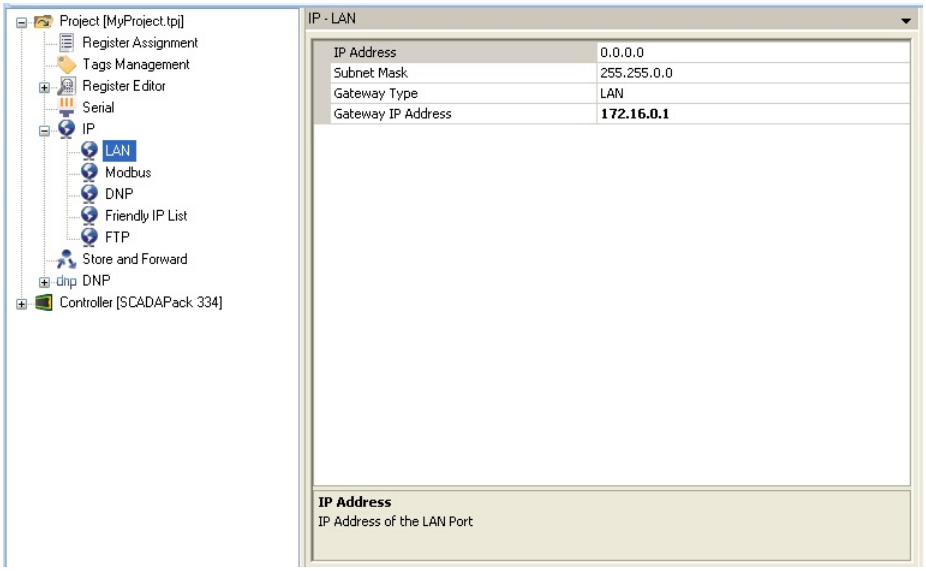
The FTP tab configures the FTP (File Transfer Protocol) server in the controller.

LAN Tab

The **LAN** tab allows configuration of the Ethernet interface of a SCADAPack controller. You configure the IP Address, Subnet Mask Gateway Type and Gateway IP Address using this tab.

IP Address

To configure the Ethernet port, the controller IP Address, Subnet Mask and Gateway Type and Gateway IP Address need to be entered in the LAN Port configuration data table.



The **IP Address** is the address of the controller LAN port. The IP address is statically assigned. Contact your network administrator to obtain an IP address for the controller. The default value is 0.0.0.0.

The **Subnet Mask** determines the subnet on which the controller LAN port is located. The subnet mask is statically assigned. Contact your network administrator to obtain the subnet mask for the controller. The default value is 255.255.0.0.

The **Gateway Type** specifies the interface type used for IP communication. The SCADAPack 32 and SCADAPack 32P support Point-To-Point Protocol (PPP) on the serial ports. See the PPP Interfaces and PPP Accounts tabs to configure the serial ports for PPP communication. When the controller type is SCADAPack 32 or SCADAPack 32P the Gateway Type selection is one of:

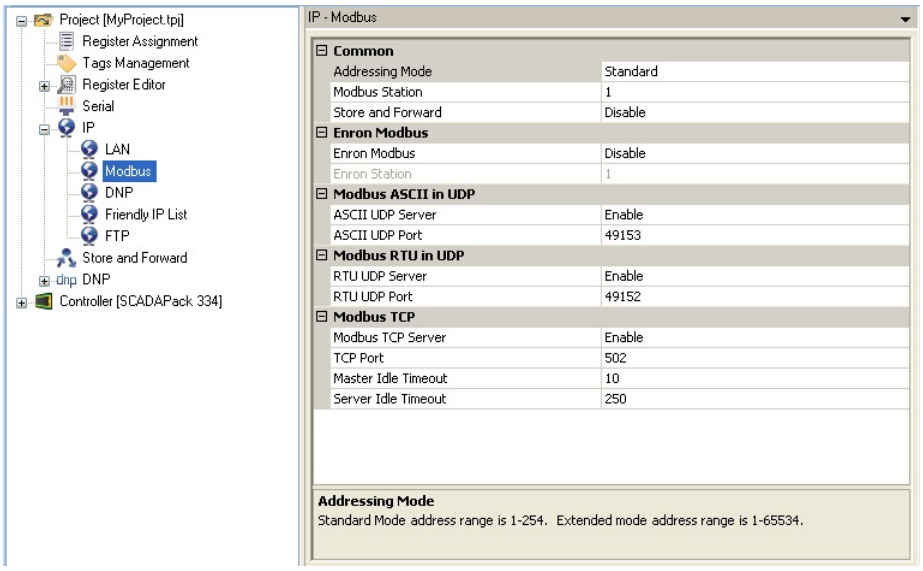
- LAN
- Com1 (SCADAPack 32 and 32P controllers)
- Com2 (SCADAPack 32 and 32P controllers)
- Com3 (SCADAPack 32 controller only)
- Com4 (SCADAPack 32 and 32P controllers)

PPP support is not available on the SCADAPack 330, SCADAPack 334 and SCADAPack 350 controllers. The headings for com 1 through com 4 are not displayed. Only the LAN option is available for these controllers.

The **Gateway** determines how your controller communicates with devices outside its subnet. The **LAN** radio button selects the gateway specified in the **LAN** edit box. Enter the IP address of the gateway. The gateway is located on the LAN port subnet. The gateway is statically assigned. Contact your network administrator to obtain the gateway IP address. The default value is 0.0.0.0.

Modbus Tab

The **Modbus** tab allows configuration of all Modbus related parameters of the IP stack.



Modbus Common

The **Addressing** selection selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default value is standard.

The **Station** selection sets the station number of the controller. The valid range is 1 to 255 if standard addressing is used and 1 to 65534 if extended addressing is used. The default value is 1.

The **Store and Forward** selection controls forwarding of messages using IP based protocols. If this option is enabled, messages will be forwarded according to the settings in the store and forward routing table. The default value is disabled.

Enron Modbus

The **Enron Modbus** box selects whether Enron Modbus is enabled for the port. If enabled, the controller, in addition to regular Modbus messages, will handle Enron Modbus messages. The default value is disabled.

The **Enron Station** box selects the Enron Modbus station address. The valid range for Enron Station is 1 to 255 if the Addressing control is set to Standard. The valid range for Enron Station is 1 to 65534 if the Addressing control is set to Extended. The Enron station needs to be different from the Modbus station set in the **Station** edit box. This ensures Enron Modbus and Modbus communication can occur on the same port.

Modbus ASCII in UDP

The **ASCII UDP Port** sets the port used by the protocol. Valid port number range is 1 to 65534. The default value is 49153. This is a recommendation only. Consult your network administrator to obtain a port if you are not using the default.

The **ASCII UDP Server** selection selects whether the server is enabled. If this option is enabled the controller supports incoming slave messages. Disabling this option prevents the controller from processing slave messages. Master messaging is always enabled.

Modbus RTU in UDP

The **RTU UDP Port** sets the port used by the protocol. Valid port number range is 1 to 65534. The default value is 49152. This is a recommendation only. Consult your network administrator to obtain a port if you are not using the default.

The **RTU UDP Server** selection selects whether the server is enabled. If this option is enabled the controller supports incoming slave messages. Disabling this option prevents the controller from processing slave messages. Master messaging is always enabled.

Modbus/TCP

The **Master Idle Timeout** determines when connections to a slave controller are closed. Setting this value to zero disables the timeout; the connection will be closed only when your program closes it. Any other value sets the timeout in seconds. The connection will be closed if no messages are sent in that time. This allows the slave device to free unused connections. Valid timeout range is 0 to 4294967295 seconds. The default value is 10 seconds.

The **Modbus TCP Server** selection selects whether the server is enabled. If this option is enabled the controller supports incoming slave messages. Disabling this option prevents the controller from processing slave messages. Master messaging is always enabled.

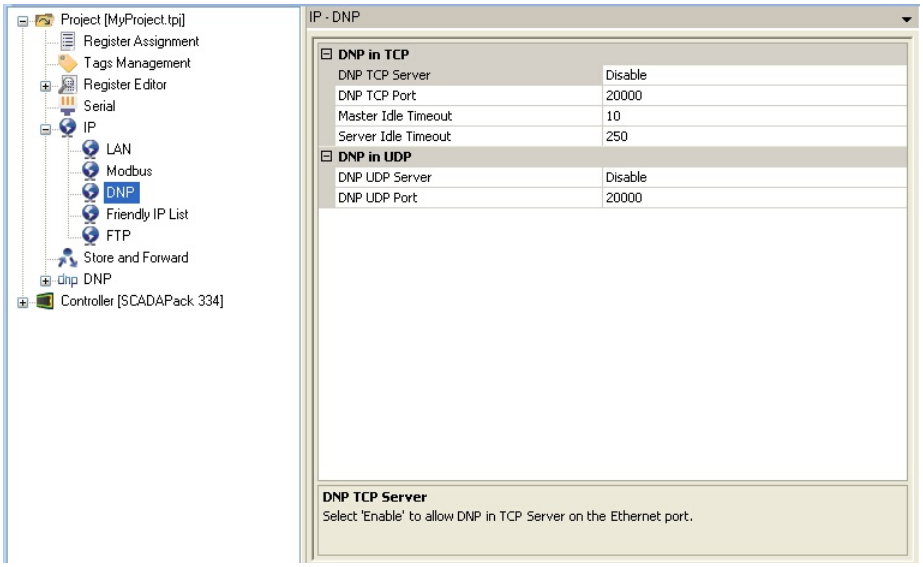
The **Server Idle Timeout** determines when connections from a remote device are closed. Setting this value to zero disables the timeout; the connection will be closed only when the remote device closes it. Any other value sets the timeout in seconds. The connection will be closed if no messages are received in that time. This allows the controller to free unused connections. Valid timeout range is 0 to 4294967295 seconds. The default value is 250 seconds.

The **TCP Port** sets the port used by the Modbus/TCP protocol. In almost all cases this should be set to 502. This is the well-known port number for Modbus/TCP. Modbus/TCP devices use 502 by default, and on many devices the value cannot be changed. It is suggested that you change this value only if this port is used by another service on your network. Valid port number range is 1 to 65534. Consult your network administrator to obtain a port if you are not using the default.

Each major category in the grid can be collapsed or expanded.

DNP Tab

The **DNP** tab allows a user to configured DNP TCP and DNP UDP protocols.



DNP in TCP

The **DNP TCP Server** selection selects whether the DNP in TCP protocol is enabled. If this option is enabled the controller supports DNP in TCP protocol. Disabling this option prevents the controller from processing DNP in TCP protocol messages. Master messaging is enabled. The default selection is disabled.

The **DNP TCP Port** sets the port used by the DNP in TCP protocol. Valid port number range is 1 to 65534. The default value is 20000. Consult your network administrator to obtain a port if you are not using the default.

The **Master Idle Timeout** determines when connections to a slave controller are closed. Setting this value to zero disables the timeout; the connection will be closed only when your program closes it. Any other value sets the timeout in seconds. The connection will be closed if no messages are sent in that time. This allows the slave device to free unused connections. Valid timeout range is 0 to 4294967295 seconds. The default value is 10 seconds.

The **Server Idle Timeout** determines when connections from a remote device are closed. Setting this value to zero disables the timeout; the connection will be closed only when the remote device closes it. Any other value sets the timeout in seconds. The connection will be closed if no messages are received in that time. This allows the controller to free unused connections. Valid timeout range is 0 to 4294967295 seconds. The default value is 250 seconds.

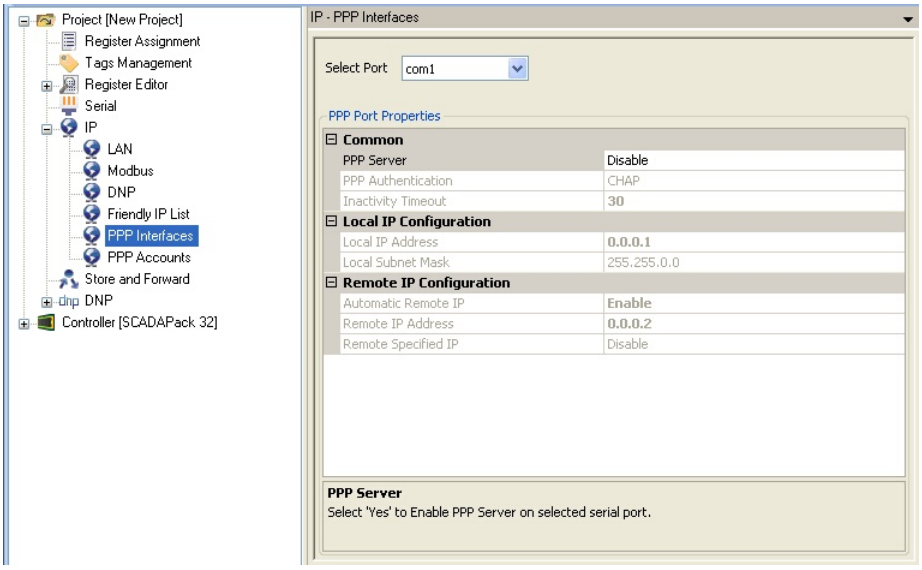
DNP in UDP

The **DNP UDP Server** selection selects whether the DNP in UDP protocol is enabled. If this option is enabled the controller supports DNP in UDP protocol. Disabling this option prevents the controller from processing DNP in UDP protocol messages and sending DNP in UDP master messages. The default selection is disabled.

The **UDP Port** sets the port used by the DNP in UDP protocol. Valid port number range is 1 to 65534. The default value is 20000. Consult your network administrator to obtain a port if you are not using the default.

PPP Interfaces Tab

The **PPP Interfaces** tab allows configuration of the PPP protocol on controller serial ports.



Select Port

Select the serial port from the **Select Port** combo box to configure PPP on that respective port. The list of available serial ports depends on the selected controller type.

- Com1 (SCADAPack 32 and 32P controllers)

- Com2 (SCADAPack 32 and 32P controllers)
- Com3 (SCADAPack 32 controller only)
- Com4 (SCADAPack 32 and 32P controllers)

Common

The **PPP Server** selection enables or disables the PPP Server on this serial port.

Select **Enable** if you want to allow a remote PPP client to connect to this port. This selection enables the remaining settings in the page.

Select **Disable** if you want to not allow a remote PPP client to connect to this port. This selection disables the remaining settings in the page.

The **PPP Authentication** selection determines the login protocol used at the start of every PPP connection.

- The **None** selection removes the login step.
- The **PAP** selection selects the Password Authentication Protocol (PAP).
- The **CHAP** selection selects the Challenge-Handshake Authentication Protocol (CHAP).

PAP and CHAP usernames and passwords are configured on the **PPP Accounts** tab.

The **Inactivity Timeout** is the inactivity timeout for this serial port. If there has been no activity on an existing PPP connection for the selected number of minutes, then the connection is automatically closed. If there is a modem connected it is hung up. Setting this value to zero disables the timeout.

Local IP Configuration

The **Local IP Address** is the address of the selected serial port. The Local IP address is statically assigned. Contact your network administrator to obtain an IP address for this serial port. The local IP Address on each PPP-enabled serial port needs to be unique. If a duplicate IP address is assigned to a serial port, an error message is displayed.

The **Local Subnet Mask** determines the subnet on which this serial port is located. The subnet mask is statically assigned. Contact your network administrator to obtain the subnet mask for this serial port. In a standard PPP configuration, a subnet mask of 255.255.255.255 is used to restrict routing on this serial port to a single host (i.e., the **Remote IP Address**).

If another subnet mask is used, all packets on that subnet will be forwarded to this serial port. Any address on that subnet in addition to the **Remote IP Address** can be used for the remote host in this case.

Remote IP Configuration

The **Automatic Remote IP** selection enables or disables the automatic selection of the Remote IP Address.

- Select **Disable** to enable entering the Remote IP Address that will be assigned to the remote client.
- Select **Enable** to automatically select the IP address to be the serial port's IP address + 1.

The **Remote IP Address** is the address that will be assigned to the remote PPP client connected to this serial port. Enter the IP address to assign to the remote client. This entry is available only when the Automatic Remote IP selection is set to **Disable**.

Set to Enable, the **Automatic** radio button automatically selects the address to be the serial port's IP address + 1.

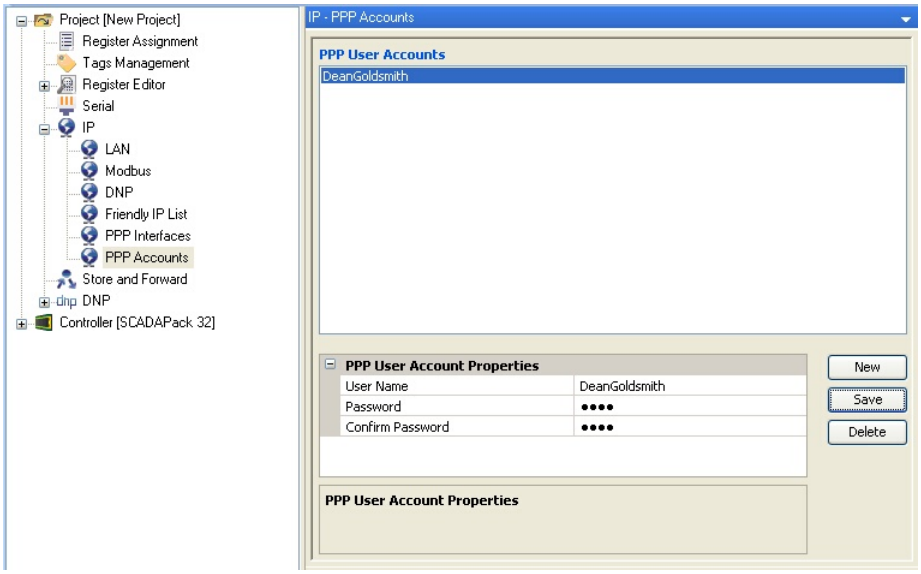
The **Remote Specified IP Address** selection enables or disables allowing the remote client to specify its own IP address.

- Select Disable to force the remote PPP client to use the entered Remote IP Address.
- Select Enable to allow the remote PPP client to assign its own IP address.

The client may or may not request its own IP address. If the client does not make this request, the PPP Server will assign the IP address in the **Remote IP Address** field.

PPP Accounts Tab

The **PPP Accounts** tab is used to configure the username and password list for PPP login authentication. The list is used only by those serial ports configured for the PPP protocol using PAP or CHAP authentication.



PPP User Accounts

This section of the PPP Accounts view displays the active PPP User Accounts. The maximum number of PPP accounts that can be added is 20. Once this limit is reached, the **Add Account** button is disabled.

PPP User Account Properties

This section of the PPP Accounts view is used to manage PPP user accounts. User accounts are created and removed using this section of the view.

The **User Name** edit box selects the User Name. A user name is any alphanumeric string 1 to 16 characters in length, and is case sensitive.

The **Password** edit box selects the password. A password is any alphanumeric string 1 to 16 characters in length, and is case sensitive.

The **Confirm Password** edit box selects the verify password. Enter the same string entered for the password.

Select the **Add Account** button to add the account entered in the PPP User Account Properties. The User Name is added to the PPP User Accounts list.

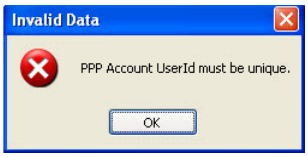
Select the **Remove Account** to remove the account entered in the PPP User Account Properties. The User Name is removed from the PPP User Accounts list.

Select the **Clear Selection** to clear the information in the PPP User Account Properties. This does not remove the account, only removes the entries in the account properties.

Add a New Account

If an account is currently selected in the PPP User Account Properties grid, click on Clear Selection.

Enter the new User Name, Password and Confirm Password into the PPP User Account Property grid. The User Names need to be unique. If the User Name is already in use the following error message is displayed.



If an error is not presented after entering the login information, click the **Add Account** button. The newly created account will be displayed in the PPP User Accounts' list box, confirming successful configuration.

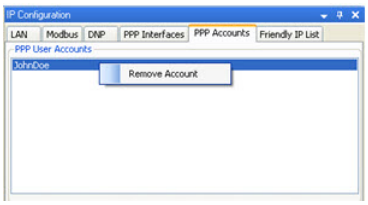
The maximum number of PPP accounts that can be added is 20. Once this limit is reached, the **Add Account** button is disabled.

Remove an Account

To delete a PPP User account, select the desired account from PPP User Accounts list box.

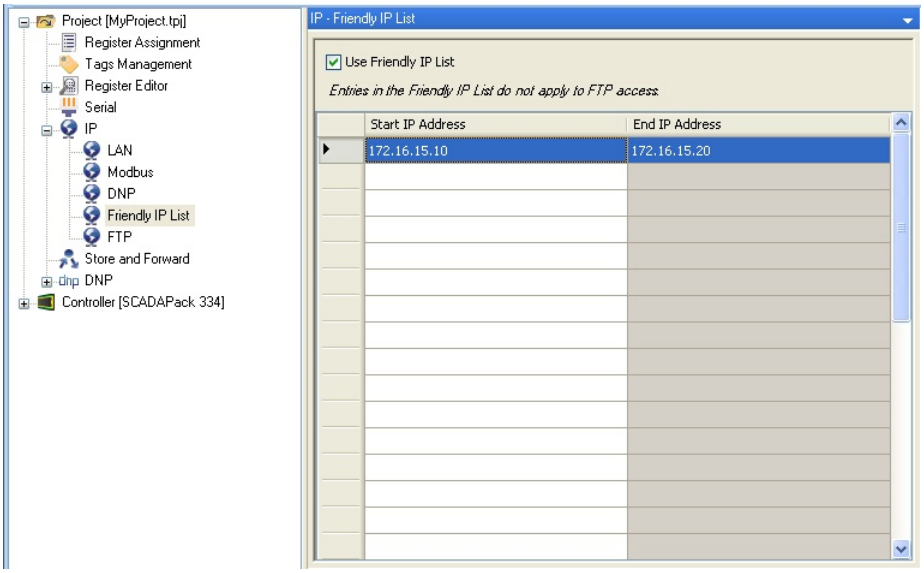
Click on **Remove Account** button. This button is disabled if there are no user accounts displayed in the PPP User Accounts list box.

Alternatively, right click on account from the PPP User Accounts list box and select **Remove Account** from the Context Sensitive menu as shown below.



Friendly IP List Tab

The **Friendly IP** data table provides a method to reject messages from any source that is not entered in the Friendly IP address table. A maximum of 32 friendly IP entries may be entered.



Each Friendly IP entry is listed as a row in the above table, with starting and ending friendly IP addresses represented in their respective columns.

The **Use Friendly IP List** checkbox enables or disables the friendly IP list. Check this box to accept messages from only the IP addresses in the list. Uncheck this to accept message from all IP addresses.

Entering a Friendly IP Address List

To enter a Friendly IP entry:

- Enter a valid IP address within the Start IP Address column.

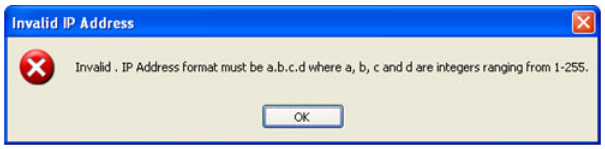
The **End IP Address** column is initially a Read-only field, until a valid starting IP address has been entered.

- Tab across to the End IP Address column of this row.

The **End IP Address** is initially set to the Start IP address. This value can be changed if required. The **End IP Address** needs to be greater than the **Start IP Address**.

Editing a Friendly IP Entry

Each IP address within the Friendly IP List table can be edited. If an invalid IP address format is specified, the following message is displayed:



Click **OK** to the data grid and make the appropriate changes.

If a valid IP address format is specified but falls within the range of already configured friendly lists, the following message is displayed:

Click on **OK** to return to the data grid, and make the appropriate changes.

The above error message is also displayed if the **End IP Address** is not valid, say less than the **Start IP Address**.

The **End IP Address** remains Read Only unless a valid **Start IP Address** has been entered.

Deleting a Friendly IP Entry

A valid or complete Friendly IP entry includes a Start IP address AND End IP Address.

An incomplete or invalid Friendly IP entry can be one of the following:

Missing Start IP Address

Missing End IP Address

Typically, to enter a Friendly IP entry, the Start IP address is entered first, followed by the End IP Address. Several simple rules have been put in place to prevent an incomplete Friendly IP List from being accepted as part of the project configuration settings.

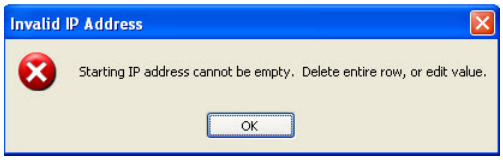
The rules are:

The End IP Address column in a row is initially set to 'Read Only', unless there is a valid Start IP address in the corresponding row.

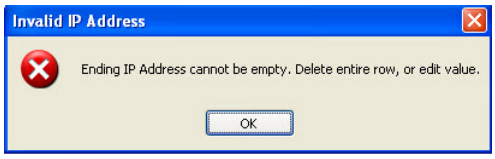
The Start or End IP address needs to be valid using format www.xxx.yyy.zzz

Once entered, the Start IP address and End IP addresses cannot be deleted independently. The entire row needs to be deleted.

If deleting the Start IP address is attempted, the following message is displayed and the previous value returned to the cell.

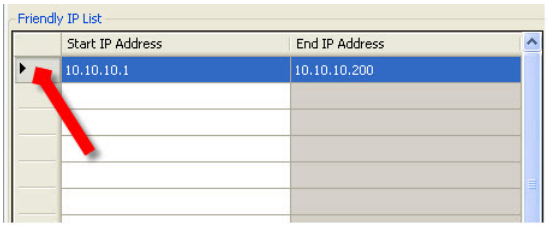


If deleting the End IP address is attempted, the following message is displayed and the previous value returned to the cell.



To delete a friendly IP list entry, highlight the entire row and select Delete. You can highlight the entire row by clicking on the row header as shown below.

Selected Friendly IP address may be deleted by pressing the delete key on the PC keyboard.



Error Messages

IP Configuration module errors are self explanatory. However, the following table list the errors produced by this module and suggested corrective actions:

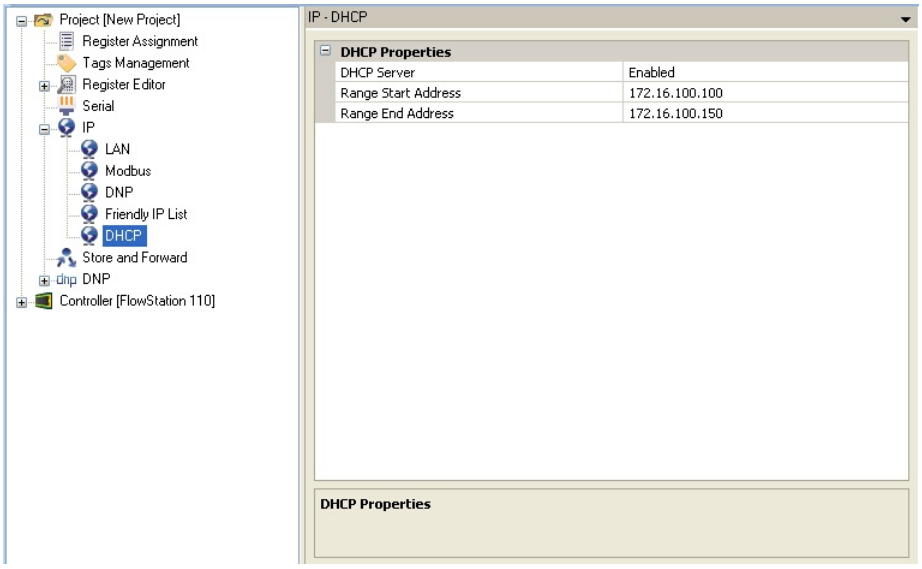
Error	Corrective Action
The controller has different IP settings. Are you sure you want to overwrite them.	Click **Yes** or **No** depending on action desired. You can also check the checkbox to always overwrite the controller's IP settings.
Invalid IP Address. IP Address format needs to be www.xxx.yyy.zzz	Specify the correct IP address, e.g., 10.10.10.1.
Invalid Modbus Station. Valid Modbus Station range is 1-255 for Standard or 1-65534 for Extended Addressing.	If the 'Addressing Mode' property is set to **Standard**, valid Modbus and Enron Modbus address range goes from 1 to 255. If 'Addressing Mode' is set to **Extended**, valid Modbus and Enron Modbus address range goes from 1 to 65534.
Invalid [Protocol] Port. Valid range is from 1-65534.	Enter proper protocol port.
Enron and Modicon Modbus addresses cannot be same	Make Enron and Modicon (regular) station addresses different.
A maximum of 32 friendly IP entries are allowed.	Avoid entering more than 32 Friendly IP entries. A friendly IP entry comprise of a Start and End Friendly IP address in one row of the Friendly IP table.
Start IP address cannot be empty. Delete entire row, or edit value.	You attempted to delete the 'Start IP Address' of a valid Friendly IP entry. This is not allowed. Either change the 'Start IP Address' of if you want to delete, you need to delete the entire row comprising the Friendly IP entry.
End IP Address cannot be empty. Delete entire row, or edit value.	You attempted to delete the 'End IP Address' of a valid Friendly IP entry. This is not allowed. Either change the 'End IP Address' of if you want to delete, you need to delete the entire row comprising the Friendly IP entry.
Proposed Friendly IP address overlaps existing range.	The proposed Start or End Friendly IP address entered overlaps an existing range.
Valid Inactivity Timeout. Inactivity Timeout range is 1 to 65534.	Enter value within specified range.
The IP Address of each serial port with PPP Server enabled must be unique.	Each serial port with PPP enabled needs to have its own unique IP address. Enter a unique local or remote IP address if enabling PPP on a serial port.
Invalid PPP login credentials entered. PPP Login credentials must be between 1 and 16 alphanumeric characters.	PPP username and passwords are restricted to 16 alphanumeric characters. In this context, alphanumeric characters include all letters of the English alphabet (Lower and Upper case), and 0 to 9, punctuation e.g., '.', ',', and so on.
PPP Account UserId must be unique.	Each PPP account Userid needs to be unique, as this is how accounts are distinguished by the PLC.
PPP Account login credentials cannot be blanks! An account has not been created.	You clicked on the 'Add Account' button before entering valid login credentials into the PPP User Accounts property grid. To add an account, enter valid login credentials, before clicking on 'Add Account'.
Both passwords must be same before	An attempt was made to change the

changes take effect. Please re-type both passwords and confirm.	password of an existing account, but the new password proposed does not match the one entered in the 'Confirm Password' field. For any changes to take effect, both passwords need to be same.

DHCP Tab

The **DHCP** tab allows configuration of the DHCP server for the **FlowStation**. This tab is only visible when the controller type is set for **FlowStation**. The server is enabled, or disabled and the Start and End address ranges are specified using this tab.

The DHCP services are used to allocate an IP address to the FlowStation local display. The FlowStation is not a router to the outside world for general web access, so no additional information is required from DHCP for connection beyond the FlowStation Ethernet interface.



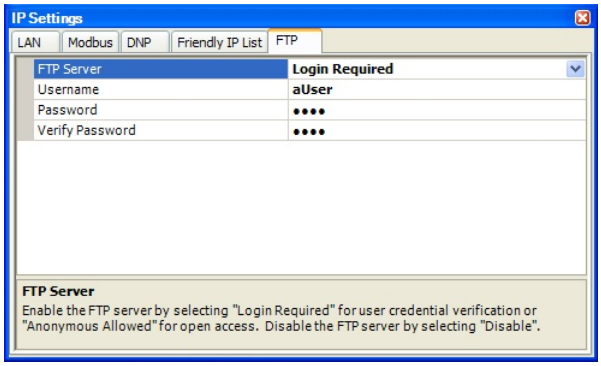
The **DHCP Server** selects whether the DHCP server is Enabled or Disabled. Select Enable to allocate an IP address the the FlowStation local display.

The **Range Start Address** is the start of the range of IP addresses that will be provided to computers connecting to the FlowStation Ethernet port. The default range start address is 172.16.100.100.

The **Range End Address** is the end of the range of IP addresses that will be provided to computers connecting to the FlowStation Ethernet port. The default range end address is 172.16.100.120.

FTP Tab

The FTP tab configures the File Transfer Protocol (FTP) server in the controller. The tab is available only if the controller supports FTP. The FTP server allows any FTP client to access the file system in the controller and on USB drives connected to the host port of the controller. See the File Management section for further information on the controller file system.



FTP Server enables the FTP server in the controller

- Select **Disabled** to disable the FTP server and prevent FTP access to the file system
- Select **Login Required** to enable the FTP server. A username and password are required to access the file system.
- Select **Anonymous Allowed** to enable the FTP server. A username and password may be used to access the file system. The server will accept an anonymous login; it will accept any username and password.

Username specifies the username for the FTP server account. It is used when **Login Required** is selected. Enter a name 1 to 16 characters long.

Password specifies the password for the FTP server account. It is used when **Login Required** is selected. Enter a password 1 to 16 characters long.

Verify Password specifies the password a second time to confirm the value. Enter the same value as for the password.

The FTP register assignment module can be used to enable and disable the FTP server using Ladder Logic.

FTP Server functions operate at a high priority in the controller. The execution of Telepace and IEC 61131-3 logic applications is impacted during file uploads and downloads. The logic applications will slow significantly during large file uploads to the controller.

To reduce the impact on logic applications:

- Use a number of small files rather than a single large file.
- Use the LS (list) command rather than the DIR (directory) command.

FTP usernames and passwords are transmitted in the clear when reading/writing controller configuration. In order to minimize the possibility of reading of credentials the controller should be kept locked against programming commands except when the configuration is intentionally being read or written.

FTP Best Practices

These best practices should be followed to maximize the effectiveness of the FTP feature.

Binary vs ASCII Mode

- Files can be transferred via either Binary or ASCII modes. Binary file transfers are recommended because they are more efficient. Most FTP clients will support both types of file transfers.

File Locations

See the File Management section for further information on the controller file system.

- The internal flash drive (d0) contains all system and user files. The system files should not be modified via FTP or user processes.
- Many system files can be found in the “d0/SYSTEM” folder. It is strongly recommended that only system files reside in this folder to prevent accidental loss/corruption of controller settings.
- C programs can be found in the root directory “d0”. These programs have a suffix of “.out” and should not be modified.
- Log files created by the data log to file functionality are stored in sub-folders of “d0/LOGS/”. These files may be read, but writing to or deleting these files is discouraged. The data log to file functionality will manage these files as required.
- Users are encouraged to create their own folders and store all files that they need to access in folders they create to minimize accidentally altering a system file. i.e. create a directory called “d0/SCADA” to keep all files that will be modified via FTP in.

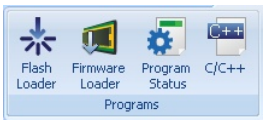
FTP Timing Consideration

The time required for FTP file upload (writing to the SCADAPack file system) or download (reading from the SCADAPack file system) between an FTP client and the SCADAPack FTP Server is dependent on a number of factors. Consideration of the these factors should be taken if it appears the file transfer process is taking longer than expected.

- The physical connections between the FTP client software running on a PC and the SCADAPack controller running the FTP server can vary between a direct tabletop Ethernet connection to a connection involving multiple Ethernet radios and routers. Each device will add some latency to the connection.
- File upload to the SCADAPack controller may be affected by the amount of memory used in the SCADAPack. Once the SCADAPack is 90% full the FTP transfer rate starts to slow down. At 90% used memory the transfer rate is two times slower than at levels less than 90%. The transfer rate continues to slow as the memory used approaches 100%. Periodically removing log files will keep the memory usage to an acceptable level for FTP uploads. File Download speeds are not significantly affected by the memory used in the SCADAPack controller.

Programs Group

The **Programs Group** is located in the Configure Ribbon.



The Programs Group contains the following commands:

- The **Flash Loader** command is used to write Ladder Logic and C/C++ programs into Flash memory on the target controller.
- The **Firmware Loader** command is used to display the controller Device Information and Current Options, update firmware in the controller and change the available options on the controller.

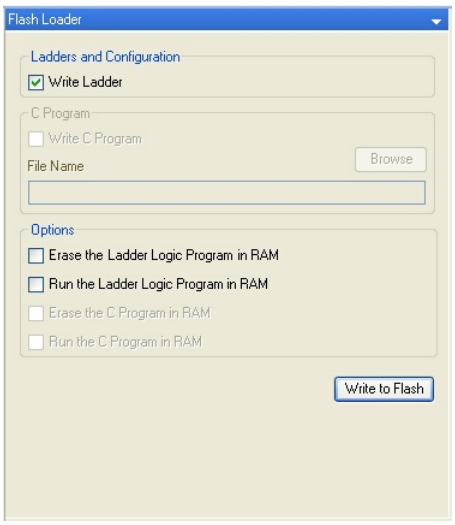
- The **Program Status** command displays information about programs loaded in the controller memory.
- The **C/C++** command is used to load C/C++ programs into the target controller.

Flash Loader

The **Flash Loader** command is used to write Ladder Logic programs into Flash memory on the target controller. This memory is permanently programmed and does not require power or a backup battery to retain the logic program. A backup battery is required to operate the controller, even if the program is in flash. The I/O database requires a backup battery to retain data. If the backup battery is not present the I/O database and forcing will power up in a random state.

The Ladder Logic program in flash is retained when a cold boot is performed. This allows the program to be restored after replacing the battery without using Telepace.

Telepace needs to be communicating with the target controller in order to use the Flash Loader functions. The Flash Loader view may be opened in Telepace projects whose controller types support this function. SCADAPack (including SCADAPack Light, SCADAPack Plus, Micro 16, SCADAPack 100 and SCADAPack 4202) controllers need to have firmware version 1.40 or newer to support this function. If the firmware version does not support Flash Loader Telepace displays the message ***"Controller does not support Flash memory"*** and does not open the Flash Loader view.



SCADAPack controllers support Ladder Logic programs in both RAM and Flash memory. The Ladder Logic programs may be identical or they may be entirely different programs. How Ladder Logic programs in RAM and Flash memory execute is subject to the following rules.

- If there is a program in RAM (and no program in Flash) the program will execute when a Run command is sent from %TP%> or the controller is power cycled.
- If there is a program in RAM and a program in Flash the program in RAM will execute when a Run command is sent from %TP%> or the controller is power cycled.
- If there is a program in RAM and a program in Flash and the program in RAM is lost due to a low RAM battery and a power loss occurs the program in Flash will execute when power to the controller is restored.
- If there is a program in Flash (and no program in RAM) the program will execute when a Run command is sent from Telepace or the controller is power cycled.

In addition to the ladder logic program the configuration data for the project is written to the controller. To see what configuration items are written see Ladder Logic and Configuration Data.

Ladder Logic and Configuration Data

When a Ladder Logic program is written to flash memory all project configuration data is included with the Ladder Logic itself. The following list shows the items that are saved to flash.

- Ladder Logic Program
- Float point Constants
- Element Configuration
- Serial Port Settings
- Outputs on Stop
- Register Assignment
- I/O error indication flag
- Configuration data for register assignment I/O modules
- IP Addressing settings

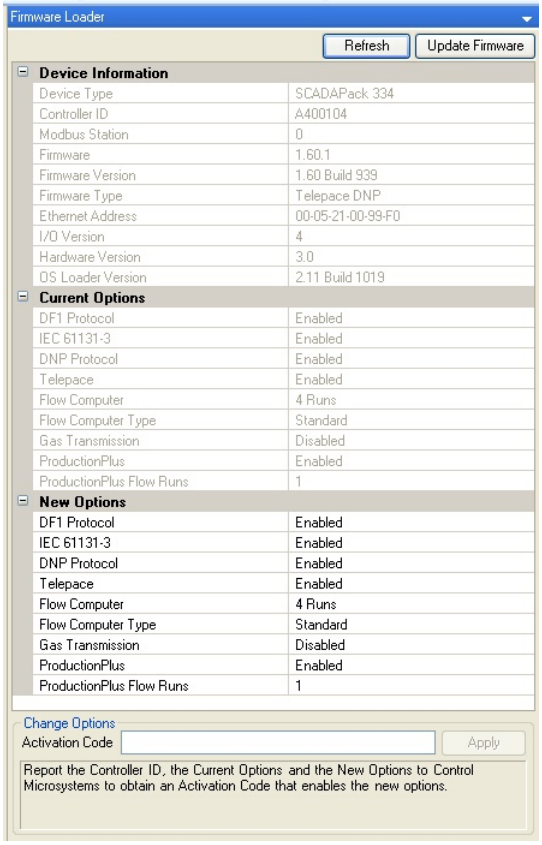
- IP Protocol settings
- Friendly IP settings
- Store and Forward Settings
- DNP configuration data
- DNP I/O points
- DNP routing
- DNP Master Application
- DNP Master Polling
- DNP Master Addressing

Firmware Loader

The **Firmware Loader** command is used to display the controller Device Information and Current Options, update firmware in the controller and change the available options on the controller.

Telepace needs to be communicating with the target controller in order to use the Firmware Loader functions. Telepace reads the target controller Device Information and Current Options when the Firmware Loader command is selected.

When you select the **Firmware Loader** command, **Firmware Loader** view opens:



The **Device Information** area of the view displays information about the controller.

The **Current Options** area of the view displays information about the controller Options.

The **New Options** are is used with the Change Options window when the controller options need to be updated.

The **Refresh button** is used to restart communication with the target controller and refresh the **Device Information** and **Current Options** information.

The **Update Firmware button** starts the update firmware procedure. See **Update Firmware** for complete information on updating the firmware in a target controller.

Device Information

The Device Information area of the **Firmware Loader** pane displays information about the controller or transmitter, its firmware version and the options installed.

Device Type

The **Device Type** window displays the type of controller or transmitter connected. The device type is one of the following:

- Boot Loader
- Boot Loader 32
- Boot Loader SCADAPack 3xx
- Boot Loader SCADAPack/SOLARPack

- FlowStation 110
- Micro16
- SCADAPack
- SCADAPack 100: 256K
- SCADAPack 100: 1024K
- SCADAPack 314
- SCADAPack 330
- SCADAPack 334
- SCADAPack 350
- SCADAPack 32
- SCADAPack 32P
- SCADAPack LIGHT
- SCADAPack LP
- SCADAPack PLUS
- SCADAPack 4202 DR
- SCADAPack 4203 DR
- SCADAPack 4202 DS
- SCADAPack 4203 DS

If the value read from the device is not one of these values, the device type is set to **No-Selection**.

If the controller ID number of the SCADAPack 100 is A182921 or less it is considered to be a SCADAPack100:256K. If the controller ID number is A182922 or greater the controller is considered to be a SCADAPack100:1024K.

Controller ID

The **Controller ID** window displays the unique ID for the controller that is set at the factory.

Station

The **Station window** displays the station address of the controller communication port currently connected. Depending on the controller type the communication port will be one of serial, Ethernet, USB or *Bluetooth.*

Firmware Type

The **Firmware Type** window displays the type of firmware currently installed in the device. It may be one of the following values: none (returned by the Boot Loader), Telepace or IEC 61131-3.

Ethernet Address

The **Ethernet Address** window displays the MAC Address for the SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 and 32P controllers. It is disabled for the Micro16, SCADAPack, SCADAPack Light, SCADAPack Plus, SCADAPack LP, SCADAPack 100 and SOLARPack 410 controllers.'

I/O Version

The **I/O Version** window displays the version number of the internal I/O controller.

Hardware Version

The **Hardware Version** window displays the hardware version of the controller board.

OS Loader Version

The **OS Loader Version** window displays the version number of the OS Loader. The OS loader executes when there is no operating system firmware in the controller and controls the loading of firmware.

DF1 Protocol

The **DF1 Protocol** option enables communication using the Allen-Bradley full duplex and half-duplex DF1 protocols. When the controller is ordered with this option it is enabled at the factory.

Flow Computer

The **Flow Computer** option enables support for the Realflo natural gas flow computer. The **Runs** window displays the number of runs available for the flow computer. When the controller is ordered with this option it is enabled at the factory.

IEC 1131

The **IEC 1131** option enables the IEC 1131 run-time engine. This option is enabled by default at the factory.

DNP Protocol

The **DNP Protocol** option enables communication using the DNP.. This option is enabled by default at the factory.

Telepace

The **Telepace** option enables the Telepace runtime option.

All SCADAPack 314, 330, 334, 350 and 357 controllers and the SCADAPack 4203 controllers have a license option for the Telepace run-time engine when the firmware is version 1.60 or newer. The option is ignored by firmware older than version 1.60. This option prevents the execution of ladder logic programs. No other operations are prevented.

Telepace checks for the Telepace option in the controller. The user is messaged if the ladder logic can't run. If all these conditions are met

- The controller is a SCADAPack 314, 330, 334, 350 and 357 controllers or SCADAPack 4203; and
- The firmware version is 1.60 or newer; and
- The firmware type is Telepace; and
- The Telepace option is disabled; and
- The ladder logic is not empty.

When all conditions are met Telepace displays the message:

"The Telepace license option in the controller is disabled. Ladder Logic programs will not execute. Contact Control Microsystems to license Telepace on this controller."

Telepace checks for the Telepace option during the following operations:

When the following operations are performed

- Writing to the controller
- Switching from offline mode to either edit on-line or monitor on-line mode
- Program status
- Flash Loader
- Run/Debug/Stop

Related Topics

Firmware Loader

New Options

Update Firmware

New Options

The **New Options** area displays controller options that are supported for the controller type. The new options are identical to the current options area when the Firmware Loader pane is opened.

The options in a controller may need to be changed if the application the controller is used in changes. When options need to be changed an Activation Code is required from the factory.

To obtain an Activation Code

1. Record the Controller ID, Current Options and New Options for each controller you wish to update.
2. Contact Control Microsystems sales department, or your local representative, and report the information gathered in step 1.
3. The sales representative will inform you of the cost of the options and arrange for payment.
4. The activation codes will be sent to you upon receipt of payment.

To change Options

1. Once an activation code has been obtained check the New Options boxes for the controller options requested in the steps above. The new options need to include the current options.
2. Enter the Activation Code in the **Activation Code** window.
3. Click the **Apply** button to enable the new options.

Related Topics

Firmware Loader

Device Information

Update Firmware

Update Firmware

The **Update Firmware** command is used to update the operating system firmware in the target controller. The firmware in a device may only be updated through serial ports on the controller board, and depending on the controller type the USB peripheral port or the Ethernet port.

Not all serial ports support firmware updating on all controllers. Allowed connections for firmware update:

- For **Micro16, SCADAPack Light, SCADAPack** and SCADAPack Plus firmware may be updated using com1 or com2 serial ports. (For 5204 controllers only com2 may be used.)
- For SCADAPack 32 and SCADAPack 32P firmware may be updated using com1, com2 or com4 serial ports and the Ethernet port.
- For the SCADAPack LP controller firmware may be updated using com2.
- For the SCADAPack 314 firmware may be updated using com1, com2 or the USB peripheral port.
- For the SCADAPack 330 firmware may be updated using com1, com2, com 3, Ethernet or the USB peripheral port.
- For the SCADAPack 334 firmware may be updated using com1, com2, com3, Ethernet or the USB peripheral port.
- For the SCADAPack 350 firmware may be updated using com1, com2, com3, Ethernet or the USB peripheral port.
- For the SCADAPack 100 firmware may be updated using com1 or com2.
- For SCADAPack 4202 firmware may be updated using com2.
- For SCADAPack 4203 firmware may be updated using com2 and com3.
- For all supported SCADAPack 4000 firmware may be updated using com2.
- For SOLARPack 410 firmware may be updated using com2 or Bluetooth connection.

- For the FlowStation 110 firmware may be updated using com1, com2, Ethernet or the USB peripheral port.

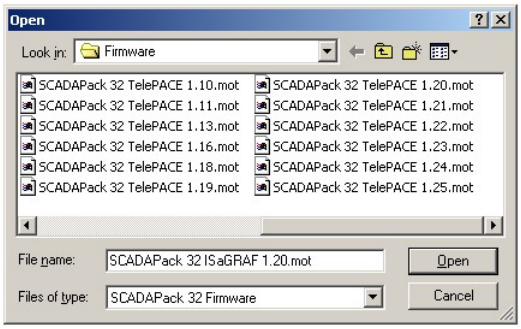
Update Firmware Procedure

All firmware files are installed to the **C:\Program Files\Control Microsystems\TelepaceStudio** folder, by default, when Telepace is installed. Newer firmware files are added to this folder when a new version of Telepace is installed. Older firmware files are not deleted when a new version of Telepace is installed.

Ensure firmware Loader is connected to a valid serial port; see above for allowed serial connections, or to an Ethernet or USB peripheral port on the target device.

- Click the **Update Firmware** button.

When the **Update Firmware** command is selected a standard open file dialog appears. Select the file containing the new operating system firmware.



Only firmware files for the device type selected will be displayed in the dialog.

For SCADAPack 32 controllers firmware files with the extension **.mot** are displayed.

For SCADAPack 4000 transmitters and SOLARPack 410 firmware files with the extension **.hex** are displayed.

For SCADAPack 314, SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 4203 and SOLARPack controllers, firmware files with the extension **.out** are displayed.

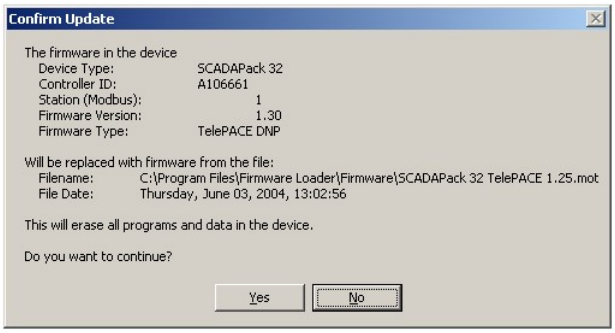
For FlowStation 110 controllers firmware files with the extension **.fwl** are displayed.

For all other controllers firmware files with the extension **.abs** are displayed.

- Select the firmware file to use and click the **Open** button.

The **Confirm Update** dialog appears displaying all the information that is available about the target device and the firmware file.

Firmware version 1.80 or newer cannot be loaded into the SCADAPack 100:256K. If download is attempted, Firmware Loader will display error message: “The device does not support the firmware version 1.80 or newer.”



Firmware Update Effect on Applications

Some SCADAPack controllers provide for the retention of Telepace Ladder Logic and C++ applications through a firmware upgrade.

- This feature is available on SCADAPack 350, SCADAPack 330, SCADAPack 334, SCADAPack 314, FlowStation 110, SCADAPack 4203 controllers and the SOLARPack 410 **firmware version 1.40 and later**. Both Telepace and IEC 61131-3 firmware versions support this feature.

This feature is not supported on SCADAPack 350, SCADAPack 330, SCADAPack 334, SCADAPack 314, FlowStation 110, SCADAPack 4203 controllers and the SOLARPack 410 unless **firmware version 1.40 and later** is currently installed on the controller.

The effect of a firmware download on IEC 61131-3, C/C++ and Telepace Ladder Logic applications are explained in the following sections.

IEC 61131-3 Applications

IEC 61131-3 applications are not supported with this feature. IEC 61131-3 applications are deleted during a firmware download.

C/C++ Applications

IEC 61131-3 C/C++ and Telepace C/C++ applications are retained during a firmware download. After a firmware download the C/C++ application needs to be re-started. When the controller restarts all statically allocated and dynamically allocated non-volatile memory is freed up.

The static non-volatile memory will be allocated on a successful start of the C application for the first time following a firmware change.

Realflo C/C++ Applications

The Realflo flow computer is a C/C++ application and is retained on a firmware download. The flow computer needs to be restarted after a firmware download. While the flow computer application is retained the following are NOT retained:

- The flow computer configuration is lost on a firmware download.
- The flow computer historic daily and hourly logs are lost on a firmware download.
- The flow computer historic event and alarm logs are lost on a firmware download.

Before beginning a firmware download on a controller running Realflo the flow computer configuration and historic logs

need to be read from the controller. This data is lost on a firmware download.

Telepace Ladder Logic Applications

Telepace Ladder Logic programs in RAM or flash, and the entire database, are retained during firmware updates. The programs are reloaded but not started as part of the firmware download. After a firmware download the Telepace Ladder Logic application needs to be re-started.

The DLOG functionality is not retained through a firmware download. The data for the **DLOG** function uses dynamic non-volatile memory. This memory is freed during a firmware download and the logged data is lost.

Before beginning a firmware download on a controller using **DLOG** the logged data logs need to be read from the controller. This data is lost on a firmware download.

Update Firmware from Boot Loader State

For Micro16, SCADAPack, SCADAPack Light, SCADAPack Plus, SCADAPack LP, SCADAPack 100, SCADAPack 4202 controllers, if the connection to the controller is broken during a firmware update, the controller will revert to the Boot Loader state.

In the Boot Loader state firmware can only be loaded through communication port 2 on the controller. The default Com 2 serial port settings of 9600 baud, 8 data bits, no parity, and 1 stop bit and Modbus station address 1 needs to be used.

SCADAPack 314, SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack32, SCADAPack32P, FlowStation 110 controllers, SCADAPack 4203 and SOLARPack 410 retain their serial or Ethernet port settings in the event of a communication loss during a firmware upgrade. These controllers also retain their original firmware properties and are not reverted to the Boot Loader state.

Related Topics

Firmware Loader

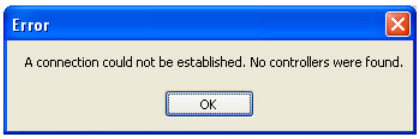
Device Information

New Options

Error Messages

Error Messages

A communication error message displays if Telepace and the target controller are not communicating.



If Telepace Studio is not communicating with a target controller when the Firmware Loader command is selected then the Device Information, Current Options and New Options fields will display **N/A**.

The project Controller Type selection determines the firmware that is supported in the controller. When the target controller is not the same type as the project controller type an error message is displayed.

Related Topics

Firmware Loader

Device Information

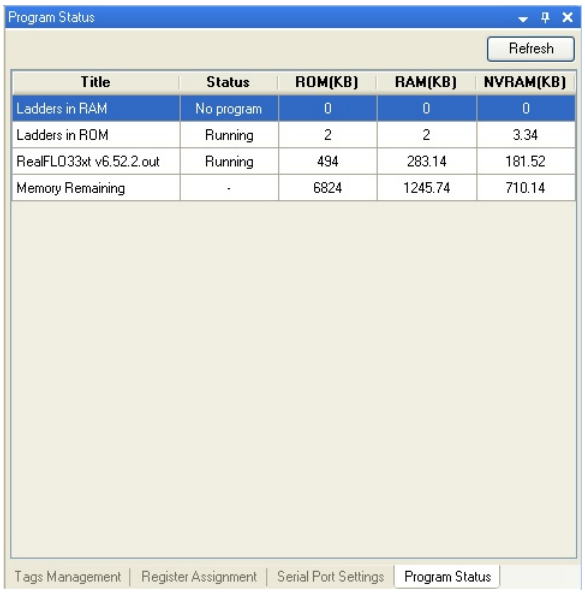
New Options

Update Firmware

Program Status

The **Program Status** dialog displays information about programs loaded in the controller memory. The Program Status View is displayed when this command is selected. The Program Status dialog displayed depends on the controller type.

Program Status View for SCADAPack 314, SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 4203 and FlowStation 110 Controllers



The Title column lists the programs in the controller:

- **Ladders in RAM** row describes the status of Ladder Logic programs loaded in RAM.
- Ladders in ROM row describes the status of Ladder Logic programs loaded in ROM (Flash).
- C/C++ applications row describes the status of any C/C++ programs loaded.
- The Memory Remaining row describes the unused memory of each type available to all programs.

The status may be one of the following:

- **Stopped** indicates the program has been downloaded and is currently stopped.
- **Running** indicates the program has been downloaded and is currently running.
- **Debug** indicates the program has been downloaded and is currently running in debug mode (Ladder Logic program only).
- **No Program** indicates that there is no program in the controller.

The memory used by each program is displayed in addition to the program status.

- The **ROM (KB)** column describes the amount of memory used in ROM by each program.
- The **RAM (KB)** column describes the amount of memory used in RAM by each program.
- The **NVRAM (KB)** column describes the amount of memory used in non-volatile RAM by each program.

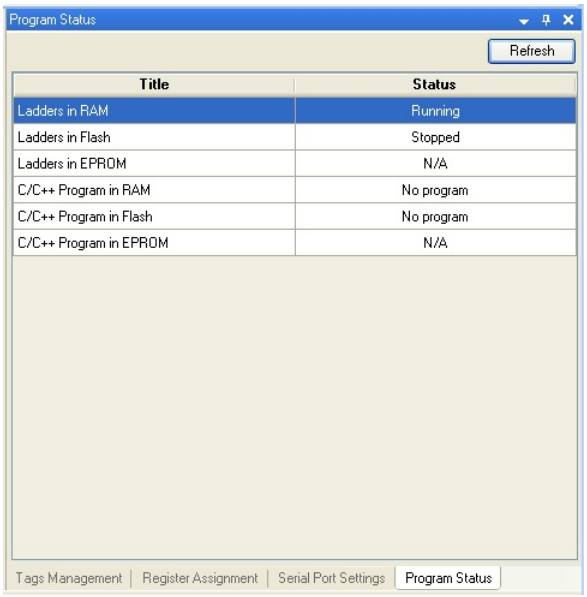
The Ladder Logic program in RAM memory takes precedence over a Ladder Logic program in ROM. When a command to run a program is received, the controller first checks for a program in RAM, then for one in ROM. A Ladder Logic program in ROM may only be run if there is no Ladder Logic program in RAM.

The **Refresh** button updates the view with the current program status.

SCADAPack and SCADAPack 32 Controllers

Programs in RAM memory take precedence over programs in Flash or EPROM memory. When a command to run a program is received, the controller first checks for a program in RAM, then for one in Flash or EPROM.

SCADAPack (16-bit)



The Title column lists the programs in the controller:

- **Ladders in RAM** row describes the status of Ladder Logic programs loaded in RAM.
- **Ladders in Flash** row describes the status of Ladder Logic programs loaded in Flash.
- **Ladders in EPROM** row describes the status of Ladder Logic programs loaded from EPROM.
- **C/C++ Program in RAM** row describes the status of any C/C++ programs loaded in RAM.
- **C/C++ Program in Flash** row describes the status of any C/C++ programs loaded in Flash.
- **C/C++ Program in RAM** row describes the status of any C/C++ programs loaded in EPROM.

The status may be one of the following:

- **Stopped** indicates the program has been downloaded and is currently stopped.
- **Running** indicates the program has been downloaded and is currently running.
- **Debug** indicates the program has been downloaded and is currently running in debug mode (Ladder Logic program only).
- **No Program** indicates that there is no program in the controller.
- **N/A** indicates the controller does not support the memory type.

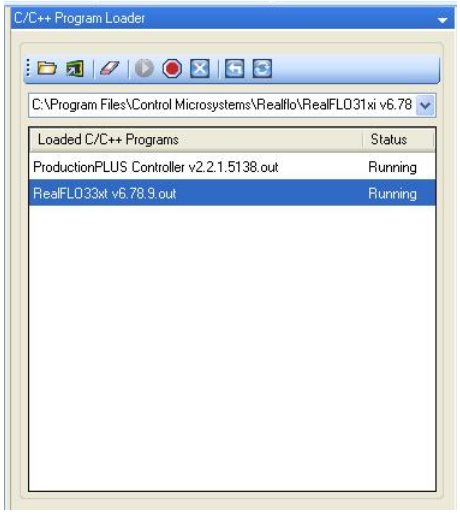
C/C++ Loader

The C/C++ Program Loader command opens the C/C++ Program Loader view. This view is used to load, run, stop, and delete C/C++ programs. The view is the same for each SCADAPack controller type.

- SCADAPack and SCADAPack 32 controllers support one C/C++ program executing, in addition to the logic application.
- SCADAPack 314, 33x, 350 and 4203 controllers support multiple C/C++ applications executing, in addition to the logic application.

Telepace Studio needs to communicating with the target controller to use the C/C++ Program Loader.

The C/C++ Loader view displays a history list of C/C++ applications that have been written to a target controller by Telepace Studio, currently loaded C/C++ programs in the controller and the status of the loaded programs. The Loaded C/C++ Programs list will be blank if there are no programs currently installed in the target controller.



The C/C++ Program Loader view is divided into three sections.

The **C/C++ Loader Toolbar** displays frequently used commands for managing C/C++ programs in the controller.

The **C/C++ Program History** section is a drop down list of C/C++ programs that have been written to the target controller since the last Clear History command was used, or a file that has been selected using the Browse command.

The **Loaded C/C++ Programs** grid displays the C/C++ program(s) loaded in the target controller and the status of the loaded programs. The **status** is one of:

- **Running** if the C/C++ application is executing in the target controller.
- **Stopped** if the C/C++ application is loaded but not executing in the target controller.

Managing C/C++ Programs

Any C/C++ programs that are in the target controller are displayed in the **Loaded C/C++ Programs** list. Each C/C++ program in the controller is individually controlled using the C/C++ Program Loader toolbar. Click on the desired C/C++ program in the list to highlight it. Any toolbar commands are performed on the highlighted program.

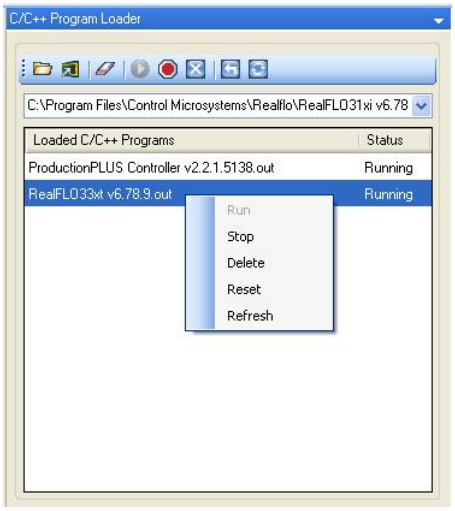
From left to right the toolbar icons are described below.

The Browse icon opens the file select navigation window to find and select a C/C++ program for writing to the target controller. The selected file and location are displayed in the C/C++ Program History window.

- The **Write** icon writes the file selected by the Browse command to the target controller.
- The **Clear History** icon erases the C/C++ program history.
- The **Run** icon starts the selected C/C++ program.
- The **Stop** icon stops the selected C/C++ program.
- The **Delete** icon deletes the selected C/C++ program from the controller memory.
- The **Reset Controller** icon refreshes the status of the loaded C/C++ program(s).

In addition to the toolbar commands may be selected from the list

- To open the command list, right click anywhere in the Loaded C/C++ Programs list. A command list is displayed. Click on any of the commands to perform the command operation.



- The **Run** command stops and then restarts the C/C++ program in the controller.
- The **Stop** command stops the selected C/C++ program in the controller.

- The **Delete** command stops and erases the selected C/C++ program in the controller.
- The Run, Stop and Delete commands are disabled if there is no C/C++ program loaded in the controller.
- The **Reset** command resets the controller. The reset command restarts the target controller, the C/C++ program(s) and the Ladder Logic program.
- The **Refresh** command refreshes the status of the loaded C/C++ program(s).

Controller Group

The **Controller Group** is located in the Configure Ribbon. The ribbon is displayed differently depending on the controller type.

For all controller types except FlowStation the ribbon is displayed as shown below.

For FlowStation controllers the ribbon is displayed as shown below.

The Controller Group contains the following commands:

- **File Management** manages files on the controller file system.
- **Controller Security** sets the controller security settings.
- **Clock** configures the target controller's internal real time clock.
- **Initialize** restores a controller to default settings.

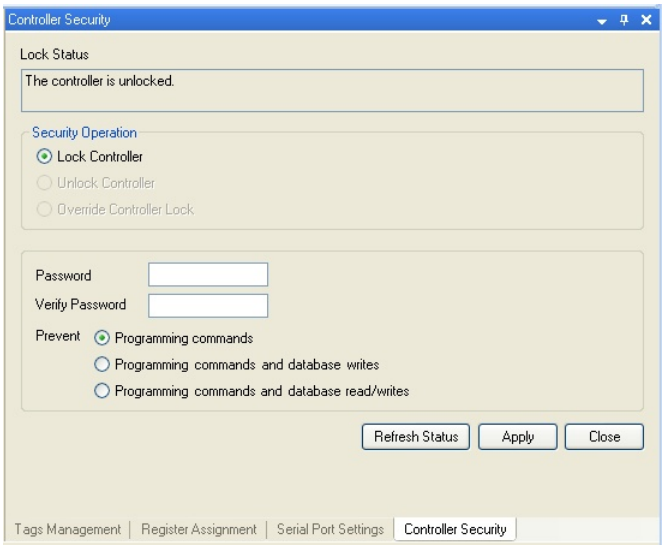
Security

The **Controller Security** command is used to set the security level for the target controller.

Telepace needs to be communicating with the target controller to use the Controller Security functions. Telepace reads the current controller security settings when the Controller Security command is selected.

A communication message will be displayed indicating the send command to controller operation did not work if Telepace and the target controller are not communicating.

When selected the Controller Security command opens the **Controller Security** pane.



Lock Status

The **Lock Status** window displays the current lock status of the of the target controller. The lock status will be one of the following.

- The **controller is unlocked** status indicates there is no security applied on the controller. The Telepace project in the controller may be edited and the controller database may be written to or read from.
- The **controller is locked against programming commands** status indicates the controller Telepace project cannot be modified or viewed without entering the correct password.
- The **controller is locked against programming commands and database write status** indicates the controller Telepace project cannot be modified and the controller will not accept writes to the data base without entering the correct password. Communication protocols can read data from the I/O database, but cannot modify any data.
- The **controller is locked against programming commands and database read/write** status indicates the controller Telepace project cannot be modified and the controller will not accept writes to or reads from the data base without entering the correct password. Communication protocols cannot read or write the I/O

database.

Controller Security

The **Security Operation** selections will be displayed depending on the lock status of the target controller.

Lock Controller

The **Lock Controller** selection is displayed only when the target controller lock status is unlocked.

To **lock** the controller:

- Enter a password in the Password edit box. Any character string up to eight characters in length may be entered. Typing in this edit box is masked. An asterisk is shown for each character typed.
- Re-enter the password in the **Verify Password** edit box.

Select the actions to **Prevent**.

- Locking the Programming Commands prevents modifying or viewing the program in the controller. Communication protocols can read and write the I/O database.
- Locking Programming commands and database write prevents modifying or viewing the program and prevents writing to the I/O database. Communication protocols can read data from the I/O database, but cannot modify any data.
- Locking Programming commands and database read/write prevents modifying or viewing the program and prevents reading and writing the I/O database. Communication protocols cannot read or write the I/O database.

To **apply** and **check** the controller security:

- Click the Apply button to write the security settings to the controller.
- A communication dialog will be displayed indicating the security settings are being written to the target controller.
- The Lock **Status** window will display the new security settings.

Click the **Refresh Status** button to check the controller lock status. The **Lock Status** window will update and display the controller lock status.

Click the **Close** button to close the **Controller Security** dialog. If changes to the **Controller Security** setting were made but not applied they are discarded when the **Close** button is clicked.

Unlock Controller

The **Unlock Controller** selection is displayed only when the target controller lock status is locked to prevent any of the following:

- Programming commands.
- Programming commands and database write.
- programming commands and database read/write.

To **unlock** the controller:

- Enter the password in the **Password** edit box.

To **apply** and **check** the controller security.

- Click the Apply button to write the password to the controller.
- A communication dialog will be displayed indicating the password is being written to the target controller.
- The Lock Status window will now display the message: **The controller is unlocked.**

Click the **Refresh Status** button to check the controller lock status. The **Lock Status** window will update and display the controller lock status.

Click the **Close** button to close the **Controller Security** dialog. If the **Password** was entered but not applied the entry is discarded when the **Close** button is clicked.

Once programming changes have been made to the Telepace project and written to the target controller the controller needs to be locked again to maintain controller security.

Override Controller

The **Override Controller Lock** selection is displayed only when the target controller lock status is locked to prevent any of the following:

- Programming commands.
- Programming commands and database write.
- Programming commands and database read/write.

The Override Controller Lock selection is used to unlock a controller without knowing the password. This can be used in the event that that the password is forgotten.

To prevent unauthorized access to the information in the controller, the C program and Ladder Logic programs are erased and all controller settings are reset to default. The Override Controller Lock popup is displayed, prompting for confirmation, whenever the override controller lock command is applied to the controller.

When the Override Controller Lock command is used on a FlowStation 110 controller the FlowStation 110 will revert all configured parameters to 0's and not the default values. This will be corrected in a later version of FlowStation. Users are urged not to use the Override Controller Lock command on the FlowStation. Instead perform a Cold Boot to unlock the controller. The Cold Boot will keep the FlowStation default parameters and not set them to zeros.

To **Override Controller Lock** in the controller:

- Click the Override Controller Lock selection.
- Click the **Apply** button to write the override controller lock command to the controller.

Confirm the action in the **Override Controller Lock** popup.

- Click **Yes** to confirm the action and override the controller lock.
- Click No to abort the override controller lock action.

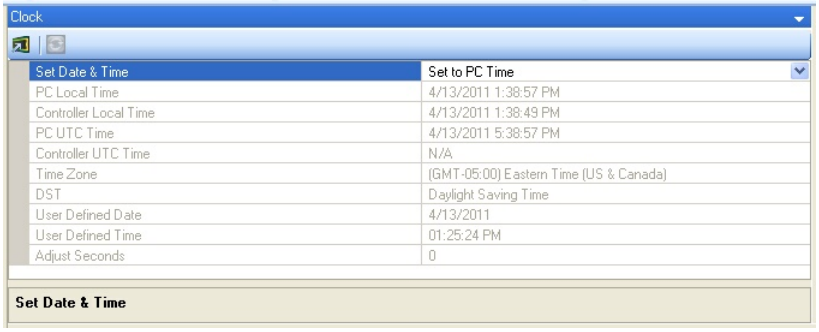
Clock

The **Clock** command is used to configure the target controller's internal real time clock.

Telepace Studio needs to be communicating with the target controller to use the Real Time Clock functions. Telepace Studio reads the current time in the controller when the Real Time Clock command is selected.

A communication message will be displayed indicating the send command to controller operation did not succeed if Telepace Studio and the target controller are not communicating.

When selected the Clock command opens the **Clock** pane.



The **Set Date & Time** section has a drop down menu control used to set the controller Real Time Clock. There are three selections available:

- The **Set to PC Time** selection is used to set the date and time in the target controller to the same time and date as the PC internal clock.
- The **Set to User Entered Time** selection is used to set the date and time in the target controller to a user defined time and date. The Time Zone, DST, User Defined Date and User Defined Time windows are available for data and time entries when this option is selected.
- The **Adjust Forward or Backward** selection is used to adjust the time in the target controller forward or backward a user defined number of seconds. The Adjust Seconds window is available for adjusting seconds when this option is selected.

The **PC Local Time** window displays the time and date in the PC internal clock. The format is DD/MM/YYYY HH:MM:SS AM or PM.

The **Controller Local Time** window displays the date and time in the controllers real time clock. The format is DD/MM/YYYY HH:MM:SS AM or PM.

The **PC UTC Time** window displays the current date & time of PC in format DD/MM/YYYY HH:MM:SS AM or PM.

The **Controller UTC Time** window displays the UTC time based on the current date & time of the controller in format DD/MM/YYYY HH:MM:SS AM or PM. The field displays “N/A” if the controller firmware version does not support this feature.

The **Time Zone** window displays the current time zone and a drop down menu for selecting different time zones. This window is greyed out if the selection for Set Date & Time is not Set to User Entered Time. The default is the PC time zone. This field is disabled if the controller firmware does not support this feature.

The **DST** window displays the DST (Daylight Saving Time) setting for the controller and a drop down menu for selecting Daylight Saving Time or Standard Time. This window is greyed out if the selection for Set Date & Time is not Set to User Entered Time. The default is the PC setting. This field is disabled if the controller type is not set as FlowStation.

The **User Defined Date** window displays the user entered date. This window is greyed out if the selection for Set Date & Time is not Set to User Entered Time. The default is the PC date. This field is disabled if the controller firmware does not support this feature.

The **User Defined Time** window displays the user entered time. This window is greyed out if the selection for Set Date & Time is not Set to User Entered Time. The default is the PC time. This field is disabled if the controller firmware does not support this feature.

The **Adjust Seconds** window displays allows the user to enter the number of seconds that will adjust the time forward or backward in the controller. The valid range is from -32768 to 32767. The default is 0.

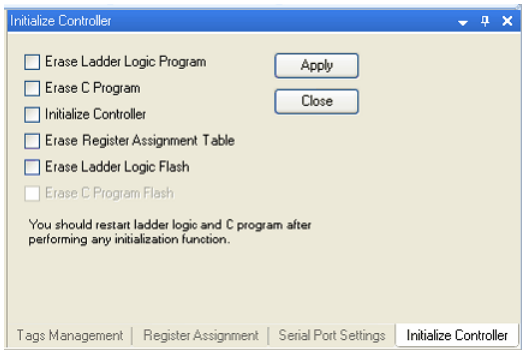
Initialize Controller

The **Initialize** command is used to restore a controller to default settings. This is typically done when starting a new project with a controller. The Initialize controller view is displayed when this command is selected. The Initialize Controller dialog presented depends on the type of controller selected.

The view displayed depends on the controller type. Each view is described below.

SCADAPack and SCADAPack 32 Controllers

The Initialize Controller view dialog shown below appears when the command is selected and the Controller Type is SCADAPack or SCADAPack 32 (**not** SCADAPack 330, SCADAPack 334, SCADAPack 350 or SCADAPack 4203).



Check the **Erase Ladder Logic Program** selection to erase the ladder logic program in the controller RAM memory.

Check the **Erase C Program** selection to erase the C application program in the controller RAM memory.

Check the **Initialize Controller** selection to reset all values in the I/O database to default settings. The Initialize Controller selection will not work on a controller that is locked. The error message “Command cannot be executed on a locked controller” will be displayed.

The Initialize controller command will do the following:

- The default communication parameters are set for all serial ports. If you are communicating using settings other than the default, the PC Communication Settings will have to be changed once the command is complete. The Ethernet port on SCADAPack 32 controllers is not changed by the initialize controller command.
- The registers in the I/O database are initialized to their default values.
- All forcing is removed.

Check the **Erase Register Assignment** selection to erase the controller register assignments. The Register Assignment is cleared and needs to be reconfigured.

Check the **Erase Ladder Logic Flash** selection to erase the Ladder Logic program in Flash. This control is grayed if the controller firmware does not support programs in Flash or if the Flash is used by the operating system.

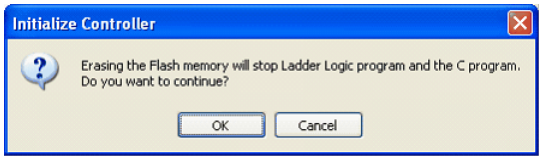
Check the **Erase C Program Flash** selection to erase the C program in Flash. This control is grayed if the controller firmware does not support programs in Flash or if the Flash is used by the operating system.

Click **Apply** to perform the requested initializations.

Click **Close** to exit without performing any action.

Erasing Programs in Flash

Erasing the Flash memory requires the Ladder Logic and C programs be stopped. The following message is displayed if the OK button is selected.

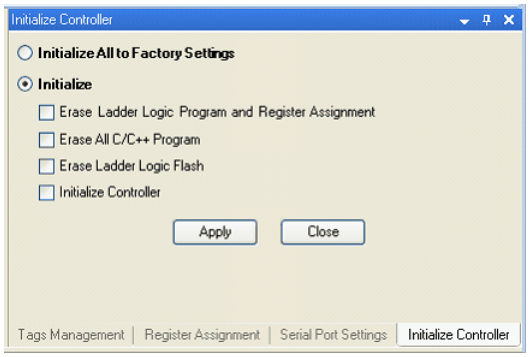


Click **OK** to stop the execution and erase the flash memory. The following actions occur.

- The Ladder Logic and C programs in the controller are stopped.
- If it was selected, the Ladder Logic program is erased.
- If it was selected, the C program is erased.
- Programs that were stopped to allow Flash to be erased are not re-started. These programs need to be re-started.

SCADAPack 330, 314, 334, 350 and SCADAPack 4203 of Controllers

For SCADAPack 330, SCADAPack 314, SCADAPack 334 , SCADAPack 350 and SCADAPack 4203 of controllers (4203 DR and 4203 DS), the Initialize view shown below appears when the Initialize command is selected.



Select the **Initialize All to Factory Settings** radio button to erase all programs in the controller memory and initialize the controller to factory settings. Please see Actions Performed when Controller is Initialized below for details. Erasing all programs includes all C/C++ programs, the Ladder Logic program in RAM, and the Ladder Logic program in ROM, if applicable.

Select the **Initialize** radio button to initialize only the selected items. The items that may be selected are as follows:

- Check t**he Erase Ladder Logic Program and Register Assignment** selection to erase the ladder logic program and register assignment in the controller RAM memory.
- Check the **Erase All C/C++ Programs** selection to erase all C/C++ programs in the controller.
- Check the **Erase Ladder Logic Flash** selection to erase the Ladder Logic program in Flash (ROM).
- Check the **Initialize Controller** selection to initialize the controller to default settings. Please see Actions Performed when Controller is Initialized below for details. The default settings used depends on the radio button selected below this selection:

Click on the **Apply** button to perform the requested initializations.

Click on the **Close** button to exit without performing any action.

Actions Performed when Controller is Initialized

The following actions are performed when the controller is initialized:

- The controller settings below are set to default values. If you are communicating using settings other than the default, the PC Communication Settings will have to be changed once the command is complete. The Ethernet port is not changed by the initialize controller command.
  - Serial port communication parameters
  - Modbus/IP and DNP/IP protocol settings
  - Outputs on Stop settings
  - HART modem configurations
  - LED power control

- All forcing is cleared.
- All Modbus registers are set to zero.
- If there is a register assignment, the assigned registers are initialized to their default values. For example, if a configuration module for a group of controller settings is in the register assignment, the assigned registers are initialized to the default values for those controller settings.
- All digital outputs and analog outputs are cleared.
- Data logged by all DLOG functions is erased.
- Alarm clock is cleared.
- Serial port event counters are cleared.
- Store and forward table is cleared.
- Friendly IP List is cleared.
- The power mode changes to full power.

File Management

The **File Management** command provides access to the controller file system on SCADAPack 300 and SCADAPack 4203 controllers. The File Management view is similar to Windows Explorer. It provides folder and file operations and management of data log files. Files can be managed using all controller communication protocols.

The File Management View may be opened without a connection to a controller. In this case only the Local view may be opened for File Management functions. Telepace needs to be communicating with the target controller to use the Controller view File Management functions.

The internal file system on the SCADAPack 330/334/350/357 controllers may use either of two available drives:

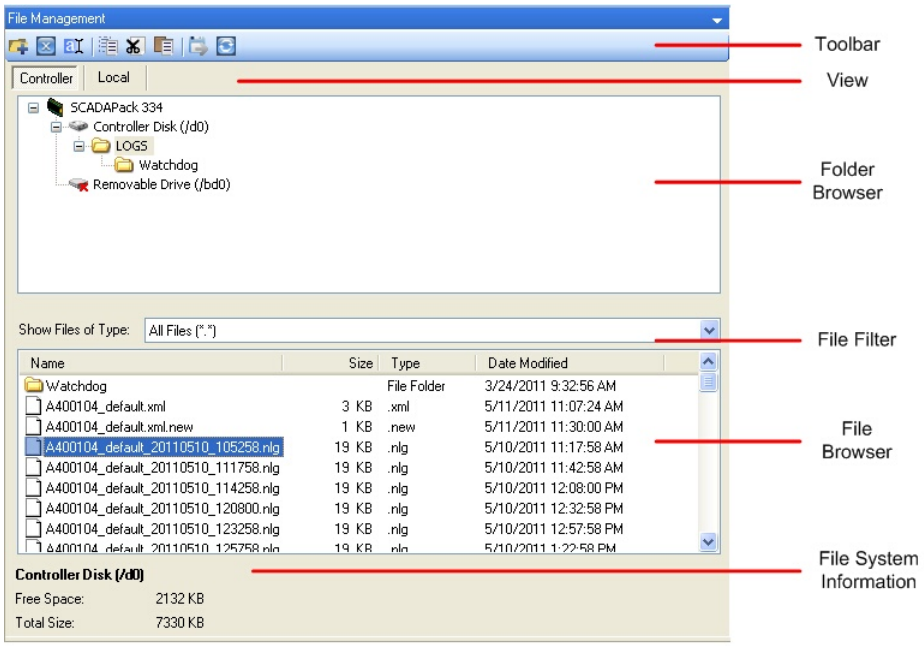
- The internal **Controller Disk Drive** is labeled **/d0**.
- The **External USB Drive** is labeled **/bd0**.

The internal file system on the SCADAPack 314 and the SCADAPack 4203 controllers use only the **Controller Disk Drive** labeled **/d0**.

The file name, including the complete path, on the internal file system cannot exceed 100 characters in total.

The **File Management** view is described below. Additional file management information is provided in the following sections:

- Folder Operations The Folder Browser Operations section describes how to manage folders in the selected file system.
- File Operations The File Browser Operations describes how to manage files in the selected file system.
- Data Log Operations describes management of data log files
- Shortcut Keys provide quick access to functions from the keyboard



The **Toolbar** is shared by the folder browser and file browser sections. From left to right the buttons are available for:

New Folder - create a new folder

Delete - delete the selected folders or files

Rename - rename the selected folder or file

Copy - copy the selected folder or file

Cut - mark to the selected folder or file to be cut

Paste - paste copied or cut folders or files

Export - export selected data log files to CSV format

Refresh - refresh the current folder

- Buttons are enabled or disabled based on the selected folder or file.
- All commands are also available by right-clicking on a file or folder and selecting the desired operation from the context menu.

The **Controller and Local tabs** select which file system is viewed.

- Select **Controller** to display files and folders on the SCADAPack 300 or SCADAPack 4203 controller.
- Select **Local** to display files and folders from the local PC.

The **Folder Browser** displays only folders in a folder tree similar to Windows Explorer. See the Folder Browser Commands section for more information on each toolbar operation.

- Folders on the file system may be expanded or collapsed by pressing the + or - icon next to the folder name. Expanding a folder will display its subfolders in the folder browser. When initially expanding a folder two levels of subfolder are expanded. Each of these subfolders may be expanded as required.
- When a folder is selected, its files and subfolders are displayed in the file browser.

The **Show Files of Type** section is used to filter the types of files that are shown.

- The default filter All Files (*.*) shows all files.
- The filter Data Log Files (*.nlg) shows only data log files with the extension NLG.

Folders are not affected by the file filter and are always visible in the file browser.

The **File Browser** displays all subfolders and files contained within the selected folder. See the File Browser Commands section for more information on each operation.

- The browser displays the contents of a single folder. If multiple folders are selected in the folder browser, only the first selected folder is displayed in the file browser.
- A file may be selected in the file browser by clicking on it. When selected, the toolbar buttons that apply to the file are enabled.
- A subfolder displayed in the file browser may be opened by double clicking it. This selects the folder in the folder browser. It updates the file browser by displaying the contents of the newly selected folder.

The **File System Information** section shows the space on the selected drive.

- **Free Space** is the amount of space available
- **Total Size** is the total space on the drive

Folder Browser Operations

The Folder Browser Operations are described below. These operations are used to manage the folders in the selected file system.

New Folder

New Folder creates a new folder with the name “New Folder” in the file system. To create a folder:

- Select the parent folder in the folder view and click *New Folder* on the toolbar, or
- Right click on the parent directory in the folder view and select *New Folder* from the menu.

Delete Folder

Delete deletes an existing folder from the file system. Deleting a folder also deletes any files and folders contained within the folder. To delete a folder:

- Select the folder from the folder or file browser and click *Delete* on the toolbar, or
- Right click the folder in the folder or file browser and select *Delete* from the menu.

All folders other than the root may be deleted. If multiple folders are selected at once, all folders and files contained within are deleted.

Rename Folder

Rename changes the name of the selected folder. To rename a folder:

- Select the folder from the folder or file browser and click *Rename* on the toolbar, or
- Right click the folder in the folder or file browser and select *Rename* from the menu.
- Type the new folder name.

Empty strings, names longer than the 100 characters (this includes the entire path and file name), and names containing the / \ : * ? < > | characters are not permitted.

Copy Folder

Copy places a copy of the selected folder onto the clipboard. To copy a folder:

- Select the folder from the folder or file browser and click *Copy* on the toolbar, or
- Right click the folder in the folder or file browser and select *Copy* from the menu.

The selected folders and contained files are copied to the clipboard where they can be pasted to a local or remote file system. If multiple folders are selected at once, all folders and files contained within are copied.

Cut Folder

Cut places a copy of the selected folder onto the clipboard and marks it to be removed after it is pasted. To cut a folder:

- Select the folder from the folder or file browser and click *Cut* on the toolbar, or
- Right click the folder in the folder or file browser and select *Cut* from the menu.

This marks the folder by making the text and icon transparent. When the cut folder is pasted to a new location, the original folder is deleted. A folder can be pasted to a local or remote file system. If multiple folders are selected at once, all folders and files contained within are cut.

Paste Folder

Paste puts data from the clipboard into the selected location. To paste a folder:

- Select a parent folder in the folder or file browser and click *Paste* on the toolbar, or
- Right click a parent folder the folder in the folder or file browser and select *Paste* from the menu.

A pasted folder is given the same name as the original folder if possible. If a folder with the same name already exists in the parent folder the name will be “Copy of *foldername*”. If multiple copies of the same folder are pasted the name will be “Copy *number* of *foldername*”.

Refresh Folder

Refresh updates the content of a folder. If the folder is on the controller file system, the folder information is read again from the controller. This may take several seconds.

Manual refreshing is required on a folder since automatic updates (when new files or folders are created by a process and not the user) can be extremely time consuming and would force communication between the PC and controller every time a folder is modified, possibly at unexpected times.

File Browser Operations

The File Browser Operations are described below. These operations are used to manage the files in the selected file system.

Delete File

Delete removes an existing file from the file system. To delete a file:

- Select the file from the file browser and click *Delete* on the toolbar, or
- Right click the file in the file browser and select *Delete* from the menu.

Any file in the file system may be deleted. Multiple files may be selected.

Rename File

Rename changes the name of the selected file. To rename a file:

- Select the file from the file browser and click *Rename* on the toolbar, or
- Right click the file in the file browser and select *Rename* from the menu.
- Type the new file name.

Empty strings, names longer than the 100 characters (this includes the entire path and file name), and names containing the / \ : * ? < > | characters are not permitted.

Copy File

Copy places a copy of the selected file or files onto the clipboard. To copy a file:

- Select the file from the file browser and click *Copy* on the toolbar, or
- Right click the file in the file browser and select *Copy* from the menu.

The selected files are copied to the clipboard where they can be pasted to a local or remote file system. If multiple file are selected at once, all files selected are copied.

Cut File

Cut places a copy of the selected file onto the clipboard and marks it to be removed after it is pasted. To cut a file:

- Select the file from the file browser and click *Cut* on the toolbar, or
- Right click the file in the file browser and select *Cut* from the menu.

This marks the file by making the text and icon transparent. When the cut file is pasted to a new location, the original file is deleted. A file can be pasted to a local or remote file system. If multiple files are selected at once, all files selectedare cut.

Paste File

Paste puts files from the clipboard into the selected folder. To paste a file:

- Select a folder in the file browser and click *Paste* on the toolbar, or
- Right click the folder the the folder or file browser and select *Paste* from the menu.

A pasted file is given the same name as the original file if possible. If a file with the same name already exists in the folder the name will be “Copy of *filename*”. If multiple copies of the same file are pasted the name will be “Copy *number* of *filename*”.

Data Log Operations

Controllers can create data log files using the Telepace Studio DLGF Element. Data log files are stored on the controller internal file system and on USB mass storage devices. Controller data logs are stored in binary format to minimize file size. Files can be exported in CSV format for import into other programs.

See Exporting Data Logs to learn how to transfer and export data logs.

A File Export Summary is produced for each export operation.

See sections Log File Format and Export File Format for information the files used with data log operations.

Log File Format

Log file names are created using the Log file name entry in the DLGF Element Configuration dialog. When the files are saved to the internal drive or the mass storage device additional information is added to the file name. Data logs files are named according to the convention:

Annnnnn_logName_YYYYMMDD_HHMMSS.nlg

Where:

- Annnnnn is the controller ID

  logName is the log file name specified in the log configuration.

-
- YYYY is the 4 digit year the file was created
- MM is the 2 digit month the file was created
- DD is the 2 digit day of the month the file was created
- HH is the 2 digit hour (24 hour clock) the file was created
- MM is the 2 digit minute the file was created
- SS is the 2 digit second the file was created

The file name is path qualified to permit similar data log names on the same controller. i.e. directory1\log and directory2\log are both unique permitted names. The drive is excluded from the file name.

Export File Format

Exported Data Log file names have the following format:

ControllerID_LogName_SessionID.CSV

Where:

ControllerID is the Controller ID contained in the data log file.

LogName is the log name contained in the data log file. If the log name contains path information this will be included in the output file name. Path separator characters are replaced with underscores so the output file name is valid.

SessionID is the Session number contained in the data log file. The session number is generated internally when the data log is created. The session number is kept constant for the duration of the log and is used to differentiate between different logs in a controller.

Output CSV file format

The output CSV file contains a two line header and then the data records for the data log.

Header

ControllerID, LogName, Description, SessionNumber, Version, Number of Fields, BatchProcessTime

“Sequence”, “Type 1 String”, “Type 2 String”,“Type 3 String”,“Type 4 String”,“Type 5 String”,“Type 6 String”,“Type 7 String”,“Type 8 String”,

Data

Sequence, Value 1, Value 2, Value 3, Value 4, Value 5, Value 6, Value 7, Value 8,

A minimum of one (1) and a maximum of eight (8) fields can be created for each data log. The number of Type String and Data Value fields will depend on the data log configuration.

Where:

- **ControllerID**: The ID of the controller
- **LogName:** Log name field from the data log file.
- **Description:** Description field from the data log file.
- **SessionNumber:** session number from the header.
- **Version:** The version number from the data logfile. This value is fixed at 1.
- **Number of Fields:** The number of fields from the data log file.
- **BatchProcessTime:** The time and date the header and records were added to the file, in the form “Processed on date time”. The time and date are output in the short time/date format for the PC.
- **Type x String:** A string describing the data type for field x.
- **Sequence:** The sequence number of the record. This is used to uniquely identify each record in the log file.
- **Value:** The value of the field in the record.

The header information in the first two lines is not repeated for each file processed. It appears only once for each batch of files. The header information will appear more than once if the output file exists when the batch of files is processed.

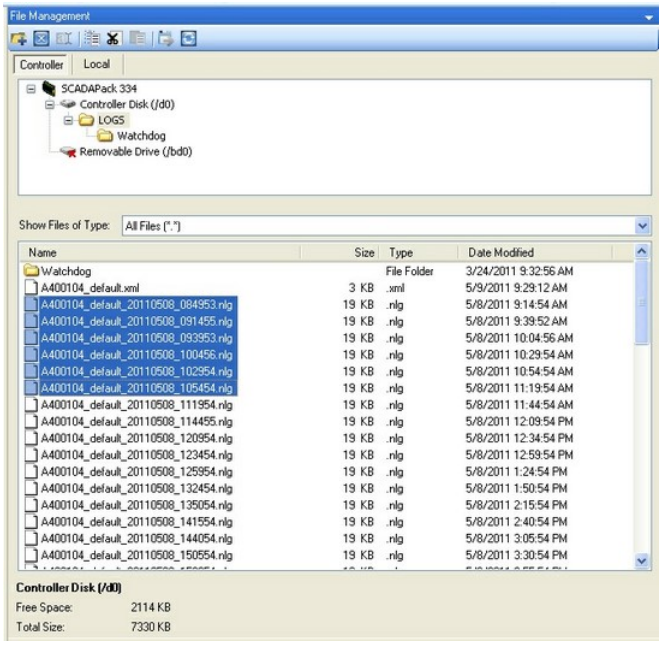
When an output file exists new data files are appended to the file.

Exporting Data Logs

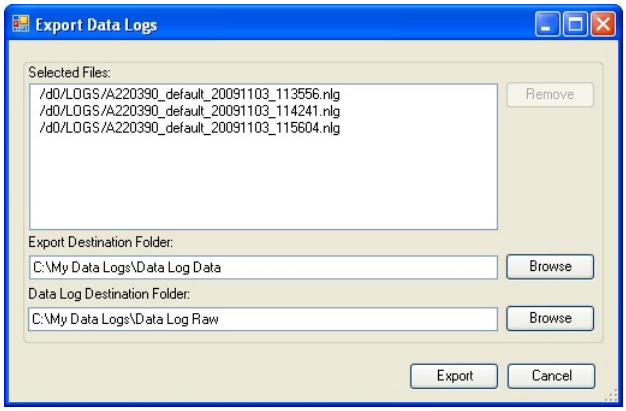
Data log files on the controller or the local file system can be exported as CSV files. The data can then be imported into other applications.

Use the following procedure to Export Data Log Files:

- From the File Management view select the **Controller** or **Local** file system and browse to the folder containing the files that are to be exported. The data log files will be listed in the File Browser section of the view.
- Select the data log .nlg files to be exported from File Browser window. Multiple files may be selected using the **CTRL** click and **Shift** click operations.



- Click the **Export** button on the File Management toolbar, or right-click the files to export and select **Export**. This opens the Export Data Logs window.



The **Selected Files** list shows which files will be exported. Files may be removed from the list if too many have been selected.

- Select one or more files.
- Click Remove, or right-click and select Remove from the menu.
- Removing a file from the list does not delete or alter the original file in any way. It simply removes it from the export list.

The **Selected Files** list needs to contain at least one file. This means that all files cannot be selected and then removed.

The **Export Destination Folder** selects where the exported files will be placed.

- Enter the destination folder, or click Browse to open a standard Windows Open Folder dialog to select a location. When manually entering a destination folder the path and folder must already exist. The destination is not automatically created by the export process. Use the Browse button and the the Create new Folder icon to create a destination folder.

The export destination folder is validated prior to commencing the export operation.

The **Data Log Destination** folder selects where raw data log files will be stored on the local PC. Files exported from a controller are copied to the local file system before exporting. This preserves a local copy of the files so they don't have to be copied from the controller more than once.

- Enter the destination folder, or click Browse to open a standard Windows Open Folder dialog to select a location. When manually entering a destination folder the path and folder needs to already exist. The destination is not automatically created by the export process. Use the Browse button and the the Create new Folder icon to create a destination folder.

The data log destination folder is available only when exporting from the controller file system. Files already on the local file system are not copied. The folder is validated prior to commencing the export operation.

Click **Export** to validate all fields and begin the export operation.

- A valid folder must be specified in the Export Destination Folder
- If exporting from the controller, a valid folder needs to be specified in the *Data Log Destination Folder*

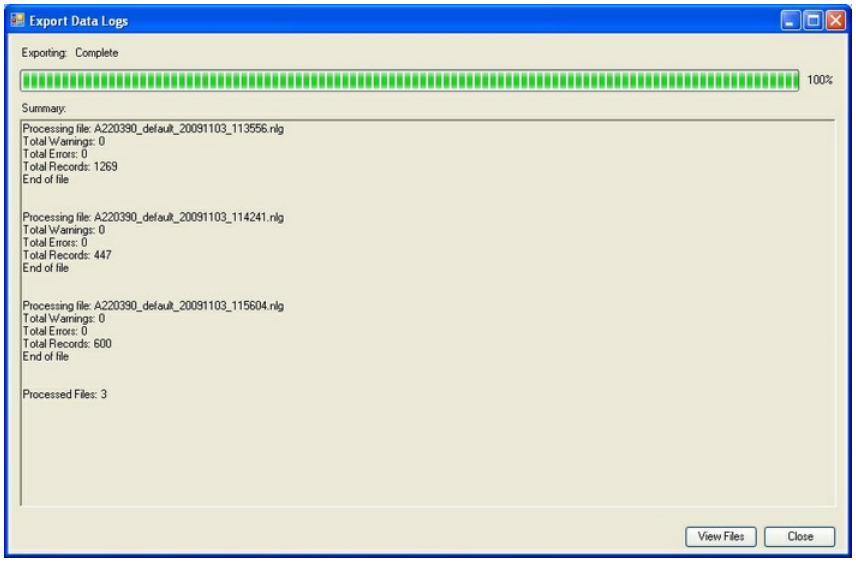
Exporting creates CSV files from the binary **.nlg** files. If files from a previous export are present in the Export Destination folder, new records are appended to the files.Files are exported to the *Export Destination Folder*. One file is created for each data log. See Export File Format for a description of the files.

If the files to be exported are located on the controller's file system, they are copied from the controller to the *Data Log Destination* folder on the local file system with their original names and in their original raw format. If a file of the same name already exists in the specified data log destination folder, the user is prompted to confirm whether or not the file should be overwritten. Files being exported from the local machine are not copied from their current location to the destination folder.

During the copy process, a communication progress dialog is displayed. Once this copy is complete, the file export begins. a progress dialog is displayed indicating which file is being exported and the percentage complete. Upon completion, the progress bar will display 100%, an export summary is displayed and the **View Files** button is enabled. Pressing the **View Files** button will open up the Export Destination folder in Windows Explorer. Pressing **Close** will close the Export Data Logs dialog.

File Export Summary

A summary is presented when Data Log Export is complete. It shows statistical data about the files processed along with warning and error messages.



Shortcut Keys

Shortcut keys are available for file management operations on both folders and files.

- Ctrl+C – Copy
- Ctrl+X – Cut
- Ctrl+V – Paste
- F2 – Rename
- Delete – Delete

FlowStation Group

The FlowStation group is located in the Configure ribbon. This group is only displayed when the controller type for the project is FlowStation 110.

The commands in the FlowStation group are used for managing FlowStation configurations. Telepace can be used to backup configuration files from a FlowStation 110 controller, restore a configuration to a FlowStation 110 controller that has no configuration, copy a configuration to a USB memory stick for automatic loading in the field or replicating a configuration to install in other FlowStation 110 controllers.

Telepace needs to be communicating with a target FlowStation 110 controller to use the FlowStation group functions. Telepace needs to reads the FlowStation controller ID when the FlowStation commands are selected. Refer to the PC Communications group description for details on establishing a connection with a target FlowStation 110 controller.

The **Backup Configuration** command reads configuration files from a FlowStation 110 and saves the files to the PC running Telepace.

The **Restore Configuration** command writes the configuration files from the PC running Telepace to a FlowStation 110.

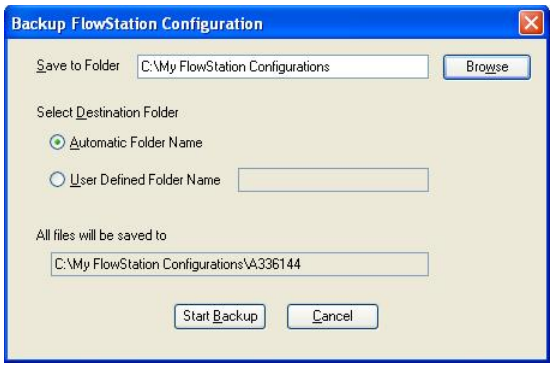
The **Copy Configuration** command copies the configuration files from the PC running Telepace to a USB memory stick.

The **Replicate Configuration** command creates FlowStation 110 configuration files from an existing configuration.

With the support of customized web content content during backup, replicate and restore functions, the number of configuration files is increased by one. These operations will take longer to complete than when custom content is not used. See the FlowStation Configuration Files for more information on the files used by the FlowStation.

Backup Configuration

The **Backup Configuration** command reads the FlowStation 110 **configuration files** from the FlowStation 110 controller and stores the files on the PC file system. When the Backup Configuration command is selected the Backup FlowStation 110 Configuration dialog is opened.



The **Save to Folder** window selects the root folder on the PC into which the backup configuration files will be saved.

The **Browse** button opens a dialog to browse for and select the root folder to be used to save the configuration files.

The **Automatic Folder Name** option sets the sub-folder name as the FlowStation 110 controller ID. The FlowStation 110 controller ID is read from the target FlowStation 110 controller when the Backup Configuration command is selected. This is the default selection when the dialog is opened.

Then **User Defined Folder Name** option sets the sub-folder name to the path entered in the User Defined Folder Name window.

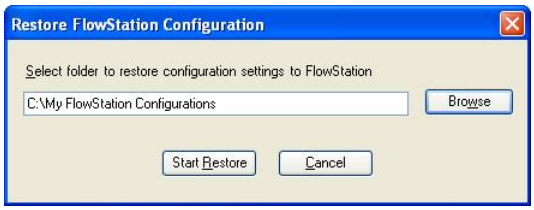
The **All files will be saved to** windows displays the full path that the backup configuration files will be saved to on the PC..

The **Start Backup** button starts the backup configuration process if full path name is valid.

The **Cancel** button closes the dialog without backing up the configuration files.

Restore Configuration

The **Restore Configuration** command reads the FlowStation 110 **configuration files** from the PC file system and writes them to the target FlowStation 110 controller. When the Restore Configuration command is selected the Restore FlowStation 110 Configuration dialog is opened.



The **Select folder to restore configuration settings to FlowStation** window is the folder on the PC that the FlowStation configuration files will be restored from.

The **Browse** button opens a dialog to browse for and select the root folder to be used to restore the FlowStation 110 configuration files from.

The **Start Restore** starts the restore process and loads the files in the selected folder to the target FlowStation 110 controller.

The **Cancel** button closes the dialog without restoring the configuration files.

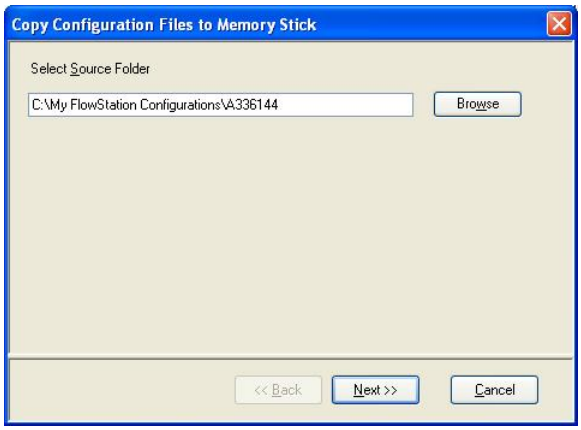
When the Restore Configuration command is executed the target FlowStation controller will reboot. This is normal operation.

Copy Configuration

The **Copy Configuration** command copies selected FlowStation **configuration files** from the PC file system and copies them to a USB memory stick. The files can then be taken to another FlowStation 110 and auto copied to the FlowStation 110.

The Copy Configuration command is used to select the PC folder where source configuration files stored and the destination folder where the configuration files will be copied to. The destination folder name may be an exact controller ID so that files in the folder can only be restored to a FlowStation 110 controller with the same controller ID. The destination folder name may also be any folder name. This allows configuration files in the folder to be restored to any FlowStation 110 controller.

When the Copy Configuration command is selected the Copy Configuration Files to Memory Stick dialog is opened. This dialog is the start of a three step wizard.



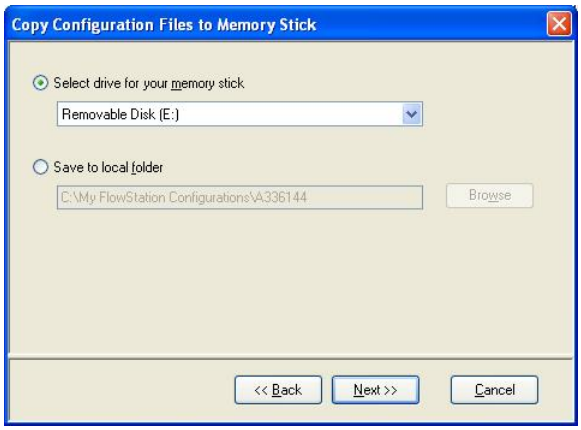
The **Select Source Folder** window selects the root folder on the PC from which the configuration files will be copied from.

The **Browse** button opens a standard dialog to select the Folder containing the configuration files.

The **Back** button is disabled.

The **Cancel** button stops the wizard process and closes the dialog.

The **Next** button starts the second step of the wizard and opens the destination folder dialog.



The **Select drive for your memory stick** drop down menu allows the selection of the destination PC drive to copy the files to.

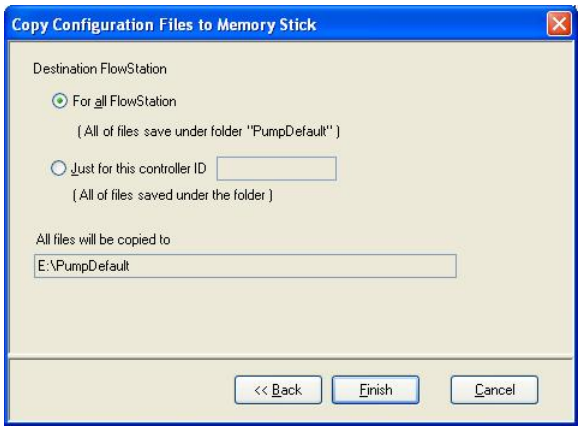
The **Save to local folder** option allows the configuration files to be copied to another folder on the PC. Enter a valid path name or use the Browse button to navigate to the folder.

The **Browse** button opens a standard dialog to select the Folder containing the configuration files.

The **Back** button returns to the first step in the Copy Configuration wizard.

The **Cancel** button stops the wizard process and closes the dialog.

The **Next** button starts the last step of the wizard and opens the destination FlowStation 110 options dialog.



When the FlowStation 110 configuration files are saved to the USB memory stick they can be available for any FlowStation 110 controller or a specific FlowStation 110 controller.

The **For all FlowStation** option sets the destination folder on the USB memory stick to PumpDefault. The configuration files in this folder can be copied to any FlowStation 110 controller.

The **Just for this controller ID** option enables the entry of a specific controller ID, or any valid folder name. When a controller ID is entered the configuration files may be auto loaded to only the FlowStation 110 controller with the entered controller ID.

The **Back** button returns to the second step in the Copy Configuration wizard.

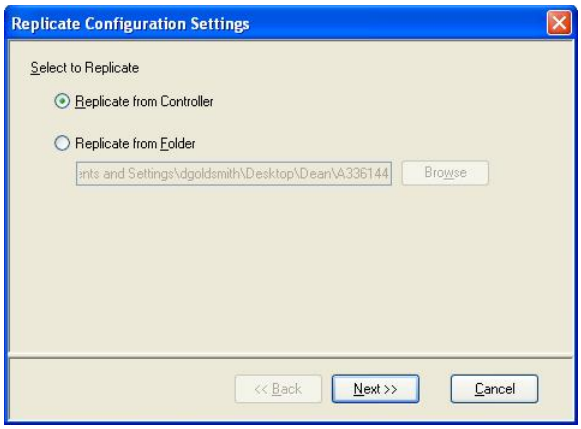
The **Finish** button completes the Copy Configuration operation.

The **Cancel** button stops the wizard process and closes the dialog.

Replicate Configuration

The **Replicate Configuration** command creates a copy of existing FlowStation **configuration files** and allows the modification of selected parameters in the configuration. The replicated configuration is then saved. This feature allows the user to copy a FlowStation 110 configuration as a starting point for a new installation.

When the Replicate Configuration command is selected the Replicate Configuration Settings dialog is opened. This dialog is the start of a three step wizard.



FlowStation 110 configurations can be replicated from a connected FlowStation 110 controller or existing configuration files on a PC.

The **Replicate from Controller** option will cause Telepace to read the configuration files from the connected FlowStation 110 controller.

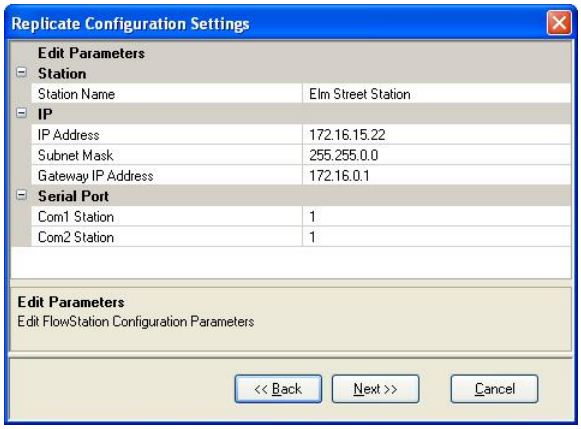
The **Replicate from Folder** option allows the selection of configuration files from the PC file system.

The **Browse** button opens a standard dialog to select the Folder containing the configuration files.

The **Back** button is disabled.

The **Cancel** button stops the wizard process and closes the dialog.

The **Next** button starts the second step of the wizard and opens the Edit Parameters dialog.



The FlowStation 110 settings that can be modified during the Replicate Configuration process are the Station Name and ID, the IP settings for the Ethernet port and the Modbus station address for Com ports 1 and 2. The settings are modified by entering the new values in the parameter window.

The **Station ID** window is used to modify the FlowStation 110 station identifier. The valid length for the station ID is from 0 to 20 characters.

The **Station Name** window is used to modify the FlowStation 110 station name. The valid length for the station name is from 0 to 20 characters.

The **IP Address** window is used to modify the FlowStation 110 IP address. The format is nnn.nnn.nnn.nnn.

The **Subnet Mask** window is used to modify the FlowStation 110 Subnet mask. The format is nnn.nnn.nnn.nnn.

The **Gateway IP Address** window is used to modify the FlowStation 110 Gateway address. The format is nnn.nnn.nnn.nnn.

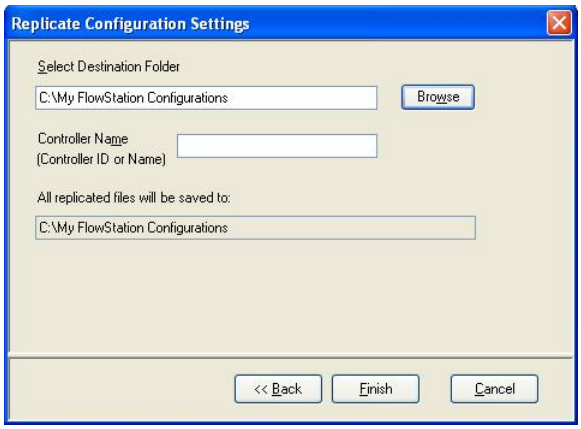
The **Com1 Station** window is used to modify the Modbus address for Com1. The valid range is from 1 to 255 if addressing is Standard, or from 1 to 65535 if addressing is Extended.

The **Com2 Station** window is used to modify the Modbus address for Com2. The range is from 1 to 255 if addressing is Standard, or from 1 to 65535 if addressing is Extended.

The **Back** button returns to the first step in the Replicate Configuration wizard.

The **Cancel** button stops the wizard process and closes the dialog.

The **Next** button starts the last step of the wizard and opens the folder to save the replicated configuration.



When a FlowStation 110 configuration file is replicated the new configuration file is stored to a folder on the PC. The last dialog in the wizard selects the folder.

The **Select Destination Folder** window selects the root folder on the PC that the replicated configuration files will save to. Type in the folder name or select the browse button to navigate to the folder.

The **Browse** button opens a standard dialog to select the Folder containing the configuration files.

The **Controller Name** window selects the sub-folder for saving the configuration. The sub-folder name may be the FlowStation 110 controller ID or any other valid folder name.

The **All replicated files will be saved to:** window displays the folder name the files will be stored in.

The **Back** button returns to the second step in the Replicate Configuration wizard.

The **Finish** button saves the replicated files to the destination folder.

The **Cancel** button stops the wizard process and closes the dialog.

FlowStation Configuration Files

Telepace uses a number of files to store the complete FlowStation configuration. Configuration can be saved toa USB memory stick or to a PC when the FlowStation Configurator or Telepace Studio Studio is used.

Inserting a USB mass storage device in the FlowStation also allows configuration to be loaded. A loaded configuration may include a variable number of configuration files depending upon what configuration is being loaded.

The files are:

customweb.dat	This file contains the custom web contents data file which contains the data of the custom web files.
customweb.toc	This file contains the custom web contents index file which contains the full path of the custom web file, the name of the file and the size of the file.
database.bin	This file contains the database values that the Ladder Logic may need to allow the Ladder Logic program to be completely restored.
FlowStation.csv	This file contains the configuration items defining the operation of the FlowStation. It contains one CSV record per line, as an address / value pair. When this file is loaded by the controller from the USB mass storage device the address, value pairs are read and applied to the FlowStation database. A complete FlowStation configuration can be created by editing this file.
Ladder.bin	This file is used during firmware replacement to store existing user Ladder Logic code (separate from the FlowStation code) so that the logic can be restored when the new firmware image is started.
Ladder.fp	This file is used to save the floating-point constants that existing user Ladder Logic needs. This allows the Ladder Logic program to be completely restored.
Securitydata.bin	This file contains FlowStation security information. It can be copied between FlowStations so the same user security is available
Tvs.bin	The tag value system file contains FlowStation settings such as serial port settings, IP settings, and register assignment settings.

FlowStation CSV File Format

The CSV file format has 1 record per line with each record using the following format:

<Register Number>, <Format>, <Value>

The <**RegisterNumber**> parameters is the Modbus address to be updated, or in the case of a multi-address parameter the first address of the group.

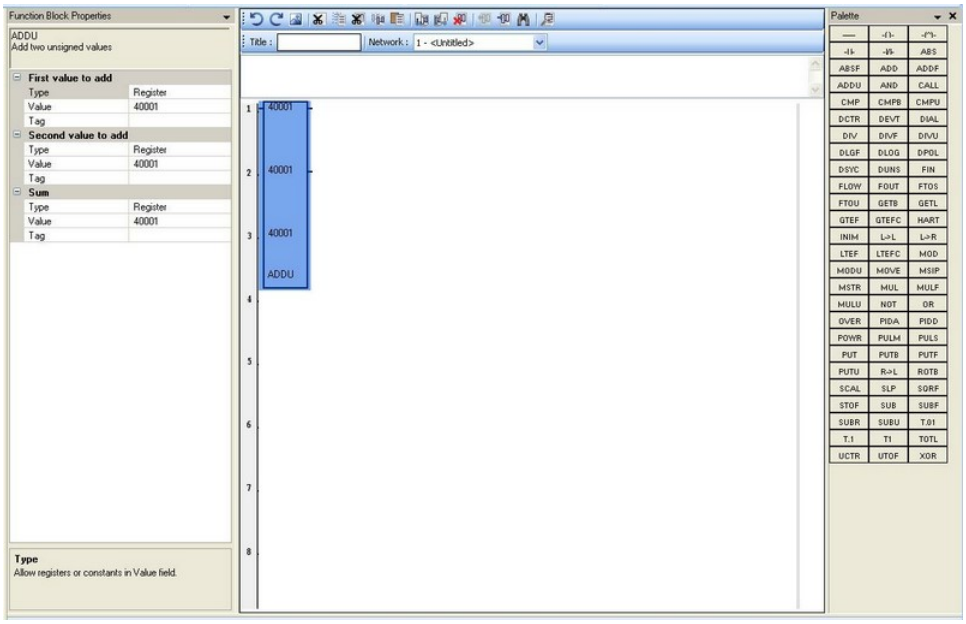
The <**Format**> parameter specifies the format that the <**Value**> is in. The format is used for converting the supplied <**Value**> to the format required by the FlowStation.

Valid formats are:

UINT16	The value is an unsigned 16 bit decimal value.
INT16	The value is a signed 16 bit decimal value.
UINT32	The value is an unsigned 32 bit decimal value.
INT32	The value is a signed 32 bit decimal value
Boolean	The value is a binary state.
FLOAT32	The value is a floating point value.
STRING[x]	x characters will be written to the database starting at the specified address, with 2 characters being written to each register.

Ladder Canvas

The ladder canvas is the area used to create and document ladder networks used in an application.



The Ladder Canvas Toolbar contains commands associated with creating and editing a Ladder Logic programs.

The Palette contains the function blocks that can be used for ladder logic applications.

The Properties view contains the editing properties for the selected function block.

The ladder logic network is the current 8 row by 10 column network.

The **Title** window is used to edit the title of the current network. The network title is entered into the window at the right of the Title icon. The network title may be a maximum of 32 characters long.

The **Network** command is used to move the Network Canvas to a desired network. A drop down list shows multiple network titles in use. Networks that have no title are displayed showing the network number only.

Monitor Element

The Monitor Element command is used to add registers used by selected elements to the Register Viewer. Registers used by the currently selected element, or elements, are added to the Register Viewer. This command is available only if an element, or multiple elements, is selected.

This command may be selected by right click your mouse button on a highlighted element, or group of elements, and selecting **Monitor Element** from the list of commands.

To add registers to the Register Viewer using the Monitor Registers command:

- Highlight an element by clicking the left mouse button on an element or dragging the cursor over a group of elements.
- Select the **Monitor Element** command from the Home View or from the mouse right click menu.

When the Monitor Element command is selected the Monitor Element dialog is displayed.

The **Existing Groups** list box selects which group the registers are added to. A drop down selection displays the created groups. If no groups have been created use the Add to Group field to create a Group.

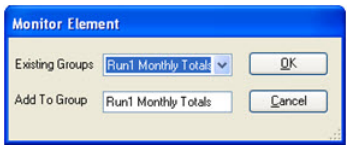
The **Add To Group** field displays the group selected in the Existing Groups list box if there are groups already created.

- To Add a new Group type a group name into the Add a Group field.

The **OK** button adds the registers to the Group and closes the dialog. The registers are added to the Register Viewer in the format used by the selected element or elements.

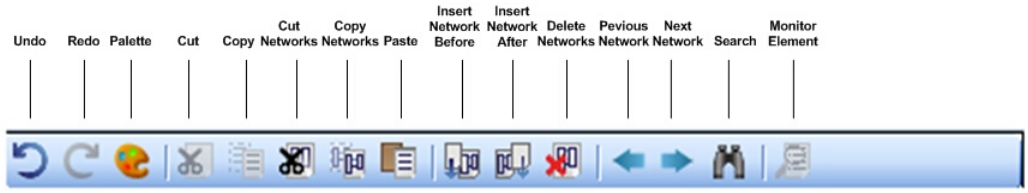
- The Available status is set to **Online** for registers except registers contained in elements that use Element Configuration.
- The Available status is set to **Always** for registers contained in elements that use Element Configuration.

The **Cancel** button closes the dialog without adding the registers to the Register Viewer.



Ladder Canvas Toolbar

The Ladder Canvas Toolbar is located at the top of the Ladder Canvas. The toolbar provides edit, navigation and search functionality for users creating Ladder Logic applications. For further details on the Ladder Canvas refer to the Ladder Canvas topic. Each of the toolbar functions is explained in the highlighted listing below.



Click on the highlighted command listing below for a detailed description of the command function.

Undo

The **Undo** command reverses changes made in the Network Canvas or Comment editor. This is a multilevel command with a virtual unlimited number of **Undo** actions.

Redo

The **Redo** command reverses the changes made with Undo command.

Palette

Cut

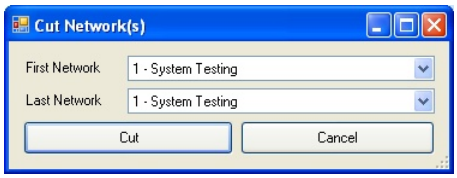
The **Cut** command removes the selected elements in the Network Canvas or the Comment editor and puts them on the Clipboard. Items on the clipboard remain there until they are replaced by other selected elements.

Copy

The **Copy** command copies the selected elements in the Network Canvas to the clipboard. Items on the clipboard remain there until they are replaced by other selected elements.

Cut Network(s)

The **Cut Network(s)** command cuts a network or sequential block of networks to the clipboard. Comments for the network are also cut to the clipboard. When the Cut Networks command is selected the Cut Networks dialog is opened. Continuous networks from the First Network selection to the Last Network selection may be cut to the clipboard.



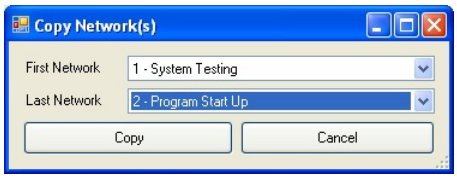
To cut a network or sequential block of networks:

- Select the **First Network** to cut using the drop down menu selection at the right of the window. The networks displayed will be the current network and any networks previous to the current network.
- Select the **Last Network** to cut using the drop down menu selection at the right of the window. The networks displayed will be the current network and any networks after the current network.
- Click the **Cut** button to cut the selected networks to the clipboard.
- Click the **Cancel** button to cancel the cut networks command.

Copy and Paste are not be supported between multiple instances of Telepace. This means that a cut or copy in one instance of Telepace will **not** be paste-able into another instance of Telepace. To Copy and Paste between Telepace projects the source project is opened and the cut or copy commands are used. The destination Telepace project needs to be opened, thus closing the source Telepace project, and then the paste action performed.

Copy Network(s)

The **Copy Network(s)** command copies a network or sequential block of networks to the clipboard. Comments for the network are also copied to the clipboard. When the Copy Networks command is selected the Copy Networks dialog is opened. Continuous networks from the First Network selection to the Last Network selection may be copied to the clipboard.



To copy a network or sequential block of networks:

- Select the **First Network** to cut using the drop down menu selection at the right of the window. The networks displayed will be the current network and any networks previous to the current network.
- Select the **Last Network** to cut using the drop down menu selection at the right of the window. The networks displayed will be the current network and any networks after the current network.
- Click the **Copy** button to copy the selected networks to the clipboard.
- Click the **Cancel** button to cancel the copy networks command.

Copy and Paste are not be supported between multiple instances of Telepace. This means that a cut or copy in one instance of Telepace will **not** be paste-able into another instance of Telepace. To Copy and Paste between Telepace projects the source project is opened and the cut or copy commands are used. The destination Telepace project needs to be opened, thus closing the source Telepace project, and then the paste action performed.

Paste

The **Paste** command is used to paste elements that have been cut or copied to the clipboard to the current cursor position. The paste command is grayed out when there are no items on the clipboard or the cursor is in the wrong pane for the clipboard items. For example text cannot be pasted into the Ladder Editor pane and network elements cannot be pasted into the Comment Editor pane.

The **Paste** command is also used to paste any networks that have been cut or copied to the clipboard. When the Paste Networks(s) command is selected the Paste Network(s) dialog is displayed to control where and how the network(s) are pasted.

The Paste Network(s) dialog selections include whether to insert before the current network of after the current network. The network title(s) and comment(s) and element configurations are pasted.

When element configuration is copied, the **values** of the element configuration will be copied, although the registers used to store these values will change. If element configuration is not copied, the default values for that element's configuration will be used.

Copy and Paste are supported between multiple instances of Telepace. This means that a cut or copy in one instance of Telepace can be pasted into another instance.

Insert Network Before

The **Insert Before** command inserts an empty network before the current network. The Network Canvas cursor moves to column 1 Row 1 of the new network.

Insert Network After

The **Insert After** command inserts an empty network after the current network. The Network Canvas cursor moves to column 1 Row 1 of the new network.

Delete Network

The **Delete Network** command deletes the current network and the elements in it. Deleting a network does not remove the Tag names from the Telepace project.

The Confirmation dialog is displayed if the **Prompt me when deleting networks** option is selected in the Telepace Options command.

- Select **Yes** to delete the network and close the dialog.
- Select **No** to close the dialog without deleting the network.
- Check the **Don't ask me this again** check box to disable the Prompt me when deleting networks option. When the option is disabled the confirmation dialog is no longer displayed.

Previous Network

Selecting the **Previous Network** command moves the Network Canvas to the previous Network in the program. The Previous command is grayed out if the current network is the first network.

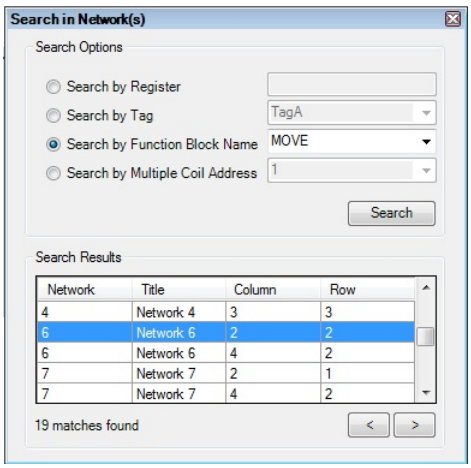
Next Network

Selecting the **Next Network** command moves the Network Canvas to the next Network in the program. The next network command is grayed out if the current network is the last network.

Search

The **Search** command is used to locate addresses, function blocks, tag names and instances of multiple coils in the current Ladder Logic program.

When the Search command is selected the Search in Network(s) dialog is opened.



Search Options

The **Search by Register** radio button activates the register select window. The user may enter a numeric register address for which to search.

The **Search by Tag** radio button activates the tag select window. Tags used in the ladder logic program are displayed when the drop down menu is selected. The user may begin typing the desired tag name to filter the drop down list items.

The **Search by Function Block** radio button activates the function block select window. Function blocks used in the ladder logic program are displayed when the drop down menu is selected. The user may begin typing the desired function block name to filter the drop down list items.

The **Search by Multiple Coil Address** radio button activates the Multiple Coil address select window. All instances of multiple coils used in the ladder logic program are displayed when the drop down menu is selected. The user may begin typing the desired multiple coil register address to filter the drop down list items.

The **Search** button begins the search of the entire program based on the parameters selected in the Search Options group, and lists the results in the table in the Search Results group.

Search Results

The table in the Search Results group will be populated with the returned search results. The first search result from the current network position will be automatically selected in the ladder canvas. Should no result be found on the current network or later, the search will wrap and the first search result in the program will be selected in the ladder canvas. The subsequent search results can be navigated using the previous (<) and next (>) buttons, or by double clicking a specific search result in the table.

The total number of search results is indicated in the lower left corner, below the search results table. Should the ladder logic or tags change after performing a search, the text indicating the number of found search results will be denoted in red with an asterisk and accompanied by a tooltip indicating that the search results may no longer be accurate and a new search should be performed.

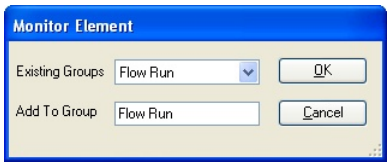
Monitor Element

The **Monitor Element** command is used to add registers used by selected elements to the Register Editor. All registers used by the currently selected element, or elements, are added to the Register Editor. This command is available only if an element, or multiple elements, is selected.

This command may be selected by right click your mouse button on a highlighted element, or group of elements, and selecting Monitor Element from the list of commands.

To add registers to the Register Editor using the Monitor Registers command first highlight an element by clicking the left mouse button on an element or dragging the cursor over a group of elements. Then select the Monitor Element command from the Program Ribbon - View group or from the mouse right click menu.

When the Monitor Element command is selected the Monitor Element dialog is displayed.



The **Existing Groups** list box selects which group the registers are added to. A drop down selection displays all created groups. If no groups have been created use the default group Group 1 is displayed.

The **OK** button adds the registers and closes the dialog. The registers are added to the Register Editor in the format used by the selected element or elements.

- The **Available** status is set to **Online** for all registers except registers contained in elements that use Element Configuration.
- The **Available** status is set to **Always** for all registers contained in elements that use Element Configuration.

The **Cancel** button closes the dialog without adding the registers to the Register Editor.

The Add To Group window displays the group selected in the Existing Groups list box. A new group may be created by simply typing the name of the new group in the Add To Group window. The group name may contain letters, numbers, and spaces. The group name may be 1 to 32 characters long and the name is not case sensitive.

The **Cancel** button closes the dialog without adding the element to a group..

Contact Monitoring

It is often necessary, while in **Monitor Online mode**, to determine whether a contact would pass power if power were supplied to it. This feature is extremely useful when testing the operation of ladder logic programs.

In **Monitor Online mode**, Telepace shows the power flow through the network. It also colors contacts that are not powered to show how power would flow if they were.

The following diagram illustrates how power flow information is displayed in Monitor Online mode:

In the above example, normally open contact 10001 and normally closed contact 10002, are not energized. The shaded background, in contact 10002, indicates power would flow through the contact if the input side were powered.

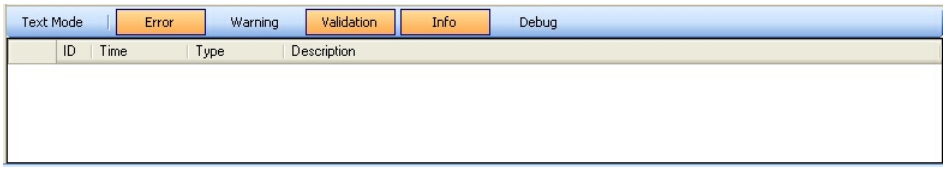
- The background of a normally open contact is colored when it is energized and its input is not powered.
- The background of a normally closed contact is colored when it is not energized and its input is not powered.

The background is only displayed on contacts if the input side of the contact is not powered.

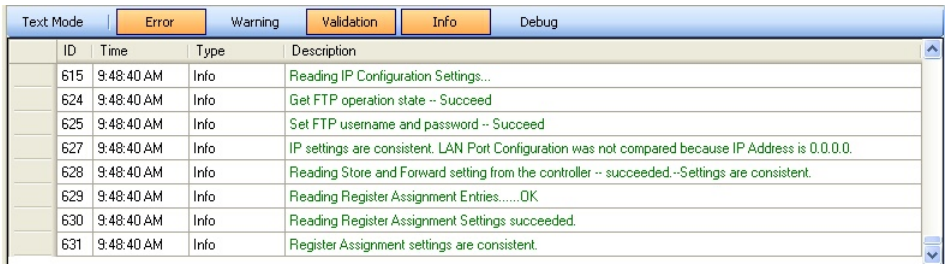
Diagnostics View

The **Diagnostics view** provides useful information that can be used to diagnose any programming or configuration conditions that arise during development. The Debug view is opened automatically when Telepace starts.

When Telepace first opens the Diagnostics view contains no data as seen in the graphic below.



As Telepace applications are Read from or Written to the target controller diagnostic data is displayed.



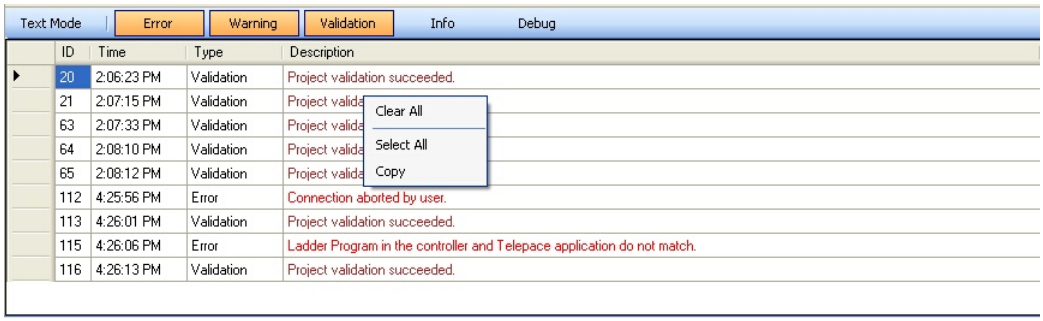
The Diagnostics view displays error, warning, validation, informational, and debug messages in one place.

- The **ID** column in the display contains an ID number to identify the diagnostic message.
- The **Time** column displays the PC time the diagnostic message occurred.
- The **Type** Column displays the type of diagnostic message, Error, Warning, Validation, Information or Debug.
- The **Description** column displays a description of the diagnostic message.

Click on any column header to sort the list by the items in that column.

The buttons across the top of the Diagnostics view display the selected diagnostic message type.

Right clicking anywhere on the diagnostics window will bring up a menu including **Clear All**, **Select All** and **Copy**. Clear All will remove all current rows from the table or text display. **Select All** will select all the rows in the table or text display. **Copy** will move all the selected text or rows, in the order they are displayed, to the Windows Clipboard allowing the user to paste into other programs. The table mode, when copied, can be pasted into a MS Excel document with the columns preserved. The right click menu will be as follows:



The Diagnostics view data is maintained during the current Telepace Studio session only.

- When Telepace is started and a project is opened, the Diagnostics window contains only new data from when Telepace Studio was started.
- The Diagnostic View data is persisted through the Current Telepace Studio session, no matter how many projects are opened or closed during the session.
- When Telepace Studio is closed all Diagnostic view data is lost.

Telepace Studio Function Block Reference

Telepace Studio ladder logic programming uses function blocks, connected in a meaningful way, to form a logic program. All available function blocks are accessed in a number of ways using the Ladder Canvas.

- Function Block Palette
- Function Block Menu
- Function Block Auto Complete Keyboard Insert

Function Block Palette

The function block palette is located at the right of the ladder canvas and contains buttons for all function blocks. To insert a function block into the ladder canvas click the button for the function block and drag it to the necessary location on the ladder canvas. The properties view will change to an edit view for the selected function block.

The palette automatically adjusts for the function blocks available to the selected controller type. For example when a SCADAPack controller is selected the MSIP function will not be displayed in the palette as the SCADAPack controller does not support ethernet communication.

Use the buttons below to select the help topic for the function block. Hover your mouse over each button to get a description of the function block.

-----	-()-	-(^)-	-||-
-|/|-	ABS	ABSF	ADD
ADDF	ADDU	AND	CALL
CMP	CMPB	CMPU	DCTR
DEVT	DIAL	DIV	DIVF
DIVU	DLGF	DLOG	DPOL
DSYC	DUNS	FIN	FLOW
FOUT	FTOS	FTOU	GETB
GETL	GTEF	GTEFC	HART
INIM	L->L	L->R	LTEF
LTEFC	MOD	MODU	MOVE
MSIP	MSTR	MUL	MULF
MULU	NOT	OR	OVER
PIDA	PIDD	POWR	PULM
PULS	PUT	PUTB	PUTF
PUTU	R->L	ROTB	SCAL
SLP	SQRF	STOF	SUB
SUBF	SUBR	SUBU	T.01
T.1	T1	TOTL	UCTR
UTOF	XOR		

Function Block Menu

Right clicking the mouse in any empty cell opens the menu selection for function blocks. The function blocks displayed will depend on the room available in the ladder canvas location.

For ease of navigation Individual Function Blocks are grouped based on their functionality.

The Logic group displays the basic logic functions used in logic programs.

The Math group displays the math functions used in logic programs.

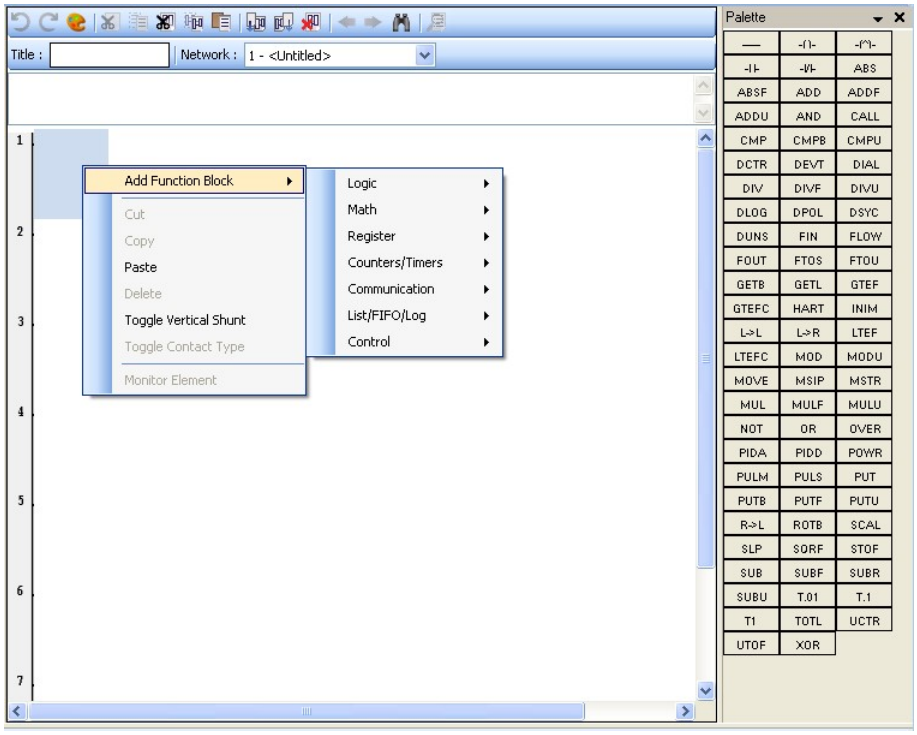
The Register group displays the register manipulation functions used in logic programs.

The Counters/Timers group displays the timer and counter functions used in logic programs.

The Communication group displays the functions used for communication with other controllers.

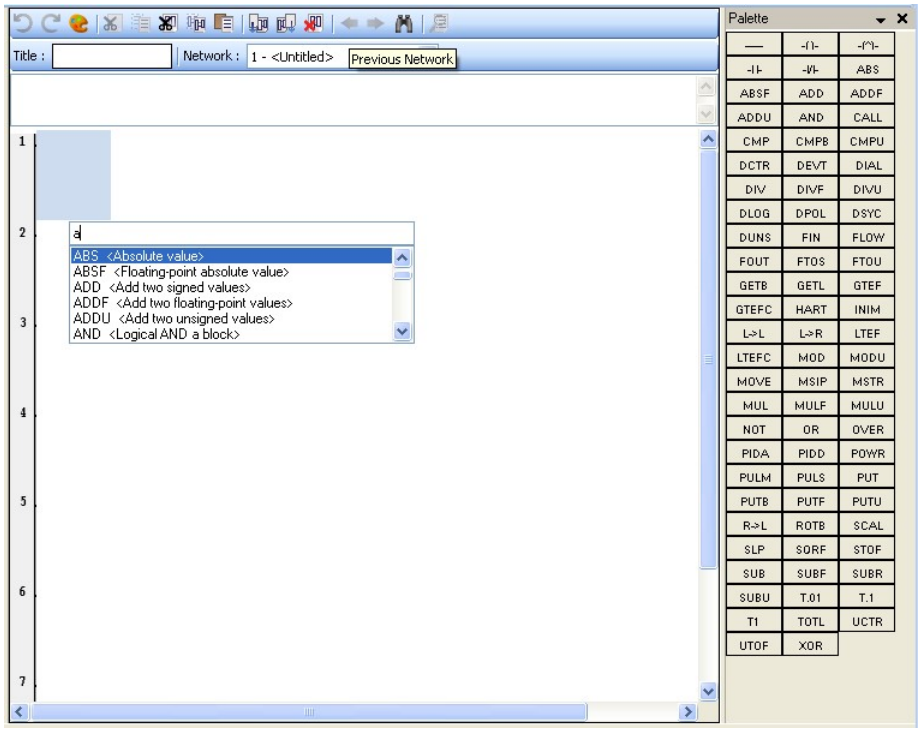
The List/FIFO/Log group displays the datalog functions and functions used to control register lists..

The Control group displays the functions used for control and in logic programs.



Function Block Auto Complete Keyboard Insert

Select a cell in the ladder canvas and then start to type the letters of the function block name. A list of function is displayed and may be selected using the mouse pointer or the up / down keys and pressing enter.



Register Types

Telepace Studio function blocks support a number of different register formats. The type of register format used will depend on the function block type. The following table defines the register formats used.

Unsigned Integer	Unsigned integer data are in the range 0 to 65535. This data requires a single 16-bit register, 3xxxx or 4xxxx.
Signed Integer	Signed integer data are in the range –32768 to 32767. This data requires a single 16-bit register, 3xxxx or 4xxxx.
Unsigned Double	Unsigned long integer data are in the range 0 to 4,294,967,295. This data requires two 16-bit registers, 3xxxx or 4xxxx. The lower numbered register contains the lower order word.
Signed Double	Signed long integer data are in the range –2,147,483,648 to 2,147,483,647. This data requires two

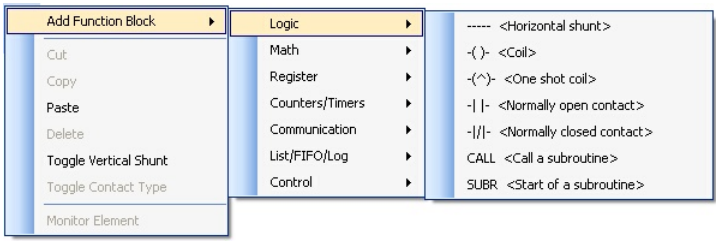
16-bit registers, 3xxxx or 4xxxx. The lower numbered register contains the lower order word.

Floating Point Floating-point data are in the IEEE single precision floating-point format. This data requires two 16-bit registers, 3xxxx or 4xxxx. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

String Format The first register contains the first two characters of the string. The first character is in the low order byte, the second in the high order byte.

Logic Functions

The Logic functions available in Telepace Studio are shown in the image below. This image is opened in Telepace Studio by right clicking on the ladder canvas and selecting the Add Function Block command. Depending on the available space on the ladder canvas some functions may not be displayed as they would not fit in the canvas at the location selected.



Hover your mouse over the icons below for a description of the function. Click the icon to go to the function description.

-----	-()-	-(^)-	-||-	-|/|-	CALL	SUBR

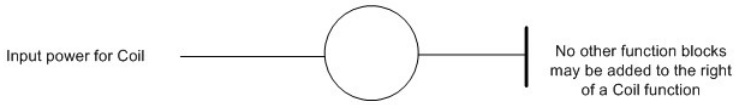
Shunts

The **horizontal** shunt conducts power from left to right.

The **vertical** shunt conducts power between rows. If any of the rows connected by vertical shunts are powered, all output rows will be powered, as shown in the example below. Use the **F5** key to insert a vertical shunt. Use the **F6** key to insert a horizontal shunt.

Coil

The **COIL** function block enables and **output** coil when its **input power** is ON.



Function Block Properties

The **COIL** function block has one parameter.

Type Register

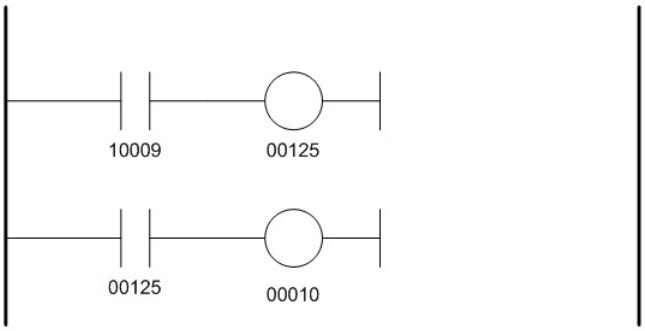
Value Any valid 0xxxx (coil register)

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

-(^)-

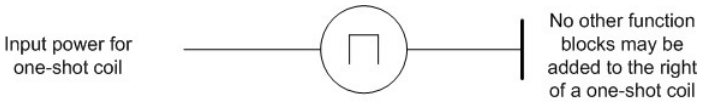
Example



When discrete input contact 10009 is closed power is applied to the **input power** input of coil 01025. This will enable the coil and the associated contact of 01025 is energized. The NO contacts 01025 are closed and this causes power to be applied to output coil 00010.

One Shot Coil

The **One Shot Coil** function block enables an output coil for one scan when the input power changes from OFF to ON.



Function Block Properties

The **One-Shot Coil** function block has one parameter.

Type	Register
Value	Any valid 0xxxx (coil register)
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

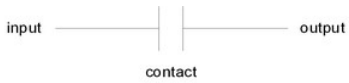
-()-

Example

One shot coil 00004 is energized for one scan on power up of the ladder logic program.

Normally Open Contact

The normally open contact, **NO**, function block conducts power when the coil or status register is ON.



Function Block Properties

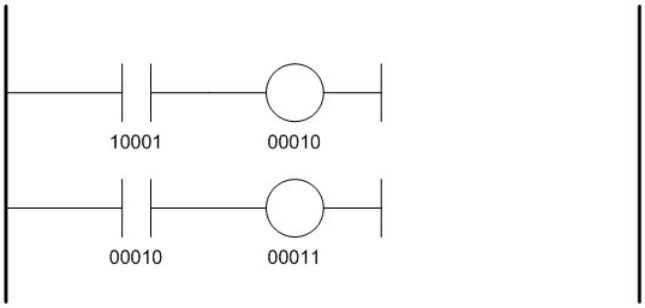
The **Normally Open contact** function block has one parameter.

Type	Register
Value	Any valid 0xxxx (coil register) or 1xxxx (status register)
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

-|/|-

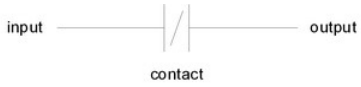
Example



This network shows **NO** contact 10001 (status register) and **NO** contact 00010 (coil register). When contact 10001 closes, power is applied to coil 00010 which causes its contact 00010 to close. This energizes coil 00011.

Normally Closed Contact

The **NC** contact function block conducts power when the coil or status register is OFF.



Function Block Properties

Type	Register
Value	Any valid 0xxxx (coil register) or 1xxxx (status register)
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

-||-

Example

Network 1 shows NO contact 10003 and **NC** contact 01041. Power conducts through **NC** contact 01041 keeping coil 00010 ON. When **NO** contact 10003 is closed coil 01041 turns ON and **NC** contact 01041 will open, turning OFF coil 00010.

CALL (Execute Subroutine)

The **CALL** function block transfers execution from the **main** program to a **subroutine**. Execution of the main program is suspended until the subroutine returns.

Subroutine calls can be used to execute parts of a program only when needed. The subroutine is called only if the input to the **CALL** function block is ON.

The function block variable *number* indicates which subroutine to call. The subroutine number needs to correspond to the number of a SUBR function block.

Function Block Properties

Type	Constant
Value	Specifies the number of the subroutine. Any number in the range 1 to 500 is valid.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

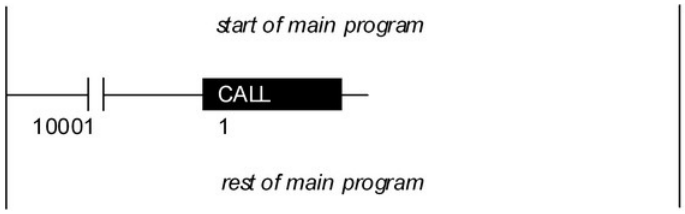
Related function blocks

SUBR

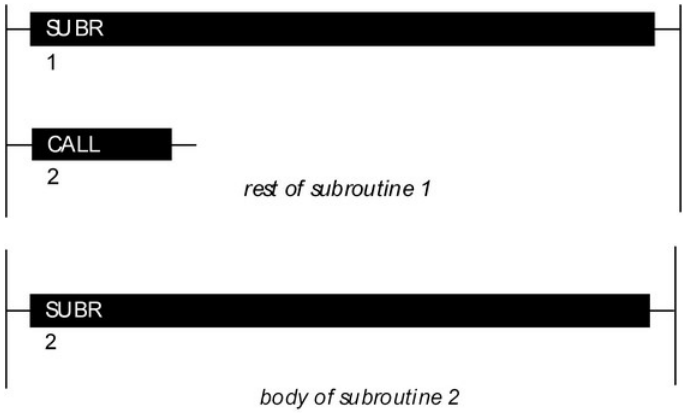
Example

This example calls subroutine 1 when input 10001 is on. Subroutine 1, in turn, calls Subroutine 2. The sequence of execution is as follows.

- The networks in the start of the main program are executed.
- If input 10001 is on, Subroutine 1 is called.
- At the beginning of Subroutine 1, Subroutine 2 is called.
- The body of Subroutine 2 is executed.
- The rest of Subroutine 1 is executed.
- The networks in the rest of the main program are executed.
- When execution reaches the start of Subroutine 1, the main program is complete.



Execution starts over at the start of the main program.



SUBR (Start of subroutine)

The **SUBR** function block defines the start of a **subroutine**.

A subroutine is a group of logic networks that can be executed conditionally. The subroutine begins with the network containing the **SUBR** function block. A subroutine ends when another subroutine function block is encountered, or when the end of the ladder logic program is reached.

The function block is a single cell element. Visually it covers the entire width of the first row of the network.

Function Block Properties

Type	Constant
Value	Specifies the number of the subroutine. Any number in the range 1 to 500 is valid.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Notes

1. The **SUBR** function block is located in row 1, column 1 of a network.
2. No other function block can be located to the right of the SUBR function block.
3. Outputs of a subroutine remain in their last state when a subroutine is not called. For example, a coil that is turned on by a subroutine remains on when the subroutine is not called. The output will only turn off when the subroutine is called and it turns the output off.
4. The subroutine number is a constant. Constant tag names are used with this value.
5. The subroutine number needs to be unique. No other subroutine can use the same number.
6. Subroutines need not be programmed in any particular numerical order. For example, subroutine 1 can follow subroutine 2, subroutine 200 can follow subroutine 400.

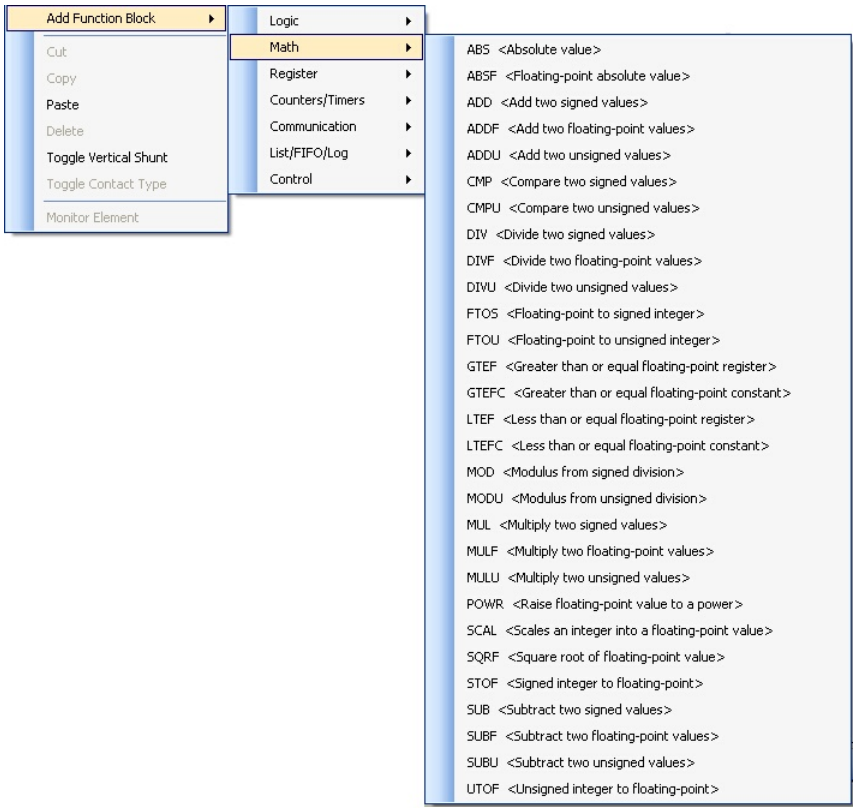
Related function blocks

CALL

Example
See the CALL (Execute Subroutine) example.

Math Functions

The Math functions available in Telepace Studio are shown in the image below. This image is opened in Telepace Studio by right clicking on the ladder canvas and selecting the Add Function Block command. Depending on the available space on the ladder canvas some functions may not be displayed as they would not fit in the canvas at the location selected.

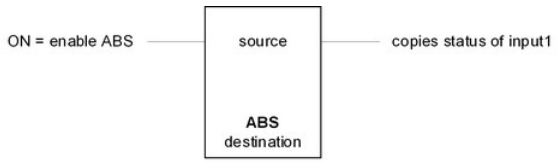


Hover your mouse over the icons below for a description of the function. Click the icon to go to the function description.

ABS	ABSF	ADD	ADDF	ADDU
CMP	CMPU	DIV	DIVF	DIVU
FTOS	FTOU	GTEF	GTEFC	LTEF
LTEFC	MOD	MODU	MULF	MULU
POWR	SCAL	SQRF	STOF	SUB
SUBF	SUBU	UTOF		

ABS (Absolute Value)

The **ABS** function block stores the absolute value of a signed constant or register into a holding register. When the **enable ABS** is ON the absolute value of the source register or constant is stored in the destination holding register.



Function Block Properties
Source

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

ABSF

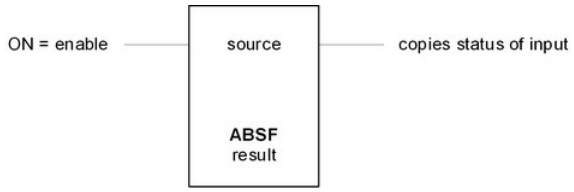
Example

network 1:

In this example, the absolute value of the **source**, holding register 44001, is stored in the **destination**, holding register 45211. Assuming holding register 44001 has a value of –4455, the value stored in holding register 45211 would be +4455.

ABSF (Floating-Point Absolute Value)

The **ABSF** function block stores the absolute value of a floating-point register or constant in a floating-point holding register. When the **enable** input is **ON** the absolute value of the source is stored in the **result** floating-point holding register. The function block output is **ON** when the input is.



Function Block Properties

Source

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination

Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

ABS

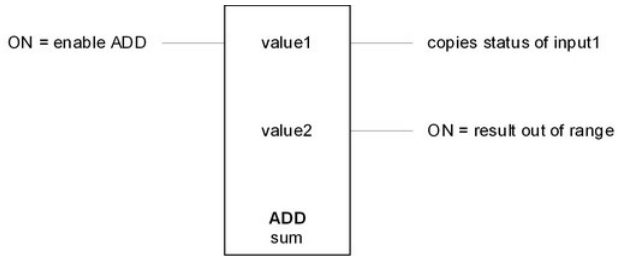
Example

In this example the **absolute value** of the **source,** holding register 40001, is stored in the **result,** floating-point register 45555. Assuming floating point register 40001 contains a floating-point value of –77988.99, the value stored in floating-point register 45555 would be +77988.99.

ADD (Add Signed Values)

The **ADD** function block adds two registers or constants and stores the result in a holding register. Signed addition is used. If the result is out of range, the sum rolls over and the out of range output is enabled.

When the **enable ADD** input is **ON** the sum of **value 1** and **value 2** is stored in **sum** holding register. If the sum is out of range (–32768...32767) the **result out of range** output is **ON**.



Function Block Properties

First Value to Add (Value 1)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Add (Value 2)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

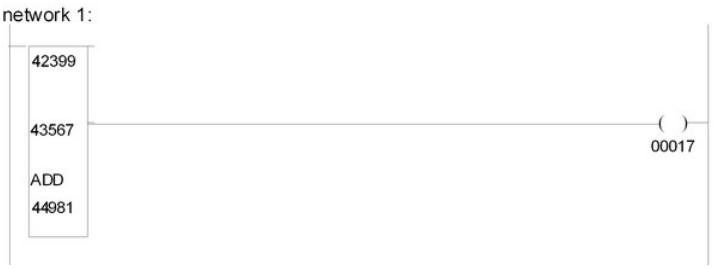
Sum

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

ADDF	**ADDU**

Example



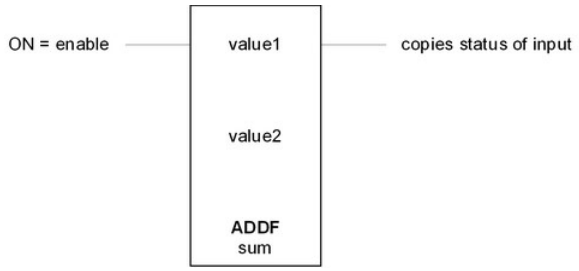
The **ADD** function block in network 1 adds holding register 42399 to holding register 43567 and stores the result is holding register 44981. When the addition is out of range – less than –32768 or greater than 32767-output coil 00017 is energized. The table below shows some examples.

First Value to Add Holding Register	Second value to Add Holding Register	Sum Holding Register	Out of Range Output Coil
42399	43567	44981	00017
23411	1098	24509	OFF
–2047	819	–1228	OFF
12767	23000	–29769	ON

ADDF (Add Floating-point Values)

The **ADDF** function block adds two floating-point registers or constants and stores the result in a floating-point holding register.

When the **enable** input is ON the sum **value 1** + **value 2** is stored in the **sum** floating point register. The function block output is ON when the input is.



Function Block Properties

First Value to Add (Value 1)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Add (Value 2)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Sum

Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

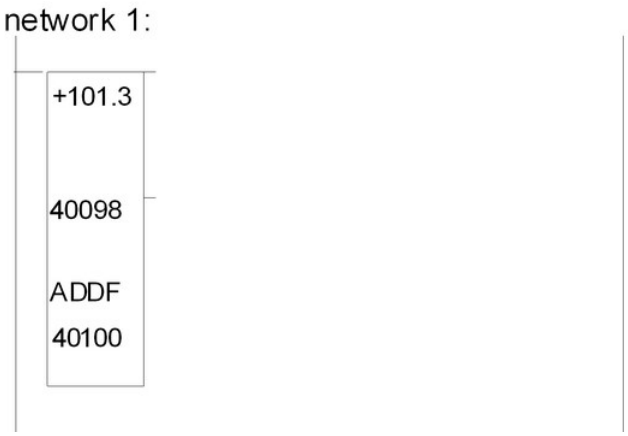
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

ADD | ADDU

Example

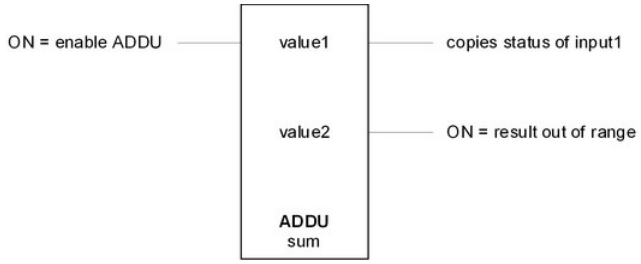


The **ADDF** function block in network 1 adds floating-point constant +101.3 to floating-point register 40098 (registers 40098 and 40099). Assuming floating point register 40098 contains a value of +1000.00 then the result is +1101.30. This value is stored in floating-point register 40100 (registers 40100 and 40101).

ADDU (Add Unsigned Values)

The **ADDU** function block adds two registers or constants and stores the result in a holding register. Unsigned addition is used. If the result is out of range, the sum rolls over and the out of range output is enabled.

When the enable ADDU input is ON the sum of value 1 and value 2 is stored in sum holding register. If the sum is out of range (> 65535) the **result out of range** output is ON.



Function Block Properties

First Value to Add (Value 1)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Add (Value 2)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

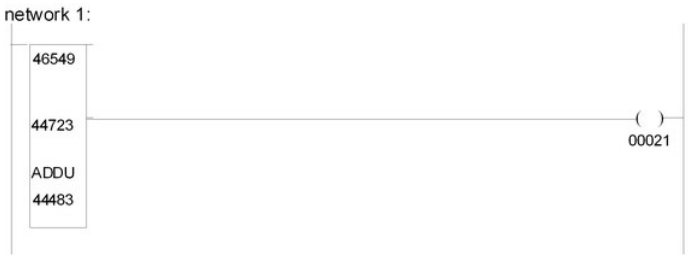
Sum

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

ADD	ADDF

Example



The **ADDU** function block in network 1 adds holding register 46549 to holding register 44723 and stores the result is holding register 44483. When the addition is out of range, greater than 65535, output coil 00021 is energized. The table below shows some examples.

First Value to Add Holding Register	Second value to Add Holding Register	Sum Holding Register	Out of Range Output Coil
46549	44723	44483	00021
2048	819	2867	OFF
23000	42530	65530	OFF
35000	35000	4464	ON

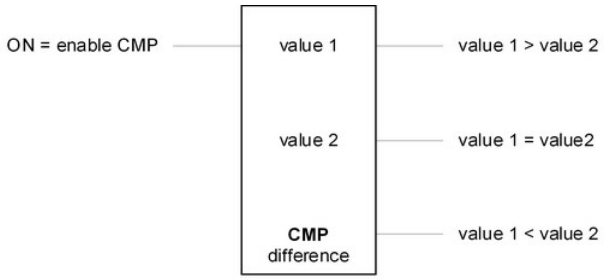
CMP (Compare Signed Values)

The **CMP** function block compares two signed registers or constants and stores the difference in a holding register. When the **enable CMP** input is energized **value 1** is compared to **value 2**.

If value 1 is greater than value 2 the **value 1 > value 2** output is energized and the difference of value 1 minus value 2 is stored in **difference** register.

If value 1 is less than value 2 the **value 1 < value 2** output is energized and the difference of value 2 minus value 1 is stored in the **difference** register.

If value 1 equals value 2 the **value 1 = value 2** output is energized and zero is stored in the **difference** register.



Function Block Properties

First Value to Compare (Value 1)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Compare (Value 2)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

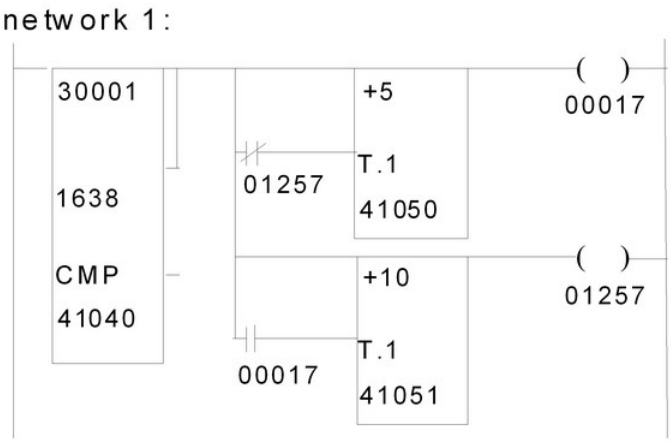
Difference

Type	Register
Value	For Register type any valid 4xxxx (holding register). if value 1 > value 2 then difference = value 1 – value 2 if value 2 > value 1 then difference = value 2 – value 1
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

CMPB	CMPU

Example



The CMP function block in network 1 above compares the signed value of Analog Input Register 30001 with the value 1638. The output coil 00017 could be tied to a light to indicate when the Input Register value is over 1638.

The **value 1 > value 2** and **value 1 = value 2** outputs are tied together. When value of register 30001 is greater than or equal to 1638 these outputs are energized. This will enable the flasher circuit of Timers 41050 and 41051 to energize and output coil 00017 will cycle at a rate of 1 second on and 0.5 seconds off.

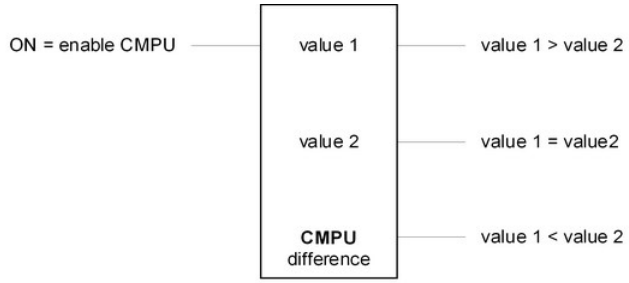
CMPU (Compare Unsigned Values)

The **CMPU** function block compares two unsigned registers or constants and stores the difference in a holding register. When **enable CMPU** input is energized **value 1** is compared to **value 2**.

If value 1 is greater than value 2 the **value 1 > value 2** output is energized and the difference of value 1 minus value 2 is stored in **difference** register.

If value 1 is less than value 2 the **value 1 < value 2** output is energized and the difference of value 2 minus value 1 is stored in the **difference** register.

If value 1 equals value 2 the **value 1 = value 2** output is energized.



Function Block Properties

First Value to Compare (Value 1)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Compare (Value 2)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Difference

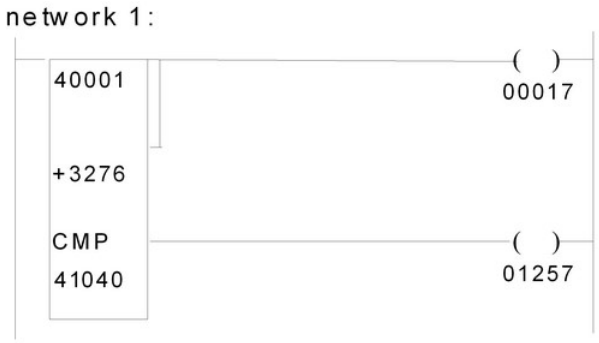
Type	Register
Value	For Register type any valid 4xxxx (holding register). if value 1 > value 2 then difference = value 1 – value 2 if value 2 > value 1 then difference = value 2 – value 1
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on

tag names.

Related function blocks

CMPB	CMP

Example



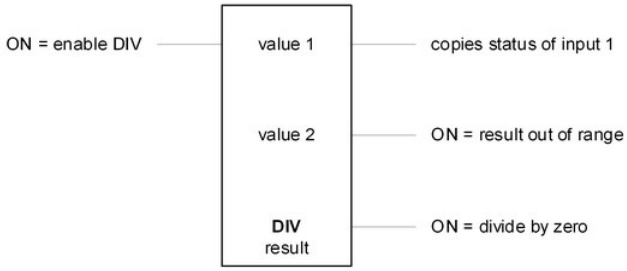
The CMPU function block in network 1 above compares the unsigned value of holding register 40001 with the value 3276. Coil 00017 is energized when holding register 40001 is **greater than or equal to** the constant 3276.

The **value 1 > value 2** and **value 1 = value 2** outputs are tied together.

DIV (Divide Signed Values)

The **DIV** function block divides two registers or a constant, **value 1**, by a register or constant, **value 2**, and stores the quotient in a Holding Register, **result**. Signed division is used. The values and result are signed numbers.

The **result out of range** output is enabled if the result is greater than 32767 or less than –32768. The **divide by zero output** is enabled if **value 2** equals 0.



Function Block Properties

Value to Divide (Value 1)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Value to Divide By (Value 2)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

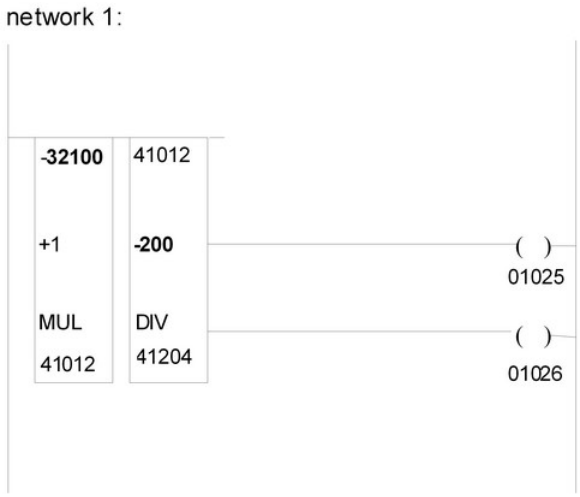
Result (Value 1 / Value 2)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

DIVF	DIVU

Example



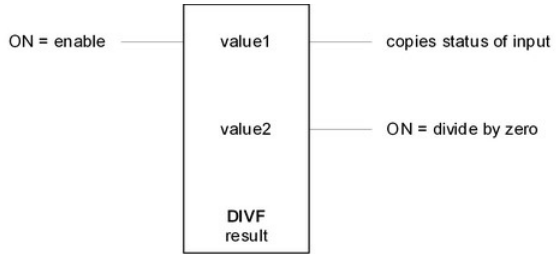
The MUL function block is used to create the double precision values required for **value 1**. A constant is multiplied by 1 with the result stored in Holding Register 41012. The low order word, 41012, and the high order word, 41013, are returned from the MUL function block. The following table illustrates various **value 1** and **value 2** values and the **result** of the division.

	high order Register 41013	low order Register 41012	Constant Value	Register 41204
		value 1	value 2	result
Example 1	-1	–32070	–200	160
Example 2	0	32070	–200	–160
Example 3	–1	–2048	16	–128
Example 4	0	3276	30	109

DIVF (Divide Floating Point Values)

The **DIVF** function block divides one floating-point register or constant by another and stores the result in a floating-point holding register.

When the **enable** input is ON the result **value 1** ÷ **value 2** is stored in the **result** floating-point register. The top output is ON when the input is. The middle output is ON if value2 equals 0 and the input is ON.



Function Block Properties

Value to Divide (Value 1)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Value to Divide By (Value 2)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Result (Value 1 / Value 2)

Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

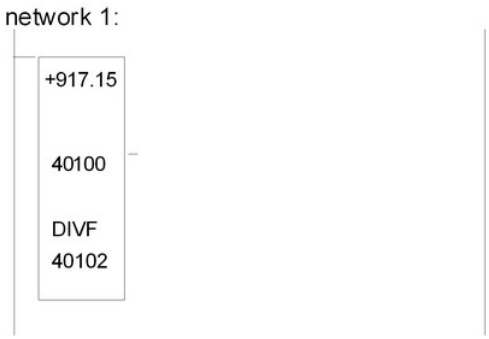
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

DIV	**DIVU**

Example

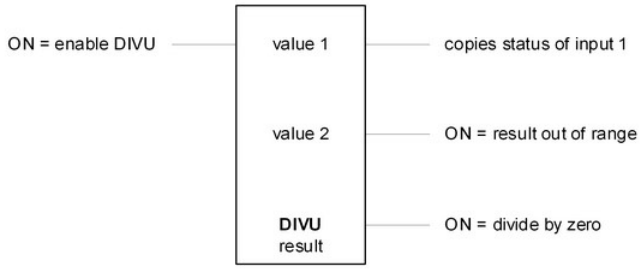


The **DIVF** function block in network 1 divides floating-point constant +917.15 by the contents of floating-point register 40100 (registers 40100 and 40101). Assuming floating-point register 40100 has a value of 5.00 then the result is +183.43. This value is stored in floating-point register 40102 (registers 40102 and 40103).

DIVU (Divide Unsigned Values)

The **DIVU** function block divides two registers or a constant, **value 1**, by a register or constant, **value 2**. The quotient is stored in a Holding Register, **result**. Unsigned division is used. The values and result are unsigned numbers.

The **result out of range output** is enabled if the result is greater than 65535 or less than 0. The **divide by zero** output is enabled if **value 2** equals 0.



Function Block Properties

Value to Divide (Value 1)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Value to Divide By (Value 2)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on

tag names.

Result (Value 1 / Value 2)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

DIV | DIVF

Example

The MULU function block is used to create the double precision values required for **value 1**. A constant is multiplied by 1 with the result stored in Holding Register 47722. The low order word, 47722, and the high order word, 47723, are returned from the MULU function block. The MULU function block ensures a value of 0 in the high order word.

The following table illustrates various **value 1** and **value 2** values and the **result** of the **DIVU** function block.

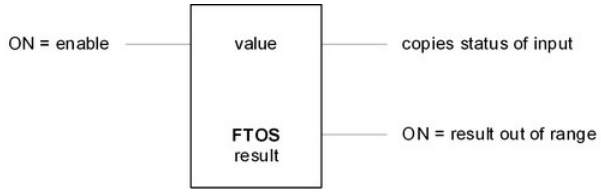
	high order Register 47723	low order Register 47722	Constant Value	Register 47722
	value 1		**value 2**	**result**
Example 1	0	53186	622	85
Example 2	0	7000	70	100
Example 3	0	8192	819	10
Example 4	0	42998	226	190

FTOS (Floating-Point to Signed Integer)

The **FTOS** function block converts a floating-point register or constant into a signed integer and stores the result in a holding register.

When the **enable** input is ON, **value** is converted into a signed number and stored in the **result** register. The value is rounded to the nearest integer. If the value is less than –32768, then –32768 is stored. If the value is greater than 32767, then 32767 is stored.

The top output is ON when the input is. The bottom output is ON if the floating-point value is less than –32768 or greater than 32767.



Function Block Properties

Source (Value)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination (Result)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered

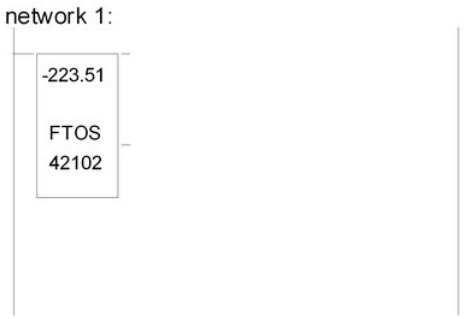
register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

UTOF	STOF	FTOU

Example



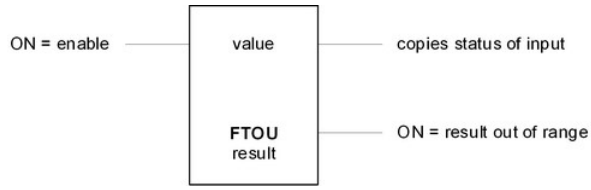
The **FTOS** function block in network 1 converts floating-point constant –223.51 to a signed integer value and puts the value into register 42102. In this example the content of register 42102 is the signed integer –224.

FTOU (Floating-point to Unsigned Integer)

The **FTOU** function block converts a floating-point register or constant into an unsigned integer and stores the result in a holding register.

When the **enable** input is ON, **value** is converted into an unsigned number and stored in the **result** register. The value is rounded to the nearest integer. If the value is less than zero, then zero is stored. If the value is greater than 65535, then 65535 is stored.

The top output is ON when the input is. The bottom output is ON if the floating-point value is less than 0 or greater than 65535.



Function Block Properties

Source (Value)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination (Result)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

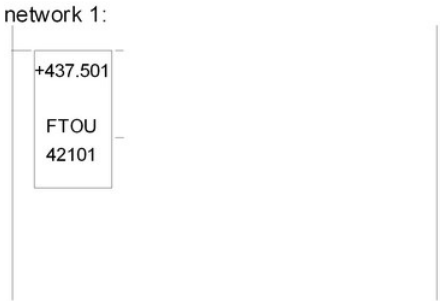
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

UTOF	STOF	FTOS

Example



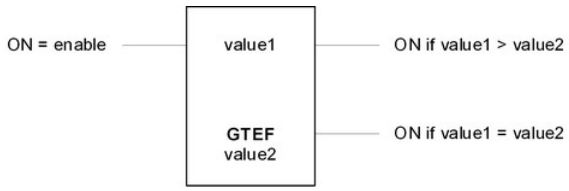
The **FTOU** function block in network 1 converts floating-point constant +437.501 to an unsigned integer value and puts the value into register 42101. In this example the content of register 42101 is the unsigned integer +438.

The *records* value is set to 31. This means that 31 sets of history registers, volume, flow time and end of period time, will be logged. In this example the accumulation input is never turned off meaning that once 31 sets of history registers are saved the next time contact 10009 is closed the 31st record is removed and the newest record is added to the history.

GTEF (Floating-point Greater than or equal Registers)

The **GTEF** function block tests if a floating-point register or constant is greater than or equal to another floating-point register.

When the **enable** input is ON, value1 and value2 are compared. If value1 is greater than value2, the top output is ON. If value1 is equal to value2 the bottom output is ON.



Function Block Properties

First Value to Compare (Value1)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Compare (Value2)

Type	Register
Value	For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

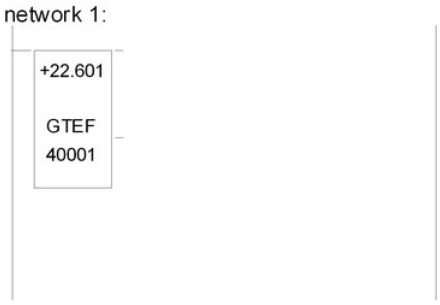
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

GTEFC

Example

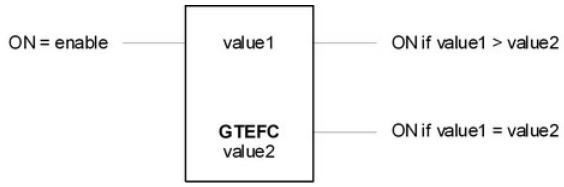


The **GTEF** function block in network 1 compares the floating-point constant +22.601 to the contents of floating-point register 40001. The top output of the GTEF function block is ON if the value of register 40001 is less than +22.601. The bottom output of the GTEF is ON if the value of register 40001 is equal to +22.601.

GTEFC (Floating-point Greater than or equal Constants)

The **GTEFC** function block tests if a floating-point register or constant is greater than or equal to a floating-point constant.

When the **enable** input is ON, value1 and value2 are compared. If value1 is greater than value2, the top output is ON. If value1 is equal to value2 the bottom output is ON.



Function Block Properties

First Value to Compare (Value1)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Compare (Value2)

Type	Constant
Value	For Constant type any IEEE single precision floating-point number.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38}.

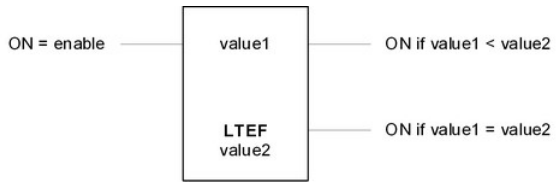
Related function blocks

GTEF

LTEF (Floating-point less than or equal Register)

The **LTEF** function block tests if a floating-point register or constant is less than or equal to another floating-point register.

When the **enable** input is ON, value1 and value2 are compared. If value1 is less than value2, the top output is ON. If value1 is equal to value2 the bottom output is ON.



Function Block Properties

First Value to Compare (Value1)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx

(holding register) registers. See Register Types for further information.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Compare (Value2)

Type Register

Value For Register type any valid 3xxxx (input register) or 4xxxx (holding register).

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

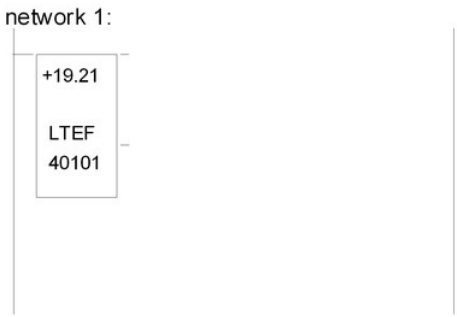
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

LTEFC

Example

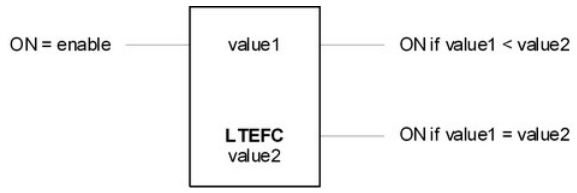


The **LTEF** function block in network 1 compares the floating-point constant +19.21 to the contents of floating-point register 40101. The top output of the LTEF function block is ON if the value of register 40101 is greater than +19.21. The bottom output of the LTEF is ON if the value of register 40001 is equal to +19.21.

LTEFC (Floating-point less than or equal Constant)

The **LTEFC** function block tests if a floating-point register or constant is less than or equal to a floating-point constant.

When the **enable** input is ON, value1 and value2 are compared. If value1 is less than value2, the top output is ON. If value1 is equal to value2 the bottom output is ON.



Function Block Properties

First Value to Compare (Value1)

Type Constant
Register

Value For Constant type any IEEE single precision floating-point number.
For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Compare (Value2)

Type Constant

Value For Constant type any IEEE single precision floating-point number.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38}.

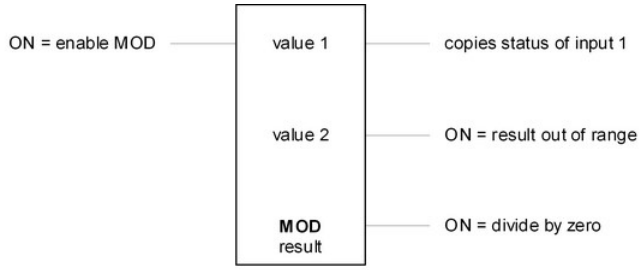
Related Elements

LTEF

MOD (Modulus of signed values)

The **MOD** function block divides two registers or a constant, **value 1**, by a register or constant, **value 2**, and stores the modulus (remainder) in a holding register, **result**. Signed division is used. The value 1, value 2 and result are signed numbers.

The **out of range** output is enabled if the result is greater than 32767 or less than –32768. The **divide by zero** output is enabled if value 2 equals 0.



Function Block Properties

Value to Divide (Value 1)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Value to Divide By (Value 2)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

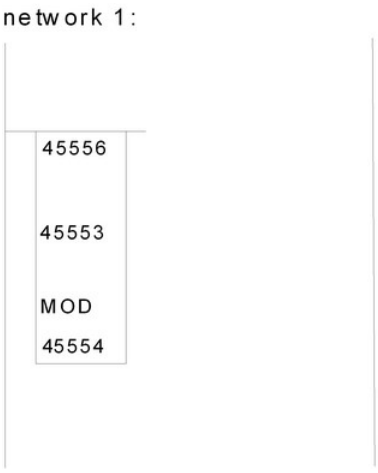
Remainder (Result)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

MODU

Example



The **MOD** element in network 1 divides **value 1**, double word registers 45556 and 45557 by **value 2**, register 45553. The modulus of this division is stored in register 45554. The table below shows some examples of different values for the registers in the **MOD** element.

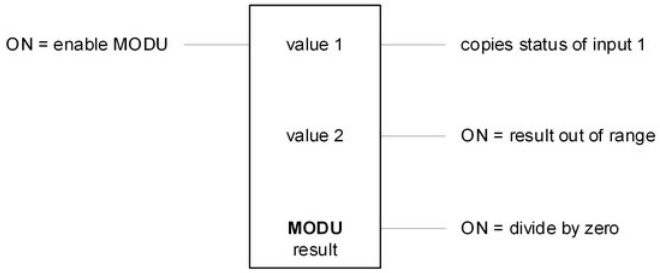
	Value of Register 45556	Value of Register 45553	Modulus Register 45554
example 1	5689	77	68
example 2	–32600	–234	–74

example 3	–22765	7659	–7447
example 4	19823	–7692	4439

MODU (Modulus of Unsigned Values)

The **MODU** function block divides two registers or a constant, **value 1**, by a register or constant, **value 2**, and stores the modulus (remainder) in a holding register, **result**. Unsigned division is used. The value 1, value 2 and result are unsigned numbers.

The **out of range output** is enabled if the result is greater than 65535 or less than 0. The **divide by zero output** is enabled if value 2 equals 0.



Function Block Properties

Value to Divide (Value 1)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Value to Divide By (Value 2)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Remainder (Result)

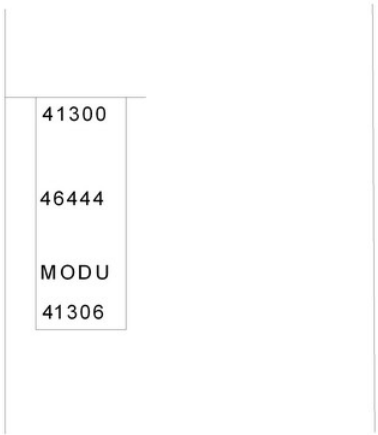
Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

MOD

Example

network 1:

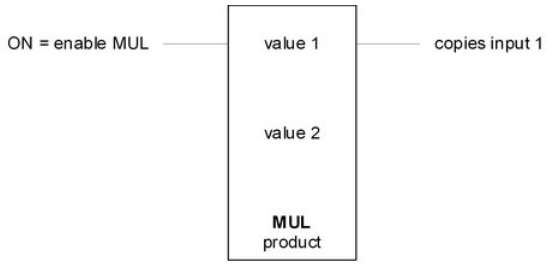


The **MODU** element in network 1 divides **value 1**, double word registers 41300 and 41301 by **value 2**, register 46444. The modulus of this division is stored in register 41306. The table below shows some examples of different values for the registers in the MOD element.

	Value of Register 45556	Value of Register 45553	Modulus Register 45554
example 1	26744	32077	184
example 2	56398	34022	20878
example 3	37697	2366	2207
example 4	33005	1100	5

MUL (Multiply Signed Values)

The **MUL** function block multiplies a signed register or constant, **value 1**, by a signed register or constant, **value 2** and stores the result in two consecutive holding registers, **product**. Signed multiplication is used.



Function Block Properties

First Value to Multiply (Value 1)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Multiply (Value 2)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Multiplication Result (Product)

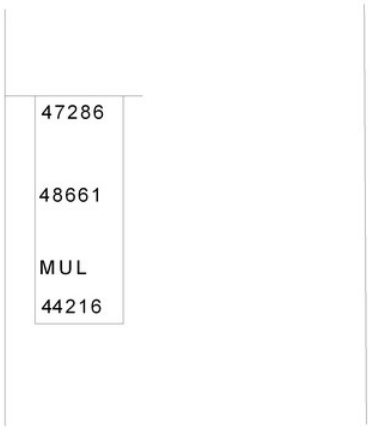
Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

MULF **MULU**

Example

network 1:



In network 1 a **MUL** element is used to multiply the contents of Holding Register 47286, **value 1**, with the contents of Holding Register 48661, **value 2**. The product is stored in Holding Registers 44216, **low order word** and Holding Register 44217, **high order word**.

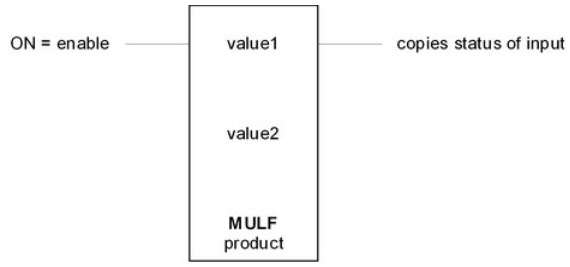
The examples in the table below show the product of the multiplication for various values in registers 47286 and 48661. The product is displayed as registers 44216, **low order word** and register 44217, **high order word**.

	Register 47286 Value 1	Register 48861 Value 2	High order Register 44217 Product	Low order Register 44316 Product
Example 1	–3057	–10	0	30570
Example 2	–15	100	–1	–1500
Example 3	1654	17	0	29118
Example 4	164	–55	-1	–9020

MULF (Multiply floating-point values)

The **MULF** function block multiplies two floating-point registers or constants and stores the result in a floating-point holding register.

When the **enable** input is ON the product **value 1** × **value 2** is stored in the **product** floating-point register. The element output is ON when the input is.



Function Block Properties

First Value to Multiply (Value 1)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Multiply (Value 2)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Multiplication Result (Product)

Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

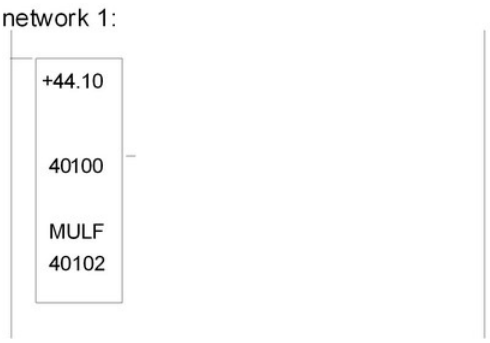
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

MUL | MULU

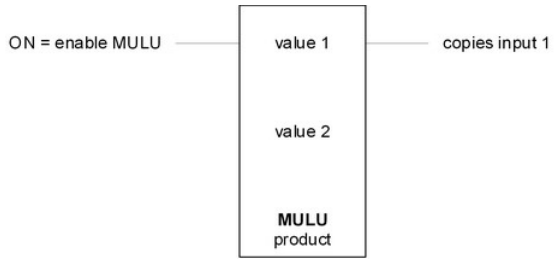
Example



In network 1 a **MULF** element is used to multiply the constant 44.10, **value 1**, with the contents of Holding Register 40100, **value 2**. The product is stored in Holding Registers 40102, **low order word** and Holding Register 40103, **high order word.**

MULU (Multiply Unsigned Values)

The **MULU** function block multiplies an unsigned register or constant, **value 1**, and an unsigned register or constant , **value 2**. The result is stored in two consecutive Holding Registers, **product**. Unsigned multiplication is used.



Function Block Properties

First Value to Multiply (Value 1)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value to Multiply (Value 2)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Multiplication Result (Product)

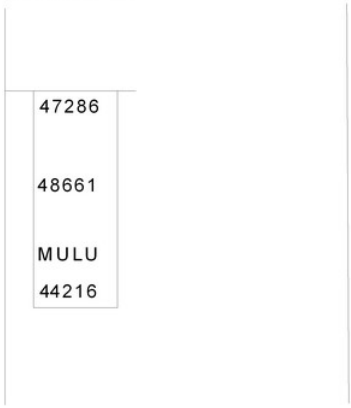
Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

MUL	MULF	MULU

Example

network 1:



In network 1 a **MULU** element is used to multiply the contents of Holding Register 47286, **value 1**, with the contents of Holding Register 48661, **value 2**. The product is stored in Holding Registers 44216, **low order word** and Holding Register 44217, **high order word**.

The examples in the table below show the product of the multiplication for various values in registers 47286 and 48661. The product is displayed as registers 44216, **low order word** and register 44217, **high order word**.

	Register 47286 Value 1	Register 48861 value 2	High order Register 44217 Product	Low order Register 44316 product
Example 1	2048	30	0	61440
Example 2	251	251	0	63001
Example 3	77	77	0	5929
Example 4	650	735	7	18998

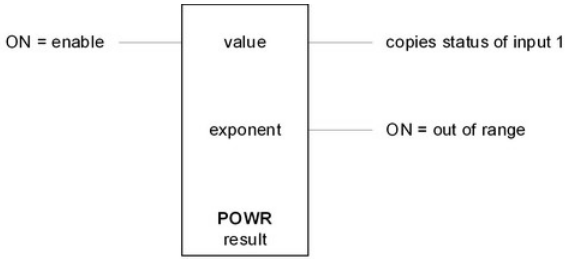
POWR (Floating-point raised to Power)

The **POWR** function block calculates the result of a value raised to an exponent and stores the result in a floating-point holding register.

When the **enable** input is ON, **value** is raised to the **exponent** and stored in the **result** floating-point register.

The top output is ON when the input is ON.

The **out-of-range** output is ON if the **value** is zero and **exponent** is less than zero; and if **value** is negative. The result is not calculated in these cases.



Function Block Properties

First Value - Base (Value)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Second Value - Exponent (Exponent)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Result

Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if

one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

MULF	SQRF

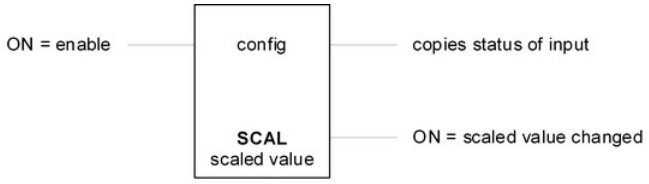
SCAL-Scale Analog Value

The SCAL element scales an integer into a floating-point value and detects changes in the floating-point value.

When the **Enable** input is ON the SCAL element calculates the scaled value.

When the **Enable** input is ON the scaled value is compared to the previous scaled value. If the difference is greater than the deadband the scaled value changed output is turned ON and the scaled value is copied to the previous scaled value. The previous scaled value does not change if scaled value is within the deadband.

When the **Enable** input changes from OFF to ON, the scaled value is copied to previous scaled value.



Function Block Properties

Configuration Block

Type	Register
Value	Address of the first register (**config**) of 9 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller. The block of registers are defined for the function block as follows: **config** + 0 = input register **config** + 1 = zero scale raw input **config** + 2 = full scale raw input **config** + 3 and 4 = zero scale output (floating point) **config** + 5 and 6 = full scale output (floating point) **config** + 7 and 8 = output deadband (floating point) See the Element Configuration section below for a description of each parameter. Floating point register types use two 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the config + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Scaled Value

Type	Register
Value	Address of the first register (**scaled value**) of 4 in a sequential block of 4xxxx (holding) registers. See Register Types for further information. **scaled Value** + 0 and 1 = Scaled value (floating point) **scaled Value** + 2 and 3 = Previous scaled value (floating point)
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration

The **SCAL** function block may be configured using the Element Configuration.

The **Addresses** window displays the registers used by the function block.

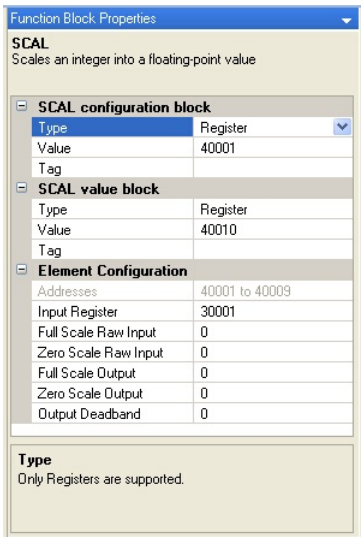
The **Input Register** specifies the register from which the input is read. Valid values are any input (3xxxx) or holding (4xxxx) register. The range depends upon the registers supported in the selected controller type.

The **Zero Scale Raw Input** specifies the minimum raw input value. Valid values are 32768 to 32767. This value needs to be less than the full-scale raw input.

The **Full Scale Raw Input** specifies the maximum raw input value. Valid values are 32768 to 32767. This value needs to be greater than the zero scale raw input.

The **Zero Scale Output** is the floating-point value calculated when the input is equal to the zero scale raw input. This value needs to be less than the full-scale output. Valid values are any floating-point number.

The **Full Scale Output** is the floating-point value calculated when the input is equal to the full-scale raw input. This value needs to be greater than the zero scale output. Valid values are



any floating-point number.

The **Output Deadband** is the amount by which the output needs to change for the **scaled value changed** output to be energized. Valid values are any floating-point number greater than or equal to 0.0.

Notes

The input is scaled using this formula.

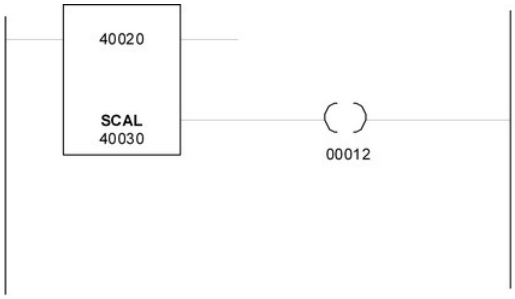
$$scaled = \frac{input - zero\_raw}{full\_raw - zero\_raw} \times (full\_output - zero\_output) + zero\_output$$

The previous scaled value changes only when difference between the current and previous values is greater than the deadband. The **scaled value changed** output is energized when this occurs.

The input register may contain values less than the zero scale raw input or greater than the full-scale raw input. The scaling calculation is still performed.

Example

In this example a SCAL block is used to convert an analog input into engineering units and close a contact each time a change of 10 units occurs.



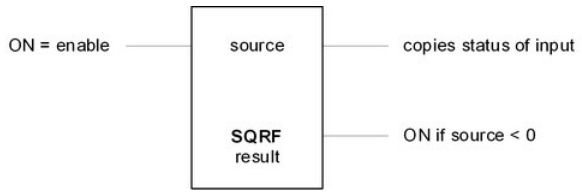
The element configuration for the SCAL block is as follows:

Parameter	Value	Notes
Input Register	30001	Assign to analog input module in register assignment.
Zero Scale Raw Input	0	Minimum value for unipolar analog input.
Full Scale Raw Input	32767	Maximum value for analog input.
Zero Scale Output	0	Corresponds to the minimum value of analog input.
Full Scale Output	1000	Corresponds to the maximum value of analog input.
Output Deadband	10	Change in output to turn on change output.

SQRF (Square root of floating-point value)

The **SQRF** function block stores the square root of a floating-point register or constant in a floating-point holding register.

When the **enable** input is ON the square root of the **source** is stored in the **result** floating-point holding register. The top output is ON when the input is. The bottom output is ON if the source is negative, and the input is ON.



Function Block Properties

Source

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination

Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

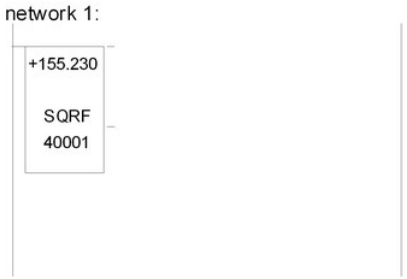
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

DIV | DIVU

Example

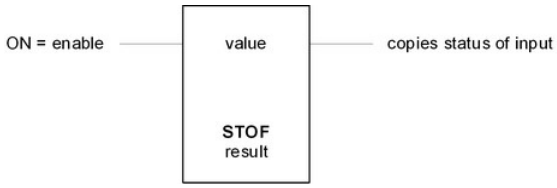


The SQRF element in network 1 takes the square root of the floating-point constant +155.23 and puts the result in floating-point register 40001 (registers 40001 and 40002). The value in 40001 is =12.459.

STOF (Signed Interger to Floating-point)

The **STOF** function block converts a signed register or constant into a floating-point number and stores the result in a floating-point holding register.

When the **enable** input is ON, **value** is converted into a floating-point number and stored in the **result** floating-point register. The element output is ON when the input is.



Function Block Properties

Source (Value)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination (Result)

Type	Register

Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

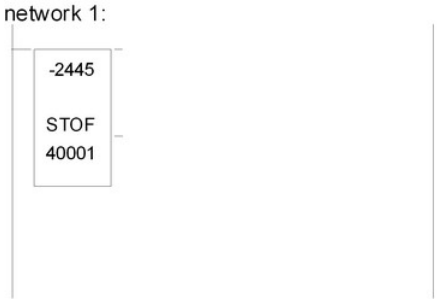
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

FTOS | FTOU | UTOF

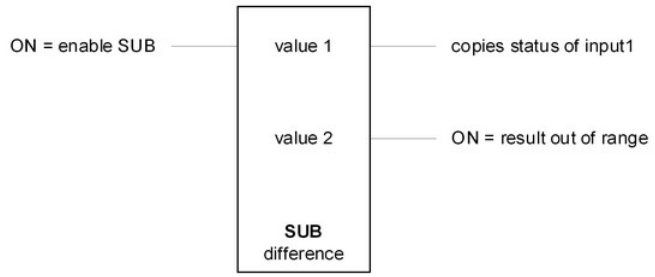
Example



The **STOF** function block in network 1 converts signed integer constant -2445 to a floating-point value and puts the value into floating point-register 40001 (registers 40001 and 40002). In this example the content of floating-point register 40001 is the floating-point value –2445.00.

SUB (Subtract Signed Values)

The **SUB** function block subtracts a signed register or constant, **value 2**, from a signed register or constant, **value 1**, and stores the result in a holding register, **result**. Signed subtraction is used. The **result out of range** output is enabled if the result is greater than 32767 or less than -32768.



Function Block Properties

Value to Subtract From (Value 1)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Value to Subtract (Value 2)

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Difference

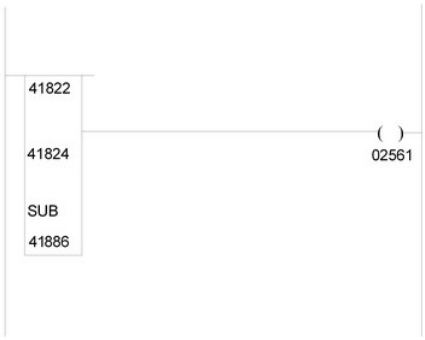
Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

SUBF **SUBU**

Example

network 1:



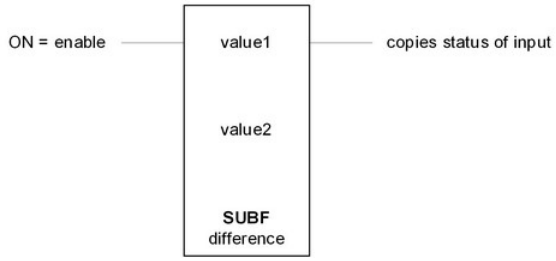
The SUB function block in network 1 subtracts the contents of register 41824, **value 2**, from the contents of register 41822, **value 1**. The **difference** is stored in register 41886. Some examples of different values for registers 41822 and 41824 are shown in the table below.

	Register 41822 value 1	Register 41824 value 2	Register 41886 difference	Coil 01561 out of range
example 1	–27765	–18755	–9010	OFF
example 2	3276	–12269	15545	OFF
example 3	–30000	27881	7655	ON

SUBF (Subtract floating-point values)

The **SUBF** function block subtracts one floating-point register or constant from another and stores the result in a floating-point holding register.

When the **enable** input is ON the difference **value 1** – **value 2** is stored in the **difference** holding register. The element output is ON when the input is.



Function Block Properties

Value to Subtract From (Value 1)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Value to Subtract (Value 2)

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Difference

Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

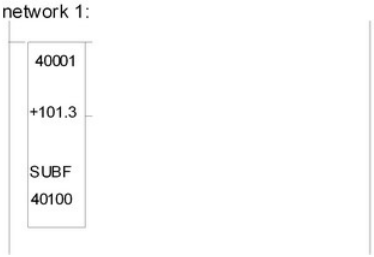
Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

SUB	SUBU

Example

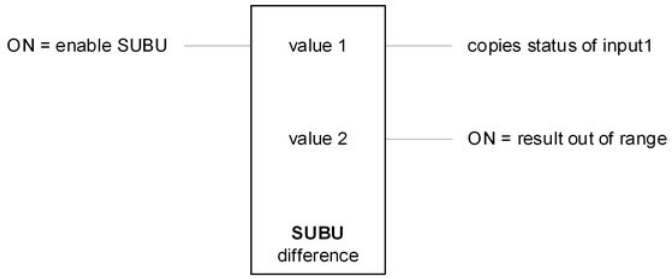
network 1:



The **SUBF** function block in network 1 subtracts floating-point constant +101.3 from floating-point register 40001 (registers 40001 and 40002). Assuming floating-point register 40001 contains a value of 1000.00 the result is +898.7. The result is stored in floating-point register 40100 (registers 40100 and 40101).

SUBU (Subtract Unsigned Values)

The **SUBU** function block subtracts an unsigned register or constant, **value 2**, from an unsigned register or constant, **value 1**, and stores the result in a holding register, difference. Unsigned subtraction is used. The **result out of range** output is enabled if the result is greater than 65535 or less than 0.



Function Block Properties

Value to Subtract From (Value 1)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Value to Subtract (Value 2)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Difference

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

SUB	SUBF

Example

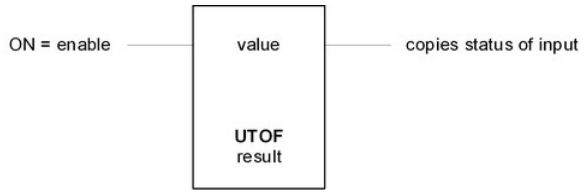
The SUBU element in network 1 subtracts the contents of register 42992, **value 2**, from the contents of register 49920, **value 1**. The **difference** is stored in register 44996. Some examples of different values for registers 49920 and 42992 are shown in the table below.

	Register 49920 value 1	Register 49922 value 2	Register 44996 difference	Coil02011 out of range
example 1	22675	1988	20687	OFF
example 2	61223	2399	58824	OFF
example 3	3877	54440	14973	ON

UTOF (Unsigned Integrator to Floating-point)

The UTOF function block converts an unsigned register or constant into a floating-point number and stores the result in a floating-point holding register.

When the enable input is ON, value is converted into a floating-point number and stored in the result floating-point register. The element output is ON when the input is.



Function Block Properties

Source (Value)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination (Result)

Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

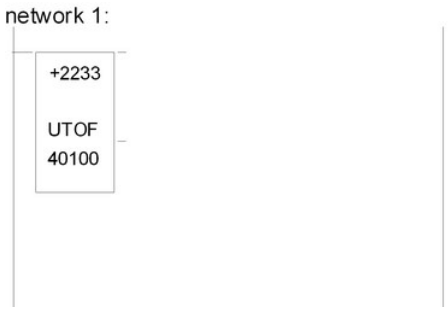
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

FTOS	FTOU	STOF

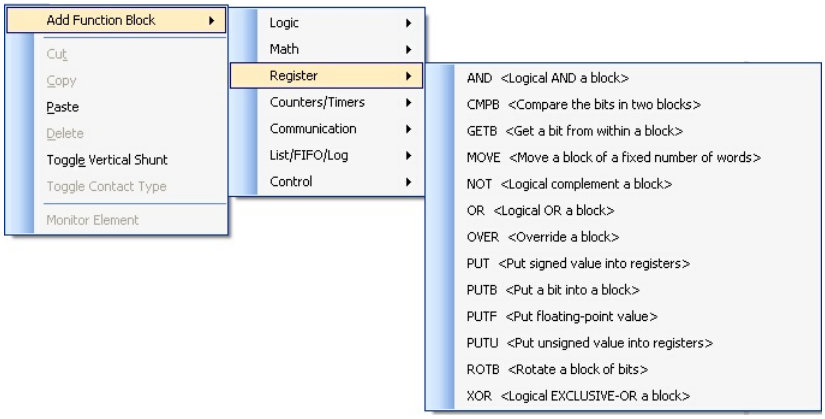
Example



The **UTOF** function block in network 1 converts unsigned constant +2233 to a floating-point number and puts the value into register 40100. In this example the content of floating point-register 40100 (registers 40100 and 40101) is 2233.00.

Register Functions

The Register functions available in Telepace Studio are shown in the image below. This image is opened in Telepace Studio by right clicking on the ladder canvas and selecting the Add Function Block command. Depending on the available space on the ladder canvas some functions may not be displayed as they would not fit in the canvas at the location selected.

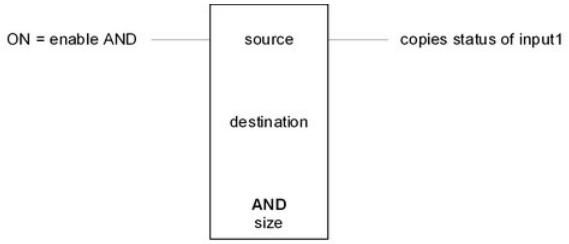


Hover your mouse over the icons below for a description of the function. Click the icon to go to the function description.

ADD	CMPB	GETB	MOVE
NOT	OR	OVER	PUT
PUTB	PUTF	PUTU	ROTB
XOR			

AND (And Block)

The **AND** function block is used to perform is a Boolean AND algebra function. The AND function block ANDs the source register(s) with the destination register(s) and stores the result in the destination register(s). The AND element requires two arguments, the source and destination. The AND function block will only turn on bits in the result that are true in both the **source** block of registers and the **destination** block of registers. The result is stored in the **destination** block of registers. The number of registers in the source and destination blocks is determined by the **size**.



Function Block Properties

Source Block (Source)

Type	Register
Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Source and Destination Block (Destination)

Type	Register
Value	For Register type any valid 0xxxx (coil register) or 4xxxx (holding register). For 4xxxx registers the result of the AND operation is stored in register (4xxxx) to register (4xxxx + size). For 0xxxx registers the result of the AND operation is stored in register (0xxxx+1) to register (0xxxx + size). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Number of Registers in Block (Size)

Type	Constant
Value	For Constant type any number in range 0 to 100. This is the number of 16-

bit registers in the queue.

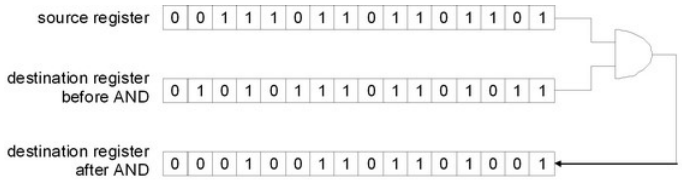
Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

NOT	OR	XOR

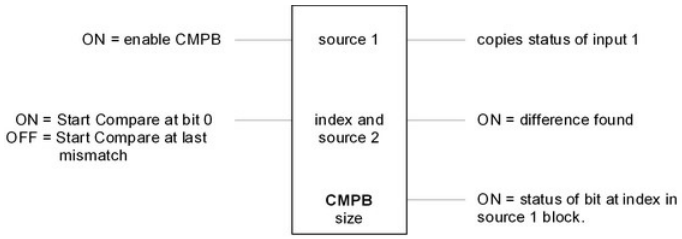
Example

In this example the source register has a value of 1521310 (0011101101101101). The destination register has a value of 2237910 (0101011101101011) before the AND element and a value of 496910 (0001001101101001) after the AND element.



CMPB (Compare Bit)

The **CMPB** element compares two blocks of registers, bit by bit and identifies the first bit that does not match. The bit index will be set to the bit offset of the first mismatched bit.



Function Block Properties

Source Block (Source 1)

Type Register

Value For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register).
This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Bit Index and Destination Block (Index and source 2)

Type Register

Value For Register type any valid 0xxxx (coil register) or 4xxxx (holding register). The address of the index register and the queue registers.
For 4xxxx registers the index is stored in register (4xxxx).
The second source register block located at register (4xxxx+1) to register (4xxxx + size).
A bit index of 0 is the most significant bit of the second source block.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

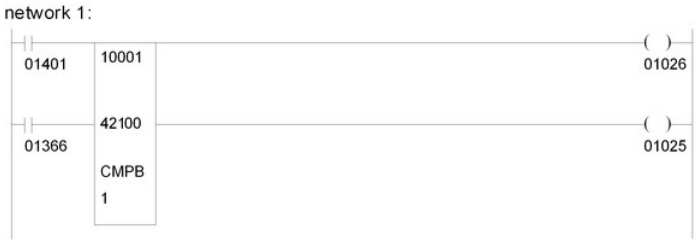
Number of Registers in Block (Size)

Type	Constant
Value	For Constant type any number in range 0 to 100. This is the number of 16-bit registers.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

CMP	CMPU

Example



The **CMPB** element in network 1 is used to check the status inputs 10001 to 10016. If Holding Register 42100 is set to zero, with a PUTU element, any status inputs in the range 10001 to 10016 that are on will be detected by the CMPB element.

Using the information in the chart below, status input registers 10004,10013 and 10014 are on. Closing contacts 01366 will start the index at bit 0 and the index will be set to the bit offset of the first mismatched bit, bit 2. Discrete output coil 01026 will be on and discrete output coil 01025 will be on.

If contacts 01366 are left closed the index will start at bit 0 and be set to the bit offset of the first mismatched bit, bit 2, on each scan of network 1. Discrete output coil 01026 will be on and discrete output coil 01025 will be on.

Opening contacts 01366 will allow the index to start at bit 2 on the next scan of network 1. The index will be set to the bit offset of the next mismatched bit, bit 3. Discrete output coil 01026 will be on and coil register 01025 will be on.

On the next scan of network 1 the index will start at bit 3. The index will be set to the bit offset of the next mismatched bit, bit 12. Coil 01026 will be on and coil register 01025 will be on.

The next scan of network 1 will have the index starting at bit 12. The remaining bits are compared and no further mismatches occur. The index will be set to bit 15 and wait for the contacts 01366 to close to start the compare at bit index 0.

Bit Index:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Discrete Input State:	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0
Discrete Input Address: (read top to bottom)	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1

GETB (Get Bit from Block)

The **GETB** function block reads the status of a bit, at the bit index, from the source block of registers.

When the **bit index** is defined as a holding register the **enable GETB** input needs to be ON to increment bit index or reset bit index to zero.

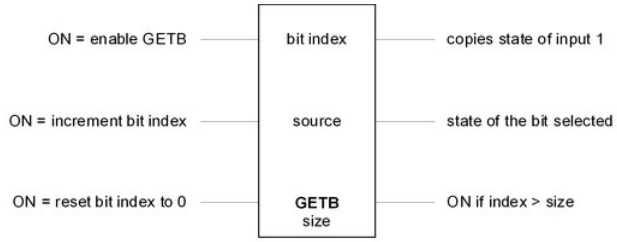
The **increment bit index** input increments the bit index once, each scan, when enabled.

The **reset bit index to 0** input resets the bit index to 0 when enabled.

When the bit index is defined as a constant, the GETB element checks the status of the bit at the bit index when **enable GETB** is ON. The **increment bit index** and **reset bit index to 0** inputs have no effect on the element in this condition.

When the **enable GETB** is ON, the **state of bit selected** output is ON if the source bit is 1 and OFF if the source bit is 0.

The **index>size** output is enabled when the bit index is equal to **size** + 1 and the **enable GETB** input is ON. The **reset bit index to 0** input needs to be set to continue element operation.



Function Block Properties

Bit Index into the Block (Bit Index)

Type	Constant

	Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 4xxxx (holding register). A bit index of 0 is the most significant bit of the second source block.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Source (Source)

Type	Register
Value	For Register type any valid 4xxxx (holding register). For Register type any valid 0xxxx (coil register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

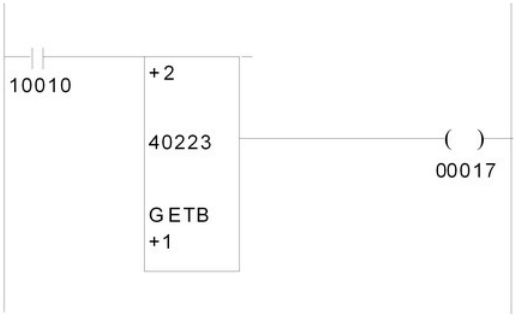
Number of Registers in Block (Size)

Type	Constant
Value	For Constant type any number in range 0 to 600. This is the number of 16-bit registers.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

CMPB	PUTB	ROTB

Example



In this example the **GETB** function block is used to test bit 13 of the PID block 0 Status register to check if the Control Block is in manual mode.

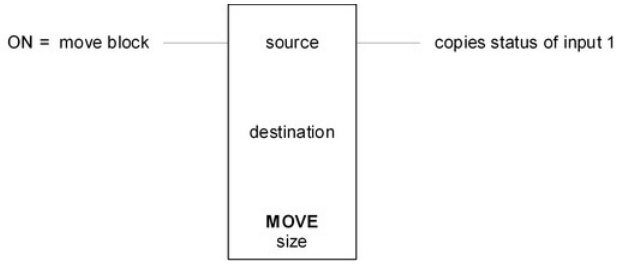
The twenty-five holding registers used for PID block 0 are assigned to the **CNFG PID Control Block I/O Module** in the Register Assignment. In this case holding registers 40220 through 40245 are assigned to the I/O module.

The Control Block status register, 40223 is tested to see if bit 13 is ON. The **GETB** element uses bit index of 0 for the most significant bit. This means that a bit index of 2 will test for bit 13 of the Control Block register.

Closing contacts 10010 sets the **enable GETB** input on. If PID block 0 is in manual mode output 00017 will turn ON.

MOVE (Move Block)

The **MOVE** function block copies the **source** block of registers into the **destination** block of coil, or holding, registers.



Function Block Properties

Source Block (Source)

Type	Register
Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must

begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination Block (Destination)

Type Register

Value For Register type any valid 0xxxx (coil register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Number of Registers in Block (Size)

Type Constant

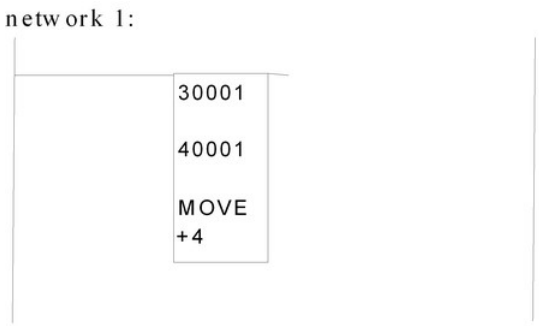
Value For Constant type any number in range 0 to 100. This is the number of 16-bit registers.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Functions

L->L L->R

Example



In the above figure the source register entered is 30001 (3reg) and the destination register entered is 40001 (4reg) and the block size entered is 4.

Block move copies 16 bit words. Coil and status register blocks are groups of 16 registers that start with the register specified as the block address. A block size of 2 corresponds to 32 coils, or two holding registers.

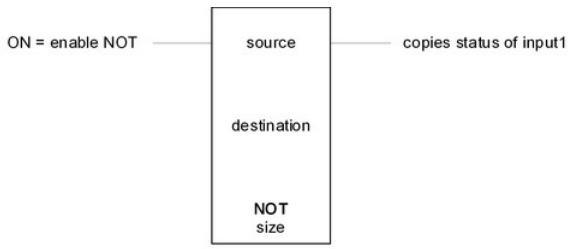
Coil and status register blocks need to begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033 and so on.

When this **MOVE** block is enabled:

- the contents of register 30001 are copied to register 40001
- the contents of register 30002 are copied to register 40002
- the contents of register 30003 are copied to register 40003
- the contents of register 30004 are copied to register 40004

NOT (Not block)

The **NOT** function block inverts the **source** block of registers and stores the result in the **destination** block of coil or Holding registers. **NOT** changes bits that are ON to OFF, and changes bits that are OFF to ON.



Function Block Properties

Source Block (Source)

Type Register

Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination Block (Destination)

Type	Register
Value	For Register type any valid 0xxxx (coil register) or 4xxxx (holding register). For 4xxxx registers the result of the AND operation is stored in register (4xxxx) to register (4xxxx + size). For 0xxxx registers the result of the AND operation is stored in register (0xxxx+1) to register (0xxxx + size). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

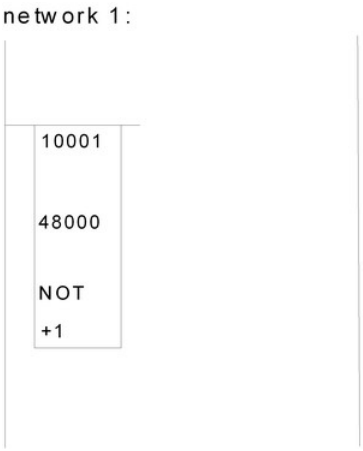
Number of Registers in Block (Size)

Type	Constant
Value	For Constant type any number in range 0 to 100. This is the number of 16-bit registers in the queue.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

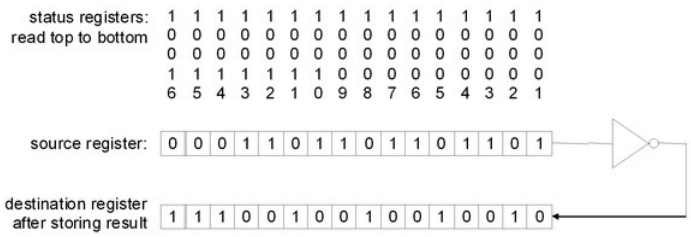
Related function blocks

AND	OR	XOR

Example

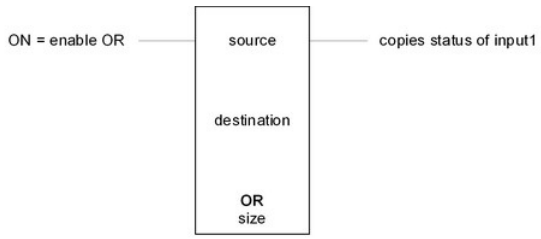


In this example the status of discrete inputs 10001 through 10016 are used as the **source** for the NOT element. These inputs are inverted and stored in the **destination** register 48000.



OR (Or Block)

The **OR** function block logically ORs the **source** block of registers with the **destination** block of registers and stores the result in the **destination** block of registers.



Function Block Properties

Source Block (Source)

Type	Register
Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Source and Destination Block (Destination)

Type	Register
Value	For Register type any valid 0xxxx (coil register) or 4xxxx (holding register). For 4xxxx registers the result of the OR operation is stored in register (4xxxx) to register (4xxxx + size). For 0xxxx registers the result of the OR operation is stored in register (0xxxx+1) to register (0xxxx + size). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

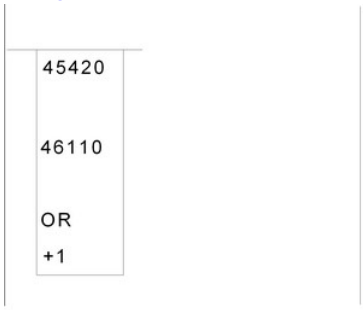
Number of Registers in Block (Size)

Type	Constant
Value	For Constant type any number in range 0 to 100. This is the number of 16-bit registers in the queue.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

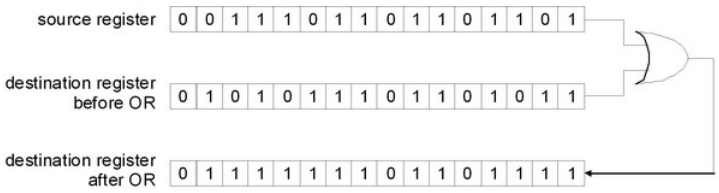
Related function blocks

AND	NOT	XOR

Example



In this example the contents of source register, 45420, are ORed with the destination register, 46110. The result is stored in the destination register. The content of register 45420 is 15213_{10} (11101101101101). The content of the destination register before the OR element is 22379_{10} (101011101101011). The content of the destination register after the

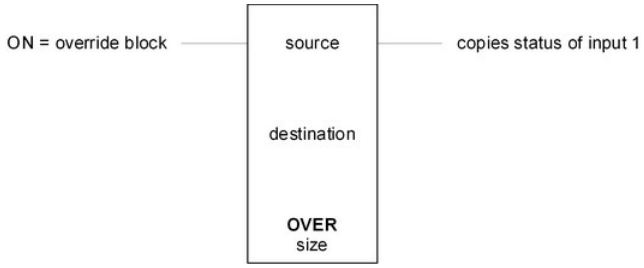


OR element is 32623_{10} (111111101101111).

OVER (Override Block)

The **OVER** function block copies, in one scan, the **source** block of registers into the **destination** block of registers and forces all registers in the **destination** block. When the **override block** input transitions from OFF to ON the destination registers are forced. The **destination** register values are overwritten with the **source** register values while the **override block** input is ON. Destination registers are unforced when the override block input transitions from ON to OFF. The element output copies the input state.

The FORCE led on the controller is turned on when the OVER element is enabled. This element is useful for simulating inputs while debugging a control program.



Function Block Properties

Source Block (Source)

Type	Register
Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Source and Destination Block (Destination)

Type	Register
Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on. so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Number of Registers in Block (Size)

Type	Constant
Value	For Constant type any number in range 0 to 100. This is the number of 16-bit registers in the queue.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Notes

1. The OVER element copies 16-bit words. Coil and status register blocks are groups of 16 registers that start with the register specified as the block address. A block size of 2 corresponds to 32 coils, or two holding registers.
2. Coil and status register blocks need to begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033, and so on.
3. An ON to OFF transition of the input needs to occur to remove the forcing on the registers. If you place an OVER element in a subroutine, be sure to execute the subroutine at least once with the input OFF to clear forcing (see Example below).
4. If forcing is cleared with a Telepace command while the OVER element is enabled, the forcing will not be set again by the OVER element. You need to disable and re-enable the element to force the registers. The element will continue to attempt to write to destination registers even though they are not forced.

Related function blocks

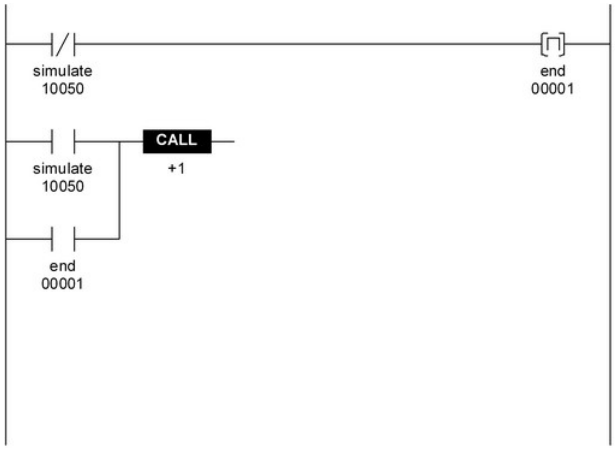
MOVE

Example

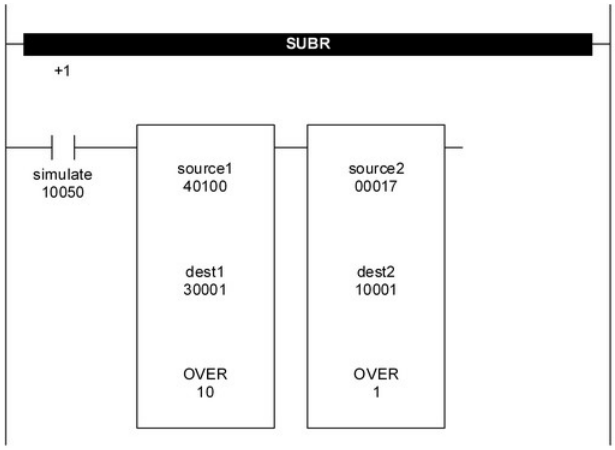
This example shows how simulated values can be written over input values in the I/O database. For this example, assume there is a register assignment that connects input hardware to registers 30001 to 30011, and to registers 10001 to 10016.

The simulate contact (10050) enables and disables the simulation. When the contact is ON simulated values are used. To simulate the values of the inputs, the program overrides the values in registers 30001 to 30011, and registers 10001 to 10016.

The first network calls the subroutine when the simulate contact is ON, and one additional time when the contact goes from ON to OFF.



The second network is a subroutine that overrides the destination registers while the simulate contact is closed. A subroutine is used so that the logic is not scanned when the simulated values are not needed. In most programs this is the normal operating state, so scanning the elements would add unnecessarily to the execution time of the program.

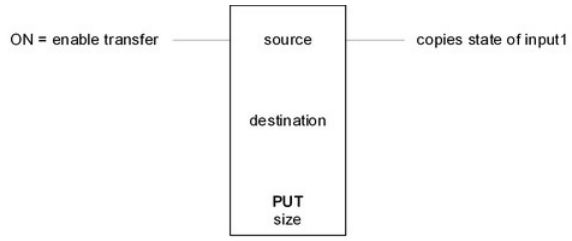


PUT (Put Signed Value Into Registers)

The **PUT** function block transfers, in one scan, the **source** holding register or signed constant into the **destination** holding registers. The number of destination registers is determined by **size**. The same value is stored in each destination register.

The source register, or signed constant, is transferred to the destination holding register when **enable transfer** is ON.

If the source is a register, this element is identical to the **PUTU** element.



Function Block Properties

Source

Type	Constant Register
Value	For Constant type any number in range -32768 to 32767.

	For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

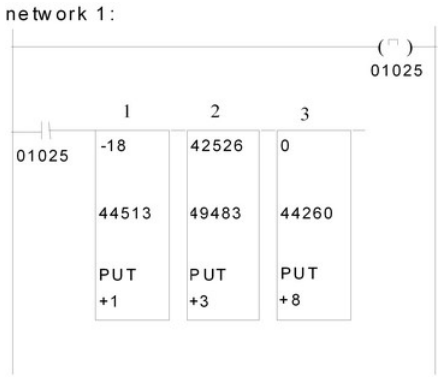
Size

Type	Constant
Value	For Constant type any number in range 1 to 9999.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

PUTF | PUTU

Example



In this example there are three PUT elements in network 1. On power up One Shot coil 01025 is energized for one scan. This will cause NO contacts 01025 to close, applying power to the enable transfer input on PUT elements 1,2 and 3.

- In PUT 1 the constant value -18 is transferred to holding register 44513.
- In PUT 2 the value of holding register 42526 (1000) is transferred to holding registers 49483 through 49485.
- In PUT 3 the constant value 0 is transferred to holding registers 44260 through 44267.

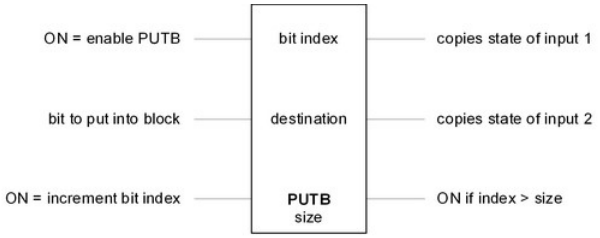
PUTB (Put Bit into Block)

The **PUTB** function block writes the status of a bit, at the **bit index**, into the **destination** block of registers.

The **enable PUTB** input needs to be energized for a PUTB element to execute.

The **bit to put into block** input is the bit value that will be written into the destination register at the location of the bit index when **enable PUTB** is energized.

The **increment bit index** input increments the index, once each scan, when **enable PUTB** is energized.



Function Block Properties

Bit Index into the Block (Bit Index)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 4xxxx (holding register). A bit index of 0 is the most significant bit of the second source block.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on

tag names.

Source (Source)

Type	Register
Value	For Register type any valid 4xxxx (holding register). For Register type any valid 0xxxx (coil register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

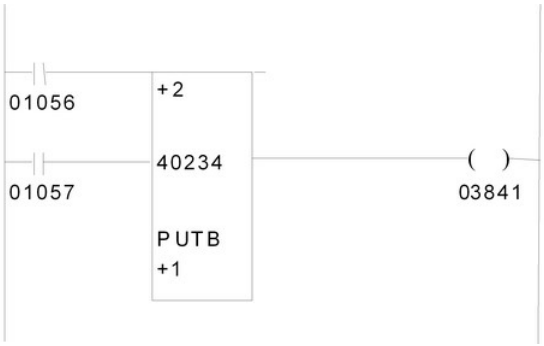
Number of Registers in Block (Size)

Type	Constant
Value	For Constant type any number in range 0 to 600. This is the number of 16-bit registers.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

CMPB | GETB | ROTB

Example



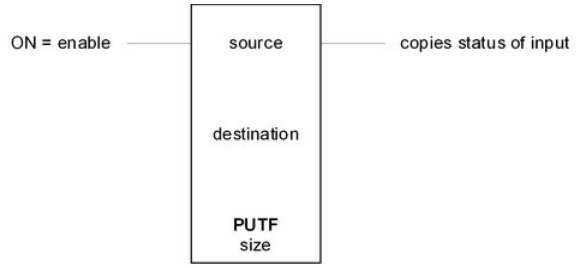
In network 1 the **PUTB** function block is being used to set the manual control bit in the PID Block 0 Control Register, bit 13 of register 40234. The bit index of the manual control bit is 2, with 0 being the most significant bit.

Closing contacts 01056 allows power to be supplied to the **enable PUTB** input of the PUTB element. With the element enabled, closing contacts 01057 allows power to the **put bit into block** input of the element. The bit at the index is then set in the destination register, the PID Block 0 Control Register. Output coil 03841 is energized.

PUTF (Put Floating-point Value)

The **PUTF** function block transfers, in one scan, the **source** floating-point holding register or constant into the **destination** floating-point holding registers. The number of floating-point destination registers is determined by **size**. The same value is stored in each floating-point register pair.

When the **enable** input is ON, the source register or constant is transferred to the destination registers. The element output is ON when the input is.



Function Block Properties

Source

Type	Constant Register
Value	For Constant type any IEEE single precision floating-point number. For Register type any two valid sequential 3xxxx (input register) or 4xxxx (holding register) registers. See Register Types for further information.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination

Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers.

See Register Types for further information.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Size

Type Constant

Value For Constant type any number in range 0 to 4999.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

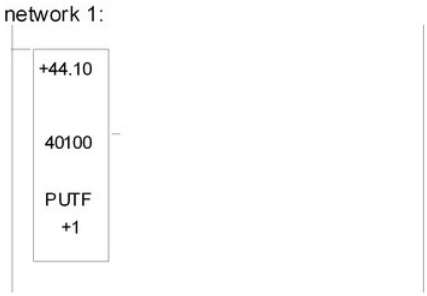
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38}.

Related function blocks

PUT | PUTF

Example



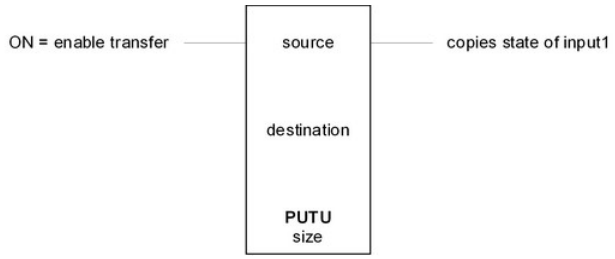
The **PUTF** function block in network 1 puts the floating-point constant +44.10 value into floating-point register 40100 (registers 40100 and 40101).

PUTU (Put Unsigned Values into Registers)

The **PUTU** function block transfers, in one scan, the **source** holding register or unsigned constant into the **destination** holding registers. The number of destination registers is determined by **size**. The same value is stored in each destination register.

The source register, or unsigned constant, is transferred to the destination holding register when **enable transfer** is ON.

If the source is a register, this element is identical to the **PUT** element.



Function Block Properties

Source

Type Constant
Register

Value For Constant type any number in range o to 65535.
For Register type any valid 3xxxx (input register) or 4xxxx (holding register).

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination

Type Register

Value For Register type any valid 4xxxx (holding register).

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Size

Type Constant

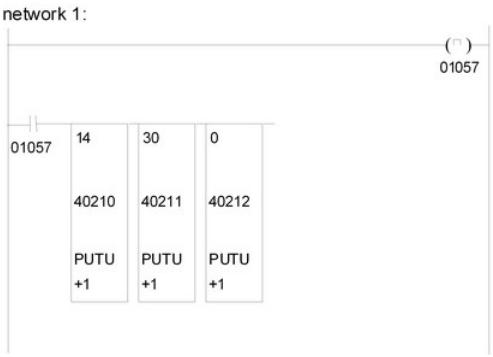
Value For Constant type any number in range1 to 9999.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

PUT | PUTF

Example



This example uses two PUTU function blocks to write the correct time to the Real Time Clock. The time for this example is 14:30:00.

The holding registers used to in this example need to be assigned to the **CNFG Real Time Clock and Alarm I/O Module** in the Register Assignment. Holding registers 40210 through 40220 are assigned to this I/O module.

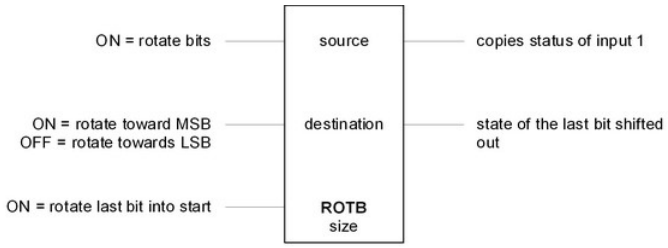
On power up One Shot coil 01057 is energized for one scan. This will cause NO contacts 01057 to close, applying power to the enable transfer inputs on the two PUTU elements.

The real time clock continues to run with the new time setting immediately after these PUTU elements are enabled. The seven clock registers need to be set to valid values for the clock to operate correctly.

Another method for setting the clock, is to use the *Edit/Force Register* dialog to write the current time to the appropriate module registers. Leave the force box in the dialog unchecked so that the data is only written, not forced.

ROTB (Rotate Bits in Block)

The **ROTB** function block rotates the bits in the **source** block by one bit and stores the result in the **destination block**. The block can be rotated towards either the most significant bit (MSB) or towards the least significant bit (LSB).



Function Block Properties

Source Block (Source)

Type	Register
Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination Block (Destination)

Type	Register
Value	For Register type any valid 0xxxx (coil register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Number of Registers in Block (Size)

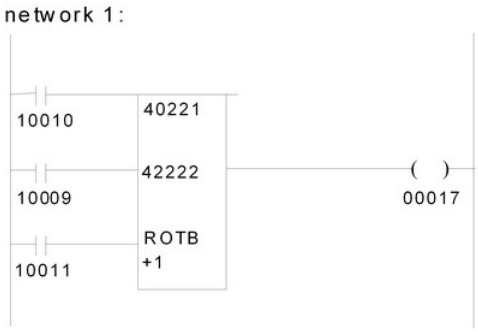
Type	Constant
Value	For Constant type any number in range 0 to 100. This is the number of 16-bit registers.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using

this window. See the Tags Management section for further information on tag names.

Related function blocks

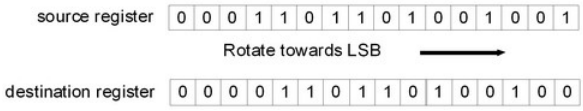
CMPB	GETB	PUTB

Example

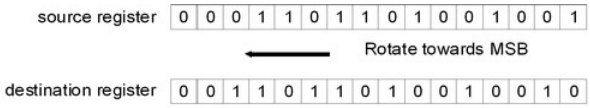


The ROTB function block in network 1 is configured to demonstrate the operation of the element. Register 40221, the source register, has a value of 6985. Register 40222 is the destination register.

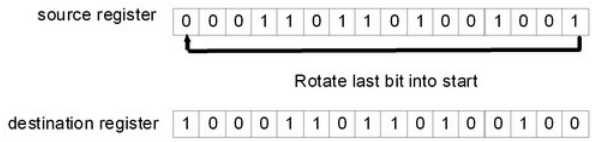
The first example is a rotate towards LSB. Closing the **rotate bits** input, contacts 10010, will cause the bits in the source register to rotate right by one bit and the result to be stored in the destination register.



The next example is a rotate towards MSB. Closing the **rotate towards MSB** input, contacts 10009 and then closing the **rotate bits** input, contacts 10010, will cause the bits in the source register to rotate once to the left. The result is stored in the destination register.

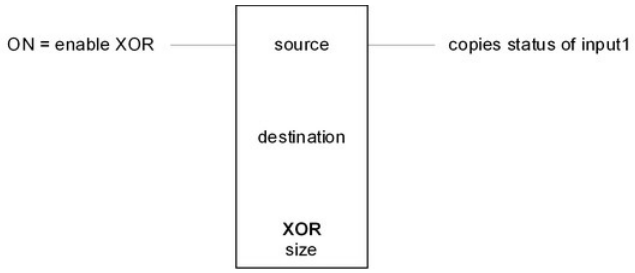


In the last example the **rotate bits** input is enabled by closing contact 10010. Closing contacts 10011 will enable the **rotate last bit to start** input. The LSB of the source register is rotated to the MSB and the remaining bits are rotated right.



XOR (Exclusive or Block)

The **XOR** function block logically EXCLUSIVE-ORs the **source** block of registers with the **destination** block of registers and stores the result in the destination block of registers.



Function Block Properties

Source Block (Source)

Type	Register
Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Source and Destination Block (Destination)

Type Register

Value For Register type any valid 0xxxx (coil register) or 4xxxx (holding register).
For 4xxxx registers the result of the XOR operation is stored in register (4xxxx) to register (4xxxx + size).
For 0xxxx registers the result of the XOR operation is stored in register (0xxxx+1) to register (0xxxx + size).
This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Number of Registers in Block (Size)

Type Constant

Value For Constant type any number in range 0 to 100. This is the number of 16-bit registers in the queue.

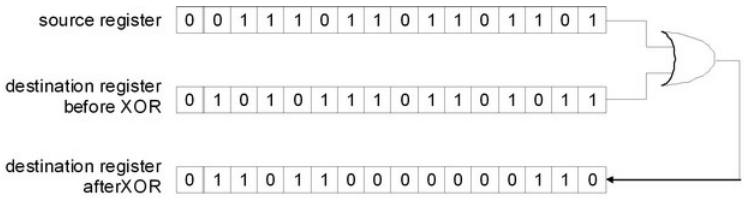
Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related function blocks

AND | NOT | OR

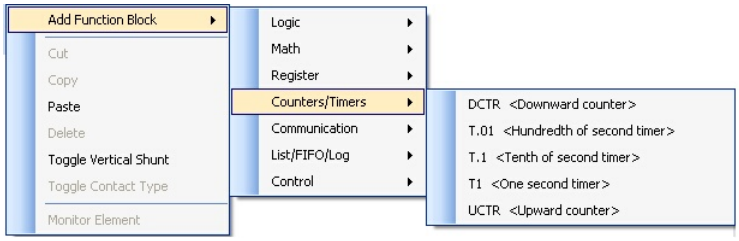
Example

In this example the source block has a value of 15213_{10} (11101101101101). The destination register has a value of 22379_{10} (101011101101011) before the XOR function block and a value of 27654_{10} after the **XOR** function block.



Counters / Timers Functions

The Counter and Timer functions available in Telepace Studio are shown in the image below. This image is opened in Telepace Studio by right clicking on the ladder canvas and selecting the Add Function Block command. Depending on the available space on the ladder canvas some functions may not be displayed as they would not fit in the canvas at the location selected.



Hover your mouse over the icons below for a description of the function. Click the icon to go to the function description.

DCTR | T.01 | T.1 | T1 | UCTR

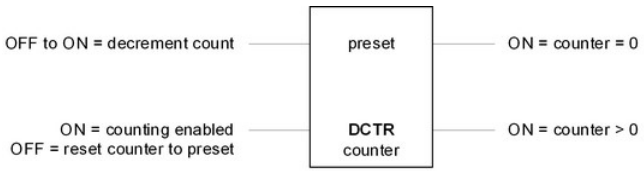
DCTR (Down Counter)

The **DCTR** function block decrements (subtracts 1 from) the value in the **preset** register or constant and stores it in the **counter** holding register when the **decrement count input** changes from OFF to ON. The counter stops counting when it reaches 0.

When the **counting enabled** input is ON and the **decrement count** goes from OFF to ON the counter decrements by one. When the **counting enabled** input is OFF the counter value is reset to preset.

When the **counting enabled** input is ON and the counter has decremented to zero then the **counter = 0** output is ON.

When the **counting enabled** input is ON and the counter is greater than the preset then the **counter > 0** output is ON.



Function Block Properties

Preset Value (Preset)

Type	Constant
	Register
Value	For Constant type any number in range 0 to 65535.
	For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

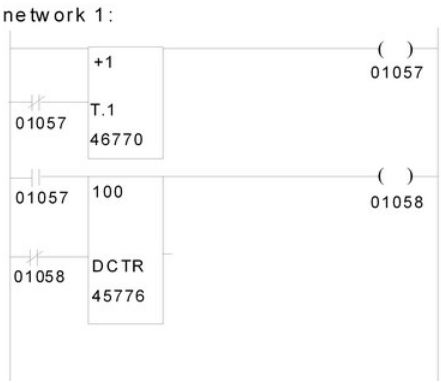
Counter Accumulator (Counter)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Function Blocks

UCTR

Example



The DCTR function block in network 1 decreases each time the timer output is enabled. The timer limit is 0.1 seconds and it will take 10 seconds for the DCTR to decrease 100 times.

When the DCTR reaches 0, output coil 01058 is energized. NC contacts 01058 will open and the DCTR will reset counter to preset. This will happen every 10 seconds.

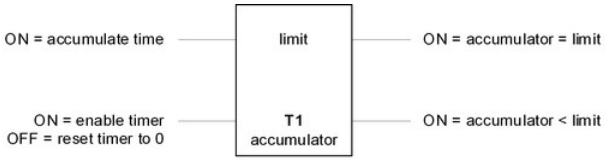
T.01 Timer (hundredths of seconds)

The **T.01** timer function block measures elapsed time and save it in a holding register, **accumulator**. The **T.01** timer measures time in 0.01 second intervals.

When the **enable timer** input is ON and the **accumulate time** is ON the accumulator increments by the Timer interval to the value of the **limit.** When the **enable timer** is off the accumulator is reset to zero.

When the **enable timer** is ON and the **accumulator** equals the **limit** then the **accumulator = limit** output is ON.

When the **enable timer** is ON and the **accumulator** is less than the **limit** then the **accumulator < limit** output is ON.



Function Block Properties

Preset Value (Limit)

Type	Constant

	Register
Value	For Constant type any number in range 0 to 65535.
	For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Timer Accumulator (Accumulator)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

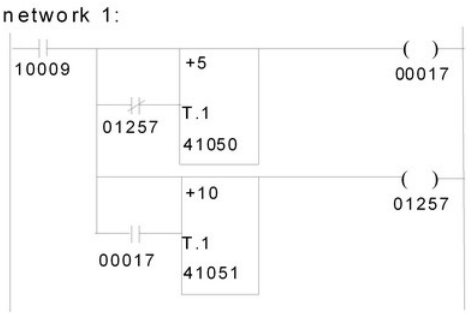
Notes

1. Timer elements are affected by the ladder logic scan time in that the Accumulator value is updated only when the Timer element is scanned.
2. For the T.01 timer the Accumulator value is updated when the element encounters a new second when it is scanned, and not necessarily after exactly 0.01 seconds has elapsed. As a result when N hundredths of seconds have been set for the Accumulator Limit value the limit will be reached anywhere between (N-1) and (N+ scan time) hundredths of seconds.

Related Function Blocks

T.01	**T.1**	**T1**

Example



In network 1 two Timer function blocks are configured such that output coil 00017 is ON for 1.0 second (10 * 0.1) and OFF for 0.5 second (5 * 0.1).

When power is applied to the circuit by closing contacts 10009, output coil 00017 is OFF and Timer 41050 starts to accumulate time. When the accumulator reaches the limit of 5, output coil 00017 turns ON. Timer 41051 is enabled by the NO contacts of output coil 00017 and starts to accumulate time. When the accumulator reaches the limit of 10, output coil 01257 turns ON and resets Timer 41050 by the NC contacts of 01257. Output coil 00017 turns OFF. The process is then repeated until contacts 10009 are opened.

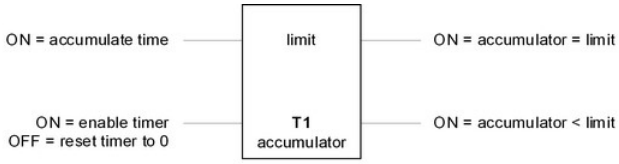
T.1 Timer (tenths of seconds)

The **T.1** timer function block measures elapsed time and save it in a holding register, **accumulator**. The **T.1** timer measures time in 0.1 second intervals.

When the **enable timer** input is ON and the **accumulate time** is ON the accumulator increments by the Timer interval to the value of the **limit.** When the **enable timer** is off the accumulator is reset to zero.

When the **enable timer** is ON and the **accumulator** equals the **limit** then the **accumulator = limit** output is ON.

When the **enable timer** is ON and the **accumulator** is less than the **limit** then the **accumulator < limit** output is ON.



Function Block Properties

Preset Value (Limit)

Type	Constant
	Register
Value	For Constant type any number in range 0 to 65535.
	For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Timer Accumulator (Accumulator)

Type	Register
Value	For Register type any valid 4xxxx (holding register).

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

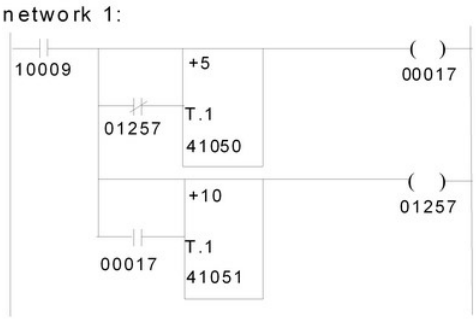
Notes

1. Timer elements are affected by the ladder logic scan time in that the Accumulator value is updated only when the Timer element is scanned.
1. For the T.1 timer the Accumulator value is updated when the element encounters a new second when it is scanned, and not necessarily after exactly 0.1 seconds has elapsed. As a result when N tenths of seconds have been set for the Accumulator Limit value the limit will be reached anywhere between (N-1) and (N+ scan time) tenths of seconds.

Related Function Blocks

T.01	T1

Example



In network 1 two Timer function blocks are configured such that output coil 00017 is ON for 1.0 second (10 * 0.1) and OFF for 0.5 second (5 * 0.1).

When power is applied to the circuit by closing contacts 10009, output coil 00017 is OFF and Timer 41050 starts to accumulate time. When the accumulator reaches the limit of 5, output coil 00017 turns ON. Timer 41051 is enabled by the NO contacts of output coil 00017 and starts to accumulate time. When the accumulator reaches the limit of 10, output coil 01257 turns ON and resets Timer 41050 by the NC contacts of 01257. Output coil 00017 turns OFF. The process is then repeated until contacts 10009 are opened.

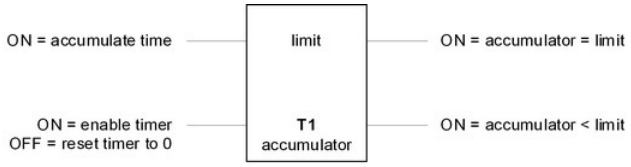
Timers

The **T1** timer function block measure elapsed time and save it in a holding register, **accumulator**. The **T1** timer measures time in 1.0 second intervals.

When the **enable timer** input is ON and the **accumulate time** is ON the accumulator increments by the Timer interval to the value of the **limit.** When the **enable timer** is off the accumulator is reset to zero.

When the **enable timer** is ON and the **accumulator** equals the **limit** then the **accumulator = limit** output is ON.

When the **enable timer** is ON and the **accumulator** is less than the **limit** then the **accumulator < limit** output is ON.



Function Block Properties

Preset Value (Limit)

Type Constant
Register

Value For Constant type any number in range 0 to 65535.
For Register type any valid 3xxxx (input register) or 4xxxx (holding register).

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Timer Accumulator (Accumulator)

Type Register

Value For Register type any valid 4xxxx (holding register).

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

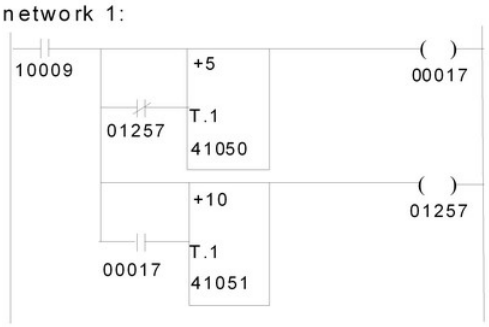
Notes

1. Timer elements are affected by the ladder logic scan time in that the Accumulator value is updated only when the Timer element is scanned.
2. For the T1 timer the Accumulator value is updated when the element encounters a new second when it is scanned, and not necessarily after exactly 1.000 seconds has elapsed. As a result when N seconds have been set for the Accumulator Limit value the limit will be reached anywhere between (N-1) and (N+ scan time) seconds.

Related Function Blocks

T.01 T.1

Example



In network 1 two Timer function blocks are configured such that output coil 00017 is ON for 1.0 second (10 * 0.1) and OFF for 0.5 second (5 * 0.1).

When power is applied to the circuit by closing contacts 10009, output coil 00017 is OFF and Timer 41050 starts to accumulate time. When the accumulator reaches the limit of 5, output coil 00017 turns ON. Timer 41051 is enabled by the NO contacts of output coil 00017 and starts to accumulate time. When the accumulator reaches the limit of 10, output coil 01257 turns ON and resets Timer 41050 by the NC contacts of 01257. Output coil 00017 turns OFF. The process is then repeated until contacts 10009 are opened.

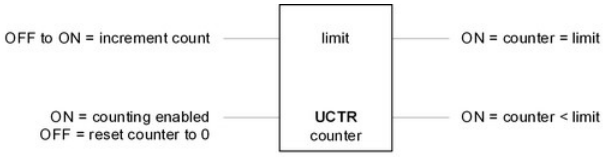
UCTR (Up counter)

The **UCTR** function block increments (adds one to) the value in the **counter** register when the **increment count** input changes from OFF to ON. The counter stops counting when the **limit** register or constant is reached.

When the **counting enabled** input is ON and the **increment count** changes from OFF to ON the counter value increments by one. When the **counting enabled** input is OFF the counter is reset to zero.

When the **counting enabled** input is ON and the counter equals the limit then the **counter = limit** output is ON.

When the **counting enabled** input is ON and the counter is less than the limit then the **counter < limit** output is ON.



Function Block Properties

Upper Limit (Limit)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

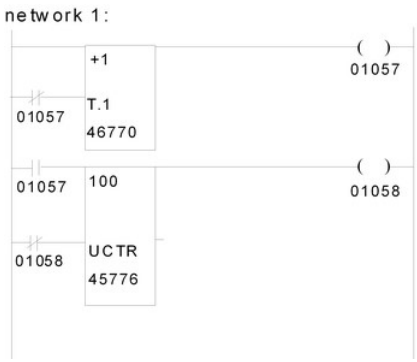
Counter Accumulator (Counter)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Function Blocks

DCTR

Example



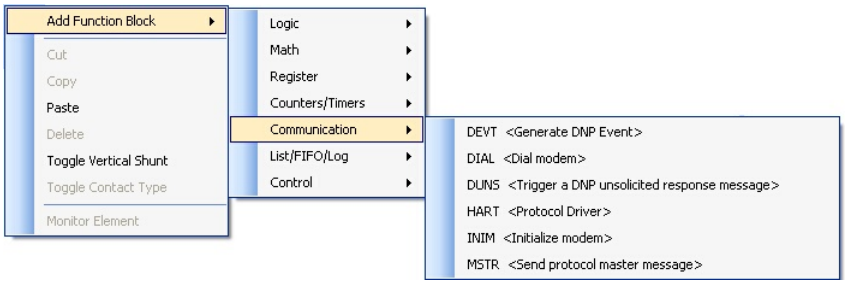
The **UCTR** function block in network 1 increments each time the timer output is enabled. The timer limit is 0.1 seconds and it will take 10 seconds for the UCTR to increment 100 times.

When the limit is reached after 10 seconds the output coil 01057 is energized. NC contact 01057 will open resetting the timer and NO contact 01057 will close incrementing the **UCTR.**

When the **UCTR** increments to 100 output coil 01058 is energized. NC contacts 01058 will open and the **UCTR** will reset counter to zero. This will happen every 10 seconds.

Communication Functions

The Communications functions available in Telepace Studio are shown in the image below. This image is opened in Telepace Studio by right clicking on the ladder canvas and selecting the Add Function Block command. Depending on the available space on the ladder canvas some functions may not be displayed as they would not fit in the canvas at the location selected.



Hover your mouse over the icons below for a description of the function. Click the icon to go to the function description.

DEVT	DIAL	DUNS	HART
INIM	MSTR		

DIAL (Control Dial-up Modem)

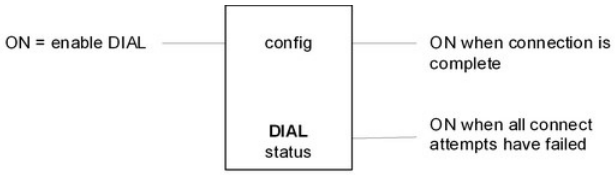
The **DIAL** element connects an internal modem or an external modem to a remote modem. Only one DIAL block may be active on each serial communication port at any one time. The **DIAL** element handles all port sharing and multiple dialing attempts.

The SCADAPack 100 does not support dial-up connections on com port 1. Dial-up connections are not supported onSCADAPack controllers.

When the **enable DIAL** input is ON the DIAL element sends the modem initialization string to the modem. The modem will attempt to connect to the remote modem. If a connection is made the **ON when connection complete** output is turned ON. If a connection is not made in the specified number of attempts the **ON when all connect attempts have failed** output is turned ON.

The **enable DIAL** input needs to be energized for the entire DIAL time, including multiple dial attempts.

So that the external modem can hang up, a pause is required after hanging up the telephone line after a DIAL connection before initiating a new DIAL attempt.



Function Block Properties

Configuration Block

Type Register

Value Address of the first register (**config**) of 39 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using

other Telepace Studio function blocks or from a Modbus device communicating with the controller.
The block of registers are defined for the function block as follows:
config + 0 = Communication port
config + 1 = Dialing attempts
config + 2 = Connect time
config + 3 = Pause time
config + 4 = Dialing type
config + 5 = Length of modem string
config + 6 to 21 = Modem initialization string
config + 22 = Length of phone number
config + 23 to 38 = Phone number
The modem initialization string and the phone number are packed registers. Two ASCII characters are stored in each register. See Register Types for further information.

Tag The tag window displays the tag name for the config + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Status Value

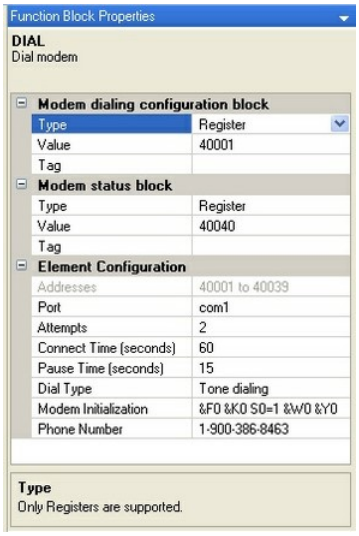
Type Register

Value Address of the first register (**status**) of 2 in a sequential block of 4xxxx (holding) registers. These registers contain the status information of the DIAL function.
Status + 0 = error code
Status + 1 = reservation identifier
See the **Error Code** section below for information on any errors returned by the function block.

Tag The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration

The **DIAL** function block may be configured using the Element Configuration.



The **Addresses** window displays the registers used by the function block.

The **Port** specifies the controller serial port to use for the DIAL function. This is port the external modem is connected to. The menu displays valid serial ports for the controller type.

The **Attempts** specifies the number of times the controller will attempt to dial before giving up and reporting an error.

The **Connect Time (seconds)** specifies the length of time, in seconds, that the controller will wait for carrier to be detected. This time is measured from the start of the dial attempt.

The **Pause Time (seconds)** is the length of time, in seconds, that the controller will wait between dial attempts.

The **Dial Type** is the type of dialing used, tone or pulse.

The **Modem Initialization** is the initialization string to be sent to the modem. The string cannot be more than 32 characters long. The function block automatically adds the "AT" and "DT" or "DP" commands. These commands are not to be included in the string.
For example the initialization string &F0 S0=1 is acceptable but he initialization string AT&F0 S0=1 is not.

The **Phone Number** is the phone number to dial. The function block automatically adds the "AT" and "DT" or "DP" commands. These commands are not to be included in the string.
For example the phone number string 555-1212 is acceptable but he initialization string AT&555-1212 is not.

Error Codes

The **error** codes are:

Error Code	Error Description
0	No error.
1	Bad configuration error occurs when an incorrect initialization string is sent to the modem. This usually means the modem does not understand a specific command in the initialization string.
2	No modem connected to the controller serial port, or the controller serial port is not set to RS-232 modem.
3	Initialization error occurs when the modem does not respond to the initialization string and may be turned off.
4	No dial tone error indicates modem is not connected to a telephone line or the S6 setting in the modem is too short.
5	Busy line indicates another device is using the phone line.
6	Call aborted by the program. This occurs if the enable DIAL input goes OFF before a modem connection occurs.
7	Failed to connect error occurs when there are no other errors but the modem failed to make a connection with the remote modem.
8	Carrier lost error occurs when carrier is lost for a time exceeding S10 setting in the modem. Phone lines with call waiting are very susceptible to this condition.
9	"serial port is not available" error occurs when the DIAL element attempts to use the serial port when another ladder communication element, C program, or an incoming call has control of the port.

Notes

1. The reservation identifier is required for the operation of the dialer but can be ignored by the ladder program.

2. The DIAL element outputs are powered only when the **enable DIAL** input is powered ON.
3. The dial-up connection handler prevents outgoing calls from using the serial port when an incoming call is in progress and communication is active. If communication stops for more than five minutes, then outgoing call requests are allowed to end the incoming call. This prevents problems with the modem or the calling application from permanently disabling outgoing calls.
4. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator display terminals, and for programming and diagnostics using the Telepace program.

Related Function Blocks

INIM

Modem Command Reference

The **DIAL** and **INIM** elements, and the Telepace program dial-up connection operate with Hayes AT command set compatible external modems. The commands specify how the modem behaves. Each modem manufacturer has a different set of modem commands required to initialize the modem.

Modem Settings

If generic modem settings are not working, fill out the table below by looking up the commands in your modem manual. Construct a command string from the results. Your modem may not require all the commands or may require additional commands. Contact our Technical Support department if you require assistance.

Command	Description
	reset modem to factory or default settings
	(This command should be sent first.)
	select extended result code set
	return verbal responses
	do not echo characters in command state
	return actual state of carrier detect (CD) signal
	hang up when **DTR** data terminal ready signal is dropped
	disable (local) flow control
	Communicate at a constant speed between **DTE** data terminal equipment and modem
	answer incoming calls on the first ring
	carrier wait time = 50 seconds

The command strings do not begin with AT. The AT will be inserted automatically when the commands are sent to the modem.

Generic Modem

This command string works with many Hayes-compatible modems.

String *X4 V1 E0 &C1 &D2 S0=1 S7=50*

Command	Description
X4	select extended result code set
V1	return verbal responses
E0	do not echo characters in command state
&C1	return actual state of carrier detect (CD) signal
&D2	hang up when **DTR** data terminal ready signal is dropped
S0=1	answer incoming calls on the first ring (range is 0 to 255)
S7=50	carrier wait time = 50 seconds (range is 1 to 255)

5901 High speed Dial-up Modem

String *&F0 &K0 S0=1 &W0 &Y0*

Command	Description
&F0	reset modem to factory or default settings (This command should be sent first.)
X4	select extended result code set
V1	return verbal responses
E0	do not echo characters in command state
&C1	return actual state of carrier detect (CD) signal
&D2	hang up when **DTR** data terminal ready signal is dropped
&K0	disable (local) flow control
Nn and S37 (see user manual)	Communicate at a constant speed between **DTE** Data terminal equipment and modem
S0=1	answer incoming calls on the first ring
S7=50	carrier wait time = 50 seconds

ATI 1440 ETC-Express

String *I&F0 X4 &C1 &D2 &K0 &Q0*

Command	Description
&F0	recall factory (configuration) profile 0
X4	select extended result code set
&C1	return actual state of carrier detect (CD) signal
&D2	hang up when DTR signal is dropped
&K0	disable flow control
&Q0	select direct asynchronous mode

ATI 14400 ETC-E, ETC-1

String *&F0 &B1 &C1 &D2 &E0*

Command	Description
&F0	recall factory (configuration) profile 0
&B1	DTE speed equals default value or the AT command speed
&C1	return actual state of carrier detect (CD) signal
&D2	hang up when DTR signal is dropped
&E0	automatic retrying disabled

Hayes Smartmodem 1200

String *F1 Q0 V1 X1*

Command	Description
F1	full duplex
Q0	result codes sent
V1	result codes transmitted as words (verbal)
X1	extended result code set

Hayes ACCURA 96, 144, 288 Modems

String *&F &C1 &D2 &K0 &Q6*

Command	Description
&F	recall factory configuration
&C1	return actual state of carrier detect (CD) signal
&D2	hang up when DTR signal is dropped
&K0	disable local flow control
&Q6	Communicate in asynchronous mode with automatic **speed buffering** Speed buffering maintains a constant speed between DTE and the modem.

Kama 2400 EI

String *&F &C1 &D2*

Command	Description
&F	recall factory profile 0
&Q0	select direct asynchronous mode
%C0	disable data compression
S7=60	wait time for carrier = 60 seconds

Megahertz XJ4288 28.8 PC Card Modem

String *&F0 &Q0 %C0 S7=60*

Command	Description
&F	recall factory profile 0
&Q0	select direct asynchronous mode
%C0	disable data compression
S7=60	wait time for carrier = 60 seconds

Multitech 224E7B

String *F Q0 V1 X4 &C1 &D2 &E3*

Command	Description
F	factory settings
Q0	send result codes
V1	verbal result codes
X4	extended result code set
&C1	**DCD** data carrier detect asserted when carrier detected
&D2	DTR controls the modem
&E3	no flow control connection

TeleSAFE 6901 Bell 212 Modem

1 String

Command	Description	Default
S0 register	rings to answer	1
S2 register	CTS-on delay measured in units of 8.3 ms.	14 (116 ms)
S3	Anti-streaming timer measured in tens of seconds	0 (disabled)
S4	CTS-off delay measured in units 8.3 ms	4 (33 ms)
S6	Dial tone wait time measured in tenths of second.	30 (3 seconds)
S7	Carrier wait time measured in seconds	15 (15 seconds)
S9	Carrier detect delay time measured in units 8.3 ms.	10 (83 ms)
S10	carrier loss time measured in units of 8.3 ms	10 (83 ms)
S11 register	tone dial speed measured in 0.050 seconds	2 (5 p/p second)

US Robotics Sportster 28,000

String *&F1 &A0 &H0 &K0 &M0 &R1 &B1 S0=1*

Command	Description
F	factory settings

Q0	send result codes
V1	verbal result codes
X4	extended result code set
&C1	DCD data carrier detect asserted when carrier detected
&D2	DTR controls the modem
&E3	no flow control connection

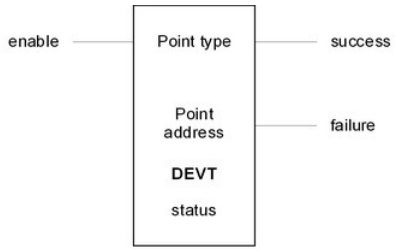
If your modem is not on the list, try the generic modem settings.

DEVT (Generate DNP Event)

The **DEVT** function block generates a DNP change event for the specified DNP point.

A change event is generated for the **Point Address** specified when the **enable** input changes from OFF to ON. The **status** register is set to 0 and the **success** output is energized if the element is successful and a change event was generated.

The status register is set to 1 and the **failure** output is energized if the element failed and a change event was not generated. The failure output indicates that the specified DNP point is invalid, or the DNP configuration has not been created.



Function Block Properties

DNP Point type (Point Type)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 4xxxx (holding register). Valid DNP Point Types: 0 = Binary Input 1 = 16-bit Analog Input 2 = 32-bit Analog Input 3 = Float Analog Input 5 = 16-bit Counter Input, 6 = 32-bit Counter Input
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

DNP Point Address (Point Address)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

DEVT Status (Status)

Type	Register
Value	For Register type any valid 4xxxx (holding register). Status Values: 0 = success 1 = failure
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Function Blocks

DPOL | DUNS

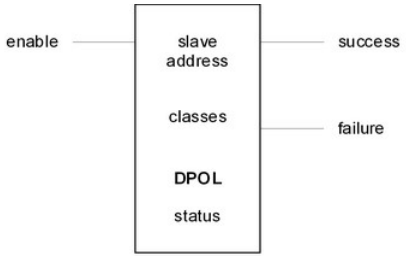
DPOL (Trigger a DNP Class Poll)

The **DPOL** function block sends a DNP Class Poll message to request the specified data classes from a DNP slave. The DNP class poll is requested when the **enable** input changes from OFF to ON.

The **status** register is set to 0 and the **success** output is energized if the element was successful.

The 'success' output is energized immediately if the poll was triggered successfully. It does not provide any information about the success or otherwise of the actual message transaction.

The **status** register is set to 1 and the **failure** output is energized if the element failed. Failure indicates the specified slave address has not been configured in the DNP Routing Table, or the DNP configuration has not been created.



Function Block Properties

DNP Slave Station Address (Slave Address)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

DNP Data Class (Classes)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 4xxxx (holding register). This is a bit-mapped register containing the classes of data to be included in the class poll message. If multiple classes are required the following values should be added together. 1 = class 0 data 2 = class 1 data 4 = class 2 data 8 = class 3 data
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

DPOL Status (Status)

Type	Register
Value	For Register type any valid 4xxxx (holding register). Status Values: 0 = success 1 = failure
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Notes

1. This element is available on the SCADAPack 314, SCADAPack 33x, SCADAPack 350 and SCADAPack 32 controllers only.

2. The element sets internal flags to trigger a DNP poll message, and returns immediately. The DNP message will be sent some time after the element call.

Related Function Blocks

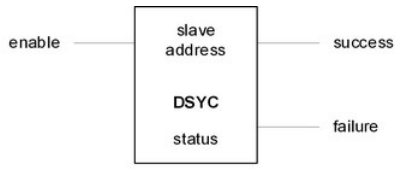
DEVT | DUNS

DSYC (Trigger a DNP Clock Synchronization)

The **DSYC** function block sends a **Clock Synchronization** message to a DNP slave.

The **Clock Synchronization** message is sent to the **slave address** when the **enable** input changes from OFF to ON. The status register is set to 0 and the **success** output is energized if the element was successful.

The **status** register is set to 1 and the **failure** output is energized if the Time Synchronization message failed. The **failure** output indicates the specified slave address has not been configured in the DNP Routing Table, or the DNP configuration has not been created.



Function Block Properties

DNP Slave Station Address (Slave Address)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

DSYC Status (Status)

Type	Register
Value	For Register type any valid 4xxxx (holding register). Status Values: 0 = success 1 = failure
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Notes

1. This element is available on the SCADAPack 314, SCADAPack 33x, SCADAPack 350 and SCADAPack 32 controllers only.

2. The element sets internal flags to trigger a DNP poll message, and returns immediately. The DNP message will be sent some time after the function block call.

Related Function Blocks

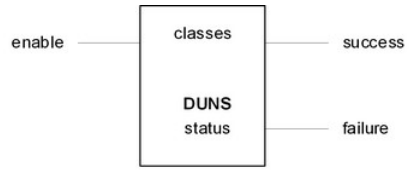
DEVT	DPOL	DUNS

DUNS (Trigger a DNP unsolicited response message)

The **DUNS** element triggers a DNP unsolicited response message of the specified class or classes.

The unsolicited response for the specified **classes** is triggered when the **enable** input changes from OFF to ON. The **status** register is set to 0 and the success output is energized if the message was triggered successfully.

The **status** register is set to 1 and the **failure** output is energized if the unsolicited response message failed. The failure output indicates master addresses have not been configured in the DNP Routing Table, or the DNP configuration has not been created.



Function Block Properties

DNP Data Class (Classes)

Type	Constant Register
Value	For Constant type any number in range 0 to 65535. For Register type any valid 4xxxx (holding register). This is a bit-mapped register containing the classes of data to be included in the unsolicited message. If multiple classes are required, the following values should be added together: 1 = class 0 data 2 = class 1 data 4 = class 2 data 8 = class 3 data
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

DUNS Status (Status)

Type	Register
Value	For Register type any valid 4xxxx (holding register). Status Values: 0 = success 1 = failure
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on

tag names.

Notes

1. The element sets internal flags to trigger a DNP unsolicited message, and returns immediately. The DNP message will be sent some time after the element call.
2. DNP unsolicited messages need not be enabled in the Application Layer configuration for events to be sent with this element. Unsolicited messages at Start Up needs to be enabled, or the master needs to enable unsolicited messages before the first message can be sent.
3. If no events are pending an empty unsolicited message will be sent.
4. The message will be sent to the configured DNP master station or stations.
5. The return status indicates whether the flags were set. The status will be FALSE if the DNP engine is not running.

Related Function Blocks

DPOL | DUNS

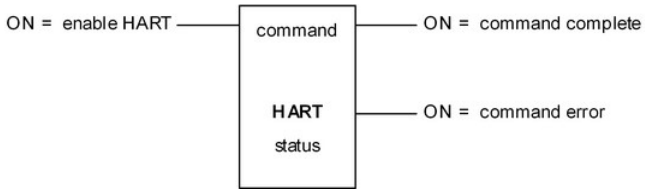
HART (Send HART command)

The **HART** element sends a HART protocol command and processes the response.

The HART element is a two-cell element, with one input and two outputs. When the **enable HART** input changes from OFF to ON, the element sends the HART command to the HART device. When the **enable HART** input changes from ON to OFF, the current command is aborted and no response is processed.

When a response is received, the data in the response is stored in the I/O database and the **command complete** output turns ON. If there is an error in the command, or if the HART device does not respond to any of the attempts, the **command error** output turns ON.

The **enable HART** input needs to remain ON until the response has been received or an error has occurred. All outputs turn OFF when the input is OFF.



Function Block Properties

Command Block

Type	Register
Value	Address of the first register (**command**) of 5 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller. The block of registers are defined for the function block as follows: **command** + 0 = 5904 HART interface module number. This is set to 0, 1, 2 or 3 and corresponds to the module number of the ***5904 HART Interface*** module in the Register Assignment. **command** + 1 = HART device address. This is in the range 0 to 15. **command** + 2 = command number **command** + 3 = command data register address **command** + 4 = response register address
Tag	The tag window displays the tag name for the command + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Status Value

Type	Register
Value	Address of the first register (**status**) of 2 in a sequential block of 4xxxx (holding) registers. These registers contain the status information of the HART function. **status** + 0 = command status **status** + 1 = additional information code See the **Status Code** section below for information on any errors returned by the function block.
Tag	The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration

The **HART** function block may be configured using the Element Configuration.

The **Addresses** window displays the registers used by the function block.

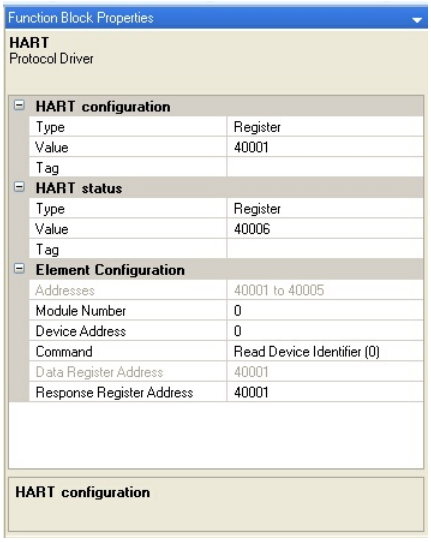
The **Module Number** specifies HART module this command is to be sent to. This is set to 0, 1, 2 or 3 and corresponds to the module number of the ***5904 HART Interface*** module in the Register Assignment.

The **Device Address** specifies HARD device address. This is a value of 0 to 15.

The **Command** specifies the HART command to send. Commands are described in the sections below.

The **Data Register Address** specifies an I/O database register where the data for the command is stored. The program needs to set the values in these registers. The commands tables below show how this address is used.

The **Response Register Address** specifies an I/O database register where the data in the response is stored. The tables below show how this address is used. Registers marked as



(float) contain floating-point values.

Read Device Identifier (0)

Command 0
Purpose Read the device identifier
Data Registers Not used
Response Registers
+0 = manufacturer ID
+1 = manufacturer Device Type
+2 = preambles Requested
+3 = command Revision
+4 = transmitter Revision
+5 = software Revision
+6 = hardware Revision
+7 = flags
+8,9 = device ID (double)
Notes This command is also the link initialization command. The HARD element performs link initialization automatically. You do not need to send this command unless the device identifier is needed.

Read PV (1)

Command 1
Purpose Read primary variable (PV)
Data Registers Not used
Response Registers
+0,1 = PV (float)
+2 = PV units code

Read PV Current and % (2)

Command 2
Purpose Read primary variable current and percent of span
Data Registers Not used
Response Registers
+0,1 = PV current (float)
+2 = PV current units code
+3,4 = PV percent (float)
+5 = PV percent units code

Read Dynamic Variables (3)

Command 3
Purpose Read dynamic variables and primary variable current.
Data Registers Not used
Response Registers
+0,1 = primary variable current (float)
+2 = primary variable current units code
+3,4 = primary variable value (float)
+5 = primary variable units code
+6,7 = secondary variable value (float)
+8 = secondary variable units code
+9,10 = tertiary variable value (float)
+11 = tertiary variable units code
+12,13 = fourth variable value (float)
+14 = fourth variable units code
Notes Not all devices return primary, secondary, tertiary and fourth variables. If the device does not support them, zero is written into the value and units code for that variable.

Read Specified Variables (33)

Command 33
Purpose Read specified transmitter variables.
Data Registers
+0 = variable code 0

Response Registers	+1 = variable code 1 +2 = variable code 2 +3 = variable code 3 +0,1 = variable 0 value (float) +2 = variable 0 units code +3,4 = variable 1 value (float) +5 = variable 1 units code +6,7 = variable 2 value (float) +8 = variable 2 units code +9,10 = variable 3 value (float) +11 = variable 3 units code

Status Codes

The **status** registers contains the current command status and additional code returned from the HART interface. The table below shows the values.

Status Code	Description	Code
0	HART interface module not communicating.	not used
1	Command ready to send device.	not used
2	Command sent to device.	current attempt number
3	Response received.	response code from HART device (see [insert anchor to notes on this page.)]
4	No valid response received after all attempts made.	0=no response from HART device. Other = error response code from HART device.
5	HART interface is not ready to transmit.	not used

HART Device Response Codes

The Device Response Code of the function block contains the response code from the HART device. It contains communication error and status information. The information varies by device, but there are some common values.

If bit 7 of the high byte is set, the high byte contains a communication error summary. This field is bit-mapped. The following table shows the meaning of each bit as defined by the HART protocol specifications. Consult the documentation for the HART device for more information.

Bit	*Description*
6	vertical parity error
5	overrun error
4	framing error
3	longitudinal parity error
2	reserved – 0
1	buffer overflow
0	undefined

If bit 7 of the high byte is cleared, the high byte contains a command response summary. The following table shows common values. Other values may be defined for specific commands. Consult the documentation for the HART device.

Code	*Description*
32	Busy – the device is performing a function that cannot be interrupted by this command
64	Command not Implemented – the command is not defined for this device.

The low byte contains the field device status. This field is bit-mapped. The following table shows the meaning of each bit as defined by the HART protocol specifications. Consult the documentation for the HART device for more information.

Bit	*Description*
7	field device malfunction
6	Configuration changed
5	cold start
4	more status available (use command 48 to read)
3	primary variable analog output fixed
2	primary variable analog output saturated
1	non-primary variable out of limits
0	primary variable out of limits

Related Topics

MSTR

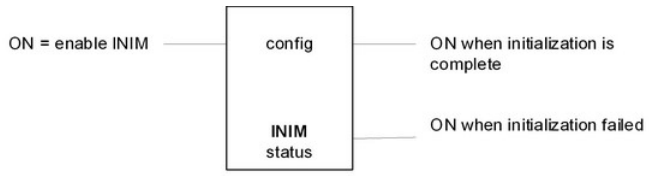
INIM (Initialize Dial-up Modem)

The **INIM** element initializes an internal modem or an external modem, typically to set it to receive calls. Only one INIM block may be active on each serial communication port at any one time.

When **enable INIM** input is ON, the INIM element sends the modem initialization string to the modem. When the initialization is complete the **ON when initialization is complete** output is turned ON. If an error occurs, the **ON when initialization failed** output is turned ON.

The **enable INIM** input needs to be energized for the entire INIM time, including multiple initialization attempts.

The SCADAPack 100 does not support dial-up connections on com port 1.



Function Block Properties

Configuration Block

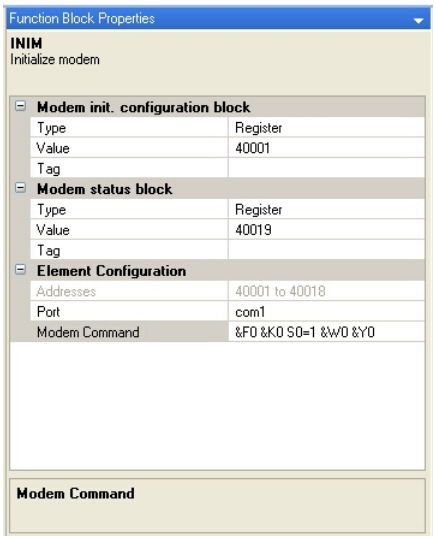
Type	Register
Value	Address of the first register (**config**) of 18 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller. The block of registers are defined for the function block as follows: **config** + 0 = Communication port **config** + 1 = Length of modem string **config** + 2 to 17 = Modem initialization string The modem initialization string uses packed registers. Two ASCII characters are stored in each register. See Register Types for further information.
Tag	The tag window displays the tag name for the config + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Status Value

Type	Register
Value	Address of the first register (**status**) of 2 in a sequential block of 4xxxx (holding) registers. These registers contain the status information of the INIM function. **Status** + 0 = error code **Status** + 1 = reservation identifier See the **Error Code** section below for information on any errors returned by the function block.
Tag	The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration

The **DIAL** function block may be configured using the Element Configuration.



The **Addresses** window displays the registers used by the function block.

The **Port** specifies the controller serial port to use for the DIAL function. This is port the external modem is connected to. Menu displays valid serial ports for the controller type.

The **Modem Command** is the initialization string to be sent to the modem. The string cannot be more than 32 characters long. The function block automatically adds the "AT" and "DT" or "DP" commands. These commands are not to be included in the string.
For example the initialization string &F0 S0=1 is acceptable but he initialization string AT&F0 S0=1 is not.

Notes

1. The modem initialization string is packed. Two ASCII characters are stored in each register.

2. The reservation identifier is required for the operation of the dialer but can be ignored by the ladder program.

Error Codes

Error Code	Description
0	No Error
1	Bad configuration error occurs when an incorrect initialization string is sent to the modem. This usually means the modem does not understand a specific command in the initialization string.
2	No modem is connected to the controller serial port, or the controller serial port is not set to RS-232 modem.
3	Initialization error occurs when the modem does not respond to the initialization string and may be turned off.
6	Call aborted by the program. This will occur if the enable INIM input goes OFF before a modem connection occurs.
9	“Serial port is not available” error occurs when the INIM element attempts to use the serial port when another ladder communication element, C program or an incoming call has control

of the port.

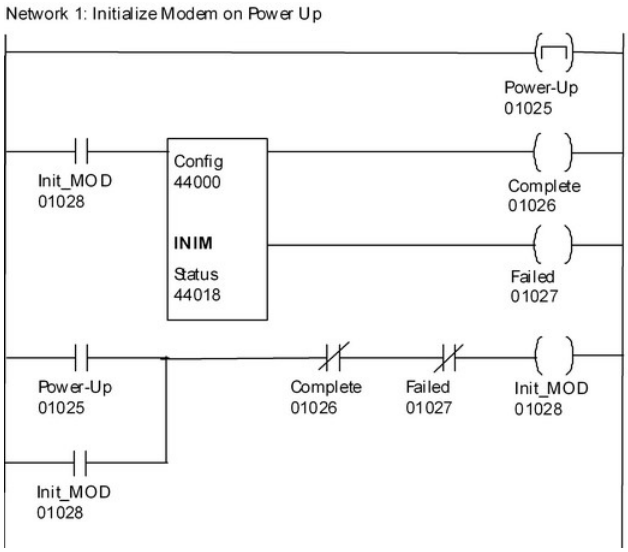
To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator display terminals, and for programming and diagnostics using the Telepace program.

Related Elements

DIAL

Example

The following network configures a generic Hayes compatible modem to answer on the second ring. The modem is initialized when the controller is powered up.



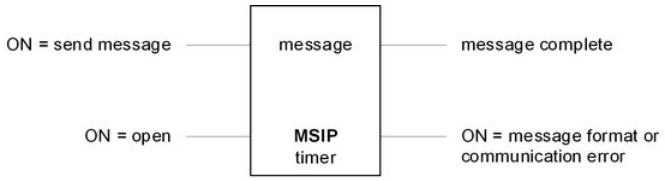
MSIP (Master IP Message)

The **MSIP** element exchanges data with another controller using the Modbus IP communication protocol. This element sends a master message to a remote IP address.

Unlike the MSTR element, multiple MSIP blocks may be active on the same communication interface at the same time. The number of active MSIP blocks is limited only by the number of available connections. A maximum of 20 connections are supported on SCADAPack controllers.

When the **open** input goes from OFF to ON the MSIP element opens an IP connection. Set the **open** input to OFF to close the connection.

When the **send message** input goes from OFF to ON, and the **open** input is ON, the MSIP element will send one Modbus IP master message to the slave address. The **send message** input must go from OFF to ON to send another MSIP message. Each master message is complete when either output is energized: The **message complete** output will be energized if the MSIP block receives a valid response from the slave station. The **message format or communication error** is energized if an error occurs. See the status codes section below for a list of possible errors.



Notes

1. The **time-out** register determines how long to wait for a response before closing the connection and reporting an error. The connection is re-connected automatically when the next message is sent.
2. The **open** input does not need to be toggled to do this.
3. Leave the **open** input ON to keep the connection open for this MSIP element.

Function Block Properties

Message Block

Type	Register
Value	Address of the first register (**message**) of 11 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller. The block of registers are defined for the function block as follows: **message** + 0 = byte1 of slave IP address (most significant byte) using format **byte1.byte2.byte3.byte4** **message** + 1 = byte2 of slave IP address **message** + 2 = byte3 of slave IP address **message** + 3 = byte4 of slave IP address (least significant byte) **message** + 4 = protocol type **message** + 5 = function code

message + 6 = slave controller address
message + 7 = slave register address
message + 8 = master register address
message + 9 = length
message + 10 = time-out in 0.1s increments

Tag The tag window displays the tag name for the message + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Timer Block Value

Type Register

Value Address of the first register (**timer**) of 5 in a sequential block of 4xxxx (holding) registers. These registers contain the accumulated time (see message + 10) for the time out and status information of the MSIP function.

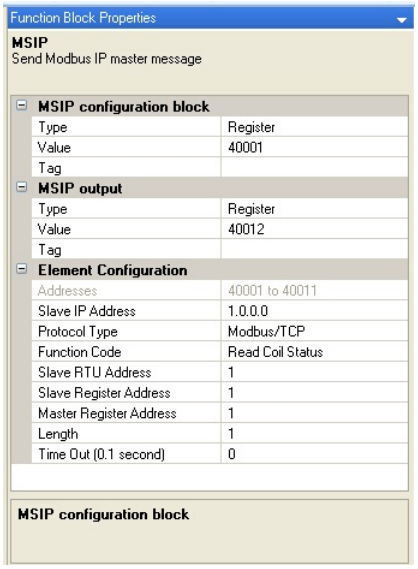
timer + 0 = timer accumulator register
timer + 1 = command status register
timer + 2 = internal: connection ID
timer + 3 = internal: closing state
timer + 4 = internal: initialized

See the **Status Code** section below for information on the current status returned by the function block.

Tag The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration

The **MSIP** function block may be configured using the Element Configuration.



The **Addresses** window displays the registers used by the function block.

The **slave IP address** has the format 255.255.255.255 where one register is used for each byte: byte1.byte2.byte3.byte4

The **protocol type** must be one of the values shown below:

- Modbus/TCP
- Modbus RTU over UDP
- Modbus ASCII over UDP

Valid **Function Codes** are shown in the Supported Function Codes table below.below.

The **slave register address** specifies the first register where data will be read from or written to in the slave controller. The following table shows the valid slave register addresses for each command.

The **master register address** specifies the first register where data will be written to or read from in this controller. The register type does not have to match the type of the slave register address. It is possible to store input registers from a slave into output registers on the master and vice-versa.

The **length** parameter specifies how many registers are to be transferred. The maximum length for each function code is shown in the above table. Functions 05 and 06 always transfer 1 register, regardless of the value of **length**.

The **time-out** is in 0.1s increments. The error output is energized if the time-out period ends without a valid response to the message.

Sending a Message

1. Set the open input to ON to open an IP connection. This input may be set to ON at the same time as the send message input in the next step.
2. If the maximum number of opened connections has been reached, an error occurs: The error output is energized and the value of the command status register indicates the maximum number of connections has been reached. Close the connection used by another MSIP element to correct the error and repeat step 1.
3. Send one command message by toggling the send message input from OFF to ON.
4. The value of the command status register is 0 indicating that the message was sent.
5. Sending is complete when either the message complete output or error output is energized. The message complete output is energized if the response is received successfully. The error output is energized if there was a command error, timeout, or connection failure. The value of the command status register is 1 for success or another value for error conditions.
6. Repeat steps 2 to 4 to send additional command messages.

Supported Function Codes

Valid **Function Codes** are shown below. Functions 05, 06, 15, and 16 support broadcast messages. Functions 129, 130, 132, 133, 135, 136, 138, and 139 may be broadcast, but some Enron Modbus slave devices may not support broadcast messages. See the ***TeleBUS Modbus Compatible Protocols User Manual*** for details on each function code.

Function Code	Description	Purpose	Maximum Registers
01	read coil status	read digital output registers	2000
02	read input status	read digital input registers	2000
03	read holding register	read analog output registers	125 see note 1 below
04	read input register	read analog input registers	125 see note 1 below
05	write single coil	write digital output register	1
06	write single register	write analog output registers	1
15	write multiple coils	write digital output registers	880

16	write multiple registers	write analog output registers	60
128	read Enron Booleans	read Enron Boolean registers	2000
129	write Enron single Boolean	write Enron Boolean register	1
130	write Enron multiple Booleans	write Enron Boolean registers	880
131	read Enron short integers	read Enron short integer register	125
132	write Enron single short integer	write Enron short integer register	1
133	write Enron multiple short integers	write Enron short integer registers	60
134	read Enron long integers	read Enron long integer register	62 see note 2 below
135	write Enron single long integer	write Enron long integer register	1
136	write Enron multiple long integers	write Enron long integer registers	30
137	read Enron floats	read Enron floating-point registers	62
138	write Enron single float	write Enron floating-point register	1
139	write Enron multiple floats	write Enron floating-point registers	30

Notes

1. For SCADAPack controllers, the MSIP element can read a maximum of 123 registers with a single MSIP element.
2. SCADAPack controllers do not support Enron Modbus commands.
3. The slave controller address edit-box sets the station number of the controller. The valid range is 0 to 255 if standard addressing is used, and 0 to 65534 if extended addressing is used. Address 0 is reserved for broadcast messages, which are directed to the IP address specified in the slave IP address field only.

Slave Register Addresses

The slave register address specifies the first register where data will be read from or written to in the slave controller. The following table shows the valid slave register addresses for each command.

Description	Slave Register Address	Length
read coil status	00001 to 09999	2000
read input status	10001 to 19999	2000
read holding register	40001 to 49999	125
read input register	30001 to 39999	125
write single coil	00001 to 09999	1
write single register	40001 to 49999	1
write multiple coils	00001 to 09999	880
write multiple registers	40001 to 49999	60
read Enron Booleans	0 to 65535	2000
write Enron single Boolean	0 to 65535	1
write Enron multiple Booleans	0 to 65535	880
read Enron short integers	0 to 65535	1251
write Enron single short integer	0 to 65535	1
write Enron multiple short integers	0 to 65535	60
read Enron long integers	0 to 65535	622
write Enron single long integer	0 to 65535	1
write Enron multiple long integers	0 to 65535	30
read Enron floats	0 to 65535	62
write Enron single float	0 to 65535	1
write Enron multiple floats	0 to 65535	30

Command Status Codes

When an error occurs, an error status code is stored in the **command status register** in the timer control block. The status codes are shown in the Status Code table below.

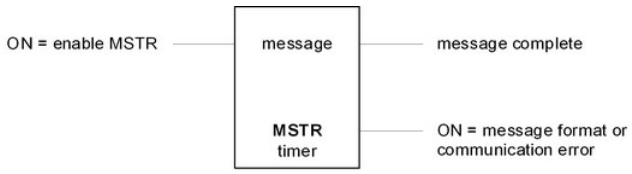
Status Code	Description
0	Valid command has been sent.
1	Response received (no error occurred).
2	No message was sent.
3	Invalid function code.
4	Invalid slave station address.
5	Invalid database address.
6	Invalid message length.
7	Serial port or protocol is invalid.
8	Connecting to slave IP address.
9	Connected to slave IP address.
10	Timeout while connecting to slave IP address.
11	TCP/IP error has occurred while sending message.
12	Time out has occurred; waiting for response. .
13	Slave has closed connection; incorrect response; or, incorrect response length.
14	Disconnecting from slave IP address is in progress.
15	Connection to slave IP address is disconnected.
16	Invalid connection ID.
17	Invalid protocol type.
18	Invalid slave IP address.
19	Last message is still being processed
20	Master connection has been released.
21	Error while connecting to slave IP address.
22	No more connections are available.
24	Exception response: invalid function code.
25	Exception response: invalid address.
26	Exception response: invalid value.
27	Protocol is invalid or serial port queue is full.
28	Slave and master stations are equal; they must be different.
29	

	Exception response: slave device failure.
30	Exception response: slave device busy.

MSTR (Master Message)

The **MSTR** element exchanges data with another controller using either the Modbus or the DF1 communication protocols.

When the **enable MSTR** goes from OFF to ON the MSTR element will send one master message to the slave address. The **enable MSTR** input must go from OFF to ON to send another MSTR message. The **message complete** output will be energized if the MSTR receives a valid response from the slave station. The **message format or communication error** output is energized if an error occurs. See the status code table below for a list of possible errors.



Function Block Properties

Message Block

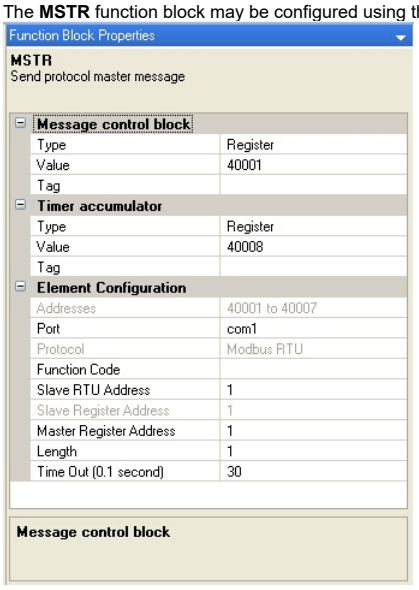
Type	Register
Value	Address of the first register (**message**) of 7 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller. The block of registers are defined for the function block as follows: **message** + 0 = serial communication port **message** + 1 = function code **message** + 2 = slave controller address **message** + 3 = slave register address **message** + 4 = master register address **message** + 5 = length **message** + 6 = time-out in 0.1s increments
Tag	The tag window displays the tag name for the message + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Timer Block Value

Type	Register
Value	Address of the **timer** 4xxxx (holding) register. This register contains the accumulated time (message + 6) for the time out.
Tag	The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration

The **MSTR** function block may be configured using the Element Configuration.



The **Address** window displays the register addresses used by the MSTR function. These addresses are entered in the Function Block Properties.

The **Port** specifies the controller serial port to use for the MSTR function. The menu displays valid serial ports for the controller type.

The **Protocol** is is not set in element configuration. The protocol used by a serial port is set in the serial port settings command.

Valid **Function Codes** are shown in the Modbus Supported Function Codes and DF1 Supported Function Codes tables below.

The **Slave RTU Address** edit-box sets the station number of the controller. For the Modbus ASCII and Modbus RTU protocols, the valid range is 0 to 255 if standard addressing is used, and 0 to 65534 if extended addressing is used. Address 0 is reserved for broadcast messages. For DF1 the address is in the range 0 to 254.

The **Slave Register Address** specifies the first register where data will be read from or written to in the slave controller. For Modbus protocol the Slave Register Address table show below displays the valid slave register addresses for each command. For DF1 this is the first DF1 physical 16-bit address where data will be read from or written to in the slave controller.

The **Master Register Address** specifies the first register where data will be written to or read from in this controller. The register type does not have to match the type of the slave register address. It is possible to store input registers from a slave into output registers on the master and vice-versa.

The **Length** parameter specifies how many registers are to be

transferred. For Modbus the maximum length for each function code is shown in the Supported Modbus Function Codes table below. Functions 05 and 06 always transfer 1 register, regardless of the value of length. For DF1 this parameter specifies how many 16-bit registers are to be transferred for functions 00, 01 and 08. For function codes 02 and 05, this field is labeled **Bit Mask**. **Bit Mask** selects the 16-bit bitmask specifying which bits in the **Master Register** to send. If a bit is set in **Bit Mask**, the corresponding bit in the register will be sent.

The **time-out** is in 0.1s increments. The error output is energized if the time-out period ends without a valid response to the message.

Notes:

To optimize serial port throughput, minimize the length of messages on communication ports that are implemented through the I/O bus. Examples of recommended uses for these communication ports are for local operator display terminals, and for programming and diagnostics using the Telepace program. this restriction applies to:

- SCADAPack LP com3
- SCADAPack 100 com3
- SCADAPack com3
- SCADAPack Light com3
- SCADAPack Plus com3 and com4
- SCADAPack 32 com3
- SCADAPack 4202 com3

Serial ports 3 and 4 operate in half duplex mode only. This is due to the fact that these serial ports use smaller buffers than serial ports 1 and 2 as they are serviced through the I/O bus. Due to the smaller buffer, data can only be stored in half duplex mode, ie. only the receive data or the transmit data, not both at the same time. As serial ports 3 and 4 are serviced through the I/O bus they are not updated as often as serial ports 1 and 2 and should be used for slower communication needs such as programming and operator interfaces.

Supported Modbus Function Codes

Functions 05, 06, 15, and 16 support broadcast messages. Functions 129, 130, 132, 133, 135, 136, 138, and 139 may be broadcast, but some Enron Modbus slave devices may not support broadcast messages.

Function Code	Description	Purpose	Maximum Registers
01	read coil status	read digital output registers	2000
02	read input status	read digital input registers	2000
03	read holding register	read analog output registers	125
04	read input register	read analog input registers	125
05	write single coil	write digital output register	1
06	write single register	write analog output registers	1
15	write multiple coils	write digital output registers	880
16	write multiple registers	write analog output registers	60
128	read Enron Boolean	read Enron Boolean registers	2000
129	write Enron single Boolean	write Enron Boolean register	1
130	write Enron multiple Booleans	write Enron Boolean registers	880
131	read Enron short integers	read Enron short integer register	125
132	write Enron single short integer	write Enron short integer register	1
133	write Enron multiple short integers	write Enron short integer registers	60
134	read Enron long integers	read Enron long integer register	62
135	write Enron single long integers	write Enron long integer register	1
136	write Enron multiple long integers	write Enron long integer registers	30
137	read Enron floats	read Enron floating-point registers	62
138	write Enron single float	write Enron floating-point register	1
139	write Enron multiple floats	write Enron floating-point registers	30

Slave Register Addresses

Description	Slave Register Address	Length
read coil status	00001 to 09999	2000
read input status	10001 to 19999	2000
read holding register	40001 to 49999	125
read input register	30001 to 39999	125
write single coil	00001 to 09999	1
write single register	40001 to 49999	1
write multiple coils	00001 to 09999	880
write multiple registers	40001 to 49999	60
read Enron Booleans	0 to 65535	2000
write Enron single Boolean	0 to 65535	1
write Enron multiple Booleans	0 to 65535	880
read Enron short integers	0 to 65535	125
write Enron single short integer	0 to 65535	1
write Enron multiple short integers	0 to 65535	60
read Enron long integers	0 to 65535	62
write Enron single long integer	0 to 65535	1
write Enron multiple long integers	0 to 65535	30
read Enron floats	0 to 65535	62
write Enron single float	0 to 65535	1
write Enron multiple floats	0 to 65535	30

Modbus Status Codes

The module Serial Port Protocol Status needs to be added to the register assignment to monitor the master command status register for the desired serial port. The

status codes are shown in the following table.

Status Code	Description
0	Message sent, waiting for response.
1	Response received (no error occurred).
3	Bad value in function code register.
4	Bad value in slave controller address register.
5	Bad value in slave register address register.
6	Bad value in length register.
7	Serial port or protocol is invalid.
12	response timeout.
24	Exception response: invalid function code.
25	Exception response: invalid address.
26	Exception response: invalid value.
27	Protocol is invalid or serial port queue is full.
28	Slave and master stations are equal; they must be different.
29	Exception response: slave device failure.
30	Exception response: slave device busy.

Supported DF1 Function Codes

Function Code	Description	Purpose	Maximum Registers
00	Protected Write	Writes words of data to limited areas of the database.	121
01	Unprotected Read	Reads words of data from any area of the database.	122
02	Protected Bit Write	Sets or resets individual bits within limited areas of the database.	1
05	Unprotected Bit Write	Sets or resets individual bits in an area of the database.	1
08	Unprotected Write	Writes words of data to any area of the database.	121

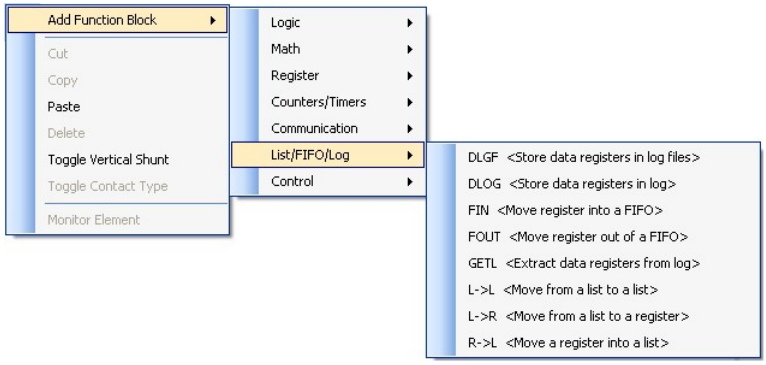
DF1 Status Codes

The module Serial Port Protocol Status needs to be added to the register assignment to monitor the master command status register for the desired serial port. The status codes are shown in the following table.

Status Code	Description
0	Message sent, waiting for response.
1	Response received (no error occurred).
3	Bad value in function code register.
4	Bad value in slave controller address register.
5	Bad value in slave register address register.
6	Bad value in length register.
7	Serial port or protocol is invalid.
8	Response received did not match command sent.
9	Specified protocol is not supported by this controller.
16	Slave error: illegal command or format.
80	Slave error: addressing problem or memory protect rungs.
Other	Error codes from other DF1 compatible controllers. Consult documentation for specific controller.

LIST / FIFO / Logging Functions

The List / FIFO / Logging functions available in Telepace Studio are shown in the image below. This image is opened in Telepace Studio by right clicking on the ladder canvas and selecting the Add Function Block command. Depending on the available space on the ladder canvas some functions may not be displayed as they would not fit in the canvas at the location selected.



Hover your mouse over the icons below for a description of the function. Click the icon to go to the function description.

DLGF	DLOG	FIN	FOUT
GETL	L->L	L->R	R->L

DLGF (Data Log to File)

The **DLGF** element records entries to a data log file. When a low to high transition occurs on the **enable DLGF input**, the data log identified by the **config structure** is created and initialized. If the data log currently exists and has a different configuration then an error will be generated. If an error is generated (see Status Registers

section below) then a different data log name must be logged to, or the first data log must be deleted. When a high to low transition occurs on the enable DLGF input the datalog previously created has its configuration deleted.

While the **enable DLGF** is ON and a low to high transition occurs on the log data input, an entry is recorded in the data log.

When the **suspend logging input** is high data will be internally buffered, but not written to file until the input goes low. Any data that is in the buffer when the suspend logging goes high will be written to file. This is useful when logging to an external mass storage device. Logging can be suspended to allow the storage device to be exchanged for another.

The DLGF element outputs are powered only when the create log config input is powered (ON).

The **copies state of input 1** output is a copy of the create/delete log config input.

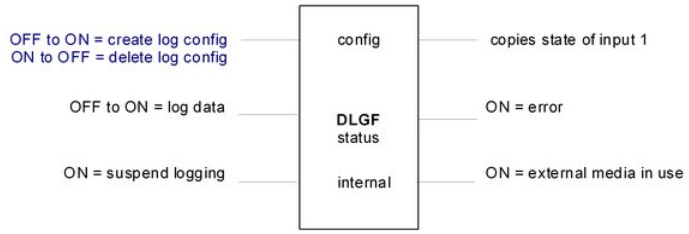
The **error** output is ON if there was a creation, configuration or data logging error. See the Status Registers section below for error codes.

The external media in use output is enabled (ON) when the data log performs write operations to external USB media. This happens under the following conditions:

- When the logging mode is selected for mass storage drive.
- When the logging mode is selected for auto transfer and the DLGF element is in the process of transferring files to the mass storage drive.

When the external media in use output is OFF the removal of the external mass storage drive is safe from the perspective of this particular DLGF element. As multiple DLGF elements can be active all DLGF element and any running C/C++ applications need to be considered when removing the external media.

The external media in use output is always OFF for DLGF element configured to use the internal drive without auto transfer. It is also OFF when the DLGF element is configured to use mass storage drive and the suspend logging input is high. The output goes ON for DLGF elements configured to use auto transfer and a new mass storage device (USB memory stick) is connected. When the output turns OFF this indicates that the write process to that media is finished.



There are limitations on the amount of file system space available for data log to file configurations. See the Determining Data Log to File Space Requirement section for information on calculating how much space your data log(s) require.

The **DLGF** element is supported on the SCADAPack 314, SCADAPack 33x, SCADAPack 350 controllers and the SCADAPack 4203 only.
The SCADAPack 314 does not support logging to a mass storage device.

Function Block Properties

Configuration Block

Type Register

Value Address of the first register (**config**) of 39 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller.

The block of registers are defined for the function block as follows:

config + 0 = Number of data fields
config + 1 = Field #1 register address. This is the address of a Modbus register, or the address of the first Modbus register of a small block of consecutive registers for floating point, signed double and unsigned double data. Valid values are any register in the range 40001 to 49999. (same for all fields)
config + 2 = Field #1 data type (See Data Types below)
config + 3 = Field #2 register address
config + 4 = Field #2 data type
config + 5 = Field #3 register address
config + 6 = Field #3 data type
config + 7 = Field #4 register address
config + 8 = Field #4 data type
config + 9 = Field #5 register address
config + 10 = Field #5 data type
config + 11 = Field #6 register address
config + 12 = Field #6 data type
config + 13 = Field #7 register address
config + 14 = Field #7 data type
config + 15 = Field #8 register address
config + 16 = Field #8 data type
config + 17 = Logging Mode
0 = internal drive
1 = internal drive, auto copy
2 = internal drive, auto move
3 = mass storage drive
config + 18 = Full File Handling
0 = overwrite oldest log file
1 = stop logging
config + 19 = Number of log files, minimum 1
config + 20 = Number of records per file. Minimum 600
config + 21 to +22 = Security Token; note that the input through Telepace only allows 8 hex characters, e.g. 1A2B3C4D; the security token is only considered for auto transfer logs which are not supported on controllers without USB host port.

	config + 23 to +38 Log Name, path qualified
Tag	The tag window displays the tag name for the config + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Status Block

Type	Register
Value	Address of the status register (see below for status code descriptions) +0 = log status code +1 = auto transfer status code +2 = media status code
Tag	The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Internal

Type	Constant
Value	Identifies the internal log. This value is used internally and must not be modified. +0, +1 = Internal Data
Tag	The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Field Data Types

See the section Register Types for information on register data types.

Data Type	Description
0	Unsigned
1	Signed
2	Unsigned Double
3	Signed Double
4	Floating point
5	Days and hundredths of seconds in two 32-bit unsigned integers. The first two registers are an Unsigned Double containing the number of complete days since 01/01/97. The last two registers are Unsigned Double containing the number of hundredths of a second since the start of the current day.

Log Status Code

The status register stores the result of a log attempt. It can have the following values. If the status register is not equal to 10 then the error output is turned on.

The **log status** codes are:

Status Codes	Description
10	The configuration is valid and data can be logged.
11	A different configuration already exists for the log.
12	The log ID is invalid or has not been created.
13	The configuration was not valid and was rejected.
14	There is not enough available memory to create a log of the requested size.
15	The number of data fields was invalid.
16	Failed to write data.
17	Invalid configuration.

Auto Transfer Status Code

The auto transfer status register indicates the status of an auto transfer operation. A status code unequal 0 signals the user that no more log files have to be transferred. This includes the case that no files can be transferred due to a missing or wrong security token. Note that the status could indicate that the auto transfer is in progress with the **external media in use** output set to OFF. This is the case if not all log files could be transferred due to full media. DLGF blocks not configured to use auto transfer always show the value 4. The same value is displayed when an auto transfer was never triggered for a configured log.

Status Code	Description
0	Auto transfer is not done or in progress.
1	Auto transfer is done; all files (at least one) were transferred.
2	Auto transfer is done; no log file had to be transferred.
3	Auto transfer is done because no valid security token could be found.
4	Auto transfer not configured or not started.

Media Transfer Status Code

The **media transfer** status codes are:

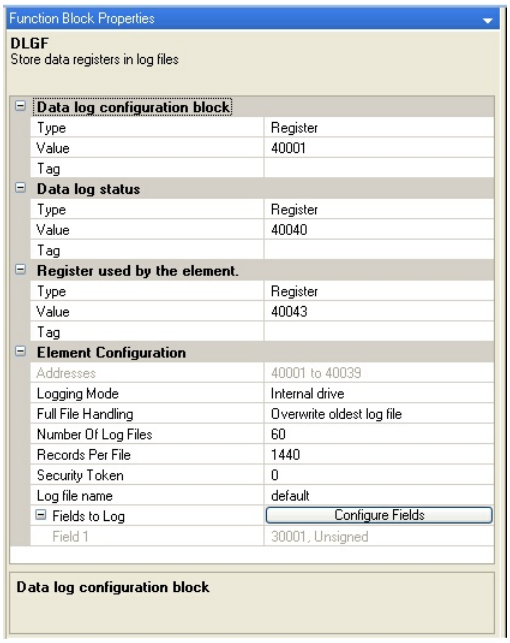
Status Code	Description
0	Media is present.
1	No external media present.
2	External media is full.
3	Internal media is full.
4	External and internal media are full.

Element Configuration

The **DLGF** element is configured using the DLGF Element Configuration view. Highlight the element by moving the cursor over the element and then use the Element Configuration view to modify the DLGF element.

If the controller is initialized, using the Initialize command in the Controller menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.

When Element Configuration is selected for the DLGF function the following dialog is displayed.



Addresses

This is the register address range that was entered for the Data log configuration block entry in the DLGF Element Properties. These registers are described in the Element Properties section above.

Logging Mode

Data logs may be logged to the internal drive or to an external mass storage drive. When stored to the internal drive data logs can be selected for automatic copying or moving to the mass storage drive when the mass storage drive is inserted into the USB host port on the controller. To allow a degree of confidentiality the data log can be configured to only transfer data if a specific security token is present on the inserted mass storage device.

Depending on the controller type the following options for logging mode are available.

Internal drive In this case multiple data logs are being saved to the internal file system. The data logs are extracted through the file transfer functions on demand by SCADALog.

Internal drive, auto copy In this case multiple data logs are being saved to the internal file system. The auto copy feature is enabled with no copy restrictions. When a USB mass storage device is inserted the data logs are copied to the mass storage device. The mass storage device is removed from the controller and then connected to the PC running SCADALog. SCADALog then imports the log files from the mass storage device.

Internal drive, auto move In this case multiple data logs are being performed to the internal file system. The auto move feature is enabled with no copy restrictions. When a USB mass storage device is inserted the data logs are moved to the mass storage device. The mass storage device is removed from the controller and then connected to the PC running SCADALog. SCADALog then imports the log files from the mass storage device.

Mass storage drive In this case there is a mass storage device already connected to the controller. Multiple data logs are configured to write the data to the mass storage device. At some point in time the mass storage device is removed. The data continues to be logged, but is stored to internal memory. When a mass storage device is inserted the buffered data is copied to the mass storage device.

In some cases the USB mass storage device may not be detected when it is inserted. It is recommended that the user provide a mechanism to manually start the auto copy move or auto move features. This may be done using the Controller Power Mode register assignment. Write a value of 1 to the start register +2 (USB Host Power State) to disable the USB host port and then writing a 0 to to the start register +2 (USB Host Power State) to enable the USB host port. The auto transfer will start when the USB port is re-enabled.

The transfer maintains any folder structure that exists on the internal file system. In other words the files will remain in the same relative position to the root of the drive. Folders are created on the mass storage device if needed.

Logs that the token grants access to are copied or moved to the mass storage device. Each data log is suspended before its log files are copied or moved. The data log is resumed if no more log files can be copied or moved. This is the case if there are no more log files to be copied or moved or if the mass storage medium is full.

If the log is configured to move the files they are deleted after the move is successful. In the event of a failure part way through the process, the files will remain on the internal file system and the operation can be attempted again later.

Full File Handling

The data is buffered in RAM before being written to file. The data buffer will hold 600 records. This is done to optimize data writes to file and reduce the total CPU load. The data is buffered in non-volatile memory so that a power cycle does not result in the loss of data.

There are two methods for handling full files:

- Logging can be configured to stop logging when the maximum number of records and maximum number of log files has been reached.
- Logging can be configured to overwrite the oldest log file, with a new file, when the maximum number of log files is reached.

In the event that no disk space remains the data will be buffered in RAM memory until that buffer fills as well, at which point new records will be discarded.

The data is written to the active log file, from buffered RAM, at 1 minute intervals.

Number of Log Files

The number of log files can parameter can be any value between 1 and 65535.

Records per File

The number of Records per file is any value between 600 and 65535.

Security Token

The security token is a 32 bit value containing 8 hex characters. This token provides for a method of authenticating the mass storage device as a trusted and authorized device. This method of authentication does meet the criteria for 2 factor security because it requires something you know (the security token), and something you have (physical access).

- If a data log is configured to have a security token value of 0 then the security is disabled on this log.

Security tokens are stored in an XML script file. The tokens are ASCII representations of a 32-bit value, i.e. if the security token value was 0x00579ACE then it would be represented as 00579ACE in the token file.

The file must be named controller.xml and reside in the root directory of the mass storage device.

Security Token File Format

```
<controller>
<securityTokens>
        <allControllers>
            <token>token1</token>
            <token>token2</token>
            …
            <token>tokenN</token>
        </allControllers>
    </securityTokens>
</controller>
```

There is no limit to the number of items that can be listed in the securityTokens section. The file may contain more information than that listed above, but any additional information will not be used. Any token listed must be represented as a hex value without any prefix or suffix, for example:

<token>A1B2C3D4</token>

The security tokens are not keyed to the mass storage device that they reside on. This improves the usability of the security token; however it does reduce the security offered by the token since the tokens could be freely copied from one device to another.

When a security token is read that matches that of a data log then the data log data will be transferred if that the data log is configured for auto transfer mode.

If a data log is configured to have a security token value of 0x00000000 then the security is disabled on this log.

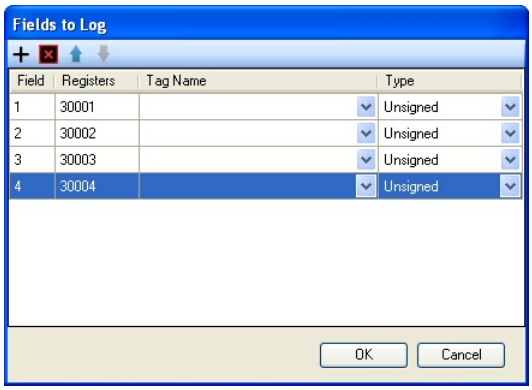
Because security tokens are checked on external USB media for auto transfer logs, the definition of the token during log creation is meaningless for pure internal or mass storage logging modes as well as for any logs on controllers which do not support an USB host.

Log File Name

Log file names are created using the Log file name entry in the DLGF Element Configuration dialog. The log file name can be up to 32 characters long and can contain folder information as well as the log name.

Configure Fields

When the Configure Fields button is clicked the Fields to Log dialog is opened. Fields may be added, deleted or moved using the icons in the tool bar at the top of the dialog.



The record configuration grid allows for entries of up to 8 fields for each record. An empty field is added below the last field each time a new field is edited, until field number 8 is entered.

The Field entries contain information about each item in the log. The fields are numbered,1 through 8, as they are added to the record. The Registers column contains the Modbus registers used in for the field, the Tag column contains the tag name for the Modbus Register and the Type column contains the Modbus register type..

The **Registers** entry contains the Modbus register address for the record if the Edit button was selected. This entry will be zero if the Add button was selected. Valid values are any register in the range 00001 to 49999. To enter a register:

- Double click anywhere is the registers window for the field being entered. If a tag name exists for the register address it will be displayed in the Tag Name window.

The **Tag Name** entry contains the tag name for the selected register address if one exists. The drop down list will display all tag names for valid address. Selecting a tag name from the list will cause the associated register address to be displayed in the Address entry. Use the drop down menu, at the right of the window, to select from the existing tag names.

The **Data Type** selection allows the selection of the Modbus register data type (see Register Types).

- Unsigned
- Signed
- Unsigned Double
- Signed Double
- Floating point
- Time and Date

Days and hundredths of seconds in two 32 bit unsigned integers

Press **OK** to validate the data and close the dialog.

Press **Cancel** to discard the changes made in the dialog.

DLGF (Data Log to File) Overview

The SCADAPack 3xx controllers support data logging to the internal file system and data logging to a mass storage device connected through the USB host port. Each data log can be configured independently to write data to either the internal file system or a mass storage drive connected through the USB host interface.

Each data log creates a configurable number of data log files. Data Log records are buffered to optimize file-writing dynamics. The log specific buffer is flushed to the active log file once per minute.

Each log can be suspended and resumed. In a suspended state data log records can still be written to the log, but the buffer won't be flushed to file. The state is useful when external medium has to be exchanged.

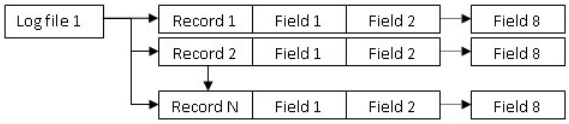
Data Log Storage Medium

Data log files are stored either on the internal file system or an external drive in form of an USB memory stick or an USB hard drive. If more than one USB drive is connected through a hub, only the first drive connected is the active USB log drive.

Logging to File Mechanism

The data is buffered in RAM before being written to file. The buffer holds a maximum of 600 records. This is done to optimize data writes to file and reduce the total CPU load. The data is buffered in non-volatile memory so that a power cycle does not result in the loss of data.

The structure for a single log file is shown below.



A single log file is made up of a configured number of records with each record containing a maximum of 8 data fields. Records are added to the buffered data in RAM each time the log data input for the DLGF element transitions from OFF to ON.

Once per minute the records that have been added to the buffered data in RAM are added to the log file in the internal drive or the mass storage drive. Based on the number of records configured and the logging interval eventually the log file will be full.

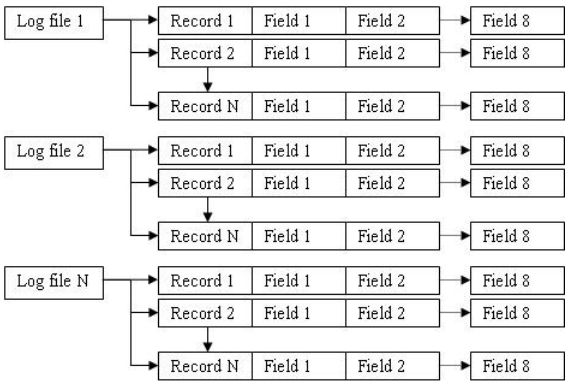
There are two methods for handling the full file:

- Logging can be configured to stop when the maximum number of records and maximum number of log files has been reached. In this case there is a single log file configured and logging will stop when all records are filled.
- Logging can be configured to overwrite the oldest log file, with a new file, when the maximum number of log files is reached. In this case there is a single log file configured. The log file will be deleted and a new log file created when all records are filled.

The way full files are handled limits the effectiveness of a single log file. The file system on the internal drive of the SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADAPack 4203 supports multiple data log files.

- The number of log files that can be configured for a each instance of the DLGF element is a maximum of 65535.
- Each log file contains a minimum of 600 records and is configurable for maximum of 65535 individual records. Each record contains a maximum of 8 configurable data fields.

The figure below shows the log file, record and field structure for data logging using multiple log files.



Records are added to the buffered data in RAM each time the log data input for the DLGF element transitions from OFF to ON.

After 1 minute Log file 1 is created and the records that have been added to the buffered data in RAM are added to Log file 1 in the internal drive or the mass storage drive.

This process repeats until the configured number of records per file, Record N, for Log file 1 is reached.

Again records are added to the buffered data in RAM each time the log data input for the DLGF element transitions from OFF to ON.

After 1 minute Log file 2 is created and the records that have been added to the buffered data in RAM are added to Log file 2 in the internal drive or the mass storage drive. Log file 2 is saved in the internal drive or the mass storage drive.

This process repeats until the configured number of records per file, Record N, for Log file 2 is reached and continues to repeat until the configured number of records, Record N, for Log file N is reached.

Now the entire data log is filled. After another 600 records the buffer will be full and no further buffering will occur. The log status register will have the number 17 indicating the status failed to write data.

If the full file handling is configured to Stop logging when the maximum number of records and maximum number of log files has been reached and the buffer is filled then data logging fails..

- If Logging mode is set for **Internal drive** then the log file is saved on the internal drive.
- If the logging mode is set for **Internal drive, auto copy** then once a mass storage drive is inserted in the USB host port the data files are copied to the mass storage device.
- If the logging mode is set for **Internal drive, auto move** then once a mass storage drive is inserted in the USB host port the data files are moved to the mass storage device.
- If the logging mode is set for **Mass storage drive** then the log file is saved on the mass storage drive.

If the full file handling is configured to Overwrite oldest log file when the maximum number of records and maximum number of log files has been reached and the buffer is filled then data logging continues by overwriting the oldest log file. In this case Log file 1 is over written and the process continues repeatedly.

File Formats

Data Log to file requires 2 types of files.

- The first is the directory file which is stored for each log in the specified path on the target drive.
- The second type of file is the actual data files. There may, and likely will be, multiple data files for each log that is created. These data files are in binary format. The SCADALog Data Converter application is used to convert the binary files to a CSV formatted file.

Directory File

The directory file contains the list of all datalog filenames, excluding path information, for the specified log. Each file name is on a new line separated by the log file tag. This file is used to determine the actual log file names so that they can be read by SCADALog, or another process.

The directory file is stored in XML format on the target drive in the same directory as the log files. Its name is controllerID_logname.xml. The path is excluded from controllerID_logname. The attribute name in the logDirectory tag contains the full log name.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<logDirectory logName="datalogName">
<!--List of data log files-->
    <logFile>A288389_ datalogName _20080324_082311.nlg</logFile>
    <logFile>A288389_ datalogName _20080324_083011.nlg</logFile>
    <logFile>A288389_ datalogName _20080324_083611.nlg</logFile>
    <logFile>A288389_ datalogName _20080324_084311.nlg</logFile>
    <logFile>A288389_ datalogName _20080324_085011.nlg</logFile>
</logDirectory>
```

Log File Names

Log file names are created using the Log file name entry in the DLGF Element Configuration dialog. When the files are saved to the internal drive or the mass storage device additional information is added to the file name. Data logs files are named according to the convention:

Annnnnn_*logfile*_YYYYMMDD_HHMMSS.nlg

Where:

- **Annnnnn** is the controller ID
- ***logfile*** is the path qualified log file name as created in the DLGF Element Configuration..
- **YYYY** is the 4 digit year the file was created
- **MM** is the 2 digit month the file was created
- **DD** is the 2 digit day of the month the file was created
- **HH** is the 2 digit hour (24 hour clock) the file was created
- **MM** is the 2 digit minute the file was created
- **SS** is the 2 digit second the file was created

Note that the file name is path qualified to permit similar data log names on the same controller. i.e. directory1\log and directory2\log are both unique permitted names. The drive is excluded from the file name.

Determining Data Log to File Space Requirements

An important part of data logging applications is determining the memory space required. There is a limit of **6,000 kB** free space available in the SCADAPack 314, SCADAPack 33x and SCADAPack 350 controllers and the SCADAPack 4203.

The following steps will assist in determining the space requirements for your Data Log application:

1. Determine the number and type of parameters for each DLGF record to get the record size.

    There are a maximum of 8 fields allowed for each record. The data types vary in the size of memory required by a field:

- Unsigned Integer type uses 2 Bytes of memory.
- Signed Integer type uses 2 Bytes of memory.
- Unsigned Double type uses 4 Bytes of memory.
- Signed Double type uses 4 Bytes of memory.
- Floating Point type uses 4 Bytes of memory.
- Time and Date type uses 8 Bytes of memory.

Record Size = Total number of Bytes for the field data plus an internal overhead of 6 bytes.

2. Determine the number of records per data log file.

Number of Records is a value between 600 and 65535.

3. Determine the number of log files.

Number of Log Files is a number between 1 and 65535.

4. Internal file header overhead is **300** bytes per data log file.

5. Number of bytes in 1kB is **1024.**

6. Determine the memory required for the data log.

Memory Required For DLGF = (**Record Size** * **Number of Records** + **300**) * **Number of Files** / **1024**.

7. Repeat steps 1 through 6 for each DLGF function in your application to determine the total memory required for data logging.

Total Memory Required = **DLGF 1** memory + **DLGF 2** memory +.....+ **DLGF n** memory.

Note: The combination of Datalog Files and Ladder Logic cannot exceed 6000kB.

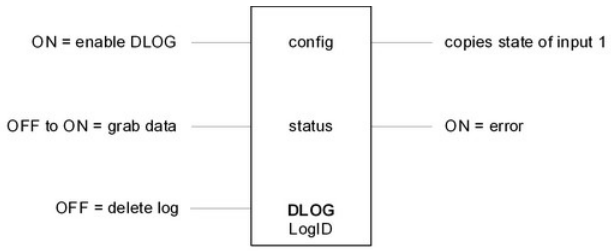
DLOG (Data Logger)

The **DLOG** element records entries in a data log. When a low to high transition occurs on the **enable DLOG** input, the data log identified by **LogID** is created and initialized. If the data log exists and has a different configuration then an error will be generated. If an error is generated (see status codes below) then a different **LogID** must be used or the log must be deleted using the **delete log** input.

While the **enable DLOG** is ON and a low to high transition occurs on the **grab data** input, an entry is recorded in the data log identified by **LogID**.

When the **delete log** input is low, all records in the data log are deleted.

The top output is a copy of the top input. The **error** output is ON if there was a creation, configuration or data logging error. The outputs are powered only when the enable input is powered.



For the changes to an existing DLOG configuration to take effect, follow these steps:

1. Read the data from the log, if desired.
2. Disable the DLOG.
3. Delete the log.
4. Write the new configuration.
5. Enable the log.

Function Block Properties

Configuration Block

Type	Register
Value	Address of the first register (**config**) of 18 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller. The block of registers are defined for the function block as follows: **config** + 0 = Maximum number of records in the log. **config** + 1 = number of data fields. **config** + 2 = Field #1 register address **config** + 3 = Field #1 data type (see data types below) **config** + 4 = Field #2 register address **config** + 5 = Field #2 data type **config** + 6 = Field #3 register address **config** + 7 = Field #3 data type **config** + 8 = Field #4 register address **config** + 9 = Field #4 data type **config** + 10 = Field #5 register address **config** + 11 = Field #5 data type **config** + 12 = Field #6 register address **config** + 13 = Field #6 data type

config + 14 = Field #7 register address
config + 15 = Field #7 data type
config + 16 = Field #8 register address
config + 17 = Field #8 data type

Tag The tag window displays the tag name for the config + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

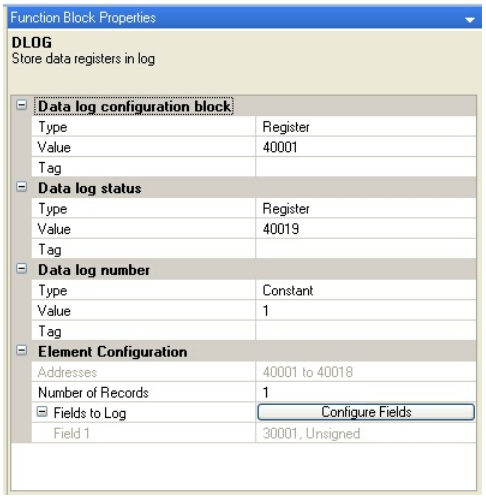
Status Block

Type Register
Value Address of the status register. See the Status Codes section below.
Tag The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Log ID Block

Type Constant
Value Identifies the internal log.Valid values are 1 to 15.
Tag The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration



The **Register Addresses** are determined from the ladder element. They cannot be modified in this dialog.

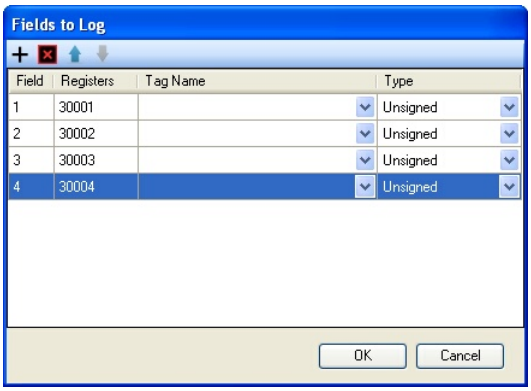
The **Number of Records** selects the number of records in the log. Each record contains the data for all the fields in the log. The number of records available will depend on the type of controller used.

- For SCADAPack 100: 256K controllers the maximum number of records is 190 records of 8 unsigned integer fields.
- For SCADAPack LP and SCADAPack 100: 1024K controllers the maximum number of records is approximately 22970 records of 8 unsigned integer fields.
- For SCADAPack and Micro16 controller the number of records depends on the available memory in socket U10 in the controller. For example a 128K RAM in U10 will allow approximately 190 records of 8 unsigned integer fields and a 512K RAM in U10 will allow approximately 22970 records of 8 unsigned integer fields.
- For SCADAPack 32 controllers the maximum number of records is 58,352 records of 8 unsigned integer fields for a total of 466816 words. The maximum number of records will be reduced if DNP protocol, Realflo or custom C++ applications are used in the controller.
- For SCADAPack 314, SCADAPack 330, SCADAPack 334 and SCADAPack 350 controllers the maximum number of records is 56,743 records of 8 unsigned integer fields for a total of 453947 words. The maximum number of records will be reduced if DNP protocol, Realflo or custom C++ applications are used in the controller.

The **Fields to Log** displays the current fields for the DLOG function. The list of fields may be edited by clicking the Configure Fields button. When this button is clicked the Fields to Log dialog is opened.

Configure Fields

When the Configure Fields button is clicked the Fields to Log dialog is opened. Fields may be added, deleted or moved using the icons in the tool bar at the top of the dialog.



The record configuration grid allows for entries of up to 8 fields for each record. An empty field is added below the last field each time a new field is edited, until field number 8 is entered.

The Field entries contain information about each item in the log. The fields are numbered,1 through 8, as they are added to the record. The Registers column contains the Modbus registers used in for the field, the Tag column contains the tag name for the Modbus Register and the Type column contains the Modbus register type..

The **Registers** entry contains the Modbus register address for the record if the Edit button was selected. This entry will be zero if the Add button was selected. Valid values are any register in the range 00001 to 49999. To enter a register:

- Double click anywhere is the registers window for the field being entered. If a tag name exists for the register address it will be displayed in the Tag Name window.

The **Tag Name** entry contains the tag name for the selected register address if one exists. The drop down list will display all tag names for valid address. Selecting a tag

name from the list will cause the associated register address to be displayed in the Address entry. Use the drop down menu, at the right of the window, to select from the existing tag names.

The **Data Type** selection allows the selection of the Modbus register data type (see Register Types).

- Unsigned
- Signed
- Unsigned Double
- Signed Double
- Floating point
- Time and Date

Days and hundredths of seconds in two 32 bit unsigned integers

Press **OK** to validate the data and close the dialog.

Press **Cancel** to discard the changes made in the dialog.

Field Data Types

See the section Register Types for information on register data types.

Data Type	Description
0	Unsigned
1	Signed
2	Unsigned Double
3	Signed Double
4	Floating point
5	Days and hundredths of seconds in two 32-bit unsigned integers. The first two registers are an Unsigned Double containing the number of complete days since 01/01/97. The last two registers are Unsigned Double containing the number of hundredths of a second since the start of the current day.

Status Register

The status register stores the result of a log attempt. It can have the following values. If the status register is not equal to 10 then the error output is turned on.

The **log status** codes are:

Status Codes	Description
10	The configuration is valid and data can be logged.
11	A different configuration already exists for the log.
12	The log ID is invalid or has not been created.
13	The configuration was not valid and was rejected.
14	There is not enough available memory to create a log of the requested size.
15	The number of data fields was invalid.
16	The log was successfully deleted. No log configuration exists.
17	Undefined status. This should never occur.

Related Elements

GETL

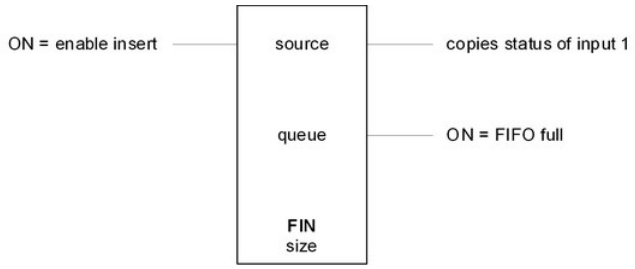
FIN (FIFO Queue Insert)

The **FIN** element inserts the contents of the source register into the First In, First Out (FIFO) queue. Values in the FIFO queue are pushed down until the queue is full.

When the **enable insert** input is ON and the **FIFO full** is not ON, the source register is inserted into the queue.

The index is incremented by one after each insert until the index value equals the size. When the index equals the size and **enable insert** is ON then **FIFO full** turns ON and no further insertions can take place.

The **FIN** element is used with the FIFO Queue Remove element to enable full control of the FIFO queue.



Function Block Properties

Index and Source Queue (queue)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination

Type	Register

Value For Register type any valid 4xxxx (holding register). The address of the index register and the queue registers.
For 4xxxx registers the index is stored in register (4xxxx). The FIFO queue is located in register (4xxxx+1) to register (4xxxx + size).
This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Number of Registers in Queue (Size)

Type Constant

Value For Constant type any number in range 0 to 9998. This is the number of 16-bit registers in the queue.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Elements

FOUT

Example

See the example for the FOUT (FIFO Queue Remove).

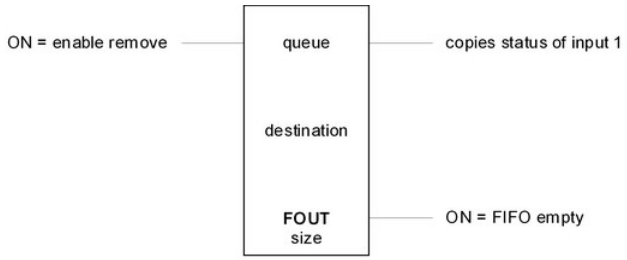
FOUT (FIFO Queue Remove)

The **FOUT** element removes a value from the first-in-first-out (FIFO) queue and transfers it into the destination register. Values in the queue are moved up, until the queue is empty.

When the **enable remove** input is ON and the **FIFO empty** is not ON, the first queue register is transferred into the destination register.

The index is decremented by one after each transfer until the index value equals zero. When the index equals zero and the **enable remove** input is ON the **FIFO empty** turns ON and no further removals can take place.

The FIFO Queue Remove element is used with the FIFO Queue Insert element to enable full control of the FIFO Queue. See FIFO Queue example.



Function Block Properties

Index and Source Queue (queue)

Type Register

Value For Register type any valid 4xxxx (holding register).

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination

Type Register

Value For Register type any valid 0xxxx (coil register) or 4xxxx (holding register). The address of the index register and the queue registers.
For 4xxxx registers the index is stored in register (4xxxx). The FIFO queue is located in register (4xxxx+1) to register (4xxxx + size).
For 0xxxx registers the FIFO queue is located in register (0xxxx+1) to register (0xxxx + size).
This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Number of Registers in Queue (Size)

Type Constant

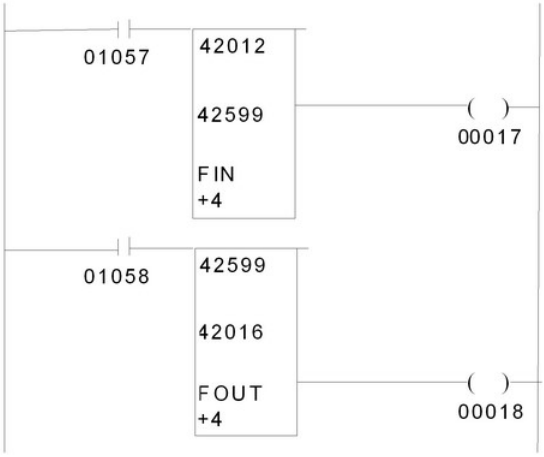
Value For Constant type any number in range 0 to 9998. This is the number of 16-bit registers in the queue.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Elements

FIN

Example



The following discussion describes the operation of a FIN element. The queue starts out empty with the FIFO index register having a value of zero.

Source Register	Source Register Data	FIFO Queue Registers
		FIFO Index --> 42599
42012	Xxxx	FIFO register 0 -->42600
		FIFO register 1 -->42601
		FIFO register 2 -->42602
		FIFO register 3 -->42603

When FIFO queue index register 42599 has a value of 0 and the FIN element is enabled, by closing contact 1057, the contents of source register 42012 is inserted into FIFO Queue register 42600 and the index is incremented to 1.

When FIFO queue index register 42599 has a value of 1 and the FIN element is enabled, by closing contact 1057, the contents of queue register 42600 is transferred to queue register 42601. The contents of source register 42012 are inserted into FIFO queue register 42600, and the index is incremented to 2.

When FIFO queue index register 42599 has a value of 2 and the FIN element is enabled, by closing contact 1057, the contents of queue register 42601 is transferred to queue register 42602. The contents of queue register 42600 is transferred to queue register 42601, the contents of source register 42012 is inserted into FIFO queue register 42600 and the index is incremented to 3.

When FIFO queue index register 42599 has a value of 3 and the FIN element is enabled, by closing contact 1057, the contents of queue register 42602 is transferred to queue register 42603. The contents of queue register 42601 is transferred to queue register 42602, the contents of queue register 42600 is transferred to queue register 42601, the contents of source register 42012 is inserted into FIFO queue register 42600 and the index is incremented to 4.

The queue is now full and no further insertions are allowed. Coil 00017 will turn ON. The FOUT Queue Remove element must be used to remove values from the queue.

The following discussion describes the operation of a FOUT element. The queue starts out full with the FIFO index register having a value of three.

	FIFO Queue Registers	FIFO Queue Register Data	Destination Register
FIFO Index -->	42599		
FIFO register 0 -->	42600		
FIFO register 1 -->	42601		
FIFO register 2 -->	42602		
FIFO register 3 -->	42603	xxxx	42016

When FIFO queue index register 42599 has a value of 4 and the FOUT element is enabled, by closing contact 1058, the contents of queue register 42603, pointed to by the index is transferred to the FIFO destination register 42016 and the index is decremented by one.

When FIFO queue index register 42599 has a value of 3 and the FOUT element is enabled, by closing contact 1058, the contents of queue register 42602 is transferred to the FIFO destination register 42016 and the index is decremented by one.

When FIFO queue index register 42599 has a value of 2 and the FOUT element is enabled the contents of queue register 42601 is transferred to the FIFO destination register 42016 and the index register is decremented by one.

When FIFO queue index register 42599 has a value of 1 and the FOUT element is enabled, by closing contact 1058, the contents of queue register 42600 is transferred to the FIFO destination register 42016 and the index is decremented to 0.

The queue is now empty and no further removals are allowed. Coil 00018 will be now be energized. The FIN Queue Insert element must be used to insert values into the queue.

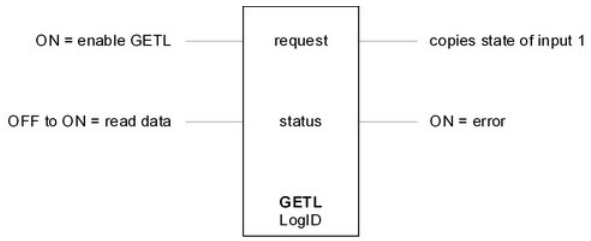
GETL (Data Logger Extractor)

The **GETL** element extracts one record from a data log and writes it into holding registers.

When a low to high transition occurs on the **enable GETL** input, the error code and sequence number status registers are updated and the data length is cleared.

When the **enable GETL** input is ON and a low to high transition occurs on the **read data** input, the status registers are updated and the data for the requested sequence number is copied to the data from log holding registers.

The **error** output is ON if the log is not configured or if there was a data reading error, see table below. The outputs are powered only when the enable input is powered.



Function Block Properties

Data Log Request Block (Request)

Type	Register
Value	For Register type any two valid sequential 4xxxx (holding register) registers. See Register Types for further information. Unsigned long sequence number requested from the log. +0 = least significant word +1 = most significant word
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Data Log Status and Data (Status)

Type	Register
Value	For Register type any valid 4xxxx (holding register). The address of the index register and the queue registers. A maximum of 35 registers are used for the status and log data. **Status** + 0 = Error Code, see below for error code list. **Status** + 1 and 20 = Sequence number. **Status** + 3 = Length of data. **Status** + 4 to 35 = Data from log.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Data Log Number (LogID)

Type	Constant
Value	For Constant type any number in range 1 to 16. This is the Data Log number, a maximum of 15 data logs may be created.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Status Register

The status register stores the result of an enable log attempt and a log data attempt. It can have the following values. If the status register is does not contain the code 20 then the error output is turned on. The **status** error codes are:

Error Code	Description
20	The configuration is valid and data can be retrieved.
21	The log ID is invalid.
22	The log is not configured.
23	The requested sequence number was not in the valid range.

Notes

To get the oldest sequence number in the log toggle the **enable GETL** input, the error code and sequence number status registers are updated and the data length is cleared. To clear the error 23 the **enable GETL** input must be turned off.

Related Elements

DLOG

L->L (List-to-List Transfer)

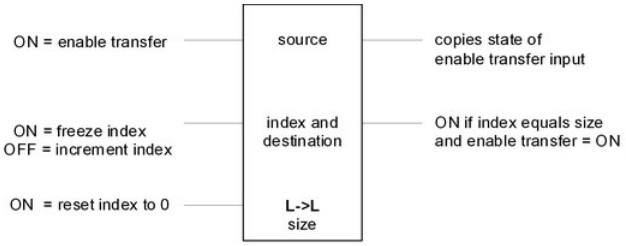
The **L–>L** element transfers the contents of one register from the source list of registers into one register from the destination list of registers, at the position of the index. The index register points to the same position in both the source and the destination list of registers.

When the **enable transfer** input is ON, and the **index equals size** output is not ON, the source list register is transferred to the destination list register at the position of the index in both lists.

The **freeze/increment index** input allows control of the index. If this input is ON the index is not incremented. If the input is OFF the index is incremented by one after each transfer.

When the **reset index** input is ON the index is reset to zero.

The index is incremented by one on each transfer until the index equals size. When the **index equals size** output is ON no further increments in index value occur until the index is reset.



Function Block Properties

Source List (Source)

Type	Register
Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Index and Destination List (Index and Destination)

Type	Register
Value	For Register type any valid 4xxxx (holding register). The address of the index register and the destination register list. The index is stored in register [4xxxx]. The data is stored in register [4xxxx+1] to register [4xxxx+size]
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

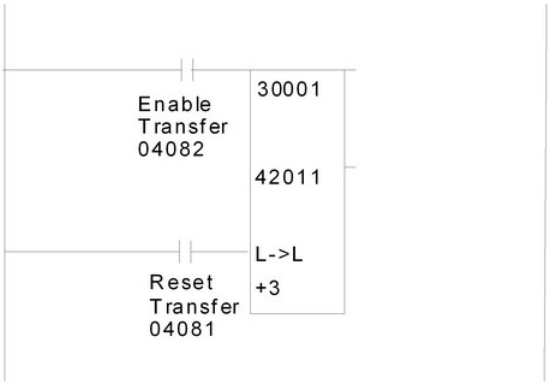
Number of Registers in List (Size)

Type	Constant
Value	For Constant type any number in range 0 to 100. This is the number of 16-bit registers.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Functions

L->L | L->R | MOVE

Example



The List to List element transfers the contents of one register in the **source** list of registers into one register of the **destination** list of registers, at the position of the index. The index register points to the same position in the source and destination lists.

Source List		Destination List	
		List Index ---->	42011
list register 0 -->	30001	list register 0 -->	42012
list register 1 -->	30002	list register 1 -->	42013
list register 2 -->	30003	list register 2 -->	42014
list register 3 -->	30004	list register 3 -->	42015

In this example the **source** register list starts at input register 30001. The **index** is holding register 42011 and the **destination** register list starts at holding register 42012. The source register list contains the first four analog input registers. The index is a general purpose analog output register that will have a value of 0, 1, 2 or 3. The destination register list starts at the next sequential register after the index register. The destination list contains four general purpose analog output registers and

has a size of four.

When list index register 42011 has a value of 0 and the L->L element is enabled the contents of source list register 30001 is transferred to destination list register 40012.

When list index register 42011 has a value of 1 and the L->L element is enabled the contents of source list register 30002 is transferred to destination list register 42013.

When list index register 42011 has a value of 2 and the L->L element is enabled the contents of source list register 30003 is transferred to destination list register 42014.

When list index register 42011 has a value of 3 and the L->L element is enabled the contents of source list register 30004 is transferred to destination register 42015.

L->R (List-to-Register Transfer)

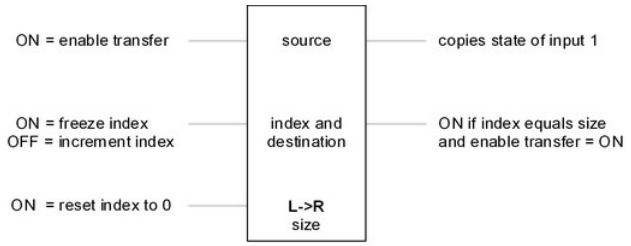
The **L->R Transfer** element transfers the contents of a register indicated by the index from the source list of registers into the destination register.

When the **enable transfer** input is ON, and the **index equals size** output is OFF, the source list register pointed to by the index is transferred to the destination register.

The **freeze/increment index** input allows control of the index. If this input is ON the index is not incremented. If the input is OFF the index is incremented by one after each transfer.

When the **reset index** input is ON the index is reset to zero.

The index is incremented by one on each transfer until the index equals size. When the **index equals size** output is ON no further increments in the index value occur until the index is reset.



Function Block Properties

Source List (Source)

Type	Register
Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Index and Destination List (Index and Destination)

Type	Register
Value	For Register type any valid 4xxxx (holding register). The address of the index register and the destination register list. The index is stored in register [4xxxx]. The data is stored in register [4xxxx+1] to register [4xxxx+size]
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

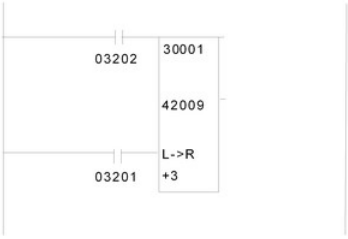
Number of Registers in List (Size)

Type	Constant
Value	For Constant type any number in range 0 to 100. This is the number of 16-bit registers.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Functions

L->L	L->R	MOVE

Example



The **List-to-Register Transfer** element transfers the contents of a register indicated by the index from the source list of registers into the destination register.

Source Register List

list register 0 --> 30001
list register 1 --> 30002
list register 2 --> 30003
list register 3 --> 30004

list index -- > 42009
Destination Register -- > 42010

An example of the registers used in a **List-to-Register** (L->R) transfer is shown in the above diagram. The source register list contains the first four analog input registers and has a size of four. The index is a general purpose analog output register that will have a value of 0, 1, 2 or 3. The destination register is a general purpose analog output register that is the next sequential register after the index register.

When list index register 42009 has a value of 0 and the **L->R** element is enabled, the contents of source register 30001 is transferred to destination register 42010.

When list index register 42009 has a value of 1 and the **L->R** element is enabled, the contents of source register 30002 is transferred to destination register 42010.

When list index register 42009 has a value of 2 and the **L->R** element is enabled, the contents of source register 30003 is transferred to destination register 42010.

When list index register 42009 has a value of 3 and the **L->R** element is enabled, the contents of source register 30004 is transferred to destination register 42010.

R->L (Register to List Transfer)

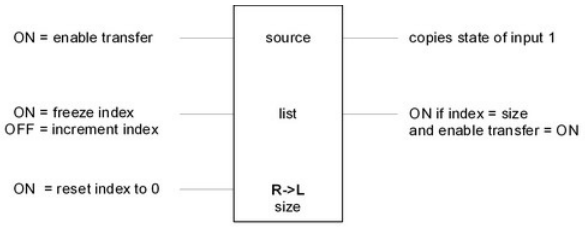
The **R->L** element transfers the contents of the source register into a register in the destination list of registers indicated by the index.

When the **enable transfer** input is ON and the **index equals size** output is OFF, the source register is transferred to the destination list register at the position of the index.

The **freeze/increment index** input allows control of the index. If this input is ON the index is not incremented. If the input is OFF the index is incremented by one after each transfer.

When the **reset index** input is ON the index is reset to zero.

The index is incremented by one on each transfer until the index equals size. When the **index equals size** output is ON no further increments of the index value occur until the index is reset.



Function Block Properties

Source (Source)

Type	Register
Value	For Register type any valid 0xxxx (coil register), 1xxxx (status register), 3xxxx (input register) or 4xxxx (holding register). This function accesses 16-bit words. Coil register blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, and so on. Input blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 10001, 10017, 10033, and so on.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Index and Destination List (Index and Destination)

Type	Register
Value	For Register type any valid 4xxxx (holding register). The address of the index register and the destination register list. The index is stored in register [4xxxx]. The data is stored in register [4xxxx+1] to register [4xxxx+size]
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Number of Registers in List (Size)

Type	Constant
Value	For Constant type any number in range 0 to 100. This is the number of 16-

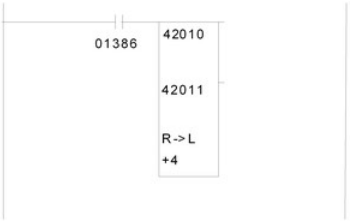
bit registers.

Tag The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Functions

L->L | L->R | MOVE

Example



The Register to List element transfers the contents of the source register 42010 into a register in the destination list of registers, 42012 to 42015, indicated by the index register 42011.

An example of the registers used in a Register to List (R->L) transfer is shown in the above diagram. The source register is a general purpose analog output register. The index is a general purpose analog output register that will have a value of 0, 1, 2 or 3. The destination register list starts at the next sequential register after the index register. The destination list contains four general purpose analog output registers and has a size of 4.

		Destination Register List
	List Index -->	42011
Source Register 42010	list register 0 -->	42012
	list register 1 -->	42013
	list register 2 -->	42014
	list register 3 -->	42015

An example of the registers used in a Register to List (R->L) transfer is shown in the above diagram. The source register is a general purpose analog output register. The index is a general purpose analog output register that will have a value of 0, 1, 2 or 3. The destination register list starts at the next sequential register after the index register. The destination list contains four general purpose analog output registers and has a size of 4.

When list index register 42011 has a value of 0 and the R->L element is enabled the contents of source register 42010 is transferred to destination list register 40012.

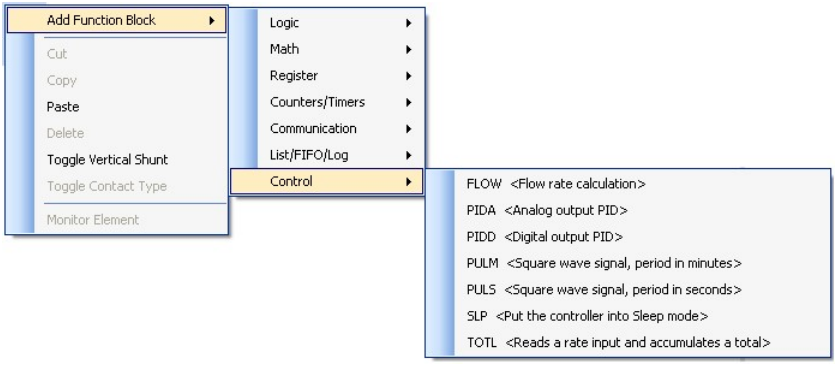
When list index register 42011 has a value of 1 and the R->L element is enabled the contents of source register 42010 is transferred to destination list register 42013.

When list index register 42011 has a value of 2 and the R->L element is enabled the contents of source register 42010 is transferred to destination list register 42014.

When list index register 42011 has a value of 3 and the R->L element is enabled the contents of source register 42010 is transferred to destination register 42015.

Control Functions

The Control functions available in Telepace Studio are shown in the image below. This image is opened in Telepace Studio by right clicking on the ladder canvas and selecting the Add Function Block command. Depending on the available space on the ladder canvas some functions may not be displayed as they would not fit in the canvas at the location selected.



Hover your mouse over the icons below for a description of the function. Click the icon to go to the function description.

FLOW | PIDA | PIDD | PULM
PULS | SLP | TOTL

FLOW (Flow Accumulator)

The **FLOW** element accumulates flow from pulse type devices such as turbine flow meters.

When the accumulate input is ON the element accumulates volume and calculates the flow rate.

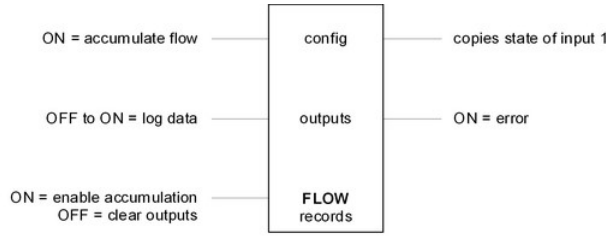
When the Log Data input goes from OFF to ON, the accumulated volume, flow time and the time at the end of the period is saved in the history registers. Older history is pushed down and the oldest record is discarded. The Log Data input must be triggered at least once every 119 hours (at the maximum pulse rate). Otherwise the

volume accumulator will overflow, and the accumulated volume will not be accurate.

The accuracy of the FLOW block improves with longer periods between logging data. For greatest accuracy avoid logging small periods of time.

When the accumulator enabled input is ON, accumulation and rate calculations are enabled. When the input is OFF, all accumulators and the rate outputs are set to zero.

The element reads and accumulates the number of pulses, and divides by the K factor to calculate the total volume. This is done on each scan of the controller logic. The element calculates the flow rate in engineering units based on the K-factor provided. The rate updates once per second if there is sufficient flow. If the flow is insufficient, the update slows to as little as once every ten seconds to maintain resolution of the calculated rate.



Function Block Properties

Configuration Block

Type Register

Value Address of the first register (**config**) of 4 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller.

The block of registers are defined for the function block as follows:

config + 0 and + 1 = K factor (floating-point)
config + 2 = input register (3xxxx or 4xxxx)
config + 3 = input type (see table below)
config + 4 = rate period (see table below)

Tag The tag window displays the tag name for the config + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Flow Rate Status and Data Block (outputs)

Type Register

Value Address of the first register (**outputs**) of 18 to 222 in a sequential block of 4xxxx (holding) registers. The number of registers used depends on the records variable.These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller.

The block of registers are defined for the function block as follows:

output + 0 = status (see table below)
output + 1 and +2 = flow rate (floating-point). These registers contain the flow rate in engineering units based on the K-factor and rate period provided. (same for all flow rate registers)
output + 3 and +4 = rate accumulator (this is an internal register used for the accumulation of the flow and calculation of the flow rate).
output + 5 and +6 = time of last rate (this is an internal register used for the accumulation of the flow and calculation of the flow rate).
output + 7 and +8 = input at last sample (this is an internal register used for the accumulation of the flow and calculation of the flow rate).
output + 9 and +10 = pulse accumulator (this is an internal register used for the accumulation of the flow and calculation of the flow rate).
output + 11 = number of records to follow. This register indicates how many sets of flow and time registers follow. This value is equal to the **Number of Register Structures in Block** variable. There are four registers in each record.
output + 12 and +13 = volume for period 1 (float). The flow volume is stored as a floating-point number in two consecutive registers.
output + 14 and +15 = time at end of period 1. The time is stored as a 32-bit integer in two consecutive registers. The registers hold the number of seconds since January 1, 1970. This is an unsigned number.
output + 16 and +17 = flow time for period 1. The flow time is stored as a 32-bit integer in two consecutive registers. The register holds the number of seconds flow was accumulated in the period. This measures the time the Accumulate input is ON, including times when the flow was zero.
output + 18 and +19= volume for period 2 (float)
output + 20 and +21 = time at end of period 2
output + 22 and +23 = flow time for period 2
output + 24 and +25= volume for period 3 (float)
output + 26 and +27 = time at end of period 3
output + 28 and +29 = flow time for period 3
... Same format for periods 4 through 34.
output + 216 and +217 = volume for period 35 (float)
output + 218 and +219 = time at end of period 35
output + 220 and +221 = flow time for period 35

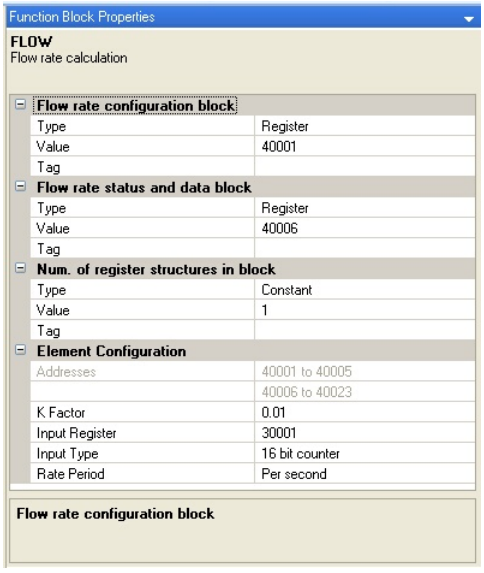
Tag The tag window displays the tag name for the output + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Number of Register Structures in Block (Records)

Type	Constant
Value	The number of measurement records stored in the output array. The valid values are 1 to 35. This value determines the number of output registers used by the element.
Tag	The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration

The FLOW element is configured using the FLOW Element Configuration view. Highlight the element by moving the cursor over the element and then use the Element Configuration view to modify the FLOW element.



The **Addresses** window displays the registers used by the function block.

The **K Factor** is a floating-point value. It may be any non-zero, positive value.

The **Input Register** is the address of the register that holds the input pulse count. Valid values are any input or holding register in the range 30001 to 49999.

The **Input Type** may be one of the following.

16 bit counter A free running 16-bit counter.
Holding register (4xxxx) = 0

32 bit counter A free running 32-bit counter with low word in the first register.
Holding register (4xxxx) = 1

16 bit difference The 16-bit difference between any two counters.
Holding Register (4xxxx) = 2

The **Rate Period** determines the units of time used to display the flow rate. It may be one of the following. Note that this value does not affect when the rate calculation is performed.

per second Holding register (4xxxx) = 0

per minute Holding register (4xxxx) = 1

per hour Holding register (4xxxx) = 2

per day Holding Resister (4xxxx) = 3

Status Codes

The status register indicates the status of the flow accumulation. It can have the following values. If the status register is non-zero, the error output is turned ON.

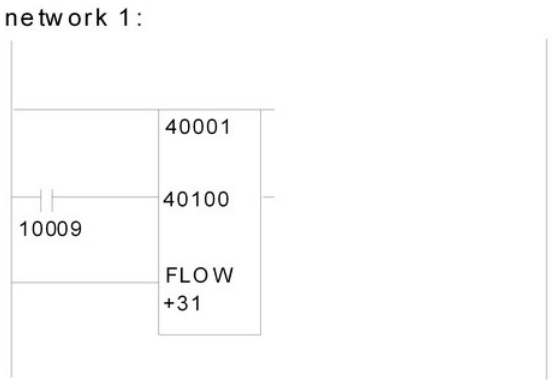
Status Register Value	Description
0	no error
1	invalid K factor configuration
2	invalid Input Register configuration
3	invalid Input Type configuration
4	invalid Rate Period configuration
5	pulse rate is too low for accurate flow rate calculation

Notes

The maximum pulse rate is 10 kHz.

The accumulated flow is calculated by measuring the accumulated pulses and dividing by the K factor. The K factor cannot be changed while an accumulation is in progress. Only change the K factor while accumulation is stopped.

Example



The **FLOW** element in network 1 has the *accumulate flow* and *enable accumulation* inputs connected to the left power rail. When these inputs are continuously powered the FLOW element accumulates volume and calculates flow rate.

Each time contact 10009 is closed, powering the log data input, the accumulated volume, flow time and the time at the end of the period is saved in the history registers.

The *records* value is set to 31. This means that 31 sets of history registers, volume, flow time and end of period time, will be logged. In this example the accumulation input is never turned off meaning that once 31 sets of history registers are saved the next time contact 10009 is closed the 31st record is removed and the newest record is added to the history.

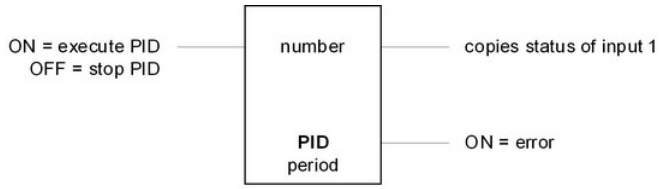
PID Controller

The **PID** element The element controls the execution of a PID controller. PID controllers execute independently of the ladder logic scan. A PID controller is configured by writing data into the PID controller registers defined in the register assignment.

The PID block starts execution of the PID controller number on the rising edge of the **execute PID** input. Execution of the PID controller is stopped when the **execute PID** is OFF.

The **execute PID** input must be toggled from OFF to ON under program control to start PID execution. The **execute PID** input cannot be directly connected to the power rail or PID execution will not restart when the program restarts.

The **error** output is enabled if the PID number is not found in the Register Assignment. The error output is also enabled if an error code is present in the PID Status Register (SR).



The SCADAPack 32, SCADAPack 32P, SCADAPack 314, SCADAPack 330/334, SCADAPack 350/357 and SCADAPack 4203 controllers do not support the PID element. See the elements PIDA – Analog Output PID and the element PIDD – Digital Output PID.

The I/O module, PID control block, must be configured in the Register Assignment when PID function blocks are are used. The I/O module PID control block provides control over the configuration of a PID control block and provides access to the control block parameters. Control block parameters are assigned to 25 consecutive holding registers. The I/O module address refers to the PID control block number.

Function Block Properties

Source

Type	Constant
Value	For Constant type any number in range 0 to 65535. This is the PID controller number.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Destination

Type	Constant
Value	For Constant type any number in range 0 to 65535. This is the PID execution period in tenths of seconds.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Tuning PID Controllers

Correct tuning of a PID controller takes into account process information such as the process variable and control information such as open/close time for valves etc. These parameters and the correct tuning requirements are outside the scope of this manual. The world wide web offers a wealth of information on tuning all types of PID controllers. Use search words such as PID controller, PID or velocity algorithm etc.

Related Elements

PIDA	PIDD

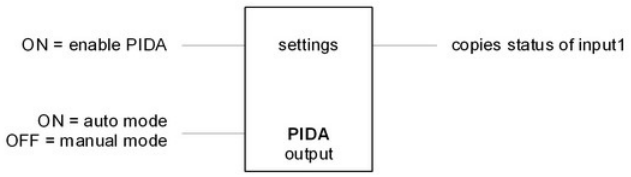
PIDA (Analog Output PID)

The **PIDA** element performs a PID algorithm and calculates an analog output.

When the **enable PIDA** input is ON, the element executes the PID algorithm. Execution of the PID algorithm is stopped when the **enable PIDA** is OFF.

When the **auto/manual mode** input is ON, the PID is placed in automatic mode. In automatic mode the **output** is calculated using the PID algorithm. A new calculation is done at the rate specified by **cycle time**.

When the **auto/manual mode** input is OFF, the PID is placed in manual mode. In manual mode the **output** is set to the value specified by **manual mode output**. The **output** is limited to the range set by **full** and **zero**.



Function Block Properties

Configuration Block (settings)

Type	Register
Value	Address of the first register (**config**) of 20 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller. The block of registers are defined for the function block as follows: **config** + 0 and +1 = process value (float). The **process value** is a value that represents the actual

state of the process being controlled. This value is typically input to the controller as an analog input signal or as a signal from a MVT. The process value is a floating-point number. If the process value is an analog input signal to the controller it must be converted to a floating-point number using the STOF (Signed Integer to Floating-point) element. If the analog input requires scaling use the SCAL-Scale Analog Value element to scale the input and provide a floating-point number for the process value.

config + 2 and +3 = setpoint (float)
config + 4 and +5 = gain (float)
config + 6 and +7 = reset time in seconds (float)
config + 8 and +9 = rate time in seconds (float)
config + 10 and +11 = deadband (float)
config + 12 and +13 = full (float)
config + 14 and +15 = zero (float)
config + 16 and +17 = cycle time in seconds (float)
config + 18 and +19 = manual mode output (float)

Tag The tag window displays the tag name for the config + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

PID Output Block (output)

Type Register

Value Address of the first register (**outputs**) of 18 to 222 in a sequential block of 4xxxx (holding) registers. The number of registers used depends on the records variable.These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller.

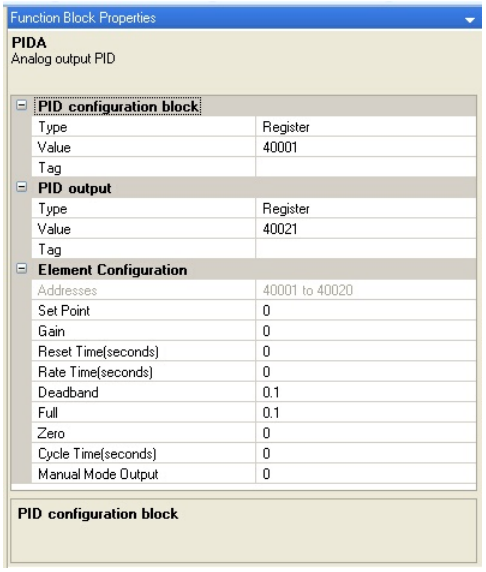
The block of registers are defined for the function block as follows:

output + 0 and +1 = PID output (float)
output + 2 and +3 = internal: process value N-1st (float)
output + 4 and +5 = internal: process value N-2nd (float)
output + 6 and +7 = internal: error N-1st (float)
output + 8 and +9 = internal: time of last scan (unsigned double)

Tag The tag window displays the tag name for the output + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration

The PIDA element is configured using the PIDA Element Configuration.



Parameter	Description
Setpoint	The **setpoint** is a floating-point value representing the desired value of the process value. The error value is the difference between the process value and the setpoint. **error = process value – setpoint (+/- deadband).**
Gain	The proportional (P) part of the PID algorithm is the **gain**. A positive value of gain configures a forward-acting PID controller and a negative value of gain configures a reverse acting controller.
Reset Time (seconds)	The integral (I) part of the PID algorithm is the **reset time**. This value, in seconds, controls the reset gain (or magnitude of integral action) in a PI or PID controller. This is typically referred to as Seconds Per Repeat. From the equation above it is seen that the integral action of the PI or PID controller is a function of the reset time and the execution period (cycle time). A smaller reset time provides more integral action and a larger reset time provides less integral action. Valid range is any value greater than 0. A value of 0 disables the reset action.
Rate Time (seconds)	The derivative (D) part of the PID algorithm is the **rate time.** This value, in seconds, controls the rate gain (or magnitude of derivative action) in a PD or PID controller. From the equation above it is seen that the derivative action of the PD or PID controller is a function of the rate gain and the execution period (cycle time). A larger rate gain provides more derivative action and a smaller rate gain provides less derivative action. Valid range is any value greater than 0. A value of 0 disables the rate action.
Deadband	The **deadband** parameter is used by the PID algorithm to determine if the process requires the control outputs to be changed. If the absolute value of the error is less than the deadband, then the function

	block skips execution of the control algorithm. This prevents changes to the output when the process value is near the setpoint and can reduce wear on the control elements. Valid range is any value greater than 0.The **setpoint** is a floating-point value representing the desired value of the process value.
Full	The **full** setting is used in limiting the maximum output value of the PIDA function. If the PID algorithm calculates an **output** quantity that is greater than the value stored in **full**, the output quantity is set equal to the value stored in **full**. The **full** setting should always be greater than the **zero** setting.
Zero	The **zero** setting is used in limiting the minimum output value of the PIDA function. If the PID algorithm calculates an **output** quantity that is less than the value stored in **zero**, the **output** quantity is set equal to the value stored in zero. The **zero** setting should always be less than the **full** setting.
Cycle Time (seconds)	The **cycle time** is the floating-point value of the PID algorithm **execution period** measured in seconds. Any value greater than or equal to 0.001 seconds (1 ms) may be specified. If the cycle time specified is less than the scan time of the Telepace program, the program scan time becomes the PID cycle time.
Manual Mode	The **manual mode output** is the value that the output is set to when the PIDA function is in manual mode.

PID Velocity Algorithm

The PIDA element uses the velocity form of the PID algorithm. The velocity form calculates the change in the output and adds it to the previous output.

$$e_n = s_n - p_n$$

$$m_n = m_{n-1} + K\left[e_n - e_{n-1} + \frac{T}{T_i}e_n - \frac{R}{T}(p_n - 2p_{n-1} + p_{n-2})\right]$$

Where: e = error
s = setpoint
p = process value
K = gain
T = execution period
T_i = integral or reset time
R = rate gain
m = output

Tuning PID Controllers

Correct tuning of a PID controller takes into account process information such as the process variable and control information such as open/close time for valves etc. These parameters and the correct tuning requirements are outside the scope of this manual. The world wide web offers a wealth of information on tuning all types of PID controllers. Use search words such as PID controller, PID or velocity algorithm etc.

Related Elements

PIDD

PIDD (Digital Output PID)

The **PIDD** element performs a PID velocity algorithm (see description below) and operates two discrete outputs to maintain PID control.

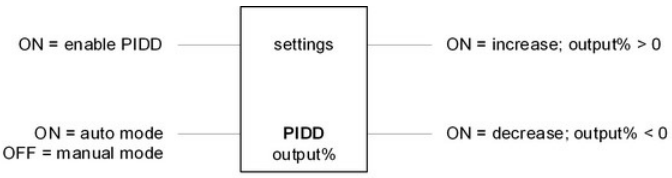
The output of the PIDD is the output percent value **output%**. The **output%** is a duty cycle of the **cycle time** as defined by the **zero%** and **full%** parameters.

The **increase** output is cycled when the **output%** is greater than zero. The **decrease** output is cycled when the **output%** is less than zero.

When the **enable PIDD** input is ON, the element executes the PID algorithm. Execution of the PID algorithm is stopped when the **enable PIDD** is OFF.

When the **auto/manual mode** input is ON, the PID is placed in automatic mode. In automatic mode the output is calculated using the PID algorithm. A new calculation is done at the rate specified by **cycle time**.

When the **auto/manual mode** input is OFF, the PID is placed in manual mode. In manual mode the **output%** is set to the value specified by **manual output%**. The output is limited to the range set by **full%** and **zero%**.



Function Block Properties

Configuration Block (settings)

Type	Register
Value	Address of the first register (**config**) of 21 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller.

The block of registers are defined for the function block as follows:

config + 0 and +1 = process value (float). The **process value** is a value that represents the actual state of the process being controlled. This value is typically input to the controller as an analog input signal or as a signal from a MVT. The process value is a floating-point number. If the process value is an analog input signal to the controller it must be converted to a floating-point number using the STOF (Signed Integer to Floating-point) element. If the analog input requires scaling use the SCAL-Scale Analog Value element to scale the input and provide a floating-point number for the process value.

config + 2 and +3 = setpoint (float)
config + 4 and +5 = gain (float)
config + 6 and +7 = reset time in seconds (float)

config + 8 and +9 = rate time in seconds (float)
config + 10 and +11 = deadband (float)
config + 12 and +13 = full % (float)
config + 14 and +15 = zero % (float)
config + 16 and +17 = cycle time in seconds (float)
config + 18 and +19 = manual mode output (float)
config + 20 = motor output enabled

Tag The tag window displays the tag name for the config + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

PID Output Block (output %)

Type Register

Value Address of the first register (**outputs**) of 18 to 222 in a sequential block of 4xxxx (holding) registers. The number of registers used depends on the records variable.These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller.

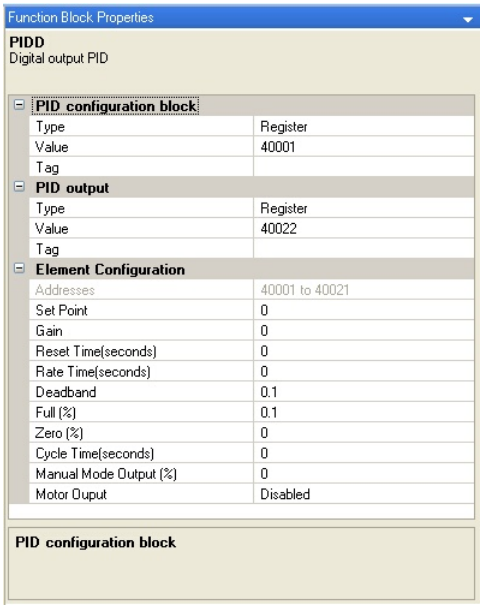
The block of registers are defined for the function block as follows:

output + 0 and +1 = PID output % (float)
output + 2 and +3 = internal: process value N-1st (float)
output + 4 and +5 = internal: process value N-2nd (float)
output + 6 and +7 = internal: error N-1st (float)
output + 8 and +9 = internal: time of last scan (unsigned double)

Tag The tag window displays the tag name for the output + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration

The PIDD element is configured using the PIDD Element Configuration.



Parameter	Description
Setpoint	The **setpoint** is a floating-point value representing the desired value of the process value. The error value is the difference between the process value and the setpoint. **error = process value – setpoint (+/- deadband)**.
Gain	The proportional (P) part of the PID algorithm is the **gain.** A positive value of gain configures a forward-acting PID controller and a negative value of gain configures a reverse acting controller.
Reset Time	The integral (I) part of the PID algorithm is the **reset time**. This value, in seconds, controls the reset gain (or magnitude of integral action) in a PI or PID controller. This is typically referred to as Seconds Per Repeat. From the equation above it is seen that the integral action of the PI or PID controller is a function of the reset time and the execution period (cycle time). A smaller reset time provides more integral action and a larger reset time provides less integral action. Valid range is any value greater than 0. A value of 0 disables the reset action. As a rule of thumb the reset time should be larger than the cycle time.
Rate TIme	The derivative (D) part of the PID algorithm is the **rate time.** This value, in seconds, controls the rate gain (or magnitude of derivative action) in a PD or PID controller. From the equation above it is seen that the derivative action of the PD or PID controller is a function of the rate gain and the execution period (cycle time). A larger rate gain provides more derivative action and a smaller rate gain provides less derivative action. Valid range is any value greater than 0. A value of 0 disables the rate action.
Deadband	The **deadband** parameter is used by the PID algorithm to determine if the process requires the control outputs to be changed. If the absolute value of the error is less than the deadband, then the function block skips execution of the control algorithm. This prevents changes to the output when the process value is near the setpoint and can reduce wear on the control elements. Valid range is any value greater than 0.The

	setpoint is a floating-point value representing the desired value of the process value.
Full (%)	The **Full (%)** setting is the full-scale output limit in percent of **cycle time**. For example if the cycle time is 10 seconds and the full% value is 100 then the maximum duty cycle of the output% is 100 percent or 10 seconds. When the output value is 0 or greater then the **increase** output is turned on for the duty cycle of the output%. When the output value is less than zero the **decrease** output is turned on for the duty cycle of the output%.
Zero (%)	The **Zero (%)** setting is the zero scale output limit in percent of **cycle time**. When the output value is 0 or greater then the **increase** output is turned on for the duty cycle of the output%. When the output value is less than zero the **decrease** output is turned on for the duty cycle of the output%.
Cycle Time	The **cycle time** is the floating-point value of the PID algorithm **execution period** measured in seconds. Any value greater than or equal to 0.001 seconds (1 ms) may be specified. If the cycle time specified is less than the scan time of the Telepace program, the program scan time becomes the PID cycle time. As a rule of thumb the cycle time should be twice as large as the controller scan time.
Manual Mode Output	The **manual mode output** is the value that the **output%** is set to when the PIDD function is in manual mode.
Motor Output	When the **motor output** is set to **enabled** the **increase** and **decrease** outputs are de-energized when error is within the deadband. When the **motor output** is set to **disabled** the **increase** and **decrease** outputs operate continuously based on the **output%**.

PID Velocity Algorithm

The **PIDD** element uses the velocity form of the PID algorithm. The velocity form calculates the change in the output and adds it to the previous output.

$$e_n = s_n - p_n$$

$$m_n = m_{n-1} + K\left[e_n - e_{n-1} + \frac{T}{T_i}e_n - \frac{R}{T}(p_n - 2p_{n-1} + p_{n-2})\right]$$

Where: e = error
s = setpoint
p = process value
K = gain
T = execution period
T_i = integral or reset time
R = rate gain
m = output

Tuning PID Controllers

Correct tuning of a PID controller takes into account process information such as the process variable and control information such as open/close time for valves etc. These parameters and the correct tuning requirements are outside the scope of this manual. The world wide web offers a wealth of information on tuning all types of PID controllers. Use search words such as PID controller, PID or velocity algorithm etc.

Related Elements

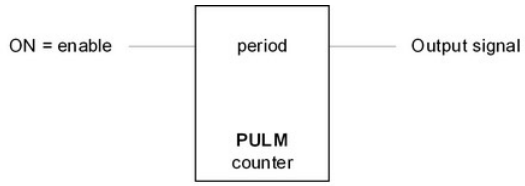
PIDA

PULM (Pulse Minutes)

The **PULM** element controls a square wave digital signal.

While the **enable** is ON, the **output** signal will be active ON or OFF based on the period. The output is powered only when the enable input is ON.

For example when the enable input is on and a period of 10 minutes is used the result will be an output square wave that is ON for 5 minutes and then OFF for 5 minutes.



Function Block Properties

Pulse Period (Period)

Type	Constant Register
Value	For Constant type any number in range 1 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register). The period is from 1 to 65535 minutes.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

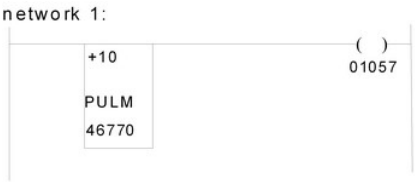
Timer Accumulator (Counter)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Elements

PULS

Example



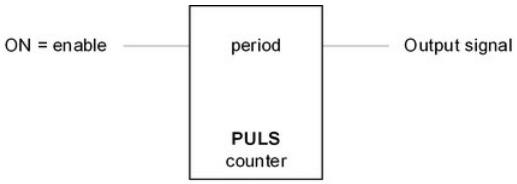
In this example the **PULM** element has a period value of 10 minutes. Coil 01057 will be on for 5 minutes and then off for 5 minutes. When the program is first run coil 01057 will start in on state for 5 minutes then go off for 5 minutes, repeating this cycle as long as the enable input is true. Registers 46770 and 46771 contain the accumulated time for the period in seconds.

PULS (Pulse Seconds)

The **PULS** element controls a square wave digital signal.

While the **enable** is ON, the **output** signal will be active ON or OFF based on the period. The output is powered only when the enable input is ON.

For example when the enable input is on and a period of 10 seconds is used the result will be an output square wave that is ON for 5 seconds and then OFF for 5 seconds.



Function Block Properties

Pulse Period (Period)

Type	Constant Register
Value	For Constant type any number in range 1 to 65535. For Register type any valid 3xxxx (input register) or 4xxxx (holding register). The period is from 1 to 65535 seconds. The smallest actual period will depend on the ladder scan time. Smaller ladder programs will be able to handle smaller periods.
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Timer Accumulator (Counter)

Type	Register
Value	For Register type any valid 4xxxx (holding register).
Tag	The tag window displays the tag name for the selected constant or register if one exists. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Related Elements

PULM

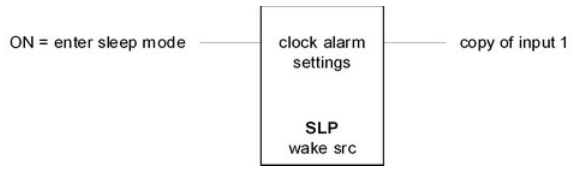
Example

In this example the **PULS** element has a period value of 10 of seconds. Coil 01057 will be on for 5 seconds and then off for 5 seconds. When the program is first run coil 01057 will start in on state for 5 seconds then go off for 5 seconds, repeating this cycle as long as the enable input is true. Registers 45660 and 45661 contain the accumulated time for the period in tenths of seconds.

SLP (Put controller into Sleep Mode)

The **SLP** element places the controller into sleep mode. Sleep mode reduces power consumption to a minimum by halting the microprocessor clock and shutting down the power supply. All programs halt until the controller resumes execution. All output points turn off while the controller is in sleep mode.

This element is not supported in the SCADAPack 32, SCADAPack 100 and FlowStation 110 controllers.



Function Block Properties

Clock and Alarm Settings Block

Type Register

Value Address of the first register (**config**) of 4 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller.

The block of registers are defined for the function block as follows:

Address of the first register in the clock alarm settings block. There are 4 registers in the block at addresses settings+0 to settings + 3

clock + 0 = type

0 = no alarm

1 = absolute time alarm

2 = elapsed time alarm

clock + 1 = hour

clock + 2 = minute

clock + 3 = second

Tag The tag window displays the tag name for the clock + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Wake Source (wake src) Block

Type Register

Value Address of the wake up source register. This register indicates the reason the controller left sleep mode. Valid values are:

1 = real time clock alarm

2 = interrupt input

4 = led power switch

8 = counter 0 overflow

16 = counter 1 overflow

32 = counter 2 overflow

256 = Assertion of:

- digital input 0 for SCADAPack 350
- Any digital input on SCADAPack LP
- INT/Cntr digital Input on SCADAPack with 5203/5204 controller board.

2048 = USB peripheral port connection.

Tag The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

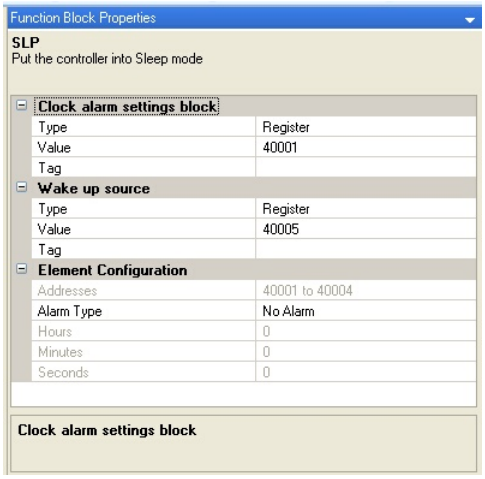
Notes

The SLP element triggers the same interrupt as the Real Time Clock and Alarm register assignment. Using both alarm settings, therefore, in the same program is not recommended.

The SCADAPack 314, SCADAPack 330, SCADAPack 334 and SCADAPack 350 controllers will not go to sleep if the USB peripheral port is connected to a host device such as a PC.

Element Configuration

The SLP element is configured using the SLP Element Configuration.



The **Addresses** window displays the registers used by the function block.

The **Alarm Type** sets the method to use for taking the controller out of sleep mode. The selections are No Alarm, Absolute Time and Elapsed Time.

- **Absolute Time** sets the wake time to a fixed time of day.
- **Elapsed Time** sets the wake time to a fixed period of time.

For Absolute Time:

The **Hours** setting is a value bet\ween 0 and 23.

The **Minutes** setting is a value bet\ween 0 and 59.

The **Seconds** setting is a value bet\ween 0 and 59.

For Elapsed Time

The **Hours** setting is a value bet\ween 0 and 23.

The **Minutes** setting is a value bet\ween 0 and 1439.

The **Seconds** setting is a value bet\ween 0 and 65535

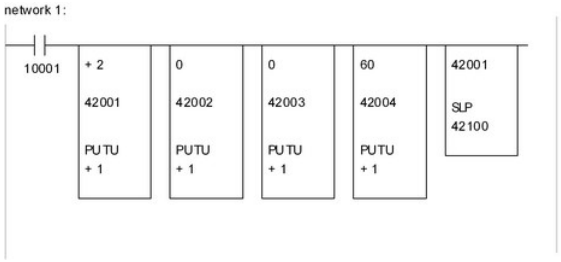
Total of times must be less than 23 hours, 59 minutes, 59 seconds

Wake up Effects on Controller

Condition	Wake-up Effects
Hardware Reset	Application programs execute from start of program.
External Interrupt	Program execution continues from point sleep element was executed.
Real time Clock Alarm	Program execution continues from point sleep element was executed.
LED Power Button	Program execution continues from point sleep element was executed.
Hardware Counter Rollover	Program execution continues from point sleep element was executed.
USB Peripheral Connection	Program execution continues from point sleep element was executed.

Example

The following network will put the controller into sleep mode. The clock alarm is set to wake the controller in 60 seconds. Other conditions may wake up the controller before the 60 seconds expires.



The **Element Configuration** dialog may be used to edit the configuration of the SLP block, instead of PUTU elements. This reduces the memory used by the program. However, the alarm settings are written to the controller only when the program is loaded. The network using the PUTU elements configures the alarm settings on every execution of the SLP element.

TOTL (Analog Totalizer)

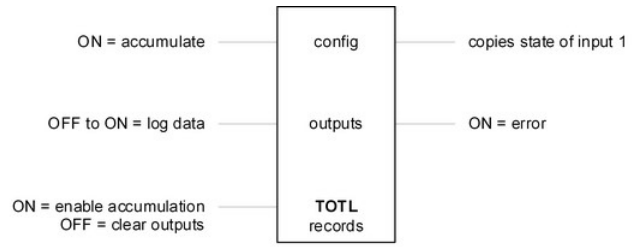
The **TOTL** element reads a rate input and accumulates (integrates) a total. It is used to measure a flow rate and accumulate a volume, or a similar calculation.

When the **Accumulate** input is ON and it has been longer than the sample interval since the last accumulation, the element reads the value of the input, scales it by time difference from the previous accumulation, and adds it into the total.

When the **Log Data** input goes from OFF to ON, the accumulated total, accumulation time, and the time at the end of the period is saved in the history registers. Older history is pushed down and the oldest record is discarded.

When the **Enable Accumulation** input is ON, accumulation is enabled. When the input is OFF, all accumulators and outputs are set to zero.

The **Error** output is ON if there is an error in the configuration registers.



Function Block Properties

Configuration Block

Type Register

Value Address of the first register (**config**) of 3 in a sequential block of 4xxxx (holding) registers. These registers contain the information used by the function block. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller.

The block of registers are defined for the function block as follows:

config + 0 = rate input register

config + 1 = input rate

0 = per second
1 = per minute
2 = per hour
3 = per day

config + 2 = sample interval (tenths of seconds)

Tag The tag window displays the tag name for the config + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

TOTL Status and Data Block (outputs)

Type Register

Value Address of the first register (**outputs**) of 10 to 214 in a sequential block of 4xxxx (holding) registers. The number of registers used depends on the records variable. The **Element Configuration** is used to initially configure the function block. As well, the information in the registers may be changed using other Telepace Studio function blocks or from a Modbus device communicating with the controller.

The block of registers are defined for the function block as follows:

Address of the first register of the output block. There are 10 to 214 registers in the block at outputs+0 to outputs+213. The number of registers used depends on the records variable.

output + 0 = number of records to follow

output + 1 = Indicates the status of the accumulation. It can have the following values. If the status register is non-zero, the error output is turned ON.. It may be one of the following.
0 = no error
1 = invalid rate units of configuration
2 = invalid sample interval

output + 2 and +3 = internal time at last sample (This is an internal register and is not intended for use by a ladder logic program. The data in this register is stored as a floating-point number in two consecutive registers:).

output + 4 and +5 = period 1 total accumulated value. (float)

output + 6 and +7 = period 1 end time stored as a 32-bit integer in two consecutive registers. These registers hold the number of seconds since January 1, 1970. This is an unsigned number..

output + 8 and +9 = period 1 accumulation time. These registers hold the number of seconds accumulation occurred in the period. This measures the time the Accumulate input is ON, including times when the rate was zero.

output + 10 and +11 = period 2 total accumulated value. (float).

output + 12 and +13 = period 2 end time stored

output + 14 and +15 = period 2 accumulation time.

output + 16 and +17 = period 3 total accumulated value. (float).

output + 18 and +19= period 3 end time stored

output + 20 and +21 = period 3 accumulation time.

… Same format for periods 4 through 34.

output + 209 and +210 = period 35 total accumulated value. (float).

output + 211 and +212 = period 35 end time stored

output + 213 and +214 = period 35 accumulation time.

Tag The tag window displays the tag name for the output + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Number of Register Structures in Block (Records)

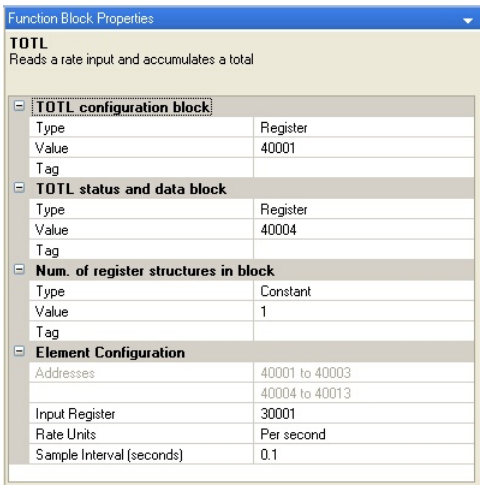
Type Constant

Value The number of measurement records stored in the output array. The valid values are 1 to 35. This value determines the number of output registers used by the element.

Tag The tag window displays the tag name for the status + 0 register above. A tag name may be entered (up to 32 characters) or edited using this window. See the Tags Management section for further information on tag names.

Element Configuration

The TOTL element is configured using the TOTL Element Configuration.



The **Addresses** window displays the registers used by the function block.

The **Input Register** specifies the register address from which the rate input is read. Valid value for this register is any floating point register. The element will use the specified register, and the next sequential register, as a floating point rate value. The address range depends upon the registers supported in the selected controller type.

The **Rate Units** specifies the units of time for the rate input. It may be one of the following.

- per second
- per minute
- per hour
- per day

The **Sample Interval (seconds)** specifies the interval at which the rate input will be sampled. A sample is taken and a calculation performed if the time since the last sample is greater than or equal to the sample interval. The exact time depends on the scan time of the logic program. The valid values are 0.1 to 6553.5 seconds (1 to 65535 tenths of a second). The dialog shows the value in seconds; the register contains an integer value in tenths of seconds.

The period over which readings are taken will vary, but will stay in the range –1 * (Expected Interval + Configured Sample Interval + Maximum time between ladder logic scans) <= Actual Sampling Interval <= +1 * (Expected Interval + Configured Sample Interval + Maximum time between ladder logic scans). As a result the total for individual periods may fluctuate despite a constant input.

Notes

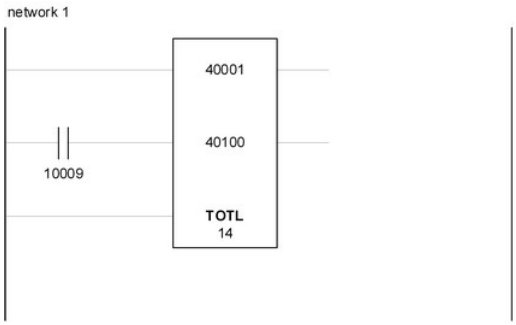
1. The accumulated value is a floating-point number. All floating-point numbers are approximations. If the accumulated value grows large, then low rate inputs will have little or no effect on the accumulated value and the accumulated value will not be accurate. Use the Log Data input to save the accumulated value and start a new accumulation when the accumulated value grows large.
2. The Log Data input must be triggered at a suitable rate or the accumulator will overflow, and the accumulated value will not be accurate.

Related Elements

FLOW

Example

In this example a **TOTL** element is used to read a flow rate from an analog input and accumulate 14 days of data. The analog input measures a flow rate in gallons per minute.



The **TOTL** element in network 1 has the **Accumulate** and **Enable Accumulation** inputs connected to the left power rail. When these inputs are continuously powered the TOTL element accumulates totals.

The **Records** variable is set to 14. This means 14 sets of history registers (total, flow time and end of period time) will be logged. In this example the accumulation input is never turned off meaning that once 14 sets of history registers are saved the next time contact 10009 is closed the 14th record is removed and the newest record is added to the history.

Each time contact 10009 is closed it powers the log data input. The accumulated total, accumulation time and the time at the end of the period is saved in the history registers. The contact should be closed by another network once per day for this example.

The element configuration for the TOTL block is as follows:

Parameter	Value	Notes
Input Register	42000	The register that contains the rate input. This is a floating point value contained in register 42000 and 42001.
Rate Units	per minute	The value in floating-point register 42000 represents the flow rate in units/minute.
Sample Interval	10	Sample, calculate, and accumulate approximately every 10.0 seconds.

Telepace Register Assignment I/O Modules

These register assignment modules are used to assign data from physical analog inputs to input registers in the I/O database. The physical analog inputs are specific SCADAPack I/O I/O modules, generic I/O modules, and internal controller data such as RAM battery voltage and board temperature.

To properly view analog input/output registers, the register type should be set to type signed. Setting the register type to anything else may result in inaccurate readings being displayed/written to these registers. to these registers.

Analog Input

These modules are used to assign data from the I/O database to physical analog outputs. The physical analog outputs are specific SCADAPack I/O I/O modules or generic I/O modules.

To properly view analog input/output registers, the register type should be set to type signed. Setting the register type to anything else may result in inaccurate readings being displayed/written to these registers. to these registers.

Analog Output

These modules are used to assign data from the I/O database to physical analog outputs. The physical analog outputs are specific SCADAPack I/O I/O modules or generic I/O modules.

To properly view analog input/output registers, the register type should be set to type signed. Setting the register type to anything else may result in inaccurate readings being displayed/written to these registers. to these registers.

Controller Configuration

These modules are used to assign data from I/O database coil and holding registers to controller configuration registers. The configuration data is used to configure controller settings such as clearing protocol and serial counters, real time clock settings, HART protocol interface, and PID control blocks.

Counter Input

These modules are used to assign data from physical counter inputs to input registers in the I/O database. The physical counter inputs are specific SCADAPack I/O I/O modules and controller counter inputs.

Controller Diagnostic

These modules are used to assign diagnostic data from the controller to input or status registers in the I/O database. The diagnostic data is used to monitor internal controller data such as controller status code, the force LED, serial port communication status and serial port protocol status.

Digital Input

These modules are used to assign data from physical digital inputs to input registers in the I/O database. The physical digital inputs are specific SCADAPack I/O I/O modules, generic I/O modules, and controller digital inputs.

Digital Output

These modules are used to assign data from the I/O database to physical digital outputs. The physical digital outputs are specific SCADAPack I/O I/O modules or generic I/O modules.

Default Register Assignment

This provides a default assignment of registers for the selected controller type.

Digital Output Mismatch

SCADAPack LP

Known outputs are compared to the corresponding inputs to detect incorrect outputs. A point is compared if it has been turned on at any time since controller reset. This input indicates if one or more outputs mismatch. The source of the mismatch can be determined by comparing each digital input against the corresponding digital output.

The SCADAPack LP on board digital I/O can be inputs or outputs. There is no configuration for the type, the SCADAPack LP accepts digital inputs on a point and will write a digital output if programmed.

For example if D I/O point 0 is assigned registers 10001 as an input it is also assigned as digital output 00001. If the application turns on 00001 then 10001 will be seen as ON. While this flexibility is useful it can cause problems in an application if users inadvertently turn ON a DIO that is used as an input in an application.

The internal DI 13 is turned ON if the SCADAPack LP detects that a point has been turned on as an input and then turned on as a digital output in the application. Each time the controller is reset it begins the internal test again.

SCADAPack 350

Known outputs are compared to the corresponding inputs to detect incorrect outputs. A point is compared if it has been turned on at any time since controller reset. This input indicates if one or more outputs mismatch. The source of the mismatch can be determined by comparing each digital input against the corresponding digital output.

The SCADAPack 350 on board digital I/O can be inputs or outputs. There is no configuration for the type, the SCADAPack 350 accepts digital inputs on a point and will write a digital output if programmed.

For example if D I/O point 0 is assigned registers 10001 as an input it is also assigned as digital output 00001. If the application turns on 00001 then 10001 will be seen as ON. While this flexibility is useful it can cause problems in an application if users inadvertently turn ON a DIO that is used as an input in an application.

The internal DI 11 is turned ON if the SCADAPack 350 detects that a point has been turned on as an input and then turned on as a digital output in the application. Each time the controller is reset it begins the internal test again.

Register Assignment Reference

Module categories comprise the root nodes of the tree view. The main categories in alphabetical order include:

Analog Input Modules
5501 Eight Channel Module
5502 Eight Channel Differential Module
5503 Four Channel RTD Module
5504 Eight Channel Thermocouple Module
5505 Eight Channel RTD Module
5506 Eight Channel Module
5521 Eight Channel Potentiometer Module
Controller Board Temperature
Controller RAM Battery Voltage
Generic Eight Channel Module

Analog Output Modules
5301 Two Channel Module
5302 Four Channel Module
5304 Four Channel Module
Generic Four Channel Module
Generic Two Channel Module
SCADAPack Two Point Module

Configuration Modules
5904 HART Interface
Clear Serial Port Counters
Clear Serial Port Protocol Counters
Controller Device Configuration
Controller IP Settings
Controller LED Power Settings
Controller Power Mode
Modbus IP Interface
Modbus IP Protocols
Modbus/TCP Settings
PID Control Block
Real Time Clock and Alarm
Save Settings to EEPROM
Serial Port DTR Off
Serial Port Protocol Settings Method 1
Serial Port Protocol Settings Method 2

Controller Default Modules

A default Register Assignment is provided for the SCADAPack of controllers. For more information, see the following topics:

4202 DR Extended/4203 DR I/O Register Assignment
SCADAPack 4202/4203 DS I/O Default Register Assignment
Micro16 Default Register Assignment (Controller I/O Only)
SCADAPack (5601 I/O Module) Default Register Assignment
SCADAPack (5604 I/O Module) Default Register Assignment
SCADAPack LIGHT Default Register Assignment
SCADAPack PLUS (5601 I/O Module) Default Register Assignment
SCADAPack PLUS (5604 I/O Module) Default Register Assignment
SCADAPack LP Default Register Assignment
SCADAPack 330 Default Register Assignment
SCADAPack 334 Default Register Assignment
SCADAPack 350 5V/20mA Default Register Assignment
SCADAPack 350 10V/40mA Default Register Assignment
SCADAPack 32 (5601 I/O Module) Default Register Assignment
SCADAPack 32 (5604 I/O Module) Default Register Assignment
SCADAPack 32P Default Register Assignment
SCADAPack 100

SCADAPack 4202 DR Extended/4203 DR I/O

Description

A default Register Assignment is provided for the SCADAPack of controllers. The table below provides the default register assignment for these controllers.

To enable the default register assignment for a SCADAPack controller:

- Select **Type** from the **Controller** menu.
- Select the appropriate SCADAPack controller.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment* dialog.
- If the controller type you selected is a SCADAPack 4202 DR, you are presented with the default register assignment for a 4202 DR and 4202 DR Extended/4203 DR. Select the appropriate module depending on your controller type.

Module	Address	Default Register Assignment
SCADAPack 4202 DR I/O	Fixed	
digital inputs		10001 to 10001
analog inputs		30001 to 30005
analog outputs		40500 to 40500
SCADAPack 4202 DR Extended/4203 DR I/O	Fixed	
digital output		00001 to 00001
digital inputs		10001 to 10001
analog inputs		30001 to 30005
analog outputs		40500 to 40500

Note:	For SCADAPack and SCADAPack 32 controllers it is necessary to create a logic application consisting of at least one element. If there is no logic program present, then the logic application ID is cleared.

Related Topics

Controller Default Register Assignments

SCADAPack 4202/4203 DS I/O

A default Register Assignment is provided for the SCADAPack DS controllers (4202 or 4203 DS). The table below contains a convenient assignment for the 4202 DS and 4203 DS controllers.

To enable the default register assignment:

- Select **Type** from the **Controller** menu.

- Select the appropriate **SCADAPack DS** controller.
- Select **Register Assignment** from the **Controller** menu.

Select the **Default** button in the *Register Assignment* dialog.

The following table summarizes the Default Register Assignment for the SCADAPack 4203 or 4202 DS.

Module	Address	Default Register Assignment
SCADAPack 4202/4203 DS I/O	Fixed	
digital output		00001 to 00002
digital inputs		10001 to 10002
analog inputs		30001 to 30007

Related Topics

Controller Default Register Assignments

Micro16 Controller I/O Only

The Micro16 Controller I/O Module contains the Controller Digital Inputs and the Controller Interrupt Input Status sub modules.

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10001. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of one sequential digital input register. The default value is 10004. Start Register is the **INT** physical digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack (5601 I/O Module) Default Register Assignment

The SCADAPack with 5601 I/O module contains the Controller Digital Inputs and the Controller Interrupt Input Status sub modules and the SCADAPack 5601**[****]**I/O Module.

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10017. Start Register + 1 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of four sequential digital input registers. The default value is 10020. Start Register is the INT physical digital input data. (0 indicates OFF and 1 indicates ON).

5601 I/O Module Properties

Field	Description
Module Name	SCADAPack 5601 I/O Module
DO Start Register	First register of any unused block of twelve consecutive digital output registers. The default setting is 1.
DI Start Register	First register of any unused block of sixteen sequential digital input registers. The default value is 10001.
AI Start Register	First register of any unused block of eight consecutive analog input registers. Default value is 30001.

SCADAPack with 5604 I/O Module

The SCADAPack with 5604 I/O module contains the Controller Digital Inputs and the Controller Interrupt Input Status sub modules and the SCADAPack 5604 I/O Module**.**

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10017. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of one sequential digital input register. The default value is 10020. Start Register is the **INT** physical digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack 5604 Module Properties

Parameter	Description
Module Name	SCADAPack 5604 I/O Module
DO Start Register	First register of any unused block of thirty-six consecutive digital output registers. The default value is 1. DO Start Register to DO Start Register + 31 are physical digital output data.(0 indicates OFF and 1 indicates ON). DO Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DO Start Register +33 (ON = VLoop output ON; OFF VLoop output OFF) DO Start Register +34 and +35 set the analog input filters:

Filter Setting	Digital Output 34	Digital Output 35
< 3 Hz	OFF	OFF
6 Hz	OFF	ON
11 Hz	ON	OFF
30 Hz	ON	ON

DI Start Register	First register of any unused block of thirty-five consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +31 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DI Start Register +33 (ON = Over current detected; OFF Over current not detected) DI Start Register +34 (ON = One or more digital outputs mismatch; OFF No mismatch)
AI Start Register	First register of any unused block of ten consecutive analog input registers. The default value is 30001. AI Start Register to AI Start Register + 7 are physical analog input data. AI Start Register +8 (external analog input for battery monitoring; 0 to 32.768V) AI Start Register +9 (internal analog input for DC/DC converter monitoring)
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001. AO Start Register to AO Start Register + 1 are physical analog output data.

SCADAPack LIGHT

The SCADAPack Light default module contains the Controller Digital Inputs and the Controller Interrupt Input Status sub modules and the SCADAPack 5602 I/O Module.

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10006. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Interrupt Status
DI Start Register	First register of any unused block of one sequential digital input register. The default value is 10009. Start Register is the **INT** physical digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack 5602 Module Properties

Field	Description
Module Name	SCADAPack 5602 I/O Module
DO Start Register	First register of any unused block of two consecutive digital output registers. The default

	setting is 1.
DI Start Register	First register of any unused block of five sequential digital input registers. The default value is 10001.
AI Start Register	First register of any unused block of five consecutive analog input registers. Default value is 30001.

SCADAPack PLUS with 5601 I/O Module

The SCADAPack Plus with 5601 I/O module contains the Controller Digital Inputs and the Controller Interrupt Input Status sub modules theSCADAPack 5601 I/O Module and the SCADAPack 5602 I/O Module.

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10022. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of one sequential digital input register. The default value is 10025. Start Register is the **INT** physical digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack 5601 Module Properties

Field	Description
Module Name	SCADAPack 5601 I/O Module
DO Start Register	First register of any unused block of twelve consecutive digital output registers. The default setting is 1.
DI Start Register	First register of any unused block of sixteen sequential digital input registers. The default value is 10001.
AI Start Register	First register of any unused block of eight consecutive analog input registers. Default value is 30001.

SCADAPack 5602 Module Properties

Field	Description
Module Name	SCADAPack 5602 I/O Module
DO Start Register	First register of any unused block of two consecutive digital output registers. The default setting is 13.
DI Start Register	First register of any unused block of five sequential digital input registers. The default value is 10017.
AI Start Register	First register of any unused block of five consecutive analog input registers. Default value is 30009.

SCADAPack PLUS with 5604 I/O Module

The SCADAPack Plus with 5604 I/O module contains the Controller Digital Inputs and the Controller Interrupt Input Status sub modules the SCADAPack 5604 I/O Module and the SCADAPack 5602 I/O Module.

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10041. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of one sequential digital input register. The default value is 10044. Start Register is the **INT** physical digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack 5604 I/O Module Properties

Parameter	Description
Module Name	SCADAPack 5604 I/O Module

DO Start Register	First register of any unused block of thirty-six consecutive digital output registers. The default value is 1. DO Start Register to DO Start Register + 31 are physical digital output data.(0 indicates OFF and 1 indicates ON). DO Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DO Start Register +33 (ON = VLoop output ON; OFF VLoop output OFF) DO Start Register +34 and +35 set the analog input filters:

Filter Setting	Digital Output 34	Digital Output 35
< 3 Hz	OFF	OFF
6 Hz	OFF	ON
11 Hz	ON	OFF
30 Hz	ON	ON

DI Start Register	First register of any unused block of thirty-five consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +31 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DI Start Register +33 (ON = Over current detected; OFF Over current not detected) DI Start Register +34 (ON = One or more digital outputs mismatch; OFF No mismatch)
AI Start Register	First register of any unused block of ten consecutive analog input registers. The default value is 30001. AI Start Register to DO Start Register + 7 are physical analog input data. AI Start Register +8 (external analog input for battery monitoring; 0 to 32.768V) AI Start Register +9 (internal analog input for DC/DC converter monitoring)
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001. AO Start Register to AO Start Register + 1 are physical analog output data.

SCADAPack 5602 I/O Module Properties

Field	Description
Module Name	SCADAPack 5602 I/O Module
DO Start Register	First register of any unused block of two consecutive digital output registers. The default setting is 37.
DI Start Register	First register of any unused block of five sequential digital input registers. The default value is 10036.
AI Start Register	First register of any unused block of five consecutive analog input registers. Default value is 30011.

SCADAPack PLUS with 5606 I/O Module

The SCADAPack Plus with 5604 I/O module contains the the Controller Digital Inputs and the Controller Interrupt Input Status sub modules the SCADAPack 5606 I/O Module and the SCADAPack 5602 I/O Module.

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10046. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of one sequential digital input register. The default value is 10049. Start Register is the **INT** physical digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack 5606 I/O Module Properties

Parameter	Description
Module Name	SCADAPack 5606 I/O Module
Module Address	The address of the module is selected using dip switches on the module. A maximum of eight SCADAPack 5606 I/O type modules may be added to a system. Default value: 0
DO Start Register	First register of any unused block of sixteen consecutive digital output registers. The default value is 1.
DI Start Register	First register of any unused block of forty consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +31 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +32 (OFF = AI1 is OK; ON AI1 is over or under range) DI Start Register +33 (OFF = AI2 is OK; ON AI2 is over or under range) DI Start Register +34 (OFF = AI3 is OK; ON AI3 is over or under range) DI Start Register +35 (OFF = AI4 is OK; ON AI4 is over or under range) DI Start Register +36 (OFF = AI5 is OK; ON AI5 is over or under range) DI Start Register +37 (OFF = AI6 is OK; ON AI6 is over or under range) DI Start Register +38 (OFF = AI7 is OK; ON AI7 is over or under range) DI Start Register +39 (OFF = AI8 is OK; ON AI8 is over or under range)

AI Start Register	First register of any unused block of eight consecutive analog input registers. The default value is 30001.
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.
AI*n* Channel Type	Defines the input measurement type for each input (AI 0 through 7). • The 0 to 5V selection sets the input type to measure 0 to 5V input signals. • The 1 to 5V selection sets the input type to measure 1 to 5V input signals. • The 0 to 20 mA selection sets the input type to measure 0 to 20mA input signals. • The 4 to 20 mA selection sets the input type to measure 4 to 20mA input signals.
Module Filter	Sets the input filter rate for all analog inputs. The filter rate is used to dampen process variations or noise. • The 3 Hz filter selection sets the response time to 155ms at 60Hz and 185ms at 50Hz. • The 6 Hz filter selection sets response time to 85ms at 60Hz and 85ms at 50Hz. • The 11 Hz filter selection sets response time to 45ms at 60Hz and 55ms at 50Hz. • The 30 Hz filter selection sets response time to 30ms at 60Hz and 30ms at 50Hz.
Module Scan Frequency	Sets the input scan rate for all analog inputs. The scan rate selection is not critical but AC noise rejection is improved at the correct frequency. If the module is used in a DC environment, the 60 Hz setting will yield slightly faster response time. • The 60 Hz selection synchronies the input scanning to 60Hz. • The 50 Hz selection synchronies the input scanning to 50Hz.
AO Channel Type	Defines the output signal type for the analog output. • The 0-20 mA selection sets the output type for 0 to 20mA signals. • The 4-20 mA selection sets the output type for 4 to 20mA signals.

SCADAPack 5602 I/O Module Properties

Field	Description
Module Name	SCADAPack 5602 I/O Module
DO Start Register	First register of any unused block of two consecutive digital output registers. The default setting is 17.
DI Start Register	First register of any unused block of five sequential digital input registers. The default value is 10041.
AI Start Register	First register of any unused block of five consecutive analog input registers. Default value is 30009.

SCADAPack LP

The **SCADAPack LP Default Module** is comprised of four different types of I/O channels. The SCADAPack LP I/O hardware contains eight analog inputs, two analog outputs, sixteen digital inputs, twelve digital outputs, and three counter inputs. In addition, the Controller Counter Inputs module is used.

The **digital input** assignment provides sixteen I/O channels for the digital input data. The digital input registers are updated continuously with data read from the digital inputs.

- The SCADAPack LP provides eight universal digital inputs or outputs. The inputs are for use with dry contacts such as switches and relay contacts. These are defined as digital input channels 0 to 7.
- Digital input channel 8 returns the Com1 (RS-485) power status.
- Digital input channel 9 returns the Com3 (HMI) power output status.
- Digital input channel 10 returns the VLOOP output status.
- Digital input channel 11 returns the dc/dc converter status. This bit reports the true status of the DC/DC converter. If over-current causes the converter to be turned off, this bit will clear.
- Digital input channel 12 returns the VLOOP over-current status. This indicates VLOOP over-current has been detected. This input clears when VLOOP output is off, or the over-current condition clears.
- Digital input channel 13 returns the Digital Output Mismatch status.
- Digital input channel 14 returns the SCADAPack Vision power.

The **digital output** assignment provides twelve I/O channels for the digital outputs of the SCADAPack LP. The digital outputs are updated continuously with data read from the coil registers.

- The SCADAPack LP provides eight universal digital inputs or outputs. Outputs are open-collector/open drain type. These are defined as digital output channels 0 to 7.
- Digital output channel 8 is not used. This is for internal use only.
- Digital output channel 9 is used to control com3 (HMI) power.
- Digital output channel 10 is used to control the VLoop power supply.
- Digital output channel 11 is used to control the dc/dc converter.

The **analog input** assignment provides seven I/O channels for the analog input data. Analog input data is a signed value in the range –32768 to 32767.

- Analog input channels 0 to 4 provide eight external analog inputs (0-10V or 0-40mA)
- Analog input channel 5 provides an external analog input for battery monitoring (0 to 32.768V)
- Analog input channel 6 provides an internal analog input for dc/dc converter monitoring.

The **analog output** assignment provides two I/O channels for the analog output data. Analog input data is a signed value in the range –32768 to 32767.

SCADAPack LP Module Properties

Parameter	Description
Module Name	SCADAPack SCADAPack LP I/O Module
DO Start Register	First register of any unused block of twelve consecutive digital output registers. The default value is 1. DO Start Register to DO Start Register + 7 are physical digital output data. (On=output transistor on; OFF=output transistor off) DO Start Register +8 (ON = com1 RS-485 power control always on;OFF= power control on form transmission plus 90 seconds) DO Start Register +9 (ON = com3 HMI power on; OFF = com3 HMI power off). DO Start Register +10 (ON = VLoop output on; OFF= VLoop output off) DC/DC converter needs to be on. DO Start Register +11 (ON = DC/DC converter on; OFF = DC/DC converter off).
DI Start Register	First register of any unused block of sixteen consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +7 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +8 (ON = Com1 RS482 power on; OFF= Com1 RS482 power off). DI Start Register +9 (ON = Com3 HMI power on; OFF = Com3 HMI power off). DI Start Register +10 (ON = VLOOP Output status on; OFF = VLOOP Output status off). DI Start Register +11 (ON = dc/dc converter on; OFF = DC/DC converter off). DI Start Register +12 (ON = VLOOP Over current; OFF = No VLOOP Over current). DI Start Register +13 (ON = One or more digital outputs mismatch; OFF = No mismatch). DI Start Register +14 (ON = Power to SCADAPack Vision; OFF = No power to SCADAPack Vision).
AI Start Register	First register of any unused block of seven consecutive analog input registers. The default value is 30001. AI Start Register to AI Start Register + 4 are physical analog input data. AI Start Register +5 (external analog input for battery monitoring; 0 to 32.768V) AI Start Register +6 (internal analog input for dc/dc converter monitoring)
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.

Controller Counter Input Module Properties

Field	Description
Module Name	Controller Counter Inputs
AI Start Register	First register of any unused blocks of 6 consecutive input registers. Default register: 30001. Start Register to Start Register +1 Controller counter input 0 (32-bit register) Start Register + 2 to Start Register +3 Controller counter input 1 (32-bit register) Start Register + 4 to Start Register +5 Controller counter input 2 (32-bit register)

SCADAPack 100

The **SCADAPack 100 Default Module** is comprised of three different types of I/O channels. The SCADAPack 100 I/O hardware contains four analog/counter inputs, six digital inputs, and six digital outputs.

The **analog input** assignment provides eight I/O channels for the analog input data. Analog input data is a signed value in the range –32768 to 32767. Counter data is a 32 bit integer assigned to two registers.

- Analog input channels 0 to 3 provide four external analog inputs.
- Analog input channel 4 provides an internal analog input for controller board temperature in degrees C.
- Analog input channel 5 provides an internal analog input for RAM battery voltage in mV.
- Analog input channels 6 and 7 provide an external counter input (32 bit counter).

The **digital input** assignment provides six I/O channels for the digital input data. The digital input registers are updated continuously with data read from the digital inputs. The inputs are for use with dry contacts such as switches and relay contacts. These are defined as digital input channels 0 to 5.

The **digital output** assignment provides six I/O channels for the digital outputs of the SCADAPack LP. The digital outputs are updated continuously with data read from the coil registers. Outputs are open-collector/open drain type. These are defined as digital output channels 0 to 5.

SCADAPack 100 Module Properties

Parameter	Description
Module Name	SCADAPack SCADAPack LP I/O Module
DO Start Register	First register of any unused block of six consecutive digital output registers. The default value is 1. DO Start Register to DO Start Register + 5 are physical digital output data.
DI Start Register	First register of any unused block of six consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +5 are physical digital input data.
AI Start Register	First register of any unused block of eight consecutive analog input registers. The default value is 30001. AI Start Register to AI Start Register + 3 are physical analog input data. AI Start Register +4 (internal analog input for controller board temperature in degrees C) AI Start Register +5 (internal analog input for RAM battery voltage in mV) AI Start Register + 6 and Start Register + 7 (controller counter input 0, 32-bit register)

SCADAPack 32 with 5601 I/O Module

The SCADAPack 32 with 5601 I/O module contains the the Controller Digital Inputs and the Controller Interrupt Input Status sub modules and the SCADAPack 5601 I/O Module.

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10017. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates

ON).
Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON).
Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of one sequential digital input register. The default value is 10020. Start Register is the **INT** physical digital input data. (0 indicates OFF and 1 indicates ON).

5601 I/O Module Properties

Field	Description
Module Name	SCADAPack 5601 I/O Module
DO Start Register	First register of any unused block of twelve consecutive digital output registers. The default setting is 1.
DI Start Register	First register of any unused block of sixteen sequential digital input registers. The default value is 10001.
AI Start Register	First register of any unused block of eight consecutive analog input registers. Default value is 30001.

SCADAPack 32 with 5604 10V/40mA I/O Module

The SCADAPack 32 with 5604 I/O module contains the the Controller Digital Inputs and the Controller Interrupt Input Status sub modules and the SCADAPack 5604 10V/40mA I/O Module.

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10017. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Interrupt Status Inputs
DI Start Register	First register of any unused block of four sequential digital input registers. The default value is 10020. Start Register is the **INT** physical digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack 5604 Module Properties

Parameter	Description
Module Name	SCADAPack 5604 I/O Module
DO Start Register	First register of any unused block of thirty-six consecutive digital output registers. The default value is 1. DO Start Register to DO Start Register + 31 are physical digital output data.(0 indicates OFF and 1 indicates ON). DO Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DO Start Register +33 (ON = VLoop output ON; OFF VLoop output OFF) DO Start Register +34 and +35 set the analog input filters:

Filter Setting	Digital Output 34	Digital Output 35
< 3 Hz	OFF	OFF
6 Hz	OFF	ON
11 Hz	ON	OFF
30 Hz	ON	ON

DI Start Register	First register of any unused block of thirty-five consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +31 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DI Start Register +33 (ON = Over current detected; OFF Over current not detected) DI Start Register +34 (ON = One or more digital outputs mismatch; OFF No mismatch)
AI Start Register	First register of any unused block of ten consecutive analog input registers. The default value is 30001. AI Start Register to AI Start Register + 7 are physical analog input data. AI Start Register +8 (external analog input for battery monitoring; 0 to 32.768V) AI Start Register +9 (internal analog input for DC/DC converter monitoring)
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.

SCADAPack 32 with 5604 10V/40mA I/O Module

The SCADAPack 32 with 5604 I/O module contains the the Controller Digital Inputs and the Controller Interrupt Input Status sub modules and the SCADAPack 5604 5V/20mA I/O Module.

Controller Digital Inputs Module Properties

Field	Description

Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10017. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Status Digital Inputs
DI Start Register	First register of any unused block of one sequential digital input register. The default value is 10020. Start Register is the INT physical digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack 5604 Module Properties

Parameter	Description
Module Name	SCADAPack 5604 I/O Module
DO Start Register	First register of any unused block of thirty-six consecutive digital output registers. The default value is 1. DO Start Register to DO Start Register + 31 are physical digital output data.(0 indicates OFF and 1 indicates ON). DO Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DO Start Register +33 (ON = VLoop output ON; OFF VLoop output OFF) DO Start Register +34 and +35 set the analog input filters: Filter Setting / Digital Output 34 / Digital Output 35 < 3 Hz / OFF / OFF 6 Hz / OFF / ON 11 Hz / ON / OFF 30 Hz / ON / ON
DI Start Register	First register of any unused block of thirty-five consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +31 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DI Start Register +33 (ON = Over current detected; OFF Over current not detected) DI Start Register +34 (ON = One or more digital outputs mismatch; OFF No mismatch)
AI Start Register	First register of any unused block of ten consecutive analog input registers. The default value is 30001. AI Start Register to AI Start Register + 7 are physical analog input data. AI Start Register +8 (external analog input for battery monitoring; 0 to 32.768V) AI Start Register +9 (internal analog input for DC/DC converter monitoring)
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.

SCADAPack 32 with 5606 I/O Module

The SCADAPack 32 with 5606 I/O module contains the the Controller Digital Inputs and the Controller Interrupt Input Status sub modules and the SCADAPack 5606 I/O Module.

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10017. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Status Digital Inputs
DI Start Register	First register of any unused block of one sequential digital input register. The default value is 10020. Start Register is the **INT** physical digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack 5606 I/O Module Properties

Parameter	Description
Module Name	SCADAPack 5606 I/O Module
Module Address	The address of the module is selected using dip switches on the module. A maximum of eight SCADAPack 5606 I/O type modules may be added to a system. Default value: 0
DO Start Register	First register of any unused block of sixteen consecutive digital output registers. The default value is 1.
DI Start Register	First register of any unused block of forty consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +31 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +32 (OFF = AI1 is OK; ON AI1 is over or under range) DI Start Register +33 (OFF = AI2 is OK; ON AI2 is over or under range)

	DI Start Register +34 (OFF = AI3 is OK; ON AI3 is over or under range) DI Start Register +35 (OFF = AI4 is OK; ON AI4 is over or under range) DI Start Register +36 (OFF = AI5 is OK; ON AI5 is over or under range) DI Start Register +37 (OFF = AI6 is OK; ON AI6 is over or under range) DI Start Register +38 (OFF = AI7 is OK; ON AI7 is over or under range) DI Start Register +39 (OFF = AI8 is OK; ON AI8 is over or under range)
AI Start Register	First register of any unused block of eight consecutive analog input registers. The default value is 30001.
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.
AI*n* Channel Type	Defines the input measurement type for each input (AI 0 through 7). • The 0 to 5V selection sets the input type to measure 0 to 5V input signals. • The 1 to 5V selection sets the input type to measure 1 to 5V input signals. • The 0 to 20 mA selection sets the input type to measure 0 to 20mA input signals. • The 4 to 20 mA selection sets the input type to measure 4 to 20mA input signals.
Module Filter	Sets the input filter rate for all analog inputs. The filter rate is used to dampen process variations or noise. • The 3 Hz filter selection sets the response time to 155ms at 60Hz and 185ms at 50Hz. • The 6 Hz filter selection sets response time to 85ms at 60Hz and 85ms at 50Hz. • The 11 Hz filter selection sets response time to 45ms at 60Hz and 55ms at 50Hz. • The 30 Hz filter selection sets response time to 30ms at 60Hz and 30ms at 50Hz.
Module Scan Frequency	Sets the input scan rate for all analog inputs. The scan rate selection is not critical but ac noise rejection is improved at the correct frequency. If the module is used in a dc environment, the 60 Hz setting will yield slightly faster response time. • The 60 Hz selection synchronies the input scanning to 60Hz. • The 50 Hz selection synchronies the input scanning to 50Hz.
AO Channel Type	Defines the output signal type for the analog output. • The 0-20 mA selection sets the output type for 0 to 20mA signals. • The 4-20 mA selection sets the output type for 4 to 20mA signals.

SCADAPack 32P Default Module

The SCADAPack 32P Controller I/O Module contains the Controller Digital Inputs and the Controller Interrupt Input Status sub modules.

Controller Digital Inputs Module Properties

Field	Description
Module Name	Controller Digital Inputs
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10001. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module Properties

Field	Description
Module Name	Controller Status Digital Inputs
DI Start Register	First register of any unused block of one sequential digital input register. The default value is 10004. Start Register is the **INT** physical digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack 314 Default Modules

The SCADAPack 314 Default Module contains the the SCADAPack 314/33x Default Module and the SCADAPack 5607 I/O Module.

SCADAPack 314/33x Module Properties

Note that the SCADAPack 314 does not use the internal DI and DO registers.

Parameter	Description
Module Name	SCADAPack 314 I/O Module
Counter Start Register	First register of any unused block of six consecutive counter input registers. The default value is 30009. Counter Start Register to Counter Start Register + 1 (controller counter input 0, 32-bit register). Counter Start Register + 2 to Counter Start Register + 3 (controller counter input 1, 32-bit register). Counter Start Register +4 to Counter Start Register + 5 (controller counter input 2, 32-bit register).

SCADAPack 5607 I/O Module Properties

Parameter	Description
Module Name	SCADAPack 5607 I/O Module

Module Address	The address of the module is selected using dip switches on the module. A maximum of eight SCADAPack 5607 I/O type modules may be added to a system. Default value: 0
DO Start Register	First register of any unused block of ten consecutive digital output registers. The default value is 1.
DI Start Register	First register of any unused block of twenty four consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +15 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +16 (OFF = AI1 is OK; ON AI1 is over or under range) DI Start Register +17 (OFF = AI2 is OK; ON AI2 is over or under range) DI Start Register +18 (OFF = AI3 is OK; ON AI3 is over or under range) DI Start Register +19 (OFF = AI4 is OK; ON AI4 is over or under range) DI Start Register +20 (OFF = AI5 is OK; ON AI5 is over or under range) DI Start Register +21 (OFF = AI6 is OK; ON AI6 is over or under range) DI Start Register +22 (OFF = AI7 is OK; ON AI7 is over or under range) DI Start Register +23 (OFF = AI8 is OK; ON AI8 is over or under range)
AI Start Register	First register of any unused block of eight consecutive analog input registers. The default value is 30001.
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.
AI*n* Channel Type	Defines the input measurement type for each input (AI 0 through 7). • The 0 to 5V selection sets the input type to measure 0 to 5V input signals. • The 1 to 5V selection sets the input type to measure 1 to 5V input signals. • The 0 to 20 mA selection sets the input type to measure 0 to 20mA input signals. • The 4 to 20 mA selection sets the input type to measure 4 to 20mA input signals.
Module Filter	Sets the input filter rate for all analog inputs. The filter rate is used to dampen process variations or noise. • The 3 Hz filter selection sets the response time to 155ms at 60Hz and 185ms at 50Hz. • The 6 Hz filter selection sets response time to 85ms at 60Hz and 85ms at 50Hz. • The 11 Hz filter selection sets response time to 45ms at 60Hz and 55ms at 50Hz. • The 30 Hz filter selection sets response time to 30ms at 60Hz and 30ms at 50Hz.
Module Scan Frequency	Sets the input scan rate for all analog inputs. The scan rate selection is not critical but AC noise rejection is improved at the correct frequency. If the module is used in a DC environment, the 60 Hz setting will yield slightly faster response time. • The 60 Hz selection synchronies the input scanning to 60Hz. • The 50 Hz selection synchronies the input scanning to 50Hz.
AO Channel Type	Defines the output signal type for the analog output. • The 0-20 mA selection sets the output type for 0 to 20mA signals. • The 4-20 mA selection sets the output type for 4 to 20mA signals.

SCADAPack 314/33x Default Module

The **SCADAPack 314/33x Default Module** is comprised of three different types of I/O channels. The SCADAPack 314/33x I/O hardware contains three counter inputs, one internal digital input, and two internal digital outputs.

The **counter input** assignment provides three I/O channels for the analog input data. The counter input registers are updated continuously with data read from the counter inputs.

- Counter input channel 0 is for counter input 0. This is a 32-bit value with automatic rollover.
- Counter input channel 1 is for counter input 1. This is a 32-bit value with automatic rollover.
- Counter input channel 2 is for counter input 2. This is a 32-bit value with automatic rollover.

The **digital input** assignment provides one internal I/O channel for the digital input data. The digital input registers are updated continuously with data read from the digital inputs.

- Digital input channel 0 is the status of COM3 HMI power.

  0 = 5V at pin 1 of connector P7 (COM3) is off.

  1 = 5V at pin 1 of connector P7 (COM3) is on.

The **digital output** assignment provides two I/O channels for the digital outputs of the SCADAPack 330 I/O hardware. The digital outputs are updated continuously with data read from the coil registers.

- Digital output channel 0 controls the USB STAT led.

  0 = USB STAT led off.

  1 = USB STAT led on.

- Digital output channel 1 controls the COM3 HMI power.

  0 = 5V at pin 1 of connector P7 (COM3) is off.

  1 = 5V at pin 1 of connector P7 (COM3) is on.

SCADAPack 330 Module Properties

Parameter	Description
Module Name	SCADAPack SCADAPack LP I/O Module
DO Start Register	First register of any unused block of two consecutive digital output registers. The default value is 1.

	DO Start Register (ON = USB status led is ON; OFF = USB status led is OFF)
	DO Start Register + 1 (ON = 5V at pin 1 of Com3 is ON; OFF = 5V at pin 1 of Com3 is OFF)
DI Start Register	First register of any unused block of six consecutive digital input registers. The default value is 10001.
	DI Start Register (ON = 5V at pin 1 of Com3 is ON; OFF = 5V at pin 1 of Com3 is OFF)
AI Start Register (Counter Inputs)	First register of any unused block of six consecutive analog input registers. The default value is 30001.
	AI Start Register to AI Start Register + 1 (controller counter input 0, 32-bit register).
	AI Start Register + 2 to AI Start Register + 3 (controller counter input 1, 32-bit register).
	AI Start Register +4 to AI Start Register + 5 (controller counter input 2, 32-bit register).

SCADAPack 334 Default Modules

The SCADAPack 334 Default Module contains the the SCADAPack 314/33x Default Module and the SCADAPack 5607 I/O Module.

SCADAPack 314/33x Module Properties

Parameter	Description
Module Name	SCADAPack 334 I/O Module
DO Start Register	First register of any unused block of two consecutive digital output registers. The default value is 11.
	DO Start Register (ON = USB status led is ON; OFF = USB status led is OFF)
	DO Start Register + 1 (ON = 5V at pin 1 of Com3 is ON; OFF = 5V at pin 1 of Com3 is OFF)
DI Start Register	First register of any unused block of six consecutive digital input registers. The default value is 10025.
	DI Start Register (ON = 5V at pin 1 of Com3 is ON; OFF = 5V at pin 1 of Com3 is OFF)
AI Start Register (Counter Inputs)	First register of any unused block of six consecutive analog input registers. The default value is 30009.
	AI Start Register to AI Start Register + 1 (controller counter input 0, 32-bit register).
	AI Start Register + 2 to AI Start Register + 3 (controller counter input 1, 32-bit register).
	AI Start Register +4 to AI Start Register + 5 (controller counter input 2, 32-bit register).

SCADAPack 5607 I/O Module Properties

Parameter	Description
Module Name	SCADAPack 5607 I/O Module
Module Address	The address of the module is selected using dip switches on the module. A maximum of eight SCADAPack 5607 I/O type modules may be added to a system. Default value: 0
DO Start Register	First register of any unused block of ten consecutive digital output registers. The default value is 1.
DI Start Register	First register of any unused block of twenty four consecutive digital input registers. The default value is 10001.
	DI Start Register to DI Start Register +15 are physical digital input data. (0 indicates OFF and 1 indicates ON).
	DI Start Register +16 (OFF = AI1 is OK; ON AI1 is over or under range)
	DI Start Register +17 (OFF = AI2 is OK; ON AI2 is over or under range)
	DI Start Register +18 (OFF = AI3 is OK; ON AI3 is over or under range)
	DI Start Register +19 (OFF = AI4 is OK; ON AI4 is over or under range)
	DI Start Register +20 (OFF = AI5 is OK; ON AI5 is over or under range)
	DI Start Register +21 (OFF = AI6 is OK; ON AI6 is over or under range)
	DI Start Register +22 (OFF = AI7 is OK; ON AI7 is over or under range)
	DI Start Register +23 (OFF = AI8 is OK; ON AI8 is over or under range)
AI Start Register	First register of any unused block of eight consecutive analog input registers. The default value is 30001.
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.
AI*n* Channel Type	Defines the input measurement type for each input (AI 0 through 7).
	• The 0 to 5V selection sets the input type to measure 0 to 5V input signals.
	• The 1 to 5V selection sets the input type to measure 1 to 5V input signals.
	• The 0 to 20 mA selection sets the input type to measure 0 to 20mA input signals.
	• The 4 to 20 mA selection sets the input type to measure 4 to 20mA input signals.
Module Filter	Sets the input filter rate for all analog inputs. The filter rate is used to dampen process variations or noise.
	• The 3 Hz filter selection sets the response time to 155ms at 60Hz and 185ms at 50Hz.
	• The 6 Hz filter selection sets response time to 85ms at 60Hz and 85ms at 50Hz.
	• The 11 Hz filter selection sets response time to 45ms at 60Hz and 55ms at 50Hz.
	• The 30 Hz filter selection sets response time to 30ms at 60Hz and 30ms at 50Hz.
Module Scan Frequency	Sets the input scan rate for all analog inputs. The scan rate selection is not critical but AC noise rejection is improved at the correct frequency. If the module is used in a DC environment, the 60 Hz setting will yield slightly faster response time.
	• The 60 Hz selection synchronies the input scanning to 60Hz.
	• The 50 Hz selection synchronies the input scanning to 50Hz.
AO Channel Type	Defines the output signal type for the analog output.
	• The 0-20 mA selection sets the output type for 0 to 20mA signals.

- The 4-20 mA selection sets the output type for 4 to 20mA signals.

SCADAPack 350 5V/20mA I/O

The **SCADAPack 350 5V/20mA I/O** is comprised of four different types of I/O channels. The SCADAPack 350 hardware contains eight analog inputs, two analog outputs, sixteen digital inputs, twelve digital outputs and three counter inputs. In addition the Controller Counter Inputs module is used.

The difference between the SCADAPack 350 10V/40mA I/O module and the SCADAPack 350 10V/40mA I/O module is how the A/D values are returned from the module. The following table displays the A/D output for both modules using the same current or voltage input. The SCADAPack 350 5V/20mA I/O module provides the same A/D values as the SCADAPack Integrated 5606 I/O module. Together these two modules comprise the SCADAPack 357 controller.

The selection for the analog input type, current or voltage, is set using jumpers on the SCADAPack 350 (see the SCADAPack 350 Hardware Manual for further information).

Current Channel 0-4	Voltage Channel 0-4	Voltage Channel 5	A/D Output 10V/40mA	A/D Output 5V/20mA
0mA	0V	0V	0	0
1.22µA	0.305mV	0.001V	1	1
4mA	1.0V	3.277V	3277	6554
10mA	2.5V	8.192V	8192	16384
20mA	5.0V	16.384V	16384	32767
39.999mA	9.9997V	32.767V	32767	NA

The **digital input** assignment provides sixteen I/O channels for the digital input data. The digital input registers are updated continuously with data read from the digital inputs.

- The SCADAPack 350 provides eight universal digital inputs or outputs. The inputs are for use with dry contacts such as switches and relay contacts. These are defined as digital input channels 0 to 7.
- Digital input channel 8 returns the VLOOP output status.
- Digital input channel 9 returns the DC/DC converter status. This bit reports the true status of the DC/DC converter. If over-current causes the converter to be turned off, this bit will clear.
- Digital input channel 10 returns the VLOOP over-current status. This indicates VLOOP over-current has been detected. This input clears when VLOOP output is off, or the over-current condition clears.
- Digital input channel 11 returns the Digital Output Mismatch status.
- Digital input channel 12 returns the Com3 (HMI) power output status.

The **digital output** assignment provides eleven I/O channels for the digital outputs of the SCADAPack 350. The digital outputs are updated continuously with data read from the coil registers.

- The SCADAPack 350 provides eight universal digital inputs or outputs. Outputs are open-collector/open drain type. These are defined as digital output channels 0 to 7.
- Digital output channel 8 is used to control the VLoop power supply.
- Digital output channel 9 is used to control the dc/dc converter.
- Digital output channel 10 used to control com3 (HMI) power.

The **analog input** assignment provides seven I/O channels for the analog input data. Analog input data is a signed value in the range –32768 to 32767.

- Analog input channels 0 to 4 provide eight external analog inputs (0-5V or 0-20mA)
- Analog input channel 5 provides an external analog input for battery monitoring (0 to 32.768V)
- Analog input channel 6 provides an internal analog input for DC/DC converter monitoring.

The **analog output** assignment provides two I/O channels for the analog output data. Analog input data is a signed value in the range –32768 to 32767.

SCADAPack 350 Module Properties

Parameter	Description
Module Name	SCADAPack SCADAPack 350 I/O Module
DO Start Register	First register of any unused block of eleven consecutive digital output registers. The default value is 1. DO Start Register to DO Start Register + 7 are physical digital output data. DO Start Register +8 (ON = VLoop output ON; OFF VLoop output OFF) DO Start Register +9 (ON = DC/DC converter ON; OFF = DC/DC converter OFF). DO Start Register +10 (ON = com3 (HMI) power ON; OFF = com3 (HMI) power OFF).
DI Start Register	First register of any unused block of sixteen consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +7 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +8 (ON = VLOOP Output status ON; OFF = VLOOP Output status OFF). DI Start Register +9 (ON = DC/DC converter ON; OFF = DC/DC converter OFF). DI Start Register +10 (ON = VLOOP Over current; OFF = No VLOOP Over current). DI Start Register +11 (ON = One or more digital outputs mismatch; OFF = No mismatch). DI Start Register +12 (ON = Com3 HMI power ON; OFF = Com3 HMI power OFF).
AI Start Register	First register of any unused block of seven consecutive analog input registers. The default value is 30001. AI Start Register to AI Start Register + 4 are physical analog input data. AI Start Register +5 (external analog input for battery monitoring; 0 to 32.768V) AI Start Register +6 (internal analog input for DC/DC converter monitoring) AI Start Register +7 (used internally by the SCADAPack 350)
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.

Controller Counter Input Module Properties

Field	Description
Module Name	Controller Counter Inputs

AI Start Register	First register of any unused blocks of 6 consecutive input registers. Default register: 30009. Start Register to Start Register +1 Controller counter input 0 (32-bit register) Start Register + 2 to Start Register +3 Controller counter input 1 (32-bit register) Start Register + 4 to Start Register +5 Controller counter input 2 (32-bit register)

SCADAPack 350 10V/40mA I/O

The **SCADAPack 350 10V/40mA I/O** is comprised of four different types of I/O channels. The SCADAPack 350 I/O hardware contains eight analog inputs, two analog outputs, sixteen digital inputs, twelve digital outputs and three counter inputs. In addition the Controller Counter Inputs module is used.

The difference between the SCADAPack 350 10V/40mA I/O module and the SCADAPack 350 5V/20mA I/O module is how the A/D values are returned from the module. The following table displays the A/D output for both modules using the same current or voltage input. The SCADAPack 350 5V/20mA I/O module provides the same A/D values as the SCADAPack Integrated 5606 I/O module. Together these two modules comprise the SCADAPack 357 controller.

The selection for the analog input type, current or voltage, is set using jumpers on the SCADAPack 350 (see the SCADAPack 350 Hardware Manual for further information).

Current Channel 0-4	Voltage Channel 0-4	Voltage Channel 5	A/D Output 10V/40mA	A/D Output 5V/20mA
0mA	0V	0V	0	0
1.22µA	0.305mV	0.001V	1	1
4mA	1.0V	3.277V	3277	6554
10mA	2.5V	8.192V	8192	16384
20mA	5.0V	16.384V	16384	32767
39.999mA	9.9997V	32.767V	32767	NA

The **digital input** assignment provides sixteen I/O channels for the digital input data. The digital input registers are updated continuously with data read from the digital inputs.

- The SCADAPack 350 provides eight universal digital inputs or outputs. The inputs are for use with dry contacts such as switches and relay contacts. These are defined as digital input channels 0 to 7.
- Digital input channel 8 returns the VLOOP output status.
- Digital input channel 9 returns the DC/DC converter status. This bit reports the true status of the DC/DC converter. If over-current causes the converter to be turned off, this bit will clear.
- Digital input channel 10 returns the VLOOP over-current status. This indicates VLOOP over-current has been detected. This input clears when VLOOP output is off, or the over-current condition clears.
- Digital input channel 11 returns the Digital Output Mismatch status.
- Digital input channel 12 returns the Com3 (HMI) power output status.

The **digital output** assignment provides eleven I/O channels for the digital outputs of the SCADAPack 350. The digital outputs are updated continuously with data read from the coil registers.

- The SCADAPack 350 provides eight universal digital inputs or outputs. Outputs are open-collector/open drain type. These are defined as digital output channels 0 to 7.
- Digital output channel 8 is used to control the VLoop power supply.
- Digital output channel 9 is used to control the dc/dc converter.
- Digital output channel 10 used to control com3 (HMI) power.

The **analog input** assignment provides seven I/O channels for the analog input data. Analog input data is a signed value in the range –32768 to 32767.

- Analog input channels 0 to 4 provide eight external analog inputs (0-10V or 0-40mA)
- Analog input channel 5 provides an external analog input for battery monitoring (0 to 32.768V)
- Analog input channel 6 provides an internal analog input for DC/DC converter monitoring.

The **analog output** assignment provides two I/O channels for the analog output data. Analog input data is a signed value in the range –32768 to 32767.

SCADAPack 350 Module Properties

Parameter	Description
Module Name	SCADAPack SCADAPack 350 I/O Module
DO Start Register	First register of any unused block of eleven consecutive digital output registers. The default value is 1. DO Start Register to DO Start Register + 7 are physical digital output data. DO Start Register +8 (ON = VLoop output ON; OFF VLoop output OFF) DO Start Register +9 (ON = DC/DC converter ON; OFF = DC/DC converter OFF). DO Start Register +10 (ON = com3 (HMI) power ON; OFF = com3 (HMI) power OFF).
DI Start Register	First register of any unused block of sixteen consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +7 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +8 (ON = VLOOP Output status ON; OFF = VLOOP Output status OFF). DI Start Register +9 (ON = DC/DC converter ON; OFF = DC/DC converter OFF). DI Start Register +10 (ON = VLOOP Over current; OFF = No VLOOP Over current). DI Start Register +11 (ON = One or more digital outputs mismatch; OFF = No mismatch). DI Start Register +12 (ON = Com3 HMI power ON; OFF = Com3 HMI power OFF).
AI Start Register	First register of any unused block of seven consecutive analog input registers. The default value is 30001. AI Start Register to AI Start Register + 4 are physical analog input data. AI Start Register +5 (external analog input for battery monitoring; 0 to 32.768V) AI Start Register +6 (internal analog input for DC/DC converter monitoring)

	AI Start Register +7 (used internally by the SCADAPack 350)
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.

Controller Counter Input Module Properties

Field	Description
Module Name	Controller Counter Inputs
AI Start Register	First register of any unused blocks of 6 consecutive input registers. Default register: 30009. Start Register to Start Register +1 Controller counter input 0 (32-bit register) Start Register + 2 to Start Register +3 Controller counter input 1 (32-bit register) Start Register + 4 to Start Register +5 Controller counter input 2 (32-bit register)

SCADAPack 4202 DR I/O

The **SCADAPack 4202 DR Default m**odule is comprised of three different types of I/O channels. The SCADAPack 4202 DR hardware contains one internal analog input, one analog output, two counter inputs, and one digital input.

The **analog output** data is assigned to a single analog output register. The analog output is updated continuously with the data read from the holding register.

The **analog and counter** input data is assigned to five consecutive input registers. One input register is used to monitor the controller input power and two counter input double registers (32-bit). These input registers are updated continuously with data read from the analog inputs and counter input.

Digital **input data** is assigned to a single status register. The status register is updated continuously with data read from the digital input.

Module Properties

Parameter	Description
Module Name	4202 DR I/O
DI Start Register	First register of any unused block of one consecutive digital input register. The default value is 10001. DI Start Register is physical digital input data. (0 indicates OFF and 1 indicates ON).
AI Start Register	First register of any unused block of five consecutive analog input registers. The default value is 30001. AI Start Register (internal analog input for input power; 0 to 32768mV) AI Start Register +1 and AI Start Register +2 (controller counter input 0, 32-bit register) AI Start Register +3 and AI Start Register +4 (controller counter input 1, 32-bit register)
AO Start Register	First register of any unused block of one consecutive analog output register. The default value is 40500.

SCADAPack 4202 DR Extended I/O

The **SCADAPack 4202 DR with Extended I/O**, provides one digital point, which may operate as a digital input/counter or as a digital output. The digital output shares the same physical connections as the digital input. When the output is turned OFF, the point may be used as input. When the output is turned on it functions only as a digital output.

The SCADAPack 4202 DR manufactured after July 13, 2004, may use the 4202 DR Extended I/O register assignment. The SCADAPack 4202 DR must have controller board version 5 and terminal board version 6.

The **digital point** data is assigned to a single coil register and a single status register. The coil register is updated continuously with data read from the coil register. The status register is updated continuously with data read from the digital input.

The **analog and counter** input data is assigned to five consecutive input registers. One input register is used to monitor the controller input power and two counter input double registers (32 bit). These input registers are updated continuously with data read from the analog inputs and counter input.

The **analog output** data is assigned to a single analog output register. The analog output is updated continuously with the data read from the holding register.

Module Properties

Parameter	Description
Module Name	4203 DR / 4202 Extended I/O
DO Start Register	First register of any unused block of one consecutive digital output register. The default value is 1.
DI Start Register	First register of any unused block of one consecutive digital input register. The default value is 10001.
AI Start Register	First register of any unused block of five consecutive analog input registers. The default value is 30001. AI Start Register (internal analog input for input power; 0 to 32768mV) AI Start Register +1 and AI Start Register +2 (controller counter input 0, 32-bit register) AI Start Register +3 and AI Start Register +4 (controller counter input 1, 32-bit register)
AO Start Register	First register of any unused block of one consecutive analog output register. The default value is 40500.

SCADAPack 4202 DS

The **SCADAPack 4202 DS Default Module** is comprised of three different types of I/O channels. The SCADAPack 4202 DS hardware contains one internal analog input, one digital input, and two counter inputs.

This module provides two digital input/output (I/O) points. The first DI/O point may be used as a digital input/counter or as a digital output. The digital output shares the same physical connections as the digital input/counter. When the output is turned OFF, the point may be used as input/counter. When the output is turned on, it

functions only as a digital output. Digital point data is assigned to a single coil register and a single status register. The coil register is updated continuously with data read from the coil register. The status register is updated continuously with data read from the digital input.

The second DI/O point may be used as a turbine meter counter input or as a digital output. The digital output shares the same physical connections as the turbine meter counter input. When the output is turned OFF, the point may be used as a turbine meter counter input. When the output is turned on, it functions only as a digital output. Digital point data is assigned to a single coil register. The coil register is updated continuously with data read from the coil register.

Analog and counter input data is assigned to seven consecutive input registers. Two input registers are used for the two analog inputs; one input register is used to monitor the controller input power; two input registers are used for the counter input (32-bit double word uses two registers) and two input registers are used for the turbine meter counter input (32-bit double word uses two registers). These input registers are updated continuously with data read from the analog inputs and counter inputs.

Module Properties

Parameter	Description
Module Name	4203 DS / 4202 DS I/O
DO Start Register	First register of any unused block of two consecutive digital output registers. The default value is 1.
DI Start Register	First register of any unused block of one consecutive digital input register. The default value is 10001.
AI Start Register	First register of any unused block of seven consecutive analog input registers. The default value is 30001. AI Start Register (analog input channel 0) AI Start Register +1 (analog input channel 1) AI Start Register +2 (internal analog input for input power; 0 to 32768mV) AI Start Register +3 and AI Start Register +4 (controller counter input 0, 32-bit register) AI Start Register +5 and AI Start Register +6 (controller counter input 1, 32-bit register)
AO Start Register	First register of any unused block of one consecutive analog output register. The default value is 40500.

SCADAPack 4203 DR

The **SCADAPack 4203 DR I/O module**, provides one digital point, which may operate as a digital input/counter or as a digital output. The digital output shares the same physical connections as the digital input. When the output is turned OFF, the point may be used as input. When the output is turned on it functions only as a digital output.

The **digital point** data is assigned to a single coil register and a single status register. The coil register is updated continuously with data read from the coil register. The status register is updated continuously with data read from the digital input.

The **analog and counter** input data is assigned to five consecutive input registers. One input register is used to monitor the controller input power and two counter input double registers (32-bit). These input registers are updated continuously with data read from the analog inputs and counter input.

The **analog output** data is assigned to a single analog output register. The analog output is updated continuously with the data read from the holding register.

Module Properties

Parameter	Description
Module Name	4203 DR / 4202 Extended I/O
DO Start Register	First register of any unused block of one consecutive digital output register. The default value is 1.
DI Start Register	First register of any unused block of one consecutive digital input register. The default value is 10001.
AI Start Register	First register of any unused block of five consecutive analog input registers. The default value is 30001. AI Start Register (internal analog input for input power; 0 to 32768mV) AI Start Register +1 and AI Start Register +2 (controller counter input 0, 32-bit register) AI Start Register +3 and AI Start Register +4 (controller counter input 1, 32-bit register)
AO Start Register	First register of any unused block of one consecutive analog output register. The default value is 40500.

SCADAPack 4203 DS

The **SCADAPack 4203 DS Default Module** is comprised of three different types of I/O channels. The SCADAPack 4203 DS hardware contains one internal analog input, one digital input, and two counter inputs.

This module provides two digital input/output (I/O) points. The first DI/O point may be used as a digital input/counter or as a digital output. The digital output shares the same physical connections as the digital input/counter. When the output is turned OFF, the point may be used as input/counter. When the output is turned on it functions only as a digital output. Digital point data is assigned to a single coil register and a single status register. The coil register is updated continuously with data read from the coil register. The status register is updated continuously with data read from the digital input.

The second DI/O point may be used as a turbine meter counter input or as a digital output. The digital output shares the same physical connections as the turbine meter counter input. When the output is turned OFF, the point may be used as a turbine meter counter input. When the output is turned on, it functions only as a digital output. Digital point data is assigned to a single coil register. The coil register is updated continuously with data read from the coil register.

Analog and counter input data is assigned to seven consecutive input registers. Two input registers are used for the two analog inputs; one input register is used to monitor the controller input power; two input registers are used for the counter input (32-bit double word uses two registers) and two input registers are used for the turbine meter counter input (32-bit double word uses two registers). These input registers are updated continuously with data read from the analog inputs and counter inputs.

Module Properties

Parameter	Description
Module Name	4203 DS / 4202 DS I/O

DO Start Register	First register of any unused block of two consecutive digital output registers. The default value is 1.
DI Start Register	First register of any unused block of one consecutive digital input register. The default value is 10001.
AI Start Register	First register of any unused block of seven consecutive analog input registers. The default value is 30001. AI Start Register (analog input channel 0) AI Start Register +1 (analog input channel 1) AI Start Register +2 (internal analog input for input power; 0 to 32768mV) AI Start Register +3 and AI Start Register +4 (controller counter input 0, 32-bit register) AI Start Register +5 and AI Start Register +6 (controller counter input 1, 32-bit register)
AO Start Register	First register of any unused block of one consecutive analog output register. The default value is 40500.

Analog Input Modules

Analog input modules are used to assign data from physical analog inputs to input registers in the I/O database. The physical analog inputs are specific SCADAPack I/O I/O modules, generic I/O modules, and internal controller data such as RAM battery voltage and board temperature.

Analog input I/O modules may assign data to any input registers in the I/O database that are not being used by another Analog Input or Counter I/O module.

All I/O database input registers that are not assigned to any other I/O modules may be used as general purpose input registers in a ladder program. The I/O modules available are described in the following topics.

Note: To properly view analog input/output registers, the register type should be set to type signed. Setting the register type to anything else may result in inaccurate readings being displayed/written to these registers.

5501 Eight Channel Module

5502 Eight Channel Differential Module

5503 Four Channel RTD Module

5504 Eight Channel Thermocouple Module

5505 Eight Channel RTD Module

5506 Eight Channel Module

5521 Eight Channel Potentiometer Module

Controller Board Temperature

Controller Input Voltage

Controller RAM Battery Voltage

Generic Eight Channel Module

5501 Eight Point Module

The **5501 Eight Point Module** provides eight analog inputs, from the 5501 analog input module, for each module address. Data from each module is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Module Properties

Field	Description
Module Name	5501 Eight Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Input type module can use this module address. Default value: 0.
AI Start	First register of any unused block of 8 consecutive input registers. Default register: 30001. Start Register (Analog Input 0) Start Register + 1 (Analog Input 1) Start Register + 2 (Analog Input 2) Start Register + 3 (Analog Input 3) Start Register + 4 (Analog Input 4) Start Register + 5 (Analog Input 5) Start Register + 6 (Analog Input 6) Start Register + 7 (Analog Input 7)

Notes

If the hardware revision of the 5501 board is 4 (there is the number 2X243120-4 beside the connector P1) and firmware revision is D (sticker on PIC chip indicates firmware version e.g. 160303D) 5501 module will work with both SP3xx and SP32. The latest version of the 5501 firmware 160303D was released on September 7, 1999.

All version 4 hardware 5501 modules that were shipped after January 1997 will work with the SP3xx or SP32 controllers.

Any previous versions of hardware or firmware (e.g. module hardware version 3 and old PIC chip firmware 160246D) will work only with 16-bit SCADAPacks.

5502 Eight Point Differential Module

The **5502 Eight Point Differential Module** provides eight analog inputs, from the 5502 analog input module, for each module address. Data from each module is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Module Properties

Field	Description
Module Name	5502 Eight Point Differential Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Input type module can use this module address. Default value: 0.
AO Start Register	First register of any unused block of 8 consecutive input registers. Default register: 30001. Start Register (Analog Input 0) Start Register + 1 (Analog Input 1) Start Register + 2 (Analog Input 2) Start Register + 3 (Analog Input 3) Start Register + 4 (Analog Input 4) Start Register + 5 (Analog Input 5) Start Register + 6 (Analog Input 6) Start Register + 7 (Analog Input 7)

5503 Four Point RTD Module

The 5503 Four Channel RTD module provides four analog inputs, from the 5503 RTD analog input module, for each module address. Data is assigned to four consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Module Properties

Field	Description
Module Name	5503 Four Point RTD Module.
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Input type module can use this module address. Default value: 0.
AI Start	First register of any unused block of 4 consecutive input registers. Default register: 30001. Start Register (RTD Analog Input 0) Start Register + 1 (RTD Analog Input 1) Start Register + 2 (RTD Analog Input 2) Start Register + 3 (RTD Analog Input 3)

5504 Eight Point Thermocouple Module

The **5504 Eight Point Thermocouple module** provides eight analog inputs, from the 5504 Thermocouple input module, for each module address. Data is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Module Properties

Field	Description
Module Name	5504 Eight Point Thermocouple Module.
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Input type module can use this module address. Default value: 0.
AI Start Register	First register of any unused block of 8 consecutive input registers. Default register: 30001. Start Register (Thermocouple Analog Input 0) Start Register + 1 (Thermocouple Analog Input 1) Start Register + 2 (Thermocouple Analog Input 2) Start Register + 3 (Thermocouple Analog Input 3) Start Register + 4 (Thermocouple Analog Input 4) Start Register + 5 (Thermocouple Analog Input 5) Start Register + 6 (Thermocouple Analog Input 6) Start Register + 7 (Thermocouple Analog Input 7)

5505 Four Point RTD Module

The **5505 Four Point RTD Module** provides four RTD analog input points from a 5505 RTD module. A maximum of sixteen 5505 RTD modules may be installed on the I/O bus. The 5505 RTD module can operate in Native mode or in 5503 Emulation Mode. The operating mode is determined by a DIP switch on the 5505 RTD module.

When using the 5505 RTD module in 5503 Emulation mode, the 5503 Four Point RTD Module is used in the register assignment. See the 5503 Four Point RTD Module topic if the module is to be used in 5503 Emulation mode.

The 5505 RTD module native mode provides enhanced capability over the 5503 emulation mode. The module operates in native mode if the 5503 Emulation DIP switch is open. This mode is recommended for all new installations.

- Each input is individually configurable for resistance measurement or RTD temperature measurement.
- Each RTD input is configurable to return the measured temperature in degrees Celsius, Kelvin, or Fahrenheit.
- All inputs have a common configurable filter rate that can be used to dampen process variations or noise.
- The module returns diagnostic and status information for each input, such as RTD type (3- or 4-wire), open RTD annunciation, and RTD out-of-range annunciation.
- Data is returned as an **IEEE 32 bit single precision floating point number** Single precision floating point is an IEEE 754 standard for encoding binary or decimal floating point numbers in 4 bytes.

that requires no additional scaling.

The RTD data is assigned to eight consecutive analog input registers, two for each floating point RTD value. The input registers are updated continuously with data read from the analog inputs.

The 5505 RTD module provides 16 internal digital input registers that return status and diagnostic information about the RTDs. The digital input registers are updated continuously with data read from the digital inputs.

Module Properties

Parameter	Description
Module Name	5505 Four Point RTD Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of sixteen consecutive digital input registers. The default value is 10001. (Reserved for future use) DI Start Register (OFF = RTD 0 is good; ON = RTD 0 is open or PWR input is OFF) DI Start Register + 1 (OFF = RTD 0 data in range; ON = RTD 0 data is out of range) DI Start Register + 2 (OFF = RTD 0 is using 3-wire; ON = RTD 0 is using 4-wire) DI Start Register + 3 (Reserved for future use) DI Start Register + 4 (OFF = RTD 1 is good; ON = RTD 1 is open or PWR input is OFF) DI Start Register + 5 (OFF = RTD 1 data in range; ON = RTD 1 data is out of range) DI Start Register + 6 (OFF = RTD 1 is using 3-wire; ON = RTD 1 is using 4-wire) DI Start Register + 7 (Reserved for future use) DI Start Register + 8 (OFF = RTD 2 is good; ON = RTD 2 is open or PWR input is OFF) DI Start Register + 9 (OFF = RTD 2 data in range; ON = RTD 2 data is out of range) DI Start Register + 10 (OFF = RTD 2 is using 3-wire; ON = RTD 2 is using 4-wire) DI Start Register + 11 (Reserved for future use) DI Start Register + 12 (OFF = RTD 3 is good; ON = RTD 3 is open or PWR input is OFF) DI Start Register + 13 (OFF = RTD 3 data in range; ON = RTD 3 data is out of range) DI Start Register + 14 (OFF = RTD 3 is using 3-wire; ON = RTD 3 is using 4-wire) DI Start Register + 15 (Reserved for future use)
AI Start Register	First register of any unused block of eight consecutive analog input registers. Default value is 30001. Start Register and Start Register + 1 (RTD 0 input in IEEE 754 floating point format) Start Register + 2 and Start Register + 3 (RTD 1 input in IEEE 754 floating point format) Start Register + 4 and Start Register + 5 (RTD 2 input in IEEE 754 floating point format) Start Register + 6 and Start Register + 7 (RTD 3 input in IEEE 754 floating point format)
AI*n* Channel Type	Defines the measurement type for each input, AIN 0 Type, AIN 1 Type, AIN 2 Type, AIN 3 Type, data in the input registers assigned to the input is set to the type selected. • The deg C selection sets the input point measurement type to degrees Celsius. • The deg F selection sets the input point data measurement to degrees Fahrenheit. • The deg K selection sets the input point data measurement to degrees Kelvin. • The ohms selection sets the input point data measurement as resistance measurement in ohms.
Module Filter	Sets the input filter rate for all RTD inputs. The filter rate is used to dampen process variations or noise. • The 0.5s selection sets the filter rate to 0.5 seconds. • The 1s selection sets the filter rate to 1 second. • The 2s selection sets the filter rate to 2 seconds. • The 4s selection sets the filter rate to 4 seconds.

5506 Eight Point Module

The **5506 Eight Point Module** provides eight analog inputs from a 5506 Analog Input module. A maximum of sixteen 5506 analog input modules may be installed on the I/O bus. The 5506 Analog Input module can operate in Native mode or in 5501 Emulation Mode. The operating mode is determined by a DIP switch on the 5506 Analog Input.

When using the 5506 Analog Input module in 5501 Emulation mode the AIN 5501 module is used in the register assignment. See 5501 Eight Point Module if the module is to be used in 5501 Emulation mode.

The 5506 native mode provides enhanced capability over the 5501 emulation mode. The module operates in native mode if the 5501 Emulation DIP switch is open.

- Each input is individually configurable for 0-5V, 1-5V, 0-20mA, or 4-20mA operation.
- Converted analog input data is returned as a signed 15-bit value, providing eight times more resolution than in 5501 mode. Negative values are possible, for example if a 4-20mA is open loop.

- The module returns status information for each analog input indicating if the analog input is in or out of range for the defined signal type.
- All inputs have a configurable filter rate.
- The input-scanning rate is software configurable to 50 or 60 hertz.

Analog input data is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

The 5506 Analog Input module provides 8 internal digital input registers that return status information about the analog inputs. The digital input registers are updated continuously with data read from the digital inputs.

Module Properties

Parameter	Description
Module Name	5506 Eight Channel Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of eight consecutive digital input registers. The default value is 10001. DI Start Register (OFF = Analog Input 0 is good; ON = Analog 0 is over or under range) DI Start Register + 1 (OFF = Analog Input 1 is good; ON = Analog 1 is over or under range) DI Start Register + 2 (OFF = Analog Input 2 is good; ON = Analog 2 is over or under range) DI Start Register + 3 (OFF = Analog Input 3 is good; ON = Analog 3 is over or under range) DI Start Register + 4 (OFF = Analog Input 4 is good; ON = Analog 4 is over or under range) DI Start Register + 5 (OFF = Analog Input 5 is good; ON = Analog 5 is over or under range) DI Start Register + 6 (OFF = Analog Input 6 is good; ON = Analog 6 is over or under range) DI Start Register + 7 (OFF = Analog Input 7 is good; ON = Analog 7 is over or under range)
AI Start	First register of any unused block of 8 consecutive input registers. Default register: 30001. Start Register (Analog Input 0) Start Register + 1 (Analog Input 1) Start Register + 2 (Analog Input 2) Start Register + 3 (Analog Input 3) Start Register + 4 (Analog Input 4) Start Register + 5 (Analog Input 5) Start Register + 6 (Analog Input 6) Start Register + 7 (Analog Input 7)
AI*n* Channel Type	Defines the input measurement type for each input (AIN 0 through 7). • The 0-5V selection sets the input type to measure 0 to 5V input signals. • The 1-5V selection sets the input type to measure 1 to 5V input signals. • The 0-20 mA selection sets the input type to measure 0 to 20mA input signals. • The 4-20 mA selection sets the input type to measure 4 to 20mA input signals.
Module Filter	Sets the input filter rate for all analog inputs. The filter rate is used to dampen process variations or noise. • The 3Hz filter selection sets the response time to 155ms at 60Hz and 185ms at 50Hz. • The 6Hz filter selection sets response time to 85ms at 60Hz and 85ms at 50Hz. • The 11Hz filter selection sets response time to 45ms at 60Hz and 55ms at 50Hz. • The 30Hz filter selection sets response time to 30ms at 60Hz and 30ms at 50Hz.
Module Scan Frequency	Sets the input scan rate for all analog inputs. The scan rate selection is not critical but ac noise rejection is improved at the correct frequency. If the module is used in a dc environment, the 60Hz setting will yield slightly faster response time. • The 60Hz selection synchronies the input scanning to 60Hz. • The 50Hz selection synchronies the input scanning to 50Hz.

5521 Eight Point Potentiometer Module

The **5521 Eight Channel Potentiometer module** provides eight analog inputs, from the 5521 Potentiometer analog input module, for each module used. Data is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Module Properties

Field	Description
Module Name	5521 Eight Channel Potentiometer Module
Module Address	This module is assigned a unique module address between 0 and 7. No other Analog Input type module can use this module address. Default value: 0.
AI Start	First register of any unused block of 8 consecutive input registers. Default register: 30001. Start Register (Potentiometer Analog Input 0) Start Register + 1 (Potentiometer Analog Input 1) Start Register + 2 (Potentiometer Analog Input 2) Start Register + 3 (Potentiometer Analog Input 3) Start Register + 4 (Potentiometer Analog Input 4) Start Register + 5 (Potentiometer Analog Input 5) Start Register + 6 (Potentiometer Analog Input 6)

Start Register + 7 (Potentiometer Analog Input 7)

Controller Board Temperature Module

The **Controller Board Temperature** module reads the circuit board temperature of the SCADAPack controller. The temperature data is assigned to two consecutive input registers. The input registers are updated continuously with data read from the controller.

Module Properties

Field	Description
Module Name	Controller Board Temperature Module
AI Start Register	First register of any unused block of 2 consecutive input registers. Default register: 30001. Start Register (Board temperature in degrees Celsius; range -40°C to 75°C) Start Register + 1 (Board temperature in degrees Fahrenheit; range -40°F to 167°F)

Controller Input Voltage Module

The **Controller Input Voltage** module reads the input voltage of the following supported controllers:

- SCADAPack 314
- SCADAPack 330
- SCADAPack 334
- SCADAPack 350
- SCADAPack 357
- SCADAPack 4203

The controller input voltage provides useful information for the power input to the controller such as if a battery back-up system is functioning correctly. The reading returned from this input is typically in the range from 9000 to 30000 representing the input supply in mV. The input supply voltage resolution is 100 millivolts.

SCADAPack controllers input voltage specification varies depending on the controller type. Refer to the user manual for your controller for the input voltage specification.

The input register is updated continuously with data read from the controller.

Module Properties

Field	Description
Module Name	Controller Input Voltage Module
AI Start Register	First register of any unused block of 1 input register. Default register: 30001. Start Register (Controller Input Voltage in millivolts)

Controller RAM Battery Voltage

The **Controller RAM Battery Voltage** module reads the RAM backup lithium battery voltage of the SCADAPack controllers. The voltage value is assigned to one input register. The input register is updated continuously with voltage data read from the RAM battery.

Module Properties

Field	Description
Modular Name	Controller RAM Battery Voltage
AI Start Register	First register of any unused block of 1 consecutive input registers. Default register: 30001. Start Register (RAM backup Battery Voltage 0-5000 mV) The 3.6V lithium battery will return a typical value of 3600 or 3700. A reading of less than 3000 (3.0V) indicates that the lithium battery requires replacement. The RAM battery voltage resolution is 100 millivolts.

Generic Eight Point Module

The **Generic Eight Point** module provides eight analog inputs. Data is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs. The **Generic Eight Point** module may be used in place of any other eight point Analog Input type module.

The **Generic Eight Point** module type is a useful selection early on in the system design stage before the final selection of the specific AIN module type is known.

Module Properties

Field	Description
Module Name	Generic Eight Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Input type module can use this module address. Default value: 0.
AI Start Register	First register of any unused block of 8 consecutive input registers. Default register: 30001. Start Register (Analog Input 0) Start Register + 1 (Analog Input 1) Start Register + 2 (Analog Input 2) Start Register + 3 (Analog Input 3) Start Register + 4 (Analog Input 4) Start Register + 5 (Analog Input 5) Start Register + 6 (Analog Input 6) Start Register + 7 (Analog Input 7)

Analog Output Modules

Analog output modules are used to assign data from the I/O database to physical analog outputs. The physical analog outputs are either specific SCADAPack I/O I/O modules or generic I/O modules.

Analog output I/O modules may assign data to any holding register in the I/O database that is not being used by another Analog output I/O module. There are 9999 holding registers available. These holding registers are numbered 40001 to 49999.

Note: Holding registers are referred to as 4nnnn registers throughout this manual.

All holding registers that are not assigned to any other I/O modules may be used as general purpose holding registers in a ladder program. The I/O modules available are described in the following pages.

Note: To properly view analog input/output registers, the register type should be set to type signed. Setting the register type to anything else may result in inaccurate readings being displayed/written to these registers.

5301 Two Point Module

The **5301 Two Point** module provides two analog outputs, to the SCADAPack I/O 5301 Analog output module, for each module used. Data is assigned from two consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers.

Module Properties

Field	Description
Module Name	5301 Two Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Output type module can use this module address. Default value: 0.
AO Start Register	First register of any unused block of 2 consecutive output registers. Default register: 40001. Start Register (Analog Output 0) Start Register + 1 (Analog Output 1)

Notes

The 5301 module was made obsolete in December 1996. The 5301 module is supported only on 16-bit SCADAPack controllers. The I/O bus hardware on the 5301 module is not capable of running at the speeds used on SCADAPack 3xx and SCADAPack 32 controllers.

The 5301 has since been replaced by:

5302 Four Channel Analog Output Module
or
5304 Four Channel Analog Output Module

5302 Four Point Module

The **5302 Four Point** module provides four analog outputs, to the SCADAPack I/O 5302 Analog output module, for each module used. Data is assigned from four consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers.

Module Properties

Field	Description
Module Name	5302 Four Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Output type module can use this module address. Default value: 0.
AO Start Register	First register of any unused block of 4 consecutive output registers. Default register: 40001. Start Register (Analog Output 0) Start Register + 1 (Analog Output 1) Start Register + 2 (Analog Output 2) Start Register + 3 (Analog Output 3)

5304 Four Channel Module

The **5304 Four Channel** module provides four analog outputs, to the SCADAPack I/O 5304 Analog output module, for each module used. Data is assigned from four consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers.

Module Properties

Field	Description
Module Name	5304 Four Channel Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Output type module

	can use this module address. Default value: 0.
AO Start Register	First register of any unused block of 4 consecutive output registers. Default register: 40001. Start Register (Analog Output 0) Start Register + 1 (Analog Output 1) Start Register + 2 (Analog Output 2) Start Register + 3 (Analog Output 3)

Generic Four Point Module

The **Generic Four Point** module provides four analog outputs for each module used. Data is assigned from four consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers. The **Generic Four Point** module may be used in place of any other 4 point Analog Output type module.

The **Generic Four Point** module type is a useful selection early on in the system design stage before the final selection of the specific AOUT module type is known.

Module Properties

Field	Description
Module Name	Generic Four Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Output type module can use this module address. Default value: 0.
AO Start Register	First register of any unused block of 4 consecutive output registers. Default register: 40001. Start Register (Analog Output 0) Start Register + 1 (Analog Output 1) Start Register + 2 (Analog Output 2) Start Register + 3 (Analog Output 3)

Generic Two Point Module

The **Generic Two Point m**odule provides two analog outputs for each module used. Data is assigned from four consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers. The **Generic Two Point m**odule may be used in place of any other 4 point Analog Output type module.

The **Generic Two Point** module type is a useful selection early on in the system design stage before the final selection of the specific AOUT module type is known.

Module Properties

Field	Description
Module Name	Generic Two Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Analog Output type module can use this module address. Default value: 0.
AO Start Register	First register of any unused block of 2 consecutive output registers. Default register: 40001. Start Register (Analog Output 0) Start Register + 1 (Analog Output 1)

SCADAPack Two Point Module

The **SCADAPack Two Point module** provides two analog outputs for each module used. Data is assigned from two consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers.

Module Properties

Field	Description
Module Name	SCADAPack Two Point Module
AO Start Register	First register of any unused block of 2 consecutive output registers. Default register: 40001. Start Register (Analog Output 0) Start Register + 1 (Analog Output 1)

Configuration Modules

Configuration modules are used to assign data from the I/O database coil and holding registers to controller configuration settings. The configuration data is used to configure controller settings such as clearing protocol and serial counters, real time clock settings, HART protocol interface, and PID control blocks.

Configuration I/O modules assign data to I/O database coil or holding registers, depending on the type of data. Configuration I/O modules may assign data to any coil or holding registers in the I/O database that are not being used by another I/O module.

All I/O database coil or holding registers that are not assigned to any other I/O modules may be used as general purpose coil or holding registers in a ladder program. The configuration modules available are described in the following pages.

PID Control Block
Real Time Clock and Alarm
Save Settings to EEPROM
Serial Port DTR Off
Serial Port Protocol Settings Method 1
Serial Port Protocol Settings Method 2
Serial Port Protocol Settings Method 3
Serial Port Settings
Store and Forward

5904 HART Interface Module

The **5904 HART Interface** module provides control of **5904 HART Interface** modules. When configured, the module maintains communication with a 5904 interface. Up to four 5904 interfaces may be attached to a controller.

The interface settings are assigned from four consecutive holding registers. The registers are initialized with the current value of the module parameters when the controller is reset, and when the **Register Assignment** is downloaded to the controller. Thereafter the data from all four holding registers are written to the module parameters whenever any of the registers change.

Module Properties

Field	Description
Module Name	5904 HART Interface Module
Module Address	This module is assigned a unique module address between 0 to 3. No other Configuration-type module can use this module address. Default value: 0.
AO Start Register	First register of any unused block of 4 consecutive input registers (AO1 through AO4) in the I/O database. Default register: 40001.

Register Data

Register Assignment	Assignment to Module Hardware
AO1	Attempts: the number of times each HART command will be sent. Valid values are 1 to 4. Default = 3
AO2	Preambles: number of preambles to send if fixed preambles selected. Valid values are 2 to 15. **Note:** This value must be set as it is used by link initialization. Default : = 15
AO3	Auto-preamble control: 0 = use fixed number of preambles 1 = use number of preambles requested by device except for link initialization Default : = 1
AO4	Device type 0 = secondary master device 1 = primary master device (recommended) Default : = 1

Notes

1. This module provides the basis for communication with HART devices connected to the 5904 interface module. It does not send commands to HART devices. Use the Ladder Logic HART element or the C Tools HART API functions to send commands to the HART devices.
2. The number of preambles must be set even if the auto-preamble control is selected. The preambles value will be used for the link initialization command that determines the number of preambles requested by the device.
3. In most sensor networks, the controller is the primary master device. A hand-held programmer is typically a secondary master device. Set the device type to primary unless another HART primary master device is connected to the network.
4. The settings can be saved to EEPROM with the *Save Settings to EEPROM* module.

Related Topics

Save Settings to EEPROM

Clear Serial Port Counters

The **Clear Serial Port Counters** module provides a way of clearing the Serial Port Comm Status diagnostic data counters for a specific serial port. The counters for the serial port are cleared when the assigned coil register is set to 1. Once the counters are cleared, the assigned coil register is automatically reset to zero.

Module Properties

Field	Description
Module Name	Clear Serial Port Protocol Counters Module
Comm Interface	Communication interface to clear Serial Port Comm Status diagnostic data counters. Depending on the controller type the selection is one of: Com1, Com2, Com3 and Com4. Default is Com1.
DO Start Register	First register of any unused block of eight sequential digital output registers. The default value is 1.

Clear Serial Port Protocol Counters

The **Clear Serial Port Protocol Counters** module provides a way of clearing the Serial Port Protocol Status counters for a specific serial port. The counters for the

serial port are cleared when the assigned coil register is set to 1. Once the counters are cleared, the assigned coil register is automatically reset to zero.

Module Properties

Field	Description
Module Name	Clear Serial Port Protocol Counters Module
Comm Interface	Communication interface to clear Serial Port Comm Status diagnostic data counters. Depending on the controller type the selection is one of: Com1, Com2, Com3 and Com4. Default is Com1.
DO Start Register	First register of any unused block of eight sequential digital output registers. The default value is 1.

Controller IP Settings Module

The **Controller IP Settings** module provides control of the IP settings for a specific communication interface. Settings are assigned from thirteen consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all thirteen holding registers is written to the module parameters, whenever any of the registers change.

Only the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers use this module.

Use this module only when IP settings need to be changed within a Ladder program. It is more convenient to use the *Controller TCP/IP Settings* dialog to download IP settings with the Ladders program.

Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users need to ensure the ladder function is only energized for one program scan. If the ladder function is always energized the TCP/IP communication will not work as expected..

Module Properties

Field	Description
Module Name	Controller IP Settings Module
Start Register	First register of any unused block of 13 consecutive output registers (AO1 through AO13) in the I/O database. Default register: 40001.

Register Data

Register Assignment	Assignment to Module Hardware
Start Register	**Byte1** of the IP Address using format: **Byte1.Byte2.Byte3.Byte4** Default: 0
Start Register + 1	**Byte2** of the IP Address. Default: 0
Start Register + 2	**Byte3** of the IP Address. Default: 0
Start Register + 3	**Byte4** of the IP Address. Default: 0
Start Register + 4	**Byte1** of the Subnet Mask address using format: **Byte1.Byte2.Byte3.Byte4** Default: 255
Start Register + 5	**Byte2** of the Subnet Mask address. Default: 255
Start Register + 6	**Byte3** of the Subnet Mask address. Default: 255
Start Register + 7	**Byte4** of the Subnet Mask address. Default: 255
Start Register + 8	**Byte1** of the Default Gateway address using format: **Byte1.Byte2.Byte3.Byte4** Default: 0
Start Register + 9	**Byte2** of the Default Gateway address. Default: 0
Start Register + 10	**Byte3** of the Default Gateway address. Default: 0
Start Register + 11	**Byte4** of the Default Gateway address. Default: 0
Start Register + 12	**IP Configuration Mode** 0 = Default gateway is on the LAN subnet 1 = Default gateway is on the com1 PPP subnet 2 = Default gateway is on the com2 PPP subnet 3 = Default gateway is on the com3 PPP subnet 4 = Default gateway is on the com4 PPP subnet

Notes

Thirteen holding registers are always assigned to the I/O database when this module is used.

Controller LED Power Settings

The **Controller LED Power Settings** module provides control of the power settings for the controller LEDs. The state of the assigned coil register is the default state for the LED power. The time to return to the Default State is contained within the assigned holding register.

The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the module parameters are updated continuously with data read from the assigned registers.

Module Properties

Field	Description

Module Name	Controller LED Power Settings Module
DO Start Register	Any unused digital output register block (DO1) in the I/O database. Default register: 1.
AO Start Register	Any unused input register (AO1) in the I/O database. Default register: 40001.

Coil Register Data

Register Assignment	Assignment to Module Hardware
Start Register	Default state for power to the controller LEDs 0 = OFF 1 = ON Default Value: 1 = ON

Holding Register Data

Register Assignment	Assignment to Module Hardware
Start Register	Time in minutes to return to the default state. 1 to 65535 minutes Default Value: 5 minutes

Notes

One holding register and one coil register are always assigned to the I/O database when this module is used.

Controller Power Mode

The **Controller Power Mode** module provides control over the power consumption of the controller. The LAN port and USB host port can be individually disabled to conserve power. The controller can also run at a reduced speed to conserve power.

The status registers are updated continuously with the current LAN, controller, and USB host power states. The current settings can be changed by writing to the appropriate holding register.

Register Assignment

Parameter	Description
Module	Controller Power Mode module
Address	fixed
Type	holding register.
Start	First register of any unused block of 3 consecutive holding registers. (4xxxx)
End	Last register of block (4xxxx + 2)
Registers	3 holding registers
Description	None

Holding Register Data

Register Assignment	Assignment to Module Hardware
Start Register	LAN Operation State (0 = LAN Enabled, 1 = LAN Disabled)
Start Register + 1	Controller Power State (0 = Normal Power Mode, 1 = Reduced Power Mode)
Start Register + 2	USB Host Port Power State (0 = USB host port enabled, 1 = USB host port disabled)

Notes

Three holding registers are always assigned to the I/O database when you use this module.

FTP

The **FTP** module provides control over the controller FTP server state. The FTP server can be disabled, enabled with anonymous login permitted, or enabled with password protected login required.

When the FTP server is to be set to enabled, password required (FTP Server State 2 below) the user name and password must be entered using the FTP tab in the IP Settings group.

The status register is updated continuously with the current FTP server status. The current settings can be changed by writing to the appropriate holding register.

Register Assignment

Parameter	Description
Module	FTP
Address	fixed
Type	holding register.
Start	Any unused holding register. (4xxxx)
End	Same as Start (4xxxx)
Registers	Single holding register

Holding Register Data

Register Assignment	Assignment to Module Hardware
Start Register	FTP Server State: 0 = FTP Disabled 1 = FTP Enabled (anonymous login permitted) 2 = FTP Enabled (password required)

Notes

One holding register is always assigned to the I/O database when you use this module.

Device Configuration Module

The **Device Configuration** module provides a method for users to assign an ID number to the applications running in a **SCADAPack** or SCADAPack controller. This module is supported on all SCADAPack controllers with 2.44 (or later) firmware; SCADAPack32 controllers with 1.92 (or later) firmware and SCADAPack 314, SCADAPack 330, SCADAPack 334, SCADAPack 350 or SCADAPack 4203 controllers with 1.25 (or later) firmware.

The Controller Device Configuration register assignment sets the Company ID, Logic Application number, and logic Application version. This information is then saved in a group of reserved registers in the I/O database. Refer to the ***Device Configuration Read Only Registers*** section for details on the Logic Application ID. These registers may be polled by SCADA Host software or other Modbus compatible devices.

Users must add a rung to the logic application that specifies the appropriate information for the Logic Application ID. For more information, see the ***Example*** below.

For SCADAPack and SCADAPack 32 controllers it is necessary to create a logic application consisting of at least one element. If there is no logic program present, then the logic application ID is cleared.

The number of applications and firmware version is the responsibility of the user.

Register Assignment

Parameter	Description
Module	CNFG Device Configuration module
Address	Fixed
Type	Coil register.
Start	First register of any unused block of 4 consecutive holding registers. (4xxxx)
End	Last register of block (4xxxx + 3)
Registers	1 coil register
Description	4 output registers.
Extended Parameters	Device information and control.

Register Data

Register Assignment	Assignment to Module Hardware
Start Register	0 = configuration registers operate as normal input registers 1 = configuration registers report device configuration
Start Register + 1	Company ID Contact Control Microsystems to obtain your Company ID.
Start Register + 2	Logic Application number
Start Register +3	Logic Application version (major *100 + minor)

Device Configuration Read Only Registers

The Device configuration is stored in Modbus input (3xxxx) registers as shown below. The registers are read with standard Modbus commands. You cannot write to these registers. Device configuration registers use fixed addresses. This facilitates identifying the applications in a standard manner.

These registers must not be used for other purposes in a logic or C application. This includes the following uses:
- **MSTR** block uses registers 39800 to 39999 (in element configuration) as a destination
- **OVER** block uses registers 39800 to 39999 as a destination
- any register assignment module uses registers 39800 to 39999

The Device configuration registers can be enabled or disabled by entering a 0 or 1 in the Start Register. They are disabled until enabled by a logic application. This provides compatibility with controllers that have already used these registers for other purposes.

The application IDs are cleared on every controller reset. Applications must run and set the application ID for it to be valid.

The following information is stored in the device configuration. Two logic application identifiers are provided for compatibility with SCADAPack ES/ER controllers that provide two IEC 61131-3 applications. The second logic application identifier is not used with other controllers. 32 application identifiers are provided to accommodate C applications in SCADAPack 33x/350 controllers.

Register	Description
39800	Controller ID (ASCII value), first byte
39801	Controller ID (ASCII value), second byte
39802	Controller ID (ASCII value), third byte
39803	Controller ID (ASCII value), fourth byte
39804	Controller ID (ASCII value), fifth byte
39805	Controller ID (ASCII value), sixth byte
39806	Controller ID (ASCII value), seventh byte
39807	Controller ID (ASCII value), eighth byte
39808	Firmware version (major*100 + minor)
39809	Firmware version build number (if applicable)
39810	Logic Application 1 - Company ID
39811	Logic Application 1 - Application number (0 to 65535)
39812	Logic Application 1 - Application version (major*100 + minor)
39813	Logic Application 2 - Company ID
39814	Logic Application 2 - Application number (0 to 65535)
39815	Logic Application 2 - Application version (major*100 + minor)
39816	Number of applications identifiers used (0 to 32) Identifiers are listed sequentially starting with identifier 1. Unused identifiers will return 0.
39817	Application Identifier 1 (see format below)
39820	Application Identifier 2 (see format below)
39823	Application Identifier 3 (see format below)
39826	Application Identifier 4 (see format below)
39829	Application Identifier 5 (see format below)
39832	Application Identifier 6 (see format below)
39835	Application Identifier 7 (see format below)
39838	Application Identifier 8 (see format below)

39841	Application Identifier 9 (see format below)
39844	Application Identifier 10 (see format below)
39847	Application Identifier 11 (see format below)
39850	Application Identifier 12 (see format below)
38853	Application Identifier 13 (see format below)
39856	Application Identifier 14 (see format below)
39859	Application Identifier 15 (see format below)
39862	Application Identifier 16 (see format below)
39865	Application Identifier 17 (see format below)
39868	Application Identifier 18 (see format below)
39871	Application Identifier 19 (see format below)
39874	Application Identifier 20 (see format below)
39877	Application Identifier 21 (see format below)
39880	Application Identifier 22 (see format below)
39883	Application Identifier 23 (see format below)
39886	Application Identifier 24 (see format below)
39889	Application Identifier 25 (see format below)
39892	Application Identifier 26 (see format below)
39895	Application Identifier 27 (see format below)
39898	Application Identifier 28 (see format below)
39901	Application Identifier 29 (see format below)
39904	Application Identifier 30 (see format below)
39907	Application Identifier 31 (see format below)
39910	Application Identifier 32 (see format below)
39913 to 39999	Reserved for future expansion

Application Identifier

The application identifier is formatted as follows.

Register	Description
Start	Company ID
Start + 1	Application number (0 to 65535)
Start + 2	Application version (major*100 + minor)

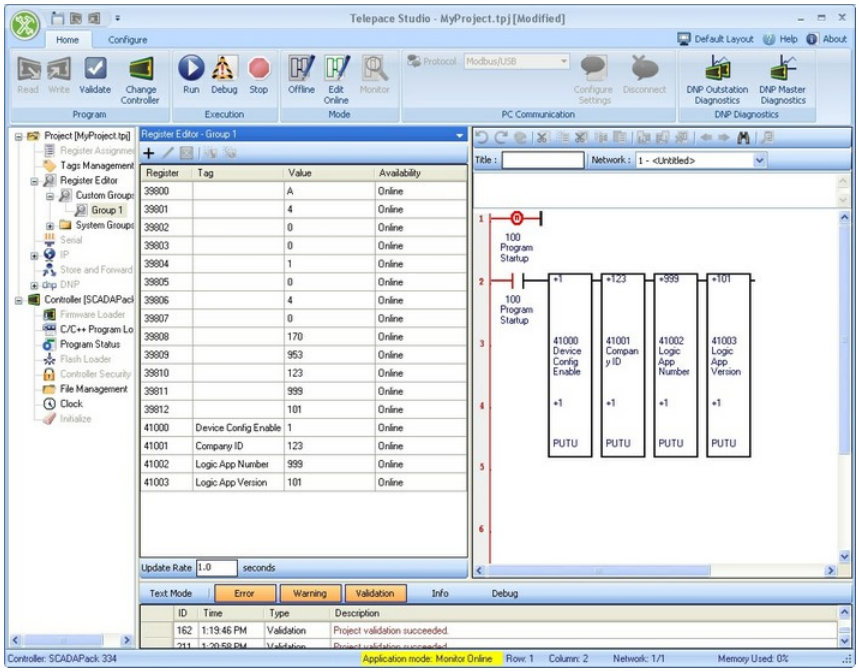
Company Identifier

Control Microsystems maintains a list of company identifiers to ensure the company ID is unique. Contact Control Microsystems for a Company ID. Company ID 0 indicates an identifier is unused.

Company IDs 1 to 100 are reserved for Control Microsystems use.

Example

The following logic enables the Device Configuration registers and sets the Company ID, Logic Application Number and Logic Application Version. The Device Configuration module uses registers 41000 through 41003. The view includes the Register Editor Group 1 containing a view of the registers associated with the Device Configuration module. Registers 39800 through 39807 contain ASCII values for the controller ID.



Modbus IP Interface Module

The **Modbus IP Interface** module provides control of the protocol interface settings used by all Modbus IP protocols on the specified communication interface. Settings are assigned from five consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all five holding registers is written to the module parameters, whenever any of the registers change.

Only the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers use this module.

Use this module only when Modbus IP protocol interface settings need to be changed within a Ladder Logic program.

Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users need to ensure the ladder function is only energized for one program scan. If the ladder function is always energized the TCP/IP communication will not work as expected..

Module Properties

Field	Description
Module Name	Modbus IP Module
AO Start Register	First register of any unused block of 5 consecutive output registers (AO1 through AO5) in the I/O database. Default register: 40001.

Register Data

Register Assignment	Assignment to Module Hardware
Start Register	Modbus station number 1 to 255 in standard Modbus 1 to 65534 in extended Modbus Default: 1
Start Register + 1	Modbus store and forward enable 0 = disabled 1 = enabled Default: 0 = disabled
Start Register + 2	Modbus addressing mode 0 = standard 1 = extended Default: 0 = standard
Start Register + 3	Enron Modbus enable 0 = disabled 1 = enabled Default: 0 = disabled
Start Register + 4	Enron Modbus station number 1 to 255 in standard Modbus 1 to 65534 in extended Modbus Default: 1

Notes

Five holding registers are always assigned to the I/O database when this module is used.

Modbus IP Protocols

The **Modbus IP Protocols** module provides control of the settings for a Modbus IP protocol. Settings are assigned from ten consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all ten holding registers is written to the module parameters, whenever any of the registers change.

Only the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers use this module.

Use this module only when Modbus IP protocol settings need to be changed within a Ladders program. It is more convenient to use the *Controller Modbus IP Protocols* dialog to download protocol settings with the Ladders program.

Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users need to ensure the ladder function is only energized for one program scan. If the ladder function is always energized the TCP/IP communication will not work as expected..

Register Assignment

Parameter	Description
Module	Modbus IP Protocols
Address	1 = Modbus/TCP 2 = Modbus RTU over UDP 3 = Modbus ASCII over UDP
Type	holding register.
Start	First Register in a Block of 10 registers.
End	Last register of block (4xxxx + 9)
Registers	10 holding registers
Description	None
Extended Parameters	None

Register Data

Register Assignment	Assignment to Module Hardware
Start Register	Protocol Port Number: 1 to 65534 Modbus/TCP Default: 502 Modbus RTU over UDP Default: 49152

	Modbus ASCII over UDP Default: 49153
Start Register + 1 and Start Register + 2	Master Idle Timeout: The length of time, in seconds, that a master connection will wait for the user to send the next command before ending the connection. This allows the slave device to free unused connections while the master application may retain the connection allocation. (32 bit register) 0 = disable timeout and let application close the connection. Default: 10 TCP protocols only. Not used by UDP protocols.
Start Register + 3 and Start Register + 4	Server Idle Timeout: The length of time, in seconds, that a server connection will wait for a message before ending the connection. (32 bit register) 0 = disable timeout and let client close connection. Default: 250 TCP protocols only. Not used by UDP protocols.
Start Register + 5	0 = disable server. 1 = enable server. Default: 1
Start Register + 6	Reserved
Start Register + 7	Reserved
Start Register + 8	Reserved
Start Register + 9	Reserved

Modbus/TCP Settings Module

The **Modbus/TCP Settings** module provides control of the Modbus/TCP protocol settings. Settings are assigned from six consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all six holding registers is written to the module parameters, whenever any of the registers change.

Only the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers use this module.

Use this module only when Modbus/TCP settings need to be changed within a Ladders program. It is more convenient to use the *Controller Modbus/TCP Settings* dialog to download Modbus/TCP settings with the Ladders program.

Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users need to ensure the ladder function is only energized for one program scan. If the ladder function is always energized the TCP/IP communication will not work as expected..

Module Properties

Field	Description
Module Name	Modbus/TCP Settings Module
Start Register	First register of any unused block of six consecutive output registers (AO1 through AO6) in the I/O database. Default register: 40001.

Register Data

Register Assignment	Assignment to Module Hardware
Start Register	Port for Modbus/TCP protocol Default: 502
Start Register + 1 and Start Register + 2	Master Idle Timeout: The length of time, in seconds, that a master connection will wait for the user to send the next command before ending the connection. (32 bit register) 0 = disable timeout and let user close the connection. Default: 10
Start Register + 3 and Start Register + 4	Server Idle Timeout: The length of time, in seconds, that a server connection will wait for a message before ending the connection. (32 bit register) 0 = disable timeout and let client close connection. Default: 250
Start Register + 5	Maximum number of server connections up to the capacity available (typical capacity is 20 connections) Default: 20

PID Control Block Module

The **PID Control Block** module provides control over the configuration of a PID control block and provides access to the control block parameters. Control block parameters are assigned to 25 consecutive holding registers. The module address refers to the PID control block number.

The holding registers are updated continuously with data read from the module parameters. Whenever a Ladder Logic program or C Application changes any of the registers, only data from the changed register is written to the corresponding module parameter.

Module Properties

Field	Description
Module Name	PID
Module Address	This module is assigned a unique module address between 0 and 31. No other PID-type module may use this module address. The default is 0.
AO Start Register	First register of any unused block of 25 consecutive output registers (AO1 through AO25) in the I/O database. Default register: 40001.

Register Data

Register Assignment	Assignment to Module Hardware

AO1	Process Value (PV)
AO2	Error (ER)
AO3	Output Quantity (OP)
AO4	Status Register (SR)
AO5	Integrated Error (IN)
AO6	Setpoint (SP)
AO7	Deadband (DB)
AO8	Gain (GA)
AO9	Reset Time (RE)
AO10	Rate Time (RA)
AO11	Full Scale Output Limit (FS)
AO12	Zero Scale Output Limit (ZE)
AO13	High Alarm Level (HI)
AO14	Low Alarm Level (LO)
AO15	Control Register (CR)
AO16	Input Source (IP)
AO17	Increase Output Address (IO)
AO18	Decrease Output Address (DO)
AO19	Input Bias (IB)
AO20	Output Bias (OB)
AO21	Inhibit Execution Address (IH)
AO22	Alarm Output Address (AO)
AO23	Cascade Setpoint Source (CA)
AO24	Execution Period from PID function.
AO25	Execution Period at Power Up in tenths of seconds.

Notes

Twenty-five holding registers are always assigned to the I/O database when you use this module.

Alarm Output Address - AO

The block alarm output address is a user defined variable which specifies the alarm output address. When a high or low alarm is detected, the digital output address specified in AO will be turned on if the block control register enables the alarms. For more information, see the Status Register section describing the alarm acknowledge bit of SR.

Method One

If the I/O Specification bit in the control register is set to 1, AO may contain the address of any valid Modbus coil register. (e.g. 00014).

Method Two

If the I/O Specification bit in the control register is cleared to 0, AO must contain an absolute address which is calculated as: channel * 8 + bit. Therefore to use channel 5, bit 3 as the alarm output, AO would be defined as 5 * 8 + 3 = 43. The absolute address method is only valid if the Default Register Assignment Table is downloaded to the controller, or if the controller is a Micro16 with firmware version 1.22 or older.

Cascaded Setpoint Source - CA

The cascaded setpoint source block is a user defined variable in the control block that defines the source of cascaded setpoints for secondary cascaded controllers. It contains the block number whose output OP, will provide the setpoint for the PID controller. The output from the block specified in CA becomes the setpoint of the secondary cascaded controller.

The block cascade setpoint is only used by the control block when the block control register is configured as a P, PI, PID controller with setpoint from block CA.

Control Register - CR

The block control register determines the function of the block.

The block control register is a special block variable which determines which functions are engaged in a control block. The block control register is a 16-bit quantity with each bit undertaking special significance. The table below lists the functions of the control register bits.

Function	Bits	Value	Options
Block Output	0,1	0 1 2 3	00 – none other than OP 01 – pulse duration 10 – analog channel 11 – motor pulse duration
Block Input	2,3	0 4 8 12	00 – none (comes from IP) 01 – from output of block IP 10 – analog channel 11 – undefined
Setpoint Source	4	0 16	0 – setpoint is stored in SP 1 – from output of block CA
Block Function	5,6	0 32 64 96	00 – alarm only 01 – P, PI, PD or PID controller 10 – ratio or ratio/bias controller 11 – undefined
Alarm Status	7	0 128	0 – not enabled 1 – alarms active
Square Root of Error	8	0 256	0 – not enabled 1 – take square root of error
Square Root of PV Input	9	0 512	0 – not enabled 1 – take square root of PV input
Alarm Type	10,11	0 1024 2048 3072	00 – absolute level 01 – deviation from setpoint 10 – rate of change 11 – undefined

Setpoint Tracking	12	0 4096	0 – not enabled 1 – SP tracks PV in manual mode
Manual Mode	13	0 8192	0 – non manual mode 1 – manual mode
I/O Specification	14	0 16384	0 – absolute addresses specified from fixed I/O map. 1 – Modbus registers specified
unused	15		

The controller configuration bits should not be changed while the controller is in operation. The only exceptions are the alarm status and manual mode bits. The recommended technique is:

- turn off the controller;
- reconfigure the control register and other variables as required; and
- re-enable the controller.

To enable a function, the corresponding bit in the control register must be set to 1. To disable any of the above functions, the corresponding bit in the control register must be cleared to 0. The simplest method of selecting the proper bits is to add their values shown in the table.

Example

A controller block is to have the following functions enabled: PID controller, analog input, pulse duration output, square root of process value, normal setpoint, alarms engaged, and Modbus I/O specification. The values of the functions are listed below. The value of the control register is the sum of the function values.

Function	Value
PID	32
Analog Input	8
Pulse Duration Output	1
Square Root of PV Input	512
Alarms Enabled	128
Modbus I/O Specification	16384
Value of CR register	17065

Manual Mode

The block manual mode suspends operation of the automatic control algorithm (PID or ratio/bias), but continues operation of other block functions. Manual mode should not be confused with the inhibit execution input function which stops all block functions.

While in manual mode, the process value (PV) is refreshed (upon each block execution) from the previously specified block input source IP. The block output, OP, is maintained at the last value it had before the switch to manual mode. An application program vary the OP register if desired. If time proportioned output is being used, the duty cycle is maintained while in manual mode; and is adjusted for changes in the OP value.

Manual mode is selected by setting the manual mode bit in the control register. The block must be enabled for automatic execution for the block to function, even if there is no intention of using automatic control.

Setpoint Tracking

Setpoint tracking provides a method for obtaining a smooth transition between manual and automatic process control. An operator may manually control an unstable process until it has stabilized; at which point, the operator will shift the block controller to automatic control. Setpoint tracking prevents a disturbance to the process at this point.

This result is accomplished by having the setpoint follow the process value as long as the block controller remains in manual. Were the setpoint not to do so, an error would exist at the time automatic control is engaged. This can lead to large fluctuations in the block controller output (OP) as the controller attempts to remove the error.

Changes to the setpoint, when the block is in manual mode, will be ignored.

Deadband - DB

The block deadband is a user defined variable in the control block that is used by the PID algorithm to determine if the process requires control outputs. If the absolute value of the block error is less than the block deadband, then the block skips execution of the control algorithm. This permits faster execution when the error is within a certain acceptable range or deadband.

To make the block perform a complete execution even on the smallest measurable error the block deadband should be set equal to 0.

To minimize background overhead, PID type blocks should use a reasonable value of deadband. Blocks execute up to five times faster if the error is within the deadband.

Decrease Output - DO

The block decrease output address is a user defined variable in the control block that is used to define a pulse duration or motorized pulse duration output. When the block output, OP is negative, the digital output at DO is turned on for a length of time (in tenths of a second) equaling the absolute value of the block output. If the block output is positive, the digital output at DO is turned off.

Method One

If the I/O Specification bit in the control register is set to 1, DO may contain the address of any valid Modbus coil register. (e.g. 00014)

Method Two

If the I/O Specification bit in the control register is cleared to 0, DO must contain an absolute address which is calculated as: channel * 8 + bit. For example, bit 7 of channel 13 will equal 13 * 8 + 7 = 111. The absolute address method is only valid if the Default Register Assignment Table is downloaded to the controller, or if the controller is a TeleSAFE Micro16 with firmware version 1.22 or older.

Error - ER

The block error is a variable generated by the control block that contains the process error from the most recent calculation. The initial calculation is

ER = SP – PV

If the absolute value of the error is less than the deadband, no further calculation is done and the output of the block does not change.

If the absolute value of the error is equal to or greater than the deadband, then the error is calculated using the formulae below.

ER = SP – PV + DB if the PV is greater than setpoint

ER = SP – PV – DB if the PV is less than the setpoint

This calculation ensures there is no large jump in the error, and a corresponding process disturbance when the process comes out of the deadband.

Full Scale Output - FS

The block full scale output is a user defined variable in the control block used in limiting the maximum block output. If the control block calculates a block output quantity that is greater than the value stored in FS, the block output quantity OP is set equal to the value stored in FS.

The units of the block full scale output vary depending whether the control block is time proportioned or analog output. For time proportioned outputs, the units are tenths of seconds and the value is usually set equal to or less than the block execution time. For analog outputs, the integer is stored in I/O units (32767 to 32767). The block full scale output should always be greater than the block zero scale output.

Gain - GA

Gain is a user defined variable in the control block. It is the proportional gain if the block control register is configured as a P, PI, PD, or PID controller. It is the ratio if the block control register is configured as a ratio or ratio/bias controller.

The value stored in the gain is a 2 decimal place fixed point integer. Since there is no actual decimal point, the value stored in the gain is 100 times the actual gain. For example a gain of 1.50 is stored as 150.

A positive value of gain configures a forward-acting PID controller and a negative value of gain configures a reverse acting controller.

High Alarm Level - HI

The block high alarm level is a user defined variable in the control block that indicates at what value the high alarm is triggered. If the block process value PV exceeds or equals the value stored in HI then the digital output specified in AO is turned on.

The block high alarm level is normally specified in the units of the process value PV. The alarm will only be announced if the block control register is configured for alarms active.

If neither a low alarm nor a high alarm exists, the output specified in AO will be turned off.

Input Bias - IB

The block input bias is a user defined variable in the control block that is used by either the PID or the ratio/bias algorithm to cancel true-zero offset in the input signal to the control block. The value stored in IB is subtracted from the block input before any of the block algorithms execute. The quantity stored in PV already has the input bias subtracted.

The block input bias is usually expressed in the units of the process value PV.

Block input bias can be useful in calibrating input signal sources by storing the actual instrument reading into the input bias under conditions of known true zero process signals.

Inhibit Execution Input - IH

The block inhibit execution input address is a user defined variable in the control block which specifies a digital input bit. It is used to disable or enable the automatic execution of a control block depending upon whether a control bit is on or off. A value of zero stored in IH disables this function.

The block will be prevented from executing whenever the bit whose address is stored in IH is on. When the bit turns off, execution will resume, but the resumption will not be bumpless. If the block input changes during the period execution is inhibited, the change will immediately appear at the block output on resumption of execution.

Method One

If the I/O Specification bit in the control register is set to 1, IH may contain the address of any valid Modbus status register (e.g. 10023).

Method Two

If the I/O Specification bit in the control register is cleared to 0, IH must contain an absolute address (i.e. channel * 8 + bit). Channel 0, bit 0 cannot be used as a valid absolute address for IH. The absolute address method is only valid if the Default Register Assignment Table is downloaded to the controller, or if the controller is a TeleSAFE Micro16 with firmware version 1.22 or older.

Integrated Error - IN

The block integrated error is a variable generated by the control block if it is configured as a PI or PID controller. The value stored in the integrated error is a 2 decimal place fixed point integer. Since there is no actual decimal point, the value stored is 100 times the actual error. For example an integrated error of 71.02 would be stored as 7102.

Changes to IN will not occur under the following conditions:

- Block output tries to exceed FS
- Block output tries to drop below ZE
- Block reset time is equal to zero
- Block inhibit execution input is ON
- The block integral is greater than 32767

- The block integral is less than –32768.

The first two conditions are known as integral anti-windup. The integrated error in a control block can be set to zero by storing 0 in the IN register.

Increase Output - IO

The block increase output address is a user defined variable in the control block that is used to define a block output point as follows:

If the I/O Specification bit in the control register is set to 1.

Output Type	Function of IO
Analog	IO contains a valid Modbus holding register.
time proportioned	IO contains a valid Modbus coil register. When the block output, OP is positive, the digital output at IO is turned on for a length of time (in tenths of a second) equaling the block output. If the block output is negative, the digital output at IO is turned off.

If the I/O Specification bit in the control register is cleared to 0. This address method is only valid if the Default Register Assignment Table is downloaded to the controller, or if the controller is a Micro16 with firmware version 1.22 or older. This is included to provide backward compatibility for older controller.

Output Type	Function of IO
Analog	IO contains the analog channel number.
time proportioned	IO contains an absolute digital address calculated as channel * 8 + bit.

Input Source - IP

The block input source is a user defined variable in the control block that is used by the control block to determine the source of the process value. The process value for the control block is taken from the source specified in IP.

The value in IP is dependent upon the configuration of the block input in the control register (see the Control Register section).

If the I/O Specification bit in the control register is set to 1.

Block Input	Function of IP
None	IP contains the process value. This is useful in running simulations.
analog	IP contains the Modbus input or holding register from which the process value is expected. This is the most often used configuration of a PID controller's process value.
block output	IP contains the control block number from whose output the process value is taken.

If the I/O Specification bit in the control register is cleared to 0. This address method is only valid if the Default Register Assignment Table is downloaded to the controller, or if the controller is a TeleSAFE Micro16 with firmware version 1.22 or older. This is included to provide backward compatibility for older controller.

Block Input	Function of IP
none	IP contains the process value. This is useful in running simulations.
analog	IP contains the analog channel from which the process value is expected. This is the most often used configuration of a PID controller's process value.
block output	IP contains the control block number from whose output the process value is taken.

Low Alarm Level - LO

The block low alarm level is a user defined variable in the control block that indicates at what value the low alarm is triggered. If the block process value PV is less than or equal to the value stored in LO then the digital output specified in AO is turned on.

The block high alarm level is normally specified in the units of the process value PV. The alarm will only be announced if the block control register is configured for alarms active.

If neither a low alarm nor a high alarm exists, the output specified in AO will be turned off.

Output Bias - OB

The block output bias is a user defined variable in the control block that is used by either the PID or the ratio/bias algorithm in calculating the output quantity. The output bias is added to the output of the control algorithm and can be used to shift the output up or down the scale.

Output bias is useful with 4-20 mA outputs. With an analog output module that generates 0-20 mA, an output bias of 6553 will ensure a 4 mA output when the algorithm output equals 0. With an analog output module that generates 4-20 mA, an output bias of 0 should be used.

Output Quantity - OP

The block output quantity is a variable generated by the control block that contains the algorithm output after the addition of output bias. It is a full range integer (–32768 to 32767) but is limited by the quantities stored in the zero scale ZE and the full scale FS.

Process Value - PV

The block process value is a variable generated by the control block that contains the block input (process value) which existed at the most recent execution of the algorithm. The block input can come from an analog channel, another block's output, or a constant generated by a program, as defined by the block control register and

IP.

Rate Time - RA

The block rate time is a user defined variable in the control block that controls the rate gain (or magnitude of derivative action) in a PD or PID controller. The possible range of values is 0 to 32767. The PID algorithm assumes that the rate time is stored in units of tenths of a second.

If RA = 0, no rate (or derivative) action will be used in the block. Maximum rate action occurs when RA = 32767. Minimum rate action occurs when RA = 1. To make a controller P or PI type, RA should equal 0.

Reset Time - RE

The block reset time is a user defined variable in the control block that controls the reset gain (or magnitude of integral action) in a PI or PID controller. The possible range of values is 0 to 32767. The PID algorithm assumes that the reset time stored in RE is in units of tenths of a second.

If RE = 0, no reset (or integral) action will be used in the block. Maximum reset action occurs when RE = 1. Minimum reset action occurs when RE = 32767. For P or PD controllers, RE should equal 0.

Setpoint - SP

The block setpoint is a user defined variable in the control block that is used to calculate the error in the PID algorithm. It is a dimension-less 16-bit signed integer (–32767 to 32767).

If the block has a cascaded setpoint, then SP is not user definable, but will be defined by the block and will equal the value of the cascaded setpoint. SP always contains the setpoint which is used by the block algorithm, regardless whether it is user defined or cascaded.

Status Register - SR

The block status register is a block variable which reports the status of certain conditions in a block. Application programs can read the status register at any time. The table below lists the individual bits of the status register and their significance.

Bit	Value	Status
0	1	reserved for future use
1	2	BAD I/O ADDRESS error on input to block
2	4	high alarm condition on input to block
3	8	low alarm condition on input to block
4	16	external inhibit execution input is on
5	32	loop is outside of setpoint deadband
6	64	derivative gain clamped at maximum (rapid PV change)
7	128	BAD I/O ADDRESS error on output from block
8	256	block output clamped at full scale limit
9	512	block output clamped at zero scale limit
10	1024	reserved for future use
11	2048	control block is executing (not-necessarily in AUTO mode)
12	4096	alarm acknowledge bit
13	8192	control block is in manual mode
14	16384	reserved for future use
15	32768	reserved for future use

An application program may test for a bit in the status register by ANDing the register with the value of the bit to be tested. If the result equals the value of the bit, the status condition signified by that bit exists.

The block status register reports the status of conditions affecting the block. Refer to the Status Register section for a complete discussion.

Zero Scale Output - ZE

The block zero scale output is a user defined variable in the control block used in limiting the minimum block output quantity. If the control block calculates a block output quantity that is less than the value stored in ZE, the block output quantity OP is set equal to the value stored in ZE.

The units of the block zero scale output vary depending whether the control block is time proportioned or analog output. For time proportioned outputs, the units are tenths of seconds and the value is usually set equal to the negative block execution time (i.e. time x –1). For analog outputs, the value is stored in I/O counts. The block zero scale output should always be less than the block full scale output.

Real Time Clock and Alarm Module

The **Real Time Clock and Alarm** module provides access to the controller real time clock data and alarm settings. Real Time Clock and Alarm settings are assigned from eleven consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller.

The clock data from the first 7 holding registers is written to the module parameters, whenever any of these registers are changed by a Ladders program or C Application. Otherwise, these holding registers are updated continuously with data read from the real time clock.

Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users need to ensure the ladder function is only energized for one program scan. If the ladder function is always energized the real time clock will not work as expected..

The alarm settings are updated continuously with data read from the last 4 holding registers.

Module Properties

Field	Description
Module Name	Real Time Clock and Alarm Module
AO Start Register	First register of any unused block of 11 consecutive output registers (A01 to A011). Default setting: 40001.

Register Data

Register Assignment	Assignment to Module Hardware	
AO1	Real time clock hour	0 to 23
AO2	Real time clock minute	0 to 59
AO3	Real time clock second	0 to 59
AO4	Real time clock year	00 to 99
AO5	Real time clock month	1 to 12
AO6	Real time clock day	1 to 31
AO7	Real time clock day of week	1 to 7 1 = Sunday 2 = Monday 7 = Saturday
AO8	Alarm hour	0 to 23
AO9	Alarm minute	0 to 59
AO10	Alarm second	0 to 59
AO11	Alarm control	0 = no alarm 1 = absolute time 2 = elapsed time alarm (relative from now)[1]

[1] The Alarm Control register (AO11) enables or disables the alarm. A value of 0 in this register disables the alarm. A value of 1 sets an alarm to be triggered in absolute time, based on the values in the preceding three registers. A value of 2 sets the alarm to be triggered at a latter time, starting now.

Notes

1. The alarm settings above and those in the Sleep (**SLP**) ladder element will trigger the same interrupt. Using both alarm settings, therefore, in the same program is not recommended.
2. SCADAPack Controllers have a hardware based real-time clock that independently maintains the time and date for the operating system. The time and date remain accurate during power-off. The calendar automatically handles leap years.
3. All seven registers for the real time clock must be set to valid values for the clock to operate properly.
4. Eleven holding registers are always assigned to the I/O database when you use this module.

Save Setting to EEPROM Module

The **Save Setting to EEPROM** module provides a way to save controller settings to EEPROM. Saving to EEPROM is controlled by one assigned coil register. The controller settings are saved when the coil register is set to 1. Once saved the coil register is automatically reset to zero.

Module Properties

Field	Description
Module Name	Save Setting to EEPROM Module
DO Start Register	Displays any unused coil register. The default setting is 1.

Register Data

Register Assignment	Assignment to Module Hardware
DO Start Register	Save the following controller settings: Serial port settings Protocol settings Enable store and forward settings LED power settings Mask for wake-up sources HART interface configuration Execution period on power-up for each PID

Serial Port DTR Off Module

The **Serial Port DTR Off** module provides a way of controlling the DTR signal for a specific serial port. When the assigned coil is energized, DTR for the serial port is de-asserted. DTR is constantly turned off when the coil is energized.

When the assigned coil is de-energized DTR for the serial port is asserted. The DTR signal is not constantly asserted. It is asserted on the transition from an energized coil to a de-energized coil. Since DTR is not constantly re-asserted other programs will be able to control the DTR setting. This way other ladder blocks and programs will be able to control the DTR settings.

Module Properties

Field	Description
Module Name	Serial Port DTR Off Module
Comm Interface	Displays the serial or IP interface (com1 or com4) associated with this module. The default setting is com1.
DO Start Register	Displays the first digital output register. The default setting is 1.

Register Data

Register Assignment	Assignment to Module Hardware
Start Register	Controls DTR signal 0 = DTR asserted 1 = DTR not asserted

Default: 0 = DTR asserted

Notes

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the Telepace program.

Serial Port Protocol Settings Method 1 Module

The **Serial Port Protocol Settings Method 1** module provides control of the protocol settings for a specific serial port. Settings are assigned from three consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all three holding registers is written to the module parameters, whenever any of the registers change.

Use this module only when protocol settings need to be changed within a Ladders program. It is more convenient to use the *Controller Serial Ports Settings* dialog to download protocol settings with the program. Refer to the *Serial Ports* section in this manual for further information on serial port settings.

Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users need to ensure the ladder function is only energized for one program scan. If the ladder function is always energized the serial communication will not work as expected..

Module Properties

Field	Description
Module Name	Protocol Settings Method 1 Module
Comm Interface	Displays the serial or IP interface (com1 or com4) associated with this module. The default setting is com1.
AO Start Register	First register of any unused block of 3 consecutive output registers (AO1 through AO3) in the I/O database. Default register: 40001.

Register Data

Register Assignment	Assignment to Module Hardware
AO1 (Start Register)	Protocol type 0 = none 1 = Modbus RTU 2 = Modbus ASCII 3 = DF1 Full Duplex BCC 4 = DF1 Full Duplex CRC 5 = DF1 Half Duplex BCC 6 = DF1 Half Duplex CRC 7 = DNP 8 = PPP Default : 1 = Modbus RTU
AO2	Protocol station number 1 to 255 in Modbus 0 to 254 in DF1 Default: 1
AO3	Store and forward enable 0 = disabled 1 = enabled Default : 0 = disabled

Notes

1. This register assignment module replaces the CNFG Protocol Settings module used in earlier versions of Telepace.
2. For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 or 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the Telepace program.
3. Three holding registers are always assigned to the I/O database when you use this module.
4. Refer to the *TeleBUS Protocols User Manual* for further information on protocols.

Related Topics

Serial Port Protocol Settings Method 2

Serial Port Protocol Settings Method 3

Serial Port Protocol Settings Method 2 Module

The **Serial Port Protocol Settings Method 2** module provides control of the protocol settings for a specific serial port. This module supports the use of extended addressing.

Settings are assigned from four consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all four holding registers is written to the module parameters, whenever any of the registers change.

Use this module only when protocol settings need to be changed within a Ladders program. It is more convenient to use the *Controller Serial Ports Settings* to download protocol settings with the program. .

Note:	Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users must ensure the ladder function is only

energized for one program scan. If the ladder function is always energized problems with the serial communication will result.

Module Properties

Field	Description
Module Name	Protocol Settings Method 1 Module
Comm Interface	Displays the serial or IP interface (com1 or com4) associated with this module. The default setting is com1.
AO Start Register	First register of any unused block of 4 consecutive output registers (AO1 through AO4) in the I/O database. Default register: 40001.

Register Data

Register Assignment	Assignment to Module Hardware
AO1 (Start Register)	Protocol type 0 = none 1 = Modbus RTU 2 = Modbus ASCII 3 = DF1 Full Duplex BCC 4 = DF1 Full Duplex CRC 5 = DF1 Half Duplex BCC 6 = DF1 Half Duplex CRC 7 = DNP 8 = PPP Default : 1 = Modbus RTU
AO2	Protocol station number 1 to 255 in standard Modbus 1 to 65534 on extended Modbus 0 to 254 in DF1 Default: 1
AO3	Store and forward enable 0 = disabled 1 = enabled Default : 0 = disabled
AO4	Protocol addressing mode 0 = standard 1 = extended Default: 0 = standard

Notes

1. For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the Telepace program.
2. Extended addressing is supported only by the Modbus RTU and Modbus ASCII protocols.
3. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the Telepace program.
4. Four holding registers are always assigned to the I/O database when you use this module.
5. Refer to the [*TeleBUS Protocols User Manual]* for further information on protocols.

Related Topics

Serial Port Protocol Settings Method 1

Serial Port Protocol Settings Method 3

Serial Port Protocol Settings Method 3 Module

The **Serial Port Protocol Settings Method 3** module rovides control of the protocol settings for a specific serial port. This module supports extended addressing and Enron Modbus parameters.

Settings are assigned from six consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all six holding registers is written to the module parameters, whenever any of the registers change.

Use this module only when protocol settings need to be changed within a Ladders program. It is more convenient to use the *Controller Serial Ports Settings* dialog to download protocol settings with the program. Refer to the *Serial Ports* section in this manual for further information on serial port settings.

Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users need to ensure the ladder function is only energized for one program scan. If the ladder function is always energized communication will not work as expected..

Module Properties

Field	Description
Module Name	Protocol Settings Method 1 Module
Comm Interface	Displays the serial or IP interface (com1 or com4) associated with this module. The default setting is com1.
AO Start Register	First register of any unused block of 6 consecutive output registers (AO1 through AO6) in the I/O database. Default register: 40001.

Register Data

Register Assignment	Assignment to Module Hardware
AO1 (Start Register)	Protocol type Protocol type 0 = none 1 = Modbus RTU 2 = Modbus ASCII 3 = DF1 Full Duplex BCC 4 = DF1 Full Duplex CRC 5 = DF1 Half Duplex BCC 6 = DF1 Half Duplex CRC 7 = DNP 8 = PPP Default : 1 = Modbus RTU
AO2	Protocol station number 1 to 255 in standard Modbus 1 to 65534 in extended Modbus 0 to 254 in DF1 Default : 1
AO3	Store and forward enable 0 = disabled
AO4	Protocol addressing mode 0 = standard 1 = extended Default: 0 = standard
AO5	Enron Modbus enable 0 = disabled 1 = enabled Default : 0 = disabled
AO6	Enron Modbus station number 1 to 255 in standard Modbus 1 to 65534 in extended Modbus Default : 1

Notes

1. For SCADAPack, SCADAPack Light and SCADAPack Plus controllers com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the Telepace program. Extended addressing is supported only by the Modbus RTU and Modbus ASCII protocols.
2. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the Telepace program.
3. Six holding registers are always assigned to the I/O database when you use this module.
4. Refer to the *TeleBUS Protocols User Manual* for further information on protocols.

Related Topics

Serial Port Protocol Settings Method 1

Serial Port Protocol Settings Method 2

Serial Port Settings Module

The **Serial Port Settings** module controls the configuration settings for a serial port. Settings are assigned from nine consecutive holding registers. The holding registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all 9 holding registers is written to the module parameters, whenever any of the registers change.

Use this module only when serial port settings need to be changed within a Ladders program. It is more convenient to use the *Controller Serial Ports Settings* dialog to download settings with the program. Refer to the *Serial Ports* section under *Controller Menu,* in this manual, for further information on serial port settings.

Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users need to ensure the ladder function is only energized for one program scan. If the ladder function is always energized communication will not work as expected..

Module Properties

Field	Description
Module Name	Protocol Settings Method 1 Module
Comm Interface	Displays the serial or IP interface (com1 or com4) associated with this module. The default setting is com1.
AO Start Register	First register of any unused block of 9 consecutive output registers (AO1 through AO9) in the I/O database. Default register: 40001.

Register Data

Register Assignment	Assignment to Module Hardware
AO1 Start Register	Baud rate 0 = 75 baud 1 = 110 baud 2 = 150 baud 3 = 300 baud 4 = 600 baud 5 = 1200 baud 6 = 2400 baud 7 = 4800 baud 8 = 9600 baud 9 = 19200 baud

	10 =38400 11 = 115200 com3, com4 only 12 = 57600 com3, com4 only Default : 8 = 9600 baud
AO2	Duplex 0 = half duplex 1 = full duplex com1, com2 Default : 1 = full com3, com4 Default : 0 = half
AO3	Parity 0 = none 1 = even 2 = odd Default : 0 = none
AO4	Data bits 0 = 7 bits 1 = 8 bits Default : 1 = 8 bits
AO5	Stop bits 0 = 1 bit 1 = 2 bits Default : 0 = 1 bit
AO6	Receiver flow control 0 = none 1 = Modbus RTU (default)
AO7	Transmitter flow control Com1, Com2 0 = none (default) 1 = XON/XOFF Com3, Com4 0 = none (default) 1 = Ignore CTS
AO8	Port type 0 = automatic 1 = RS232 3 = RS485 6 = RS232 MODEM 7 = RS232 Collision Avoidance Default : 1 = RS232
AO9	Serial time-out delay 0 to 65535 (x 0.1 seconds) Default : 600 = 60 seconds

Notes

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the Telepace program.

Store and Forward Module

The **Store and Forward** module controls the store and forward table. Store and Forward table data is assigned from 512 consecutive holding registers. One coil register is assigned to clear the translation table.

This module is used by the SCADAPack controller only. For SCADAPack 32 controllers use the Store and Forward command in Telepace.

The holding registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the table is updated continuously with data read from the holding registers. The table is cleared when the coil register is set to 1. Once cleared the coil register is automatically reset to 0.

Module Properties

Field	Description
Module Name	Protocol Settings Method 1 Module
AO Start Register	First register of any unused block of 512 consecutive output registers (AO1 through AO512) in the I/O database. Default register: 40001.
DO Start Register	Any unused digital output register block (DO1) in the I/O database. Default register: 00001.

Coil Register Data

Register Assignment	Assignment to Module Hardware
AO Start Register	Clear Translation table Translation table entry 0 Port A 0 = com1 1 = com2 2 = com3 3 = com4 Default : 0
AO2	Translation table entry 0 Station A 0 to 255 standard addressing 0 to 65534 extended addressing 65535 = disable Default : 65535
AO3	Translation table entry 0 Port B

	0 = com1 1 = com2 2 = com3 3 = com4 Default : 0
AO4	Translation table entry 0 Station B 0 to 255 standard addressing 0 to 65534 extended addressing 65535 = disable Default : 65535
AO5 to AO8	Translation table entry 1
AO9 to AO12	Translation table entry 2
AO13 to AO512	Translation table entries 3 to 127

Notes

1. Refer to the *TeleBUS Protocols User Manual* for more information on Store and Forward messaging.
2. For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the Telepace program.
3. Store and forward messaging is enabled on the required serial ports using the module **CNFG Protocol settings**.

Related Topics

CNFG Protocol Settings Method 1

Serial Port Protocol Settings Method 1

Serial Port Protocol Settings Method 2

Serial Port Protocol Settings Method 3

Counter Modules

Counter input modules are used to assign data from physical counter inputs to input registers in the I/O database. The physical counter inputs are specific SCADAPack I/O I/O modules and controller counter inputs.

Counter input modules may assign data to any input registers that are not used by another Analog Input module or Counter input module.

All I/O database input registers that are not assigned to any other I/O modules may be used as general purpose input registers in a ladder program.

Related Topics

5410 Four Channel Module

Controller Counter Inputs

Controller Interrupt Inputs

Controller Counter Inputs Module

The **Controller Counter Inputs** module reads the controller board counter inputs on the the following SCADAPack controllers. The number of supported Controller Counter Inputs is also listed.

- SCADAPack (3 Counter Inputs)
- SCADAPack Light (3 Counter Inputs)
- SCADAPack Plus (3 Counter Inputs)
- SCADAPack 32P (3 Counter Inputs)
- SCADAPack 100 (1 Counter Input)
- SCADAPack LP (3 Counter Inputs)

Data is assigned to six consecutive input registers. The input registers are updated continuously with data read from the counters.

Each counter is a 32-bit number, stored in two 16-bit registers. The first register holds the less significant 16 bits of the counter. The second register holds the more significant 16 bits of the counter. To obtain the correct count, both registers must be read with the same protocol read command. Ladder Logic programs can read the registers in any order, provided both registers are examined on the same pass through the program.

The maximum count is 4,294,967,295. Counters roll over to 0 when the maximum count is exceeded.

Module Properties

Field	Description
Module Name	Controller Counter Inputs
AI Start Register	First register of any unused blocks of 6 consecutive input registers. Default register: 30001. **Note:** The number of digital inputs that may be used depends on the controller type. See list above. Start Register to Start Register +1 Controller counter input 0 (32-bit register) Start Register + 2 to Start Register +3 Controller counter input 1 (32-bit register) Start Register + 4 to Start Register +5 Controller counter input 2 (32-bit register)

Controller Interrupt Input Counter Module

The **Controller Interrupt Counter Status** module reads the interrupt input on the SCADAPack controllers as a counter input. Data is assigned to two consecutive input registers. The input registers are updated continuously with data read from the counter.

The counter is a 32-bit number, stored in two 16-bit registers. The first register holds the least significant 16 bits of the counter. The second register holds the most significant 16 bits of the counter. To obtain the correct count, both registers must be read with the same protocol read command. Ladder Logic programs can read the registers in any order, provided both registers are examined on the same pass through the program.

The maximum count is 4,294,967,295. The counter rolls over to 0 when the maximum count is exceeded.

Module Properties

Field	Description
Module Name	Controller Interrupt Input Counter
AI Start Register	First register of any unused blocks of 2 consecutive input registers. Default register: 30001. Start Register to Start Register +1 Interrupt (INT) input (32-bit register)

5410 Sixteen Channel Module

The **5410 Sixteen Channel module** module provides four counter inputs. Data is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the counters.

Each counter is a 32-bit number, stored in two 16-bit registers. The first register holds the least significant 16 bits of the counter. The second register holds the most significant 16 bits of the counter. To obtain the correct count, both registers must be read with the same Modbus protocol read command. Ladder Logic programs can read the registers in any order, provided both registers are examined on the same pass through the program.

The maximum count is 4,294,967,295. Counters roll back to 0 when the maximum count is exceeded.

Module Properties

Field	Description
Module Name	5410 Sixteen Channel Module
Module Address	This module is assigned a unique module address between 0 and 15. No other CNTR-type module may use this module address. The default is 0.
AI Start Register	First register of any unused block of 8 consecutive input registers (AI1 to AI8) in the I/O database. Default register: 30001.

Register Data

Register Assignment	Assignment to Module Hardware
AO1 and AO2	Counter input 0 (32-bit register)
AO3 and AO4	Counter input 1 (32-bit register)
AO5 and AO6	Counter input 2 (32-bit register)
AO7 and AO8	Counter input 3 (32-bit register)

Notes

Refer to the *5410 High Speed Counter Input Module User Manual* for more information.

Diagnostic Modules

Diagnostic modules are used to assign diagnostic data from the controller to input or status registers in the I/O database. The diagnostic data is used to monitor internal controller data such as controller status code, the force LED, serial communication status and protocol status.

Diagnostic I/O modules assign data to I/O database input or status registers, depending on the type of data. Diagnostic I/O modules may assign data to any input or status registers in the I/O database that is not being used by another I/O module.

All I/O database input and status registers that are not assigned to any other I/O modules may be used as general purpose input or status registers in a ladder program.

Related Topics

Controller IP Connections

Controller Logic Status

Controller Status Code

DNP Connection Status

DNP Event Status

DNP Port Status

DNP Station Status

Modbus Protocol Status

Register Force Indicator

Serial Port Comm Status

Serial Port Protocol Status

Controller IP Connections Module

The **Controller IP Connections** module provides a summary of the IP connections. The connection information is assigned to three consecutive input registers. The

input registers are updated continuously with connection data.

Note:	Only the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers use this module.

The connection statistics include all IP connection types.

Module Properties

Field	Description
Module Name	Controller IP Connections
AI Start Register	First register of any unused blocks of three consecutive input registers. Default register: 30001. Start Register + 0 (Number of current slave IP connections) Start Register + 1 (Number of current master IP connections) Start Register + 2 (Number of free IP connections)

Controller Logic Status Module

The **Controller Logic Status** module reads the Ladder Logic Application status in the RAM and FLASH memory and assigns it to two input registers. The input registers are updated continuously with the current application status.

Module Properties

Field	Description
Module Name	Controller Logic Status Module
AI Start Register	First register of any unused blocks of two consecutive input registers. Default register: 30001. Start Register +0 (Ladder Logic Program Status) 0 = No Program 1 = Program Stopped 2 = Program Running in Debug Mode 3 = Program Running Start Register + 1 (Type of memory where Ladder Logic program is loaded) 0 = No program in RAM or FLASH; 1 = Program loaded in RAM 2 = Program loaded in FLASH 3 = Program loaded in RAM and FLASH

Controller Status Code

The **Controller Status Code** module reads the controller status code and assigns it in one input register. The input register is updated continuously with the status code.

The Status LED blinks a binary sequence equal to the controller status code. Note that if a "*Register assignment table checksum error*" occurs, only the Status LED will indicate this status code. The I/O scan is disabled if this error occurs, so that the **Controller Status Code** module is no longer updated.

Module Properties

Field	Description
Module Name	Controller Status Code
AI Start Register	Any unused input register. Default register: 30001. Controller status code: 0 = Normal 1 = I/O module communication failure 2 = Register Assignment Checksum error

Status LED I/O Error Indication

When the Status LED flashes the error code 1 (i.e., a short flash, once every second), there is a communication failure with one or more I/O module. To correct the problem, do one of the following:

1. Ensure that every module contained in the Register Assignment is connected to the controller. Check that the module address selected for each module agrees with the selection made in the Register Assignment.
2. If a module is currently not connected to the controller, delete it from the Register Assignment. Download the new Register Assignment to the controller.
3. If a module is still suspect of having failed, confirm the failure by removing the module from the Register Assignment. Download the new Register Assignment to the controller. The Status LED should stop flashing.
4. If unused modules must be intentionally left in the Register Assignment, the I/O error indication may be disabled from a selection box on the Register Assignment dialog.

DNP Connection Status Module

The **DNP Connection Status** module determines the connection status of a remote DNP station, by repeatedly sending a short message to the selected remote station and monitoring the response. The connection status information is assigned to an input register, which is updated continuously.

One holding register is assigned to specify the repeat time period for polling the remote station. A second holding register specifies the remote station address.

Changing the station number (AO Start Register +0) does not stop the poll to the previous station. Another poll to the new station value is added to the DNP routing table. Only downloading the DNP configuration will reset all previous polls.

This module is used by the SCADAPack 32, SCADAPack 314, SCADAPack 330/334 and SCADAPack 350 controllers only

Module Properties

Field	Description
Module Name	Controller Status Code
AI Start Register	Any unused input register. Default register: 30001. Connection Status Code:

	0 = Connection is OK 1 = Connection has failed
AO Start Register	First register of any unused blocks of two consecutive holding registers. Default register: 40001. Start Register + 0 (remote DNP station address) Start Register + 1 (Repeat rate, in seconds, for connection request message)

DNP Event Status

The **DNP Event Status** module provides diagnostic data about DNP change events. The number of change events stored in the event buffers is totaled for all DNP classes, and for all DNP point types.

The status information is assigned to twelve consecutive input registers. The input registers are updated continuously with data concerning the DNP change events.

One coil register is assigned to clear the change event buffers. The data is cleared when the coil register is set to 1. Once the data is cleared the coil register is automatically reset to 0.

Module Properties

Field	Description
Module Name	DNP Event Status Module
AI Start Register	First register of any unused block of 12 consecutive input registers. Default register: 30001. Start Register + 0 (DNP event count for Binary Inputs) Start Register + 1 (DNP event count for 16-bit Counter Inputs) Start Register + 2 (DNP event count for 32-bit Counter Inputs) Start Register + 3 (DNP event count for 16-bit Analog Inputs) Start Register + 4 (DNP event count for 32-bit Analog inputs) Start Register + 5 (DNP event count for short-float analog inputs) Start Register + 6 (DNP event count for Class 1 Data) Start Register + 7 (DNP event count for Class 2 Data) Start Register + 8 (DNP event count for Class 3 Data) Start Register + 9 (Reserved for future use) Start Register + 10 (Reserved for future use) Start Register + 11 (Reserved for future use)
DO Start Register	First register of any unused block of one sequential digital output registers. Start Register (ON = Clear event change counters; OFF do not clear event change buffers)

DNP Port Status

The **DNP Port Status** module provides DNP diagnostic data for a specific communication port. Diagnostic information is totaled for all DNP connections on the selected communication port.

The status information is assigned to ten consecutive input registers. The input registers are updated continuously with the DNP diagnostic data.

One coil register is assigned to clear the diagnostic data. The data is cleared when the coil register is set to 1. Once the data is cleared the coil register is automatically reset to 0.

Module Properties

Field	Description
Module Name	DNP Port Status Module
Comm Interface	Communication interface to monitor for DNP diagnostic data. Depending on the controller type the selection is one of: Com1, Com2, Com3, Com4 and LAN/PPP
AI Start Register	First register of any unused block of 10 consecutive input registers. Default value: 30001. Start Register + 0 (**DNP message successes:** The number of successful DNP message transactions initiated by this station) Start Register + 1 (**DNP message failures:** The number of failed DNP message transactions initiated by this station) Start Register + 2 (**DNP fails since last success:** Number of consecutive failed DNP message transactions initiated by this station since the last success) Start Register + 3 (**DNP format errors:** The number of received DNP message frames with formatting errors (including checksum errors and unrecognized commands) Start Register + 4 (**DNP message frames received:** The number of DNP message frames received) Start Register + 5 (**DNP message frames sent:** The number of DNP message frames sent. This includes any message retries at application or data link layer.) Start Register + 6 (**DNP messages received:** The number of complete DNP messages received) Start Register + 7 (**DNP messages sent:** The number of complete DNP messages sent. This includes any message retries at the application layer. Start Register + 8 (Reserved for future use) Start Register + 9 (Reserved for future use)
DO Start Register	First register of any unused block of one consecutive digital output register. The default setting is 1. Start Register (ON = Clear diagnostic data; OFF = Do not clear diagnostic data)

DNP Station Status

The **DNP Station Status** module provides DNP diagnostic data for a specific DNP remote station address. Diagnostic information is totaled for all DNP communication with the station.

The status information is assigned to nine consecutive input registers. The input registers are updated continuously with data concerning the DNP status of the specific interface.

One coil register is assigned to clear the diagnostic data. The data is cleared when the coil register is set to 1. Once the data is cleared the coil register is automatically reset to 0.

One holding register specifies the remote DNP station address.

Module Properties

Field	Description
Module Name	DNP Port Status
AI Start Register	First register of any unused block of 10 consecutive input registers. Default value: 30001. Start Register + 0 (**DNP message successes:** The number of successful DNP message transactions initiated by this station) Start Register + 1 (**DNP message failures:** The number of failed DNP message transactions initiated by this station) Start Register + 2 (**DNP fails since last success:** Number of consecutive failed DNP message transactions initiated by this station since the last success) Start Register + 3 (**DNP format errors:** The number of received DNP message frames with formatting errors, including checksum errors and unrecognized commands) Start Register + 4 (**DNP message frames received**: The number of DNP message frames received) Start Register + 5 (**DNP message frames sent**: The number of DNP message frames sent. This includes any message retries at application or data link layer.) Start Register + 6 (**DNP messages received**: The number of complete DNP messages received) Start Register + 7 (**DNP messages sent:** The number of complete DNP messages sent. This includes any message retries at the application layer. Start Register + 8 (Reserved for future use) Start Register + 9 (Reserved for future use)
DO Start Register	First register of any unused block of one consecutive digital output register. The default setting is 1. Start Register (ON = Clear diagnostic data; OFF = Do not clear diagnostic data)
AO Start Register	First register of any unused block of one consecutive holding register. The default value is 40001. Start Register (DNP remote station address)

Modbus Protocol Status Module

The **Modbus Protocol Status** module provides diagnostic data about Modbus protocols for a specific communication interface. Diagnostic information is totaled for all Modbus protocols on the selected communication interface. When multiple Modbus IP connections exist on the same interface, diagnostic information is totaled for all connections.

This module is required when the communication interface is a LAN connection. For serial port communication interfaces this module and the Serial Port Protocol Status register assignment may be used.

The status information is assigned to nine consecutive input registers. The input registers are updated continuously with data concerning the Modbus protocol status of the specific interface.

One coil register is assigned to clear the diagnostic data. The data is cleared when the coil register is set to 1. Once the data is cleared the coil register is automatically reset to 0.

Module Properties

Field	Description
Module Name	Modbus Protocol Status
Comm Interface	Communication interface to monitor for Modbus protocol diagnostic data. Depending on the controller type the selection is one of: Com1, Com2, Com3, Com4 and LAN/PPP
AI Start Register	First register of any unused block of 9 consecutive input registers. Default register: 30001. Start Register + 0 (**Modbus command errors:** This counts the number of received messages containing an unsupported Modbus function code. It also counts the number of otherwise valid commands that are rejected because of a controller lock.) Start Register + 1 (**Modbus format errors:** This counts the number of received messages that don't have the correct framing or internal formatting. Basically this is a count of how many messages are garbled. For Modbus ASCII this includes: • start of message missing • end of message missing • unexpected start of message inside message • unexpected end of message • invalid characters in message For Modbus RTU this includes: • received data is too short to be a message • received data is too long to be a message (it didn't fit in the 256 byte buffer).) Start Register + 2 (**Modbus checksum errors:** This counts the number of received messages with a bad checksum.)

	Start Register + 3 (**Modbus slave commands received**: This counts the number of slave commands received)
	Start Register + 4 (**Modbus master commands sent:** This counts the number of master commands sent)
	Start Register + 5 (**Modbus master responses received:** The number number of master responses received)
	Start Register + 6 (**Modbus slave responses sent:** The number number of slave responses sent)
	Start Register + 7 (**Modbus stored messages:** number of messages stored (received) for forwarding)
	Start Register + 8 (**Modbus forwarded messages:** number of messages forwarded (transmitted))
DO Start Register	First register of any unused block of one consecutive digital output register. The default setting is 1.
	Start Register (ON = Clear diagnostic data; OFF = Do not clear diagnostic data)

Register Force Status

The **Register Force Status** module is used to determine if a Modbus register in a SCADAPack controller is forced to some state or value. Il user accessible Modbus registers in a SCADAPack controller can be forced to a desired state or value. This module reads the 'forced' state of a SCADAPack controller, and assigns it to one status register. The status register is updated continuously with the current 'forced' state of the controller.

Note that the Force LED on SCADAPack controllers, excluding the SCADAPack programmable controllers, also indicates if a Modbus register is forced.

Module Properties

Field	Description
Module Name	Register Force Indicator Module
DI Start Register	Any unused digital input register. Default register: 10001. 0 = Modbus register is not forced 1 = Modbus register is forced Force LED: 0 = Force LED is OFF 1 = Force LED is ON

Serial Port Comm Status

The **Serial Port Comm Status** module provides diagnostic data about a specific serial port. The status information is assigned to five consecutive input registers which act as counters for the diagnostic data. The input registers are updated continuously with data concerning the communication status of the serial port.

To clear the diagnostic data counters use the Clear Serial Port Counters register assignment.

Module Properties

Field	Description
Module Name	Serial Port Comm Status Module
Comm Interface	Communication interface to monitor for Serial Port Comm Status diagnostic data. Depending on the controller type the selection is one of: Com1, Com2, Com3 and Com4. The default is Com1.
AI Start Register	First register of any unused block of 5 consecutive analog input registers. Default register: 30001.
	Start Register + 0 (**Framing errors:** The number of received bytes where too few or too many bits were received. This indicates that data is being garbled or lost on the communication medium or that there is noise on the medium)
	Start Register + 1 (**Parity errors:** The number of received bytes where the parity bit did not match the parity of the data bits. This indicates that data is being garbled or lost on the communication medium or that there is noise on the medium)
	Start Register + 2 (**Character overrun errors:** The number of times a character was received before the previous character could be read. If this occurs, the processor is too busy to read the serial port. It could indicate a problem with the firmware (since this should never happen) but it might be caused by noise, especially if it occurs in conjunction with parity and framing errors)
	Start Register + 3 (**Buffer overrun errors:** The number of characters discarded because the 512 byte receive buffer is full. If a protocol is running on the port, this indicates that the message sent to the controller is too long. If no protocol is running, this indicates that the application program that should be reading the data from the port isn't)
	Start Register + 4 (Integer value indicating the status of serial port I/O lines CTS and DCD: The LSB = state of the CTS line and Second to LSB = state of DCD input) Example: Value of 3 implies CTS and DCD is ON. Value of 1 implies CTS is ON and DCD is OFF.

Serial Port Protocol Status Module

The **Serial Port Protocol Status** module provides diagnostic data about a specific serial port. The status information is assigned to ten consecutive input registers which act as counters for the diagnostic data. The input registers are updated continuously with data concerning the protocol status of the serial port.

To clear the counters use the Clear Serial Port Protocol Counters register assignment.

Module Properties

Field	Description
Module Name	Serial Port Protocol Status Module
Comm Interface	Communication interface to monitor for Serial Port Protocol Status diagnostic data. Depending on the controller type the selection is one of: Com1, Com2, Com3 and Com4.
AI Start Register	First register of any unused block of 10 consecutive analog input registers. Default register: 30001.
	First register of any unused block of 9 consecutive input registers. Default register: 30001.
	Start Register + 0 (**Modbus command errors:** This counts the number of received messages containing an unsupported Modbus function code. It also counts the number of otherwise valid commands that are rejected because of a controller lock.)
	Start Register + 1 (**Modbus format errors:** This counts the number of received messages that don't have the correct framing or internal formatting. Basically this is a count of how many messages are garbled. For Modbus ASCII this includes: • start of message missing • end of message missing • unexpected start of message inside message • unexpected end of message • invalid characters in message For Modbus RTU this includes: • received data is too short to be a message • received data is too long to be a message (it didn't fit in the 256 byte buffer).)
	Start Register + 2 (**Modbus checksum errors:** This counts the number of received messages with a bad checksum.)
	Start Register + 3 (**Modbus slave commands received**: This counts the number of slave commands received)
	Start Register + 4 (**Modbus master commands sent:** This counts the number of master commands sent)
	Start Register + 5 (**Modbus master responses received:** The number number of master responses received)
	Start Register + 6 (**Modbus slave responses sent:** The number number of slave responses sent)
	Start Register + 7 (**Protocol master command status:** Status codes are shown below: 0 = message sent, waiting for response 1 = response received, no error occurred 2 = not used 3 = bad value in function code register 4 = bad value in slave controller address register 5 = bad value in slave register address 6 = bad value in length register 7 = serial port or protocol is invalid 8 = not used 9 = specified protocol is not supported by this controller 11 = an unexpected length response was received. This means corrupt messages for regular Modbus. Check the format and checksum counters. For Enron Modbus, this is normal. 12 = response timeout 24 = exception response: invalid function code 25 = exception response: invalid address 26 = exception response: invalid value 27 = protocol is invalid or serial port queue is full 28 = slave and master stations are equal; they must be different 29 = exception response: slave device failure 30 = exception response: slave device busy
	Start Register + 8 (**Modbus stored messages:** number of messages stored (received) for forwarding)
	Start Register + 9 (**Modbus forwarded messages:** number of messages forwarded (transmitted))

Digital Input Modules

Digital input modules are used to assign data from physical digital inputs to status registers in the I/O database. The physical digital inputs are specific SCADAPack I/O I/O modules, generic I/O modules, and controller digital inputs.

Digital input modules may assign data to any status registers in the I/O database that are not being used by another Digital input module. Status registers are referred to as 1nnnn registers throughout this manual.

All I/O database status registers that are not assigned to any other I/O modules may be used as general-purpose status registers in a ladder program.

Related Topics

5401 Eight Channel Module

5402 Sixteen Channel Module

Controller Digital Inputs Module

The **Controller Digital Inputs** module reads the controller board digital inputs on the the following SCADAPack controllers. The number of supported Controller Digital Inputs is also listed.

- SCADAPack (3 Digital Inputs)
- SCADAPack Light (3 Digital Inputs)
- SCADAPack Plus (3 Digital Inputs)
- SCADAPack 32P (3 Digital Inputs)
- SCADAPack 100 (6 Digital Inputs)
- SCADAPack LP (8 Digital Inputs)

Data is assigned to three consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

The Controller Digital Inputs on the SCADAPack, SCADAPack Light, SCADAPack Plus and SCADAPack 32 can be used as counter inputs, see Controller Counter Inputs.

Module Properties

Field	Description
Module Name	Controller Digital Inputs Module
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10001. **Note:** The number of digital inputs that may be used depends on the controller type. See list above. Start Register + 0 is the DIN 0 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the DIN 1 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the DIN 2 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the DIN 3 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the DIN 4 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the DIN 5 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the DIN 6 physical digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the DIN 7 physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Interrupt Input Status Module

The **Controller Interrupt Input Status** module reads the Interrupt (INT) digital input on the SCADAPack controllers. Data is assigned to a single status register. The status register is updated continuously with data read from the digital input. The Controller Interrupt Status Digital Input can be used as a counter input, see counter module Controller Interrupt Input Counter Module.

Module Properties

Field	Description
Module Name	Controller Interrupt Input Status Module
DI Start Register	First register of any unused block of one sequential digital input registers. The default value is 10001. Start Register + 0 is the INT physical digital input data. (0 indicates OFF and 1 indicates ON).

Controller Option Switches Module

The **Controller Option Switches** module reads the state of the three option switches on the SCADAPack controllers. Data is assigned to three status registers. The status registers are updated continuously with data read from the option switches.

Module Properties

Field	Description
Module Name	Controller Option Switches Module
DI Start Register	First register of any unused block of three sequential digital input registers. The default value is 10001. Start Register + 0 is the Option Switch 1 digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the Option Switch 2 digital input data. (0 indicates OFF and 1 indicates

ON).
Start Register + 2 is the Option Switch 3 digital input data. (0 indicates OFF and 1 indicates ON).

SCADAPack 32 Option Switches Module

The **SCADAPack 32 Option Switches** module reads the state of the four option switches on SCADAPack 32 controllers. Data is assigned to four status registers. The status registers are updated continuously with data read from the option switches.

Module Properties

Field	Description
Module Name	SCADAPack 32 Option Switches Module
DI Start Register	First register of any unused block of four sequential digital input registers. The default value is 10001. Start Register + 0 is the Option Switch 1 digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the Option Switch 2 digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the Option Switch 3 digital input data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the Option Switch 4 digital input data. (0 indicates OFF and 1 indicates ON).

5401 Eight Point Module

The **5401 Eight Point m**odule provides eight digital inputs from the 5401 Digital I/O module. Data is assigned to eight consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

When there are digital outputs being used in combination with digital inputs on the 5401, a full 8 status registers must still be assigned to the **5401 Eight Point Module**. When this is the case, a digital output **5401 Eight Point Module** must also be added to the Register Assignment with the same module address.

Module Properties

Field	Description
Module Name	5401 Eight Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of eight sequential digital input registers. The default value is 10001. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON).

5402 Sixteen Point Module

The **5402 Sixteen Point m**odule provides sixteen inputs from the 5402 Digital I/O module. Data is assigned to sixteen consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Module Properties

Field	Description
Module Name	5402 sixteen Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of sixteen sequential digital input registers. The default value is 10001. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON). Start Register + 8 is the digital input 8 data. (0 indicates OFF and 1 indicates ON).

Start Register + 9 is the digital input 9 data. (0 indicates OFF and 1 indicates ON).
Start Register + 10 is the digital input 10 data. (0 indicates OFF and 1 indicates ON).
Start Register + 11 is the digital input 11 data. (0 indicates OFF and 1 indicates ON).
Start Register + 12 is the digital input 12 data. (0 indicates OFF and 1 indicates ON).
Start Register + 13 is the digital input 13 data. (0 indicates OFF and 1 indicates ON).
Start Register + 14 is the digital input 14 data. (0 indicates OFF and 1 indicates ON).
Start Register + 15 is the digital input 15 data. (0 indicates OFF and 1 indicates ON).

5403 Eight Point Module

The **5403 Eight Point** module provides eight digital inputs from the 5403 Digital Input module. Data is assigned to eight consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Module Properties

Field	Description
Module Name	5403 Eight Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of eight sequential digital input registers. The default value is 10001. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON).

5404 Sixteen Point Module

The **5404 Sixteen Point** module provides sixteen digital inputs. Data is assigned to sixteen consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Module Properties

Field	Description
Module Name	5404 Sixteen Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of sixteen sequential digital input registers. The default value is 10001. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON). Start Register + 8 is the digital input 8 data. (0 indicates OFF and 1 indicates ON). Start Register + 9 is the digital input 9 data. (0 indicates OFF and 1 indicates ON). Start Register + 10 is the digital input 10 data. (0 indicates OFF and 1 indicates ON). Start Register + 11 is the digital input 11 data. (0 indicates OFF and 1 indicates ON). Start Register + 12 is the digital input 12 data. (0 indicates OFF and 1 indicates ON). Start Register + 13 is the digital input 13 data. (0 indicates OFF and 1 indicates ON). Start Register + 14 is the digital input 14 data. (0 indicates OFF and 1 indicates ON). Start Register + 15 is the digital input 15 data. (0 indicates OFF and 1 indicates ON).

Module Properties

Field	Description

Module Name	5404 Sixteen Channel Module
Module Address	This module is assigned a unique module address between 0 and 15. No other DIN-type module may use this module address. The default is 0.
DI Start Register	First register of any unused block of 16 consecutive digital input (status) registers (DI1 to DI16) in the I/O database. Default register: 10001.

Register Data

Registers	Assignment to Module Hardware
DI1 (DI Start Register)	Digital input 0
DI2	Digital input 1
DI3	Digital input 2
DI4	Digital input 3
DI5	Digital input 4
DI6	Digital input 5
DI7	Digital input 6
DI8	Digital input 7
DI9	Digital input 8
DI10	Digital input 9
DI11	Digital input 10
DI12	Digital input 11
DI13	Digital input 12
DI14	Digital input 13
DI15	Digital input 14
DI16	Digital input 15

Notes

Refer to the *5404 Digital I/O Module User Manual* for further information.

5405 Thirty Two Point Module

The **5405 Thirty-two Point** module provides thirty-two digital inputs. Data is assigned to thirty-two consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Module Properties

Field	Description
Module Name	5404 Sixteen Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of sixteen sequential digital input registers. The default value is 10001. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON). Start Register + 8 is the digital input 8 data. (0 indicates OFF and 1 indicates ON). Start Register + 9 is the digital input 9 data. (0 indicates OFF and 1 indicates ON). Start Register + 10 is the digital input 10 data. (0 indicates OFF and 1 indicates ON). Start Register + 11 is the digital input 11 data. (0 indicates OFF and 1 indicates ON). Start Register + 12 is the digital input 12 data. (0 indicates OFF and 1 indicates ON). Start Register + 13 is the digital input 13 data. (0 indicates OFF and 1 indicates ON). Start Register + 14 is the digital input 14 data. (0 indicates OFF and 1 indicates ON). Start Register + 15 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 16 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 17 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 18 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 19 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 20 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 21 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 22 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 23 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 24 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 25 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 26 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 27 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 28 is the digital input 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 29 is the digital input 15 data. (0 indicates OFF and 1 indicates ON).

Start Register + 30 is the digital input 15 data. (0 indicates OFF and 1 indicates ON).
Start Register + 31 is the digital input 15 data. (0 indicates OFF and 1 indicates ON).

5414 Sixteen Point Module

The **5414 Sixteen Point** module provides sixteen digital inputs from a 5414 Digital Input module. A maximum of sixteen 5414 digital input modules may be installed on the I/O bus. Note that this I/O module is not supported by SCADAPack, SCADAPack Light, SCADAPack Plus, SCADAPack LP, SCADAPack 100, Micro16 or SCADAPack 4202 controllers.

Module Properties

Field	Description
Module Name	5414 Sixteen Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of sixteen sequential digital input registers. The default value is 10001. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON). Start Register + 8 is the digital input 8 data. (0 indicates OFF and 1 indicates ON). Start Register + 9 is the digital input 9 data. (0 indicates OFF and 1 indicates ON). Start Register + 10 is the digital input 10 data. (0 indicates OFF and 1 indicates ON). Start Register + 11 is the digital input 11 data. (0 indicates OFF and 1 indicates ON). Start Register + 12 is the digital input 12 data. (0 indicates OFF and 1 indicates ON). Start Register + 13 is the digital input 13 data. (0 indicates OFF and 1 indicates ON). Start Register + 14 is the digital input 14 data. (0 indicates OFF and 1 indicates ON). Start Register + 15 is the digital input 15 data. (0 indicates OFF and 1 indicates ON).
AC/DC	The AC/DC selection for all inputs defines the input measurement type for the digital inputs. • The **DC** selection sets the input to measure DC input signals. • The **AC** selection sets the input to measure AC input signals.
Scan Frequency	The Scan Frequency selection set the scan rate for all digital inputs. The scan rate selection is not critical but AC noise rejection is improved at the correct frequency. If the module is used in a DC environment, the 60 Hz setting will yield slightly faster response time. • The **60 Hz** selection synchronizes the input scanning to 60 Hz. • The **50 Hz** selection synchronizes the input scanning to 50 Hz.

5421 Eight Point Toggle Switch Module

The **5421 Eight Point** module provides eight digital inputs from the 5421 Toggle Switch Digital Input. Data is assigned to eight consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Module Properties

Field	Description
Module Name	5421 Eight Point Toggle Switch Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of eight sequential digital input registers. The default value is 10001. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON).

Generic Eight Point Module

The **Generic Eight Point** module provides eight digital inputs. Data is stored in eight consecutive status registers. The status registers are updated continuously with

data read from the digital inputs. The **Generic Eight Channel** module may be used in place of any other 8 point Digital Input type module.

The **Generic Eight Channel** module type is a useful selection early on in the system design stage before the final selection of the specific Digital Input module type is known.

Module Properties

Field	Description
Module Name	Generic Eight Channel Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of eight sequential digital input registers. The default value is 10001. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON).

Generic Sixteen Channel Module

The **Generic Sixteen Point Module** module provides sixteen digital inputs. Data is assigned to sixteen consecutive status registers. The status registers are updated continuously with data read from the digital inputs. The **Generic Sixteen Channel** module may be used in place of any other 16 point Digital Input type modules.

The **Generic Sixteen Point Module** type is a useful selection early on in the system design stage before the final selection of the specific Digital Input module type is known.

Module Properties

Field	Description
Module Name	Generic Sixteen Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of sixteen sequential digital input registers. The default value is 10001. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON). Start Register + 8 is the digital input 8 data. (0 indicates OFF and 1 indicates ON). Start Register + 9 is the digital input 9 data. (0 indicates OFF and 1 indicates ON). Start Register + 10 is the digital input 10 data. (0 indicates OFF and 1 indicates ON). Start Register + 11 is the digital input 11 data. (0 indicates OFF and 1 indicates ON). Start Register + 12 is the digital input 12 data. (0 indicates OFF and 1 indicates ON). Start Register + 13 is the digital input 13 data. (0 indicates OFF and 1 indicates ON). Start Register + 14 is the digital input 14 data. (0 indicates OFF and 1 indicates ON). Start Register + 15 is the digital input 15 data. (0 indicates OFF and 1 indicates ON).

Digital Output Modules

Digital output modules are used to assign data from the I/O database to physical digital outputs. The physical digital outputs are specific SCADAPack I/O I/O modules or generic I/O modules.

Digital output modules may assign data to any coil registers in the I/O database that are not being used by another Digital output module. Coil registers are referred to as 0nnnn registers throughout this manual.

All I/O database coil registers that are not assigned to any other I/O modules may be used as general-purpose coil registers in a ladder program.

Related Topics

5401 Eight Channel Module

5402 Sixteen Channel Module

5406 Sixteen Channel Relay Module

5407 Eight Channel Relay Module

5408 Eight Channel Triac Module

5409 Eight Channel FET Module

5411 Thirty Two Channel Module

5415 Twelve Channel Relay Module

Generic Eight Channel Module

Generic Sixteen Channel Module

5401 Eight Point Module

The **5401 Eight Point m**odule provides eight digital outputs to the 5401 Digital I/O module. Data is assigned from eight consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

When there are digital outputs being used in combination with digital inputs on the 5401, a full 8 status registers must still be assigned to the **5401 Eight Point** module. When this is the case, a digital input **5401 Eight Point m**odule must also be added to the Register Assignment with the same module address.

Module Properties

Field	Description
Module Name	5401 Eight Channel Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Output type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of eight sequential digital output registers. The default value is 1. Start Register + 0 is the digital output 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital output 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital output 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital output 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital output 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital output 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital output 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital output 7 data. (0 indicates OFF and 1 indicates ON).

5402 Sixteen Point Module

The **5402 Eight Point** module provides sixteen digital outputs to the 5402 Digital I/O module. Data is assigned from sixteen consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Module Properties

Field	Description
Module Name	5401 Eight Channel Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Output type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of sixteen sequential digital output registers. The default value is 1. Start Register + 0 is the digital output 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital output 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital output 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital output 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital output 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital output 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital output 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital output 7 data. (0 indicates OFF and 1 indicates ON). Start Register + 8 is the digital output 8 data. (0 indicates OFF and 1 indicates ON). Start Register + 9 is the digital output 9 data. (0 indicates OFF and 1 indicates ON). Start Register + 10 is the digital output 10 data. (0 indicates OFF and 1 indicates ON). Start Register + 11 is the digital output 11 data. (0 indicates OFF and 1 indicates ON). Start Register + 12 is the digital output 12 data. (0 indicates OFF and 1 indicates ON). Start Register + 13 is the digital output 13 data. (0 indicates OFF and 1 indicates ON). Start Register + 14 is the digital output 14 data. (0 indicates OFF and 1 indicates ON). Start Register + 15 is the digital output 15 data. (0 indicates OFF and 1 indicates ON).

5406 Sixteen Point Relay Module

The **5406 Sixteen Point Relay** module provides sixteen digital outputs. Data is assigned from sixteen consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Module Properties

Field	Description
Module Name	5406 Sixteen Point Relay Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Output type module can use this module address. Default value: 0.
DO Start Register	First register of any unused block of sixteen sequential digital output registers. The default value is 1. Start Register + 0 is the digital output 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital output 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital output 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital output 3 data. (0 indicates OFF and 1 indicates ON).

Start Register + 4 is the digital output 4 data. (0 indicates OFF and 1 indicates ON).
Start Register + 5 is the digital output 5 data. (0 indicates OFF and 1 indicates ON).
Start Register + 6 is the digital output 6 data. (0 indicates OFF and 1 indicates ON).
Start Register + 7 is the digital output 7 data. (0 indicates OFF and 1 indicates ON).
Start Register + 8 is the digital output 8 data. (0 indicates OFF and 1 indicates ON).
Start Register + 9 is the digital output 9 data. (0 indicates OFF and 1 indicates ON).
Start Register + 10 is the digital output 10 data. (0 indicates OFF and 1 indicates ON).
Start Register + 11 is the digital output 11 data. (0 indicates OFF and 1 indicates ON).
Start Register + 12 is the digital output 12 data. (0 indicates OFF and 1 indicates ON).
Start Register + 13 is the digital output 13 data. (0 indicates OFF and 1 indicates ON).
Start Register + 14 is the digital output 14 data. (0 indicates OFF and 1 indicates ON).
Start Register + 15 is the digital output 15 data. (0 indicates OFF and 1 indicates ON).

5407 Eight Point Relay Module

The **5407 Eight Point Relay** module provides eight digital outputs. Data is assigned from eight consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Module Properties

Field	Description
Module Name	5407 Eight Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Output type module can use this module address. Default value: 0.
DO Start Register	First register of any unused block of eight sequential digital input registers. The default value is 1. Start Register + 0 is the digital output 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital output 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital output 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital output 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital output 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital output 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital output 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital output 7 data. (0 indicates OFF and 1 indicates ON).

5408 Eight Point Triac Module

The **5408 Eight Point Triac** module provides eight digital outputs. Data is assigned from eight consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Module Properties

Field	Description
Module Name	5407 Eight Channel Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Output type module can use this module address. Default value: 0.
DO Start Register	First register of any unused block of eight sequential digital input registers. The default value is 1. Start Register + 0 is the digital output 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital output 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital output 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital output 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital output 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital output 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital output 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital output 7 data. (0 indicates OFF and 1 indicates ON).

5409 Eight Point FET Module

The **5409 Eight Point FET Channel** module provides eight digital outputs. Data is assigned from eight consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Module Properties

Field	Description
Module Name	5407 Eight Point FET Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Output type module can use this module address. Default value: 0.
DO Start Register	First register of any unused block of eight sequential digital input registers. The default value is 1. Start Register + 0 is the digital output 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital output 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital output 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital output 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital output 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital output 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital output 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital output 7 data. (0 indicates OFF and 1 indicates ON).

5411 Thirty-two Point Module

The **5411 Thirty-two Point** module provides thirty-two digital outputs. Data is assigned from thirty-two consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Module Properties

Field	Description
Module Name	5411 Thirty-two Channel Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Output type module can use this module address. Default value: 0.
DO Start Register	First register of any unused block of thirty two sequential digital output registers. The default value is 1. Start Register + 0 is the digital output 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital output 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital output 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital output 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital output 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital output 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital output 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital output 7 data. (0 indicates OFF and 1 indicates ON). Start Register + 8 is the digital output 8 data. (0 indicates OFF and 1 indicates ON). Start Register + 9 is the digital output 9 data. (0 indicates OFF and 1 indicates ON). Start Register + 10 is the digital output 10 data. (0 indicates OFF and 1 indicates ON). Start Register + 11 is the digital output 11 data. (0 indicates OFF and 1 indicates ON). Start Register + 12 is the digital output 12 data. (0 indicates OFF and 1 indicates ON). Start Register + 13 is the digital output 13 data. (0 indicates OFF and 1 indicates ON). Start Register + 14 is the digital output 14 data. (0 indicates OFF and 1 indicates ON). Start Register + 15 is the digital output 15 data. (0 indicates OFF and 1 indicates ON). Start Register + 16 is the digital output 16 data. (0 indicates OFF and 1 indicates ON). Start Register + 17 is the digital output 17 data. (0 indicates OFF and 1 indicates ON). Start Register + 18 is the digital output 18 data. (0 indicates OFF and 1 indicates ON). Start Register + 19 is the digital output 19 data. (0 indicates OFF and 1 indicates ON). Start Register + 20 is the digital output 20 data. (0 indicates OFF and 1 indicates ON). Start Register + 21 is the digital output 21 data. (0 indicates OFF and 1 indicates ON). Start Register + 22 is the digital output 22 data. (0 indicates OFF and 1 indicates ON). Start Register + 23 is the digital output 23 data. (0 indicates OFF and 1 indicates ON). Start Register + 24 is the digital output 24 data. (0 indicates OFF and 1 indicates ON). Start Register + 25 is the digital output 25 data. (0 indicates OFF and 1 indicates ON). Start Register + 26 is the digital output 26 data. (0 indicates OFF and 1 indicates ON). Start Register + 27 is the digital output 27 data. (0 indicates OFF and 1 indicates ON). Start Register + 28 is the digital output 28 data. (0 indicates OFF and 1 indicates ON). Start Register + 29 is the digital output 29 data. (0 indicates OFF and 1 indicates ON). Start Register + 30 is the digital output 30 data. (0 indicates OFF and 1 indicates ON). Start Register + 31 is the digital output 31 data. (0 indicates OFF and 1 indicates ON).

5415 Twelve Point Relay Module

The **5415 Twelve Point Relay** module provides twelve relay digital outputs. Data is assigned from twelve consecutive coil registers. The 5415 digital outputs are updated continuously with data read from the coil registers.

The 5415 module provides two digital inputs to monitor the relay power status and power jumper position.The 5415 digital inputs are updated continuously with data read from the input registers.

Note that this I/O module is not supported by SCADAPack, SCADAPack Light, SCADAPack Plus, SCADAPack LP, SCADAPack 100, Micro16 or SCADAPack 4202 controllers.

Module Properties

Field	Description
Module Name	5415 Twelve Point Relay Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Output type module can use this module address. Default value: 0.
DO Start Register	First register of any unused block of sixteen sequential digital output registers. The default value is 1. Start Register + 0 is the digital output 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital output 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital output 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital output 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital output 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital output 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital output 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital output 7 data. (0 indicates OFF and 1 indicates ON). Start Register + 8 is the digital output 8 data. (0 indicates OFF and 1 indicates ON). Start Register + 9 is the digital output 9 data. (0 indicates OFF and 1 indicates ON). Start Register + 10 is the digital output 10 data. (0 indicates OFF and 1 indicates ON). Start Register + 11 is the digital output 11 data. (0 indicates OFF and 1 indicates ON). Start Register + 12 is the digital output 12 data. (0 indicates OFF and 1 indicates ON). Start Register + 13 is the digital output 13 data. (0 indicates OFF and 1 indicates ON). Start Register + 14 is the digital output 14 data. (0 indicates OFF and 1 indicates ON). Start Register + 15 is the digital output 15 data. (0 indicates OFF and 1 indicates ON).
DI Start Register	First register of any unused block of two sequential digital input registers. The default value is 1. Start Register + 0 is the board relay power status. (0 indicates no power to relays and 1 indicates power to relays). Start Register + 1 is the relay power jumper position. (0 indicates internal power from 5V Bus and 1 indicates external power).

Generic Eight Point Module

The **Generic Eight Point** module provides eight digital outputs. Data is assigned from eight consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers. The **Generic Eight Point** module may be used in place of any other 8 point Digital Output type module.

The **Generic Eight Point Module** type is a useful selection early on in the system design stage before the final selection of the specific Digital Output module type is known.

Module Properties

Field	Description
Module Name	Generic Eight Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of eight sequential digital input registers. The default value is 1. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON). Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON). Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON). Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON). Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON). Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON). Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON).

Generic Sixteen Point Module

The **Generic Sixteen Point** module provides sixteen digital outputs. Data is assigned from sixteen consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers. The **Generic Sixteen Point** module may be used in place of any other 16 point Digital Output type module.

The **Generic Sixteen Point m**odule module type is a useful selection early on in the system design stage before the final selection of the specific Digital Output module type is known.

Module Properties

Field	Description
Module Name	Generic Sixteen Point Module
Module Address	This module is assigned a unique module address between 0 and 15. No other Digital Input type module can use this module address. Default value: 0.
DI Start Register	First register of any unused block of eight sequential digital input registers. The default value is 1. Start Register + 0 is the digital input 0 data. (0 indicates OFF and 1 indicates ON). Start Register + 1 is the digital input 1 data. (0 indicates OFF and 1 indicates ON).

Start Register + 2 is the digital input 2 data. (0 indicates OFF and 1 indicates ON).
Start Register + 3 is the digital input 3 data. (0 indicates OFF and 1 indicates ON).
Start Register + 4 is the digital input 4 data. (0 indicates OFF and 1 indicates ON).
Start Register + 5 is the digital input 5 data. (0 indicates OFF and 1 indicates ON).
Start Register + 6 is the digital input 6 data. (0 indicates OFF and 1 indicates ON).
Start Register + 7 is the digital input 7 data. (0 indicates OFF and 1 indicates ON).
Start Register + 8 is the digital input 8 data. (0 indicates OFF and 1 indicates ON).
Start Register + 9 is the digital input 9 data. (0 indicates OFF and 1 indicates ON).
Start Register + 10 is the digital input 10 data. (0 indicates OFF and 1 indicates ON).
Start Register + 11 is the digital input 11 data. (0 indicates OFF and 1 indicates ON).
Start Register + 12 is the digital input 12 data. (0 indicates OFF and 1 indicates ON).
Start Register + 13 is the digital input 13 data. (0 indicates OFF and 1 indicates ON).
Start Register + 14 is the digital input 14 data. (0 indicates OFF and 1 indicates ON).
Start Register + 15 is the digital input 15 data. (0 indicates OFF and 1 indicates ON).

Integrated I/O Modules

The following Integrated I/O Modules are supported by the controller types listed:

SCADAPack 5601 I/O Module

SCADAPack

SCADAPack Plus

SCADAPack 32

SCADAPack 5602 I/O Module

SCADAPack

SCADAPack Light

SCADAPack Plus

SCADAPack 5604 I/O Module

SCADAPack

SCADAPack 32

SCADAPack 5606 I/O Module

SCADAPack

SCADAPack 32

SCADAPack 330

SCADAPack 350

SCADAPack 5607 I/O Module

SCADAPack 32

SCADAPack 314

SCADAPack 350

SCADAPack 350

SCADAPack 330

SCADAPack 5601 I/O Module

The **SCADAPack 5601 I/O** module is an integrated component of a SCADAPack controller and is not available as a stand alone unit. The following controllers support the SCADAPack 5601 I/O module:

- SCADAPack - A 520x controller board with an integrated 5601 I/O Module.
- SCADAPack 32 - A 5232 controller board with an integrated 5601 I/O module.

The SCADAPack 5601 I/O module provides eight analog inputs may be configured for a 0 or 20% offset in the measuring range. All analog input channels are transient protected, optically isolated from the main logic supply and share a common return (COM) that is connected to the chassis. The input registers are updated continuously with data read from the analog inputs.

The SCADAPack 5601 I/O module provides sixteen digital input channels. The digital inputs can be configured in four standard voltage ranges for both AC and DC type applications as requested at the time of purchase. The digital inputs are optically isolated from the logic power. A current limiting resistor, on each input, determines the voltage range. Light Emitting Diodes (LEDs) on the digital inputs show the status of each of the input. The status registers are updated continuously with data read from the digital inputs.

The SCADAPack 5601 I/O provides twelve dry contact digital (mechanical relay) outputs are provided onboard this I/O module. Ten outputs are Form A (normally open NO) and two outputs are Form C (both normally open NO and normally closed NC) contacts. The Form C outputs and two of the Form A outputs are individually isolated allowing multiple power sources to be used within the same module. Loads can be connected to either output terminal and to either the high or the low side of the power source. The digital outputs are updated continuously with data read from the coil registers.

Module Properties

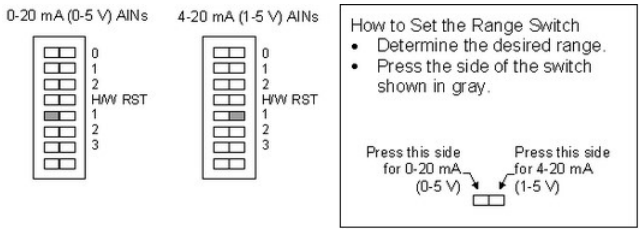
Field	Description
Module Name	SCADAPack 5601 I/O Module
DO Start Register	First register of any unused block of twelve consecutive digital output registers. The default setting is 1.

DI Start Register	First register of any unused block of sixteen sequential digital input registers. The default value is 10001.
AI Start Register	First register of any unused block of eight consecutive analog input registers. Default value is 30001.

Analog Input Measurement Range Selection

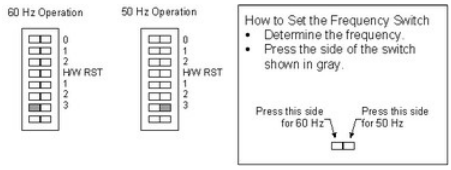
The SCADAPack Option Switch 1 selects the measurement range. All 8 analog inputs are set to the same range.

The figure below shows the switch settings for selecting the measurement range.



Line Frequency Selection

The SCADAPack I/O board may select 50 or 60 Hz line frequency for digital and analog input processing. The SCADAPack Option Switch 3 selects this option. The figure below shows the switch settings for selecting the line frequency.



Notes

1. Only one SCADAPack 5601 I/O module may be assigned to the same controller.

2. Refer to the SCADAPack Hardware Manual for information on the option switches.

SCADAPack 5602 I/O Module

The **SCADAPack 5602 I/O** module has five analog/digital inputs and two digital outputs. This I/O module is found in the following SCADAPack configurations:

- SCADAPack Plus – 5203/5204 controller board with 5601 and 5602 I/O module
- SCADAPack Light - 5203/5204 controller board with a 5602 I/O module

The SCADAPack 5602 I/O module provides five single ended analog inputs that are configured for either 20mA current or 5V operation with a 0 or 20% offset. The current input version has 250W current sense resistors on the inputs. It is otherwise identical to the voltage input version. The 20mA analog inputs can also be used as digital inputs to monitor DC voltages by installing a resistor in series with the analog input. The input registers are updated continuously with data read from the analog inputs.

The SCADAPack 5602 I/O module has two dry contact, digital (mechanical relay) outputs. The outputs are Form C type (both normally open NO and normally closed NC contacts). The Form C outputs are individually isolated allowing multiple power sources to be used within the same module. The digital outputs are updated continuously with data read from the coil registers.

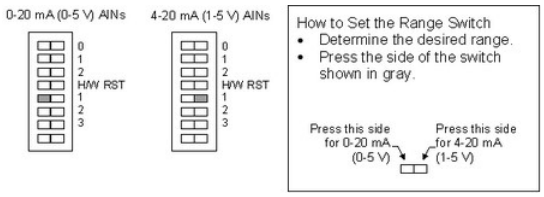
Module Properties

Field	Description
Module Name	SCADAPack 5602 I/O Module
DO Start Register	First register of any unused block of two consecutive digital output registers. The default setting is 1.
DI Start Register	First register of any unused block of five sequential digital input registers. The default value is 10001.
AI Start Register	First register of any unused block of five consecutive analog input registers. Default value is 30001.

Analog Input Measurement Range Selection

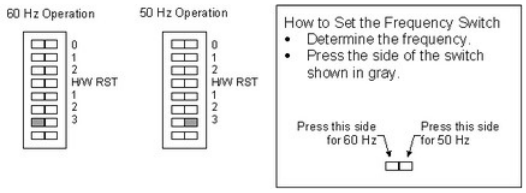
The SCADAPack Option Switch 1 selects the measurement range. All 5 analog inputs are set to the same range.

The figure below shows the switch settings for selecting the measurement range.



Line Frequency Selection

The SCADAPack I/O board may select 50 or 60 Hz line frequency for analog input processing. The SCADAPack Option Switch 3 selects this option. The figure below shows the switch settings for selecting the line frequency.



Notes

1. Only one **SCADAPack 5602 I/O module** may be assigned to the same controller. Option switches 1 and 3 function as described above regardless of whether the module, **DIN 5203/4 option switches**, is added to the Register Assignment or not.
2. Refer to the **SCADAPack Hardware Manual** for information on the option switches.

SCADAPack 5604 10V/40mA I/O Module

The **SCADAPack 5604 10V/40mA I/O** module is comprised of four different types of I/O channels. The SCADAPack 5604 I/O hardware contains ten analog inputs, two analog outputs, thirty-five digital inputs and thirty-six digital outputs.

The **digital input** assignment provides thirty-five I/O channels for the digital input data from the SCADAPack 5604 I/O Module hardware. The digital input registers are updated continuously with data read from the digital inputs.

- The SCADAPack 5604 I/O Module provides thirty-two universal digital inputs or outputs. The inputs are for use with dry contacts such as switches and relay contacts.
- The SCADAPack 5604 I/O Module also provides three internal digital inputs.
  - Digital input channel 32 returns the DC/DC converter status.
  - Digital input channel 33 returns the DC/DC converter over current status.
  - Digital output channel 34 returns the digital output mismatch status.

The **digital output** assignment provides thirty-six I/O channels for the digital outputs of the SCADAPack 5604 I/O Module I/O hardware. The digital outputs are updated continuously with data read from the coil registers.

- The SCADAPack 5604 I/O Module provides thirty-two universal digital inputs or outputs. Outputs are open-collector/open drain type.
- The SCADAPack 5604 I/O Module also provides four internal digital outputs.
  - ○ Digital output channel 32 is used to control the DC/DC converter.
  - ○ Digital output channel 33 is used to control the VLoop power supply.
  - ○ Digital output channels 34 and 35 control the SCADAPack 5604 I/O Module Analog Input filters.

Filter Setting	Digital Output 34	Digital Output 35
< 3 Hz	OFF	OFF
6 Hz	OFF	ON
11 Hz	ON	OFF
30 Hz	ON	ON

The **analog input** assignment provides ten I/O channels for the analog input data from the SCADAPack 5604 I/O Module hardware. The analog input registers are updated continuously with data read from the analog inputs.

- Analog input channels 0 to 7 provide eight external analog inputs.
- Analog input channel 8 provides an external analog input for battery monitoring (0 to 32.768V)
- Analog input channel 9 provides an internal analog input for DC/DC converter monitoring.

The **analog output** assignment provides two I/O channels for the analog output data for the SCADAPack 5604 I/O Module hardware when the optional analog output module is installed.The analog output registers are updated continuously with data read from the analog output registers.

Module Properties

Parameter	Description
Module Name	SCADAPack 5604 I/O Module
DO Start Register	First register of any unused block of thirty-six consecutive digital output registers. The default value is 1. DO Start Register to DO Start Register + 31 are physical digital output data (0 indicates OFF and 1 indicates ON). DO Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DO Start Register +33 (ON = VLoop output ON; OFF VLoop output OFF) DO Start Register +34 and +35 set the analog input filters: Filter Setting / Digital Output 34 / Digital Output 35 < 3 Hz / OFF / OFF 6 Hz / OFF / ON 11 Hz / ON / OFF 30 Hz / ON / ON
DI Start Register	First register of any unused block of thirty-five consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +31 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DI Start Register +33 (ON = Over current detected; OFF Over current not detected) DI Start Register +34 (ON = One or more digital outputs mismatch; OFF No mismatch)
AI Start Register	First register of any unused block of ten consecutive analog input registers. The default value is 30001. AI Start Register to AI Start Register + 7 are physical analog input data. AI Start Register +8 (external analog input for battery monitoring; 0 to 32.768V) AI Start Register +9 (internal analog input for DC/DC converter monitoring)
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.

SCADAPack 5604 5V/20mA I/O Module

The **SCADAPack 5604 5V/20mA I/O** module is comprised of four different types of I/O channels. The SCADAPack 5604 I/O hardware contains ten analog inputs, two analog outputs, thirty-five digital inputs and thirty-six digital outputs.

The **digital input** assignment provides thirty-five I/O channels for the digital input data from the SCADAPack 5604 I/O Module hardware. The digital input registers are updated continuously with data read from the digital inputs.

- The SCADAPack 5604 I/O Module provides thirty-two universal digital inputs or outputs. The inputs are for use with dry contacts such as switches and relay contacts.
- The SCADAPack 5604 I/O Module also provides three internal digital inputs.
  - Digital input channel 32 returns the DC/DC converter status.
  - Digital input channel 33 returns the DC/DC converter over current status.
  - Digital output channel 34 returns the digital output mismatch status.

The **digital output** assignment provides thirty-six I/O channels for the digital outputs of the SCADAPack 5604 I/O Module I/O hardware. The digital outputs are updated continuously with data read from the coil registers.

- The SCADAPack 5604 I/O Module provides thirty-two universal digital inputs or outputs. Outputs are open-collector/open drain type.
- The SCADAPack 5604 I/O Module also provides four internal digital outputs.
  - Digital output channel 32 is used to control the DC/DC converter.
  - Digital output channel 33 is used to control the VLoop power supply.
  - Digital output channels 34 and 35 control the SCADAPack 5604 I/O Module Analog Input filters.

Filter Setting	Digital Output 34	Digital Output 35
< 3 Hz	OFF	OFF
6 Hz	OFF	ON
11 Hz	ON	OFF
30 Hz	ON	ON

The **analog input** assignment provides ten I/O channels for the analog input data from the SCADAPack 5604 I/O Module hardware. The analog input registers are updated continuously with data read from the analog inputs.

- Analog input channels 0 to 7 provide eight external analog inputs (0-5V or 0-20mA)
- Analog input channel 8 provides an external analog input for battery monitoring (0 to 32.768V)
- Analog input channel 9 provides an internal analog input for DC/DC converter monitoring.

The **analog output** assignment provides two I/O channels for the analog output data for the SCADAPack 5604 I/O Module hardware when the optional analog output module is installed.The analog output registers are updated continuously with data read from the analog output registers.

Module Properties

Parameter	Description
Module Name	SCADAPack 5604 I/O Module
DO Start Register	First register of any unused block of thirty-six consecutive digital output registers. The default value is 1. DO Start Register to DO Start Register + 31 are physical digital output data.(0 indicates OFF and 1 indicates ON). DO Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DO Start Register +33 (ON = VLoop output ON; OFF VLoop output OFF) DO Start Register +34 and +35 set the analog input filters: Filter Setting / Digital Output 34 / Digital Output 35 < 3 Hz / OFF / OFF 6 Hz / OFF / ON 11 Hz / ON / OFF 30 Hz / ON / ON
DI Start Register	First register of any unused block of thirty-five consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +31 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +32 (ON = DC/DC converter ON; OFF DC/DC converter OFF) DI Start Register +33 (ON = Over current detected; OFF Over current not detected) DI Start Register +34 (ON = One or more digital outputs mismatch; OFF No mismatch)
AI Start Register	First register of any unused block of ten consecutive analog input registers. The default value is 30001. AI Start Register to AI Start Register + 7 are physical analog input data. AI Start Register +8 (external analog input for battery monitoring; 0 to 32.768V) AI Start Register +9 (internal analog input for DC/DC converter monitoring)
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.

SCADAPack 5606 I/O Module

The **SCADAPack 5606 I/O** module is comprised of four types of I/O channels, digital inputs, digital outputs, analog inputs, and analog outputs.

The **digital input** assignment provides forty I/O channels for the digital input data from the SCADAPack 5606 I/O Module hardware. The digital input registers are updated continuously with data read from the digital inputs.

- The 5606 I/O Module provides thirty-two external digital inputs. These are externally wetted inputs.
- The module provides 8 internal digital inputs, which indicate if the corresponding analog input is in or out of range.

The **digital output** assignment provides sixteen digital outputs. All are digital output relays. The digital outputs are updated continuously with data read from the coil registers.

The **analog input** assignment provides eight I/O channels for the analog input data from the SCADAPack 5606 I/O Module hardware. The analog input registers are updated continuously with data read from the analog inputs.

The **analog output** assignment provides two I/O channels for the analog output data for the SCADAPack 5606 I/O Module hardware when the optional analog output module is installed.The analog output registers are updated continuously with data read from the analog output registers.

Module Properties

Parameter	Description
Module Name	SCADAPack 5606 I/O Module
Module Address	The address of the module is selected using dip switches on the module. A maximum of eight SCADAPack 5606 I/O type modules may be added to a system. Default value: 0
DO Start Register	First register of any unused block of sixteen consecutive digital output registers. The default value is 1.
DI Start Register	First register of any unused block of forty consecutive digital input registers. The default value is 10001. DI Start Register to DI Start Register +31 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +32 (OFF = AI1 is OK; ON AI1 is over or under range) DI Start Register +33 (OFF = AI2 is OK; ON AI2 is over or under range) DI Start Register +34 (OFF = AI3 is OK; ON AI3 is over or under range) DI Start Register +35 (OFF = AI4 is OK; ON AI4 is over or under range) DI Start Register +36 (OFF = AI5 is OK; ON AI5 is over or under range) DI Start Register +37 (OFF = AI6 is OK; ON AI6 is over or under range) DI Start Register +38 (OFF = AI7 is OK; ON AI7 is over or under range) DI Start Register +39 (OFF = AI8 is OK; ON AI8 is over or under range)
AI Start Register	First register of any unused block of eight consecutive analog input registers. The default value is 30001.
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.
AI*n* Channel Type	Defines the input measurement type for each input (AI 0 through 7). • The 0 to 5V selection sets the input type to measure 0 to 5V input signals. • The 1 to 5V selection sets the input type to measure 1 to 5V input signals. • The 0 to 20 mA selection sets the input type to measure 0 to 20mA input signals. • The 4 to 20 mA selection sets the input type to measure 4 to 20mA input signals.
Module Filter	Sets the input filter rate for all analog inputs. The filter rate is used to dampen process variations or noise. • The 3 Hz filter selection sets the response time to 155ms at 60Hz and 185ms at 50Hz. • The 6 Hz filter selection sets response time to 85ms at 60Hz and 85ms at 50Hz. • The 11 Hz filter selection sets response time to 45ms at 60Hz and 55ms at 50Hz. • The 30 Hz filter selection sets response time to 30ms at 60Hz and 30ms at 50Hz.
Module Scan Frequency	Sets the input scan rate for all analog inputs. The scan rate selection is not critical but AC noise rejection is improved at the correct frequency. If the module is used in a DC environment, the 60 Hz setting will yield slightly faster response time. • The 60 Hz selection synchronies the input scanning to 60Hz. • The 50 Hz selection synchronies the input scanning to 50Hz.
AO Channel Type	Defines the output signal type for the analog output. • The 0-20 mA selection sets the output type for 0 to 20mA signals. • The 4-20 mA selection sets the output type for 4 to 20mA signals.

SCADAPack 5607 I/O Module

The **SCADAPack 5607 I/O** module is comprised of four types of I/O channels, digital inputs, digital outputs, analog inputs, and analog outputs.

The **digital input** assignment provides twenty four I/O channels for the digital input data from the SCADAPack 5607 I/O Module hardware. The digital input registers are updated continuously with data read from the digital inputs.

- The 5607 I/O Module provides sixteen external digital inputs. These are externally wetted inputs.
- The module provides 8 internal digital inputs, which indicate if the corresponding analog input is in or out of range.

The **digital output** assignment provides ten digital outputs. All are digital output relays. The digital outputs are updated continuously with data read from the coil registers.

The **analog input** assignment provides eight I/O channels for the analog input data from the SCADAPack 5607 I/O Module hardware. The analog input registers are updated continuously with data read from the analog inputs.

The **analog output** assignment provides two I/O channels for the analog output data for the SCADAPack 5607 I/O Module hardware when the optional analog output module is installed.The analog output registers are updated continuously with data read from the analog output registers.

Module Properties

Parameter	Description
Module Name	SCADAPack 5607 I/O Module
Module Address	The address of the module is selected using dip switches on the module. A maximum of eight SCADAPack 5607 I/O type modules may be added to a system. Default value: 0
DO Start Register	First register of any unused block of ten consecutive digital output registers. The default value is 1.
DI Start Register	First register of any unused block of twenty four consecutive digital input registers. The default value is 10001.

	DI Start Register to DI Start Register +15 are physical digital input data. (0 indicates OFF and 1 indicates ON). DI Start Register +16 (OFF = AI1 is OK; ON AI1 is is over or under range) DI Start Register +17 (OFF = AI2 is OK; ON AI2 is is over or under range) DI Start Register +18 (OFF = AI3 is OK; ON AI3 is is over or under range) DI Start Register +19 (OFF = AI4 is OK; ON AI4 is is over or under range) DI Start Register +20 (OFF = AI5 is OK; ON AI5 is is over or under range) DI Start Register +21 (OFF = AI6 is OK; ON AI6 is is over or under range) DI Start Register +22 (OFF = AI7 is OK; ON AI7 is is over or under range) DI Start Register +23 (OFF = AI8 is OK; ON AI8 is is over or under range)
AI Start Register	First register of any unused block of eight consecutive analog input registers. The default value is 30001.
AO Start Register	First register of any unused block of two consecutive analog output registers. The default value is 40001.
AI*n* Channel Type	Defines the input measurement type for each input (AI 0 through 7). • The 0 to 5V selection sets the input type to measure 0 to 5V input signals. • The 1 to 5V selection sets the input type to measure 1 to 5V input signals. • The 0 to 20 mA selection sets the input type to measure 0 to 20mA input signals. • The 4 to 20 mA selection sets the input type to measure 4 to 20mA input signals.
Module Filter	Sets the input filter rate for all analog inputs. The filter rate is used to dampen process variations or noise. • The 3 Hz filter selection sets the response time to 155ms at 60Hz and 185ms at 50Hz. • The 6 Hz filter selection sets response time to 85ms at 60Hz and 85ms at 50Hz. • The 11 Hz filter selection sets response time to 45ms at 60Hz and 55ms at 50Hz. • The 30 Hz filter selection sets response time to 30ms at 60Hz and 30ms at 50Hz.
Module Scan Frequency	Sets the input scan rate for all analog inputs. The scan rate selection is not critical but AC noise rejection is improved at the correct frequency. If the module is used in a DC environment, the 60 Hz setting will yield slightly faster response time. • The 60 Hz selection synchronies the input scanning to 60Hz. • The 50 Hz selection synchronies the input scanning to 50Hz.
AO Channel Type	Defines the output signal type for the analog output. • The 0-20 mA selection sets the output type for 0 to 20mA signals. • The 4-20 mA selection sets the output type for 4 to 20mA signals.

Telebus Protocols Reference

The Telebus Protocol communication protocols provide a standard communication interface to SCADAPack controllers.

There are two Telebus Protocol serial protocols. The Telebus Protocol RTU protocol is compatible with the Modbus RTU protocol and the Telebus Protocol ASCII protocol is compatible with the Modbus ASCII protocol.

There are six Internet protocols. They are Modbus/TCP; Modbus/UDP; Modbus RTU in TCP; Modbus ASCII in TCP; Modbus RTU in UDP; and Modbus ASCII in UDP. These six protocols use the Internet Protocol (IP) at the Network layer. They can be used on any IP network. They differ from each other at the transport and presentation layers. All six protocols can use any IP compatible data link and physical layers. These layers are not part of this document.

Compatibility

Compatibility refers to communication only. The protocol defines communication aspects such as commands, syntax, message framing, error handling and addressing. The controllers do not mimic the internal functioning of any programmable controller. Device specific functions that relate to the hardware or programming of a specific programmable controller are not implemented.

Serial Protocols

There are two Telebus Protocol serial protocols. The Telebus Protocol RTU protocol is compatible with the Modbus RTU protocol and the Telebus Protocol ASCII protocol is compatible with the Modbus ASCII protocol.

Internet Protocols

There are six Internet protocols. They are Modbus/TCP; Modbus/UDP; Modbus RTU in TCP; Modbus ASCII in TCP; Modbus RTU in UDP; and Modbus ASCII in UDP.

These six protocols use the Internet Protocol (IP) at the Network layer. They can be used on any IP network. They differ from each other at the transport and presentation layers. All six protocols can use any IP compatible data link and physical layers. These layers are not part of this document.

Serial Protocols

The Telebus Protocol supports two serial protocols. They are compatible with the **Modbus ASCII** and **Modbus RTU** protocols. The protocols differ in the message framing, method of data encoding, and checksum calculation

- Message framing allows the receiving device to locate the start and end of a protocol message.
- The data encoding describes how the message data are transmitted.
- The checksum is used for error detection. The checksum is calculated by the transmitter and verified by the receiver.

Modbus RTU

The **Modbus RTU** protocol is the more efficient of the two serial protocols. It sends about one-half the number of bytes as the ASCII mode. The checksum detects more

errors. However, the framing method uses message timing so is more sensitive to out-of-message noise and timing on the communication network.

Modbus RTU messages are framed by timing. A timing gap of 3.5 character times, at the current data rate, determines the end of a message. This is shown in the figure below. The potential message is all bytes between the two timing gaps.

The message consists of the addressing, command or response data and a checksum.

- Data are encoded as eight-bit binary values.
- The maximum number of bytes in a message, including address and checksum, is 256.
- The Modbus RTU checksum is the last two bytes of the message.
- The checksum is calculated using a 16-bit Cyclic Redundancy Check (CRC).

Modbus ASCII

The **Modbus ASCII** protocol is the less efficient of the two serial protocols. It sends about twice the number of bytes as the RTU mode. The checksum detects fewer errors. However, the framing method is less sensitive to out-of-message noise and timing on the communication network. The messages are also printable characters so errors can be easier to diagnose.

Unique characters at the start and end of each message frame Modbus ASCII messages.

- Each Modbus ASCII message starts with the colon (:) character. In the ASCII character set, this is character 58 (0x3A).
- Each Modbus ASCII message ends with a carriage return (CR) and line feed (LF) characters. In the ASCII character set, CR is character 13 (0x0D) and LF is character 10 (0x0A).
- The CR terminates the message. The LF indicates to the receiving station that the transmitting station is ready to receive a response.

The message consists of the addressing, command or response data and a checksum.

Modbus ASCII messages are encoded in ASCII Hexadecimal notation. The message data consists entirely of the printable ASCII characters 0 to 9 and A to F.

Each byte of the message is transmitted as two characters, representing the value of the byte in hexadecimal. To determine the characters to transmit, convert the byte into hexadecimal, and transmit the two characters that correspond to it.

For example, suppose the message consists of three bytes with the values 1, 7 and 248. The string transmitted would be 0107F8. This is shown in the figure below.

Message (decimal)	1		7		248	
Message (hexadecimal)	01		07		F8	
Characters transmitter	0	1	0	7	F	8
ASCII value of character	48	49	48	55	70	56

The maximum number of characters in a message, including all address and checksum, is 512. This corresponds to the Modbus RTU limit of 256 bytes, as two characters are transmitted for each byte of the message.

The Modbus ASCII checksum is the last byte of the message. The checksum is calculated using a Longitudinal Redundancy Check (LRC) algorithm. Note that the checksum is calculated on the message before it is encoded. The checksum is not the sum of the characters transmitted.

Serial Port Configuration

This section describes the parameters you need to set when configuring a serial port.

Communication Parameters

The Telebus Protocol protocols are, in general, independent of the serial communication parameters. The baud rate, word length and parity may be chosen to suit the host computer and the characteristics of the data link.

The port configuration can be set in four ways:

- using the Telepace program;
- using the **set_port** function from a C application program;
- writing to the I/O database from a C or ladder logic application program; or
- writing to the I/O database remotely from a Modbus compatible device.

To configure a serial port through the I/O database, add the module and CNFG Serial port settings to the Register Assignment Table.

RTU Protocol Parameters

The TeleBUS RTU protocol is an eight-bit binary protocol. The table below shows possible and recommended communication parameters.

Parameter	Possible Settings	Recommended Setting
Baud rate	see Baud Rate	see Baud Rate
Data bits	8	8
Parity	none	none

	even	
	odd	
Stop bits	1 stop bit	1 stop bit
	2 stop bits	
Flow control	disabled	disabled
Duplex	see Duplex mode	see Duplex mode

Note that the selection for 2 Stop bits is not available for SCADAPack 314, SCADAPack 33x, SCADAPack 350, or SCADAPack 4203.

ASCII Protocol Parameters

The TeleBUS ASCII protocol is an seven bit character based protocol. The table below shows possible and recommended communication parameters.

Parameter	Possible Settings	Recommended Setting
Baud rate	see Baud Rate	see Baud Rate
Data bits	7 data bits	7 data bits
	8 data bits	
Parity	none	none
	even	
	odd	
Stop bits	1 stop bit	1 stop bit
	2 stop bits	
Flow control	disabled	disabled
Duplex	see Duplex mode	see Duplex mode

Note that the selection for 2 Stop bits is not available for SCADAPack 314, SCADAPack 33x, SCADAPack 350, or SCADAPack 4203.

Note:	Flow control should never be enabled with modems or in noisy environments. Noise can result in the accidental detection of an XOFF character, which shuts down communication. Flow control is not recommended for any environment, but can be used on high quality, full duplex, direct wiring where speeds greater than 4800 baud are required.

Baud Rate

The baud rate sets the communication speed. The type of serial data link used determines the possible settings. The table below shows the possible settings for SCADAPack and <%UNIT1R%> controllers. Note that not all port types and baud rates are available on all controller ports.

Note:	All port types and baud rates are not available on all controller ports.

Port Type	Possible baud Settings	Recommended Setting
RS-232 or RS-232 Dial-up modem	75, 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200	Use the highest baud setting supported by all devices on your network.
RS-485	75, 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600. 115200	Use the highest baud setting supported by all devices on your network.

Duplex Mode

The Telebus Protocol protocols communicate in one direction at a time. However the type of serial data link used determines the duplex setting. The table below shows the possible settings forSCADAPack and <%UNIT1R%> controllers.

Note:	All port types and baud rates are not available on all controller ports.

Port Type	Possible baud Settings	Recommended Setting
RS-232 or RS-232 Dial-up modem	half duplex	Use full duplex wherever possible. Use half duplex for most external modems.
	full duplex	
RS-485	half duplex	Slave stations always use half duplex.
	full duplex	Master stations can use full duplex only on 4-wire systems.

Protocol Parameters

Description

The Telebus Protocol protocols operate independently on each serial port. Each port may set the protocol type, station number, protocol task priority, and store-and-forward messaging options.

The port configuration can be set in four ways:

- using the Telepace or IEC 61131-3 programs
- using the **set_protocol** function from a C or C++ application program
- writing to the I/O database from a C, C++, IEC 61131-3 or Ladder Logic application program
- writing to the I/O database remotely from a Modbus compatible device

To configure protocol settings through the I/O database, add the module and CNFG Protocol settings to the Register Assignment for Telepace applications or use the **setprot** function inIEC 61131-3 applications.

Related Topics

Telebus Protocols Overview

Communication Parameters

Protocol Type

The protocol type may be set to Modbus ASCII and Modbus RTU protocols, or it may be disabled. When the protocol is disabled, the port functions as a normal serial port.

Related Topics

Station Number

Task Priority

Store and Forward Messages

Station Number

The Telebus Protocol protocol allows up to 254 devices on a network using standard addressing and up to 65534 devices using extended addressing. Station numbers identify each device. A device responds to commands addressed to it, or to commands broadcast to all stations.

The station number is in the range 1 to 254 for standard addressing and 1 to 65534 for extended addressing. Address 0 indicates a command broadcast to all stations, and cannot be used as a station number. Each serial port may have a unique station number.

Related Topics

Protocol Parameters

Protocol Type

Task Priority

Store and Forward Messages

Task Priority

A task is responsible for monitoring each serial port for messages. The real time operating system (RTOS) schedules the tasks with the application program tasks according to the task priority. The priority can be changed only with the **set_protocol** function from an application program.

The default task priority is 3. Changing the priority is not recommended.

Related Topics

Protocol Parameters

Protocol Type

Task Priority

Store and Forward Messages

Store and Forward Messaging

Store and forward messaging re-transmits messages received by a controller. Messages may be re-transmitted on any serial port, with or without station address translation. A user-defined translation table determines actions performed for each message.

Store and forward messaging may be enabled or disabled on each port. It is disabled by default.

Related Topics

Protocol Parameters

Protocol Type

Station Number

Task Priority

Internet Protocols

There are six Internet protocols. They are Modbus/TCP; Modbus/UDP; Modbus RTU over TCP; Modbus ASCII over TCP; Modbus RTU over UDP; and Modbus ASCII over UDP. All six protocols use the Internet Protocol (IP) at the Network layer. They can be used on any IP network. They differ from each other at the transport and presentation layers. All six protocols can use any IP compatible data link and physical layers. These layers are not part of this document.

Modbus/TCP Protocol

Modbus/TCP is an extension of serial Modbus, which defines how Modbus messages are encoded within and transported over TCP/IP-based networks. The Modbus/TCP protocol uses a custom Modbus protocol layer on top of the TCP protocol. Its request and response messages are prefixed by six bytes. These six bytes consist of three fields: transaction ID field, protocol ID field and length field. The encapsulated Modbus message has exactly the same layout and meaning, from the function code to the end of the data portion, as other Modbus messages. The Modbus 'CRC-16' or 'LRC' check fields are not used in Modbus/TCP. The TCP/IP and link layer (e.g. Ethernet) checksum mechanisms instead are used to verify accurate delivery of the packet.

The Modbus/TCP protocol is a standard protocol. It was originally defined by Schneider Automation and made available as an open protocol. It has been implemented by a number of manufacturers.

Modbus/TCP adds a presentation level header to the application layer message.

Field	Transaction ID	Protocol ID	Length	message
Size (B)	2	2	2	<250

The **Transaction ID** is a unique number for each new command. It allows the receiver to differentiate between new messages and retransmissions. The receiver responds with the same transaction ID as the command. This allows the sender to match responses to commands. This is an important improvement over the serial commands as it allows the sender to reject out of sequence responses.

The P**rotocol ID** identifies the protocol. At present there is only one protocol. The ID is always 0.

The **Length** is the length of the message.

As the name suggests, Modbus/TCP uses TCP as the transport layer. TCP ensures reliable transport. This reliability comes at the cost of setting up and maintaining TCP sessions, and overhead in acknowledging and re-transmitting failed packets.

By default requests are sent via TCP on registered port 502. The implementation allows end users to select the port number.

Modbus/UDP Protocol

Modbus/UDP communication mode is similar to Modbus/TCP communication mode. It has the same message format with the Modbus/TCP. The only difference between them is one uses TCP protocol and another uses UDP protocol.

The Modbus/UDP protocol is a Control Microsystems invention. It is very similar to Modbus/TCP, but uses UDP as the transport layer for higher efficiency.

Modbus/UDP adds a presentation level header to the application layer message.

Field	Transaction ID	Protocol ID	Length	message
Size (B)	2	2	2	<250

The **Transaction ID** is a unique number for each new command. It allows the receiver to differentiate between new messages and retransmissions. The receiver responds with the same transaction ID as the command. This allows the sender to match responses to commands. This is an important improvement over the serial commands as it allows the sender to reject out of sequence responses.

The **Protocol ID** identifies the protocol. At present there is only one protocol. The ID is always 0.

The **Length** is the length of the message.

As the name suggests, Modbus/UDP uses UDP as the transport layer. UDP is more efficient that TCP. There are no retransmissions and no acknowledgement messages are returned. This is especially useful in radio and satellite links where there is a long turn around time or high cost per message.

By default requests are sent via UDP on registered port 502. The implementation allows end users to select the port number.

Modbus RTU in TCP Protocol

Modbus RTU in TCP message format is exactly same as that of the Modbus RTU protocol. The main difference is that Modbus RTU in TCP protocol communicates with a controller through the Internet and Modbus RTU protocol through the serial port. The Modbus RTU in TCP protocol does not include a six-byte header prefix, as with the Modbus\TCP, but does include the Modbus ‘CRC-16’ or ‘LRC’ check fields. The Modbus RTU in TCP message format supports Modbus RTU message format.

The Modbus RTU over TCP protocol is a de-facto standard protocol implemented by many manufacturers. It encapsulates a Modbus RTU message in a TCP packet.

The presentation layer appends a Modbus RTU checksum to the message. See the Modbus RTU section for details on calculating the checksum.

The Modbus RTU encoding requires approximately half as many bytes as does Modbus ASCII encoding.

TCP is the transport layer. TCP ensures reliable transport. This reliability comes at the cost of setting up and maintaining TCP sessions, and overhead in acknowledging and re-transmitting failed packets.

By default requests are sent via TCP on port 49152. The implementation allows end users to select the port number.

Modbus ASCII in TCP Protocol

Modbus ASCII in TCP message format is exactly same as that of the Modbus ASCII protocol. The main difference is that Modbus ASCII in TCP protocol communicates with a SCADAPack controller through the Internet and Modbus ASCII through the serial port. The Modbus ASCII in TCP protocol does not include a six-byte header prefix, as with the Modbus\TCP, but does include the Modbus ‘CRC-16’ or ‘LRC’ check fields.

The Modbus ASCII over TCP protocol is a de-facto standard protocol implemented by many manufacturers. It encapsulates a Modbus ASCII message in a TCP packet.

The presentation layer applies Modbus ASCII framing and checksum to the message. Data is encoded as ASCII Hexadecimal. See the Modbus ASCII section for details.

The Modbus ASCII encoding requires approximately twice as many bytes as the Modbus RTU encoding.

TCP is the transport layer. TCP ensures reliable transport. This reliability comes at the cost of setting up and maintaining TCP sessions, and overhead in acknowledging and re-transmitting failed packets.

By default requests are sent via TCP on port 49153. The implementation allows end users to select the port number.

Modbus RTU in UDP Protocol

Modbus RTU in UDP protocol is similar to Modbus RTU in TCP protocol. It has the same message format as the RTU in TCP message. The only difference between them is one uses TCP protocol and another uses UDP protocol.

The Modbus RTU over UDP protocol is a de-facto standard protocol implemented by many manufacturers. It encapsulates a Modbus RTU message in a UDP packet.

The presentation layer appends a Modbus RTU checksum to the message. See the Modbus RTU section for details on calculating the checksum.

The Modbus RTU encoding requires approximately half as many bytes as does Modbus ASCII encoding.

UDP is the transport layer. UDP is more efficient that TCP. There are no retransmissions and no acknowledgement messages are returned. This is especially useful in radio and satellite links where there is a long turn around time or high cost per message.

By default requests are sent via UDP on registered port 49152. The implementation allows end users to select the port number.

Modbus ASCII in UDP Protocol

Modbus ASCII in UDP protocol is similar to Modbus ASCII in TCP protocol. It has the same message format as the Modbus ASCII in TCP. The only difference between them is one uses TCP protocol and another uses UDP protocol.

The Modbus ASCII over UDP protocol is a de-facto standard protocol implemented by many manufacturers. It encapsulates a Modbus ASCII message in a UDP packet.

The presentation layer applies Modbus ASCII framing and checksum to the message. Data is encoded as ASCII Hexadecimal. See the Modbus ASCII section for details.

The Modbus ASCII encoding requires approximately twice as many bytes as the Modbus RTU encoding.

UDP is the transport layer. UDP is more efficient that TCP. There are no retransmissions and no acknowledgement messages are returned. This is especially useful in radio and satellite links where there is a long turn around time or high cost per message.

By default requests are sent via UDP on registered port 49153. The implementation allows end users to select the port number.

I/O Database

The Telebus Protocol protocols read and write information from the I/O database. The I/O database contains user-assigned registers and general purpose registers.

User-assigned registers map directly to the I/O hardware or system parameter in the controller. Assigned registers are initialized to the default hardware state or system parameter when the controller is reset. Assigned output registers do not maintain their values during power failures. However, output registers do retain their values during application program loading.

General purpose registers are used by ladder logic and C application programs to store processed information and to receive information from remote devices. General purpose registers retain their values during power failures and application program loading. The values change only when written by an application program or a communication protocol.

- Telepace ladder logic programs access the I/O database through function blocks. All function blocks can access the I/O database.
- C language programs access the I/O database with two functions. The **dbase** function reads a value from the I/O database. The **setdbase** function writes a value to the I/O database.

The I/O database is divided into four sections.

Coil registers are single bits which the protocols can read and write. Coil registers are located in the digital output section of the I/O database. The number of registers depends on the controller. Coil registers are numbered from 1 to the maximum for the controller. Coil and status registers contain one bit of information, that is, whether a signal is on or off.

- Writing any non-zero value to the register turns the bit on. Writing zero to the register turns the bit off. If the register is assigned to an I/O module, the bit status is written to the module output hardware or parameter.
- Reading a coil register returns –1 if the bit is on, or 0 if the bit is off. The stored value is returned from general purpose registers. The I/O module point status is returned from assigned registers.

Status registers are single bits which the protocol can read. Status registers are located in the digital input section of the I/O database. The number of registers depends on the controller. Status registers are numbered from 10001 to the maximum for the controller.

- Reading a status register returns –1 if the bit is on, or 0 if the bit is off. The stored value is returned from general purpose registers. The I/O module point status is returned from assigned registers.

Input registers are 16-bit registers which the protocol can read. Input registers are located in the analog input section of the I/O database. The number of registers depends on the controller. Input registers are numbered from 30001 to the maximum for the controller.

- Reading an input register returns the value stored in the register. Reading an assigned register returns the value read from the I/O module.

Holding registers are 16 bit registers that the protocol can read and write. Holding registers are located in the analog output section of the I/O database. The number of registers depends on the controller. Holding registers are numbered from 40001 to the maximum for the controller.

- Writing any value to a holding register stores the value in the register. Writing a value to an assigned register, writes the value to the assigned I/O module.
- Reading a holding register returns the value stored in the register. Reading an assigned register returns the value read from the I/O module.

Extended Station Addressing

The TeleBUS serial and Internet protocols support two type of Modbus station addressing. Standard Modbus addressing allows a maximum of 255 stations and is compatible with standard Modbus devices.

Extended Modbus addressing allows a maximum of 65534 stations. Extended Modbus addressing is fully compatible with standard Modbus addressing for addresses between 0 and 254.

Theory of Operation

The address field of a Modbus message is a single byte. Address 0 is a broadcast address; messages sent to this address are sent to all stations. Addresses 1 to 255 are station addresses. Figure 1 shows the format of a standard Modbus message.

Field	Address	Function	...
Size	1	1	N

The address field extension adds a two-byte extended address field to the message. Figure 2 shows the format of an extended address Modbus message.

Field	Address = 255	Extended Address (high)	Extended Address (low)	Function	...
Size	1	1	1	1	n

Messages for addresses 0 to 254 use the standard format message. The station address is stored in the address byte.

Messages for stations 255 to 65534 use the extended address format message. The address byte is set to 255. This indicates the extended address format is used. The actual address is stored in the two extended address bytes.

Station address 65535 is reserved and cannot be used as a station number. This station address is used in store-and-forward tables to indicate a disabled station.

Slave, master and store-and-forward stations treat the addresses in the same manner. The application program controls the use of the extended addressing format. It may enable or disable the extended addressing.

Modbus Master Mode

The Telebus Protocol protocol can act as a communication master on any serial port. In master mode, the controller sends commands to other devices on the network. Simultaneous master messages may be active on all ports.

The protocol cannot support master mode and store-and-forward mode simultaneously on a serial port. Enabling store and forward messaging disables processing of responses to master mode commands. Master mode may be used on one port and store-and-forward mode on another port.

Modbus Function Codes

The table shows the implemented function codes. The maximum number of registers that can be read or written with one message is shown in the maximum column. The slave device may support fewer registers than shown; consult the manual for the device for details.

Function	Name	Description	Maximum
01	Read Coil Status	Reads digital output registers.	2000
02	Read Input Status	Reads digital input registers.	2000
03	Read Holding Register	Reads analog output registers.	125
04	Read Input Register	Reads analog input registers.	125
05	Force Single Coil	Writes digital output register.	1
06	Preset Single Register	Writes analog output registers.	1
15	Force Multiple Coils	Writes digital output registers.	880
16	Preset Multiple Registers	Writes analog output registers.	60

Read Coil Status

The Read Coil Status function reads data from coil registers in the remote device. Data can be written into the digital input or the digital output sections of the I/O database.

Any number of registers may be read up to the maximum number supported by the slave device or the maximum number above, whichever is less. The read may start at any address, provided the entire block is within the valid register range. Each register is one bit.

Read Input Status

The Read Input Status function reads data from input registers in the remote device. Data can be written into the digital input or the digital output sections of the I/O database.

Any number of registers may be read up to the maximum number supported by the slave device or the maximum number above, whichever is less. The read may start at any address, provided the entire block is within the valid register range. Each register is one bit.

Read Holding Register

The Read Holding Register function reads data from holding registers in the remote device. Data can be written into the analog input or the analog output sections of the I/O database.

Any number of registers may be read up to the maximum number supported by the slave device or the maximum number above, whichever is less. The read may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

Read Input Register

The Read Input Register function reads data from input registers in the remote device. Data can be written into the analog input or the analog output sections of the I/O database.

Any number of registers may be read up to the maximum number supported by the slave device or the maximum number above, whichever is less. The read may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

Force Single Coil

The Force Single Coil function writes one bit into a coil register in the remote device. The data may come from the digital input or digital output sections of the I/O database.

The write may specify any valid coil register in the remote device.

Preset Single Register

The Preset Single Register function writes one 16 bit value into a holding register in the remote device. The data may come from the analog input or output sections of the I/O database.

The write may specify any valid holding register in the remote device.

Force Multiple Coils

The Force Multiple Coils function writes single bit values coil registers in the remote device. The data may come from the digital input or digital output sections of the I/O database.

Any number of registers may be written up to the maximum number supported by the slave device or the maximum number above, which ever is less. The write may start at any address, provided the entire block is within the valid register range of the remote device. Each register is 1 bit.

Preset Multiple Registers

The Preset Multiple Register function writes 16 bit values into holding registers of the remote device. The data may come from the analog input or output sections of the I/O database.

Any number of registers may be written up to the maximum number supported by the slave device or the maximum number above, which ever is less. The write may start at any address, provided the entire block is within the valid register range of the remote device. Each register is 16 bits.

Enron Modbus Master Mode

The Enron Modbus protocol is based on the Modbus ASCII and RTU protocols. Message framing is identical to the Modbus protocols. However, there are many differences in message formatting and register numbering, at both the logical and protocol levels.

Variable Types

There are ranges of Enron registers to hold short integers, long integers and single precision floats. The ranges are as follows.

Range	Data Type

1001 - 1999	Boolean
3001 - 3999	Short integer
5001 - 5999	Long integer
7000 - 9999	Float

In general, both numeric and Boolean function codes can be used to read and write all types of registers.

Boolean Registers

Enron Modbus Boolean registers are usually numbered 1001 to 1999.

Boolean registers are read using Modbus command 1 and written using Modbus command 5 for single registers and 15 for multiple registers.

The address offset in the message is equal to the register number.

The number of Modbus registers is equal to the number of Enron registers.

The response format is identical to the Modbus response format.

Short Integer Registers

Enron Modbus Short Integer registers are usually numbered 3001 to 3999.

Short Integer registers are read using Modbus command 3. Short Integer registers are written using Modbus command 6 for single registers and 16 for multiple registers.

The address offset in the message is equal to the register number.

The number of Modbus registers is equal to the number of Enron registers.

The response format is identical to the Modbus response format.

Long Integer Registers

Enron Modbus Long Integer registers are usually numbered 5001 to 5999.

Long Integer registers are read using Modbus command 3. Long Integer registers are written using Modbus command 6 for single registers and 16 for multiple registers.

The address offset in the message is equal to the register number.

The number of Modbus registers requested is equal to the number of Enron registers.

The number of Modbus registers expected in the response is equal to two times the number of Enron registers.

Floating Point Registers

Enron Modbus Floating-point registers are usually numbered 7001 to 7999.

Floating-point registers are read using Modbus command 3. Floating-point registers are written using Modbus command 6 for single registers and 16 for multiple registers.

The address offset in the message is equal to the register number.

The number of Modbus registers requested is equal to the number of Enron registers.

The number of Modbus registers expected in the response is equal to two times the number of Enron registers.

Enron Modbus Function Codes

The following table shows the implemented function codes for Enron Modbus. The maximum number of registers that can be read or written with one message is shown in the maximum column. The slave device may support fewer registers than shown; consult the manual for the device for details.

Functions 129, 130, 132, 133, 135, 136, 138, and 139 may be broadcast, but some Enron Modbus slave devices may not support broadcast messages. Consult the manual for the device for details.

Function	Name	Description	Maximum
128	Read Enron Boolean	Read Enron Boolean registers	2000
129	Write Enron Boolean	Write Enron Boolean register	1
130	Write Enron Multiple Boolean	Write Enron Boolean registers	880
131	Read Enron Short Integer	Read Enron short integer register	125
132	Write Enron Short Integer	Write Enron short integer register	1
133	Write Enron Multiple Short Integer	Write Enron short integer registers	60
134	Read Enron Long Integer	Read Enron long integer register	62
135	Write Enron Long Integer	Write Enron long integer register	1
136	Write Enron Multiple Long Integer	Write Enron long integer registers	30
137	Read Enron Floating Point	Read Enron floating-point register	62
138	Write Enron Floating Point	Write Enron floating-point register	1
139	Write Enron Multiple Floating Point	Write Enron floating-point registers	30

Sending Messages

Description

A master message is initiated in one of five ways:

- using the **master_message** function from a C or C++ application program; or
- using the **MSTR** function block from a Telepace ladder logic program; or
- using the **MSIP** function block from a Telepace ladder logic program; or

- using the master function in an IEC 61131-3 program; or
- using the **masterip** function in an IEC 61131-3 program.

These functions specify the port on which to issue the command, the function code, the type of station addressing, the slave station number, and the location and size of the data in the slave and master devices. The protocol driver, independent of the application program receives the response to the command.

The application program detects the completion of the transaction by:

- calling the **get_protocol_status** function in a C application program; or
- using the output of the **MSTR** function block in a Telepace ladder logic program; or
- using the output of the **master** function in an IEC 61131-3 program.

A communication error has occurred if the slave does not respond within the expected maximum time for the complete command and response. The application program is responsible for detecting this condition. When errors occur, it is recommended that the application program retry several times before indicating a communication failure.

The completion time depends on the length of the message, the length of the response, the number of transmitted bits per character, the transmission baud rate, and the maximum message turn-around time. One to three seconds is usually sufficient. Radio systems may require longer delays.

Related Topics

Modbus Master Mode

Modbus Slave Mode

The TeleBUS protocols operate in slave and master modes simultaneously. In slave mode the controller responds to commands sent by another device. Commands may be sent to a specific device or broadcast to all devices.

The TeleBUS protocols emulate the Modbus protocol functions required for communication with a host device. These functions are described below. It also implements functions for programming and remote diagnostics. These functions are not required for host communication, so are not described here.

Modbus Function Codes

The table summarizes the implemented function codes. The maximum number of registers that can be read or written with one message is shown in the maximum column.

Function	Name	Description	Maximum
01	Read Coil Status	Reads digital output registers.	2000
02	Read Input Status	Reads digital input registers.	2000
03	Read Holding Register	Reads analog output registers.	125
04	Read Input Register	Reads analog input registers.	125
05	Force Single Coil	Writes digital output register.	1
06	Preset Single Register	Writes analog output registers.	1
07	Read Exception Status	Reads special information.	N/A
15	Force Multiple Coils	Writes digital output registers.	880
16	Preset Multiple Registers	Writes analog output registers.	60
17	Report Slave ID	Reads controller type information	N/A

Functions 5, 6, 15, and 16 support broadcast messages. The functions are described in detail below.

Read Coil Status

The Read Coil Status function reads data from the digital output section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is one bit.

Read Input Status

The Read Input Status function reads data from the digital input section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is one bit.

Read Holding Register

The Read Holding Register function reads data from the analog output section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

Read Input Register

The Read Input Register function reads data from the analog input section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

Force Single Coil

The Force Single Coil function writes one bit into the digital output section of the I/O database. The write may specify any valid register.

Preset Single Register

The Preset Single Register function writes one 16 bit value into the analog output section of the I/O database. The write may specify any valid register.

Read Exception Status

The Read Exception Status function reads a single byte containing controller specific status information. The information is defined by the application program. This function is included for compatibility with devices expecting to communicate with a Modicon PLC.

Force Multiple Coils

The Force Multiple Coils function writes single bit values into the digital output section of the I/O database. Any number of registers may be written up to the maximum number. The write may start at any address, provided the entire block is within the valid register range. Each register is 1 bit.

Preset Multiple Registers

The Preset Multiple Register function writes 16 bit values into the analog output section of the I/O database. Any number of registers may be written up to the maximum number. The write may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

Report Slave ID

The Report Slave ID function reads a variable length message containing controller specific information. The information and the length of the message is defined by the application program. This function is included for compatibility with devices expecting to communicate with a Modicon PLC.

Function Codes

The table summarizes the implemented function codes. The maximum number of registers that can be read or written with one message is shown in the maximum column.

Function Code	Function	Description	Maximum Registers
01	Read Coil Status	Reads digital output registers.	2000
02	Read Input Status	Reads digital input retgisters.	2000
03	Read Holding Register	Reads analog output registers.	125
04	Read Input Register	Reads analog input registers.	125
05	Force Single Coil	Writes digital output register.	1
06	Preset Single Register	Writes analog output registers.	1
07	Read Exception Status	Reads special information.	N/A
15	Force Multiple Coils	Writes digital output registers.	880
16	Preset Multiple Registers	Writes analog output registers.	60
17	Report Slave ID	Reads controller type information.	N/A

Functions 5, 6, 15, and 16 support broadcast messages. For more information on these functions, see:

Read Coil Status

The **Read Coil Status** function reads data from the digital output section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is one bit.

Command Message Format

This command reads the values of coil registers in the slave device. Coil registers are single bit outputs. They may be on or off.

The command allows up to 2000 registers to be read, but the slave device may impose a smaller limit.

The starting address is the offset from the first coil register. For example the address for coil 1 is 0; coil 2 is 1, etc. The slave device determines the limit on the address.

Byte	Description	Valid Values
0	function code	1 = Read Coil Status
1	start register offset high byte	0 to 9998
2	start register offset low byte	
3	number of registers high byte	1 to 2000
4	number of registers low byte	

Response Message Format

The response is formatted as follows. If the number of registers requested is not a multiple of eight, the high order bits of the last data byte will be filled with zeros.

Byte	Description	Valid Values
0	function code	1 = Read Coil Status
1	length = n	1 to 250
2	coil register values for 8 coils	bit 0 = first coil value bit 7 = eighth coil value
…		
1+n	coil register values for 8 coils	bit 0 = first coil value bit 7 = eighth coil value

Read Input Status

The **Read Input Status** function reads data from the digital input section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is one bit.

Command Message Format

This command reads the values of input status registers in the slave device. Input status registers are single bit inputs. They may be on or off.

The command allows up to 2000 registers to be read, but the slave device may impose a smaller limit.

The starting address is the offset from the first input status register. For example the address for input 10001 is 0; input 10002 is 1, etc. The slave device determines the limit on the address.

Byte	Description	Valid Values
0	function code	2 = Read Input Status
1	start register offset high byte	0 to 9998
2	start register offset low byte	
3	number of registers high byte	1 to 2000
4	number of registers low byte	

Response Message Format

The response is formatted as follows. If the number of registers requested is not a multiple of eight, the high order bits of the last data byte will be filled with zeros.

Byte	Description	Valid Values
0	function code	2 = Read Input Status
1	length = n	1 to 250
2	input register values for 8 coils	bit 0 = first input value bit 7 = eighth input value
…		
1+n	input register values for 8 coils	bit 0 = first input value bit 7 = eighth input value

Read Holding Register

The **Read Holding Register** function reads data from the analog output section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

Command Message Format

This command reads the values of holding registers in the slave device. Holding registers are 16-bit outputs. They may hold a value from 0 to 65535.

The command allows up to 125 registers to be read, but the slave device may impose a smaller limit.

The starting address is the offset from the first holding register. For example the address for input 40001 is 0; input 40002 is 1, etc. The slave device determines the limit on the address.

Byte	Description	Valid Values
0	function code	3 = Read Holding Register
1	start register offset high byte	0 to 9998
2	start register offset low byte	
3	number of registers high byte	1 to 125
4	number of registers low byte	

Response Message Format

Byte	Description	Valid Values
0	function code	3 = Read Holding Register
1	length = n	1 to 250
2	holding register value high byte	0 to 65535
3	holding register value low byte	
…		
n	holding register value high byte	0 to 65535
n+1	holding register value low byte	

Read Input Register

The **Read Input Register** function reads data from the analog input section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

Command Message Format

This command reads the values of input registers in the slave device. Input registers are 16-bit inputs. They may hold a value from 0 to 65535.

The command allows up to 125 registers to be read, but the slave device may impose a smaller limit.

The starting address is the offset from the first input register. For example the address for input 30001 is 0; input 30002 is 1, etc. The slave device determines the limit on the address.

Byte	Description	Valid Values
0	function code	4 = Read Input Register
1	start register offset high byte	0 to 9998
2	start register offset low byte	
3	number of registers high byte	1 to 125
4	number of registers low byte	

Response Message Format

Byte	Description	Valid Values
0	function code	4 = Read Input Register

1	length = n	1 to 250
2	input register value high byte	0 to 65535
3	input register value low byte	
…		
n	input register value high byte	0 to 65535
n+1	input register value low byte	

Force Single Coil

The **Force Single Coil** function writes one bit into the digital output section of the I/O database. The write may specify any valid register.

Command Message Format

This command writes to one coil register. Coil registers are single bit outputs. They may be on or off.

This command supports broadcast mode. All slaves will write to the specified coil register if the broadcast address is used.

The starting address is the offset from the first coil register. For example the address for coil 1 is 0; coil 2 is 1, etc. The slave device determines the limit on the address.

Byte	Description	Valid Values
0	function code	5 = Force Single Coil
1	coil register offset high byte	0 to 9998
2	coil register offset low byte	
3	coil value	0x00 = off 0xFF = on
4	data	must be set to 0

Response Message Format

Note that the response is identical to the command.

Byte	Description	Valid Values
0	function code	5 = Force Single Coil
1	coil register offset high byte	same as command
2	coil register offset low byte	same as command
3	coil value	same as command
4	data	same as command

If the slave device supports forcing of registers, the response is identical, but a forced register is not modified.

Preset Single Register

The **Preset Single Register** function writes one 16 bit value into the analog output section of the I/O database. The write may specify any valid register.

Command Message Format

This command writes to one holding register. Holding registers are 16-bit outputs. They may hold a value from 0 to 65535.

This command supports broadcast mode. All slaves will write to the specified holding register if the broadcast address is used.

The starting address is the offset from the first holding register. For example the address for input 40001 is 0; input 40002 is 1, etc. The slave device determines the limit on the address.

Byte	Description	Valid Values
0	function code	6 = Load Single Register
1	holding register offset high byte	0 to 9998
2	holding register offset low byte	
3	register value high byte	0 to 65535
4	register value low byte	

Response Message Format

Note that the response is identical to the command.

Byte	Description	Valid Values
0	function code	6 = Load Single Register
1	holding register offset high byte	same as command
2	holding register offset low byte	same as command
3	register value high byte	same as command
4	register value low byte	same as command

If the slave device supports forcing of registers, the response is identical, but a forced register is not modified.

Read Exception Status

The **Read Exception Status** function reads a single byte containing controller specific status information. The information is defined by the application program. This function is included for compatibility with devices expecting to communicate with a Modicon PLC.

The exception status is not the same as an exception response to a command. The values of the exception status are controller specific.

Command Message Format

Byte	Description	Valid Values
0	function code	7 = Read Exception Status

Response Message Format

Byte	Description	Valid Values
0	function code	7 = Read Exception Status
1	exception status	0 to 255

Force Multiple Coils

The **Force Multiple Coils** function writes single bit values into the digital output section of the I/O database. Any number of registers may be written up to the maximum number. The write may start at any address, provided the entire block is within the valid register range. Each register is 1 bit.

Command Message Format

This command writes to multiple coil registers. Coil registers are single bit outputs. They may be on or off.

This command supports broadcast mode. All slaves will write to the specified coil registers if the broadcast address is used.

The command allows up to 880 registers to be written, but the slave device may impose a smaller limit.

The starting address is the offset from the first coil register. For example the address for coil 1 is 0; coil 2 is 1, etc. The slave device determines the limit on the address.

Byte	Description	Valid Values
0	function code	15 = Force Multiple Coils
1	coil register offset high byte	0 to 9998
2	coil register offset low byte	
3	number of coils high byte	1 to 880
4	number of coils low byte	
5	length = n	1 to 110
6	coil register values for 8 coils	bit 0 = first coil value bit 7 = eighth coil value
…		
5+n	coil register values for 8 coils	bit 0 = first coil value bit 7 = eighth coil value

Response Message Format

Byte	Description	Valid Values
0	function code	15 = Force Multiple Coils
1	coil register offset high byte	same as command
2	coil register offset low byte	same as command
3	number of coils high byte	same as command
4	number of coils low byte	same as command

If the slave device supports forcing of registers, the response is identical, but any forced registers are not modified.

Preset Multiple Registers

The **Preset Multiple Register** function writes 16 bit values into the analog output section of the I/O database. Any number of registers may be written up to the maximum number. The write may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

Command Message Format

This command writes to multiple holding registers. Holding registers are 16-bit outputs. They may hold a value from 0 to 65535.

This command supports broadcast mode. All slaves will write to the specified holding registers if the broadcast address is used.

The command allows up to 60 registers to be written, but the slave device may impose a smaller limit.

The starting address is the offset from the first holding register. For example the address for input 40001 is 0; input 40002 is 1, etc. The slave device determines the limit on the address.

Byte	Description	Valid Values
0	function code	16 = Load Multiple Registers
1	holding register offset high byte	0 to 9998

2	holding register offset low byte	
3	number of registers high byte	1 to 60
4	number of registers low byte	
5	length = n	2 to 120
6	holding register value high byte	0 to 65535
7	holding register value low byte	
…		
4+n	holding register value high byte	0 to 65535
5+n	holding register value low byte	

Response Message Format

Byte	Description	Valid Values
0	function code	16 = Load Multiple Registers
1	holding register offset high byte	same as command
2	holding register offset low byte	same as command
3	number of registers high byte	same as command
4	number of registers low byte	same as command

If the slave device supports forcing of registers, the response is identical, but any forced registers are not modified.

Report Slave ID

The **Report Slave ID** function reads a variable length message containing controller specific information. The information and the length of the message is defined by the application program. This function is included for compatibility with devices expecting to communicate with a Modicon PLC.

Command Message Format

This command reports the slave ID data. The implementation of this function is specific to the controller.

Command Message Format		
Byte	Description	Valid Values
0	function code	17 = Report Slave ID

Response Message Format

Response Message Format		
Byte	Description	Valid Values
0	function code	17 = Report Slave ID
1	length = n	1 to 200
2	data byte 0	0 to 255
…		
1+n	data byte n	0 to 255

Broadcast Messages

Some commands support broadcast addressing. A broadcast command is sent to all stations. The receiving stations process the command, but do not respond to it. The master device must poll the slaves individually to obtain data resulting from the action of a broadcast command. Address 0 is the broadcast address.

Store and Forward Messages

Store and forward messaging is required on systems where there is no direct link between a host computer and all the remote sites. This occurs on radio systems where the host computer transmission cannot be heard by all remote sites. It occurs on systems where one controller is used as a data concentrator for several remote units. With store and forward messaging, a request to a controller that cannot be directly accessed by a host is routed through an intermediate controller, which can communicate with both the host and the remote controller.

The Telebus Protocol protocol provides store and forward messaging through address translation. A controller configured for store and forward operation receives messages destined for a remote station, re-addresses them according to translation table, and forwards the message to the remote station. Responses from the remote station are processed in the same manner.

The Telebus Protocol protocol allows messages to be re-transmitted on the same port with address translation. This is used with radio systems. The radio at the intermediate site is used as a type of repeater. The protocol allows messages to be re-transmitted on a different port, with or without address translation. This is used where the intermediate controller is a bridge between two networks.

The Telebus Protocol protocol driver maintains diagnostics counters at the store and forward site on the number of messages received and transmitted to aid in the diagnosing of communication problems.

On SCADAPack, SCADAPack Plus, SCADAPack Light, SCADAPack LP, SCADAPack 100, and SCADAPack 4203 controllers the protocol cannot support master mode and store-and-forward mode simultaneously on a serial port. Enabling store and forward messaging disables processing of responses to master mode commands.

Master mode may be used on one port and store-and-forward mode on another port. Applications requiring both modes on a single port must switch the modes under control of the application program. This restriction does not apply to SCADAPack 32 and SCADAPack 300 and SCADAPack 4203.

Related Topics

Translation Table

Invalid Translations

Store and Forward Configuration

Translation Table

The translation table specifies address and communication port translation. The translation table differs for SCADAPack and SCADAPack32 controllers. Each entry in the translation table for SCADAPack controllers has four components, as shown in the table entry below.

Port A	Station Address A	Port B	Station Address B
The Port the slave message is received from for each translation.	The station address of the slave message.	The Port the message is forwarded from.	The station address of the forwarded message.

The entry defines a bi-directional transfer. A message (poll or reply) received for station A on port A is re-transmitted to station B on port B. A message received for station B on port B is re-transmitted to station A on port A.

Each entry in the translation table forSCADAPack32 controllers has five components, as shown in the table entry below.

Slave Interface	Slave Station	Forward Interface	Forward Station	Forward IP address
The Slave Interface entry contains the receiving slave interface the message is received from for each translation.	The Slave Station entry contains the Modbus station address of the slave message.	The Forward Interface entry contains the interface the message is forwarded from. When forwarding to a TCP or UDP network, the protocol type is selected for the Forward Interface. The IP Stack automatically determines the exact interface (e.g. Ethernet1) to use when it searches the network for the Forward IP Address.	The Forward Station entry contains the Modbus station address of the forwarded message.	The Forward IP Address entry contains the IP address of the Forward Station. This field is blank unless a TCP or UDP network is selected for Forward Interface.

Table Size

The translation table holds 128 translation entries. This is sufficient to re-transmit one-half of 256 possible addresses. On a single port controller only 128 translations are required since each address must translate to a different address for re-transmission on the same port see Invalid Translations.

Related Topics

Invalid Translations

Store and Forward Configuration

Invalid Translations

The following translations are not valid. The described action is taken when these translations are encountered.

- Re-transmission on the same port with the same address is not valid, except for broadcast messages. This restriction is required because many message responses are identical to the command. It is impossible for the master station to distinguish between the re-transmitted message and the response from the slave. The re-transmitted message would appear to be the response.
- The protocol re-transmits broadcast messages on the same port. Some stations will receive the broadcast message twice. The master station will also receive the message and may execute it if it is able to operate as a slave. The user must bear these consequences in mind when forwarding broadcast messages.
- The store and forward controller also processes broadcast messages.
- Translations where either of the station addresses are the same as the controller station address for the port, are not valid. The protocol processes these messages as if they were directed to the controller. It does not look up the address in the translation table.
- Translations with non-existent port numbers or invalid addresses are not valid.
- Multiple translations for a port and station address combination are not valid.
- Translations where one station is DISABLED and the other station is not, are not valid. A DISABLED translation is a valid translation.

Related Topics

Store and Forward Messages

Translation Table

Diagnostics Counters

The TeleBUS protocol provides diagnostics counters for each serial port. The counters aid in determining the source of communication errors. Store and forward messaging provides the following counters for each communication port. All counters have a maximum count of 65535. Counters roll back to zero on the next event.

- **Stored Message Counter**: the number of messages received, which qualified for forwarding. A message qualifies for forwarding if a valid translation is found for the port and station in the translation table.
- **Forwarded Message Counter**: the number of messages forwarded (transmitted) on this port.

Refer to the user manual for the controller and programming environment you are using for information on the diagnostics counters.

Store and Forward Configuration

The Store and Forward configuration varies depending on the controller you are configuring. See the following topics for the configuration for each type of controller.

SCADAPack Light Controller

SCADAPack Light Controller

SCADAPack Plus Controller

SCADAPack LP Controller

SCADAPack 100 Controller

SCADAPack 3xx, SCADAPack 32 and 32P Controller

SCADAPack Controller

An application program, written inTelepace Ladder Logic or Telepace C Tools and IEC 61131-3 IEC 61131-3 or IEC 61131-3 IEC 61131-3 C tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

Telepace Ladder Logic

1. To enable the use of store and forward messaging on one or more serial ports the **Configuration Module - Serial Port Protocol Settings Method 1, 2 or 3** must be added to the register assignment. The store and forward enable register must be set to enable.
2. Add the **Configuration Module - Store and Forward** to the register assignment to configure the translation table.
3. Configure the translation table by writing the necessary translation table entries to the registers defined in the CNFG Store and Forward I/O module.

Telepace C Tools

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when Telepace programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

The Telepace C language application program interface provides the following functions. Refer to the Telepace C Tools Reference and User Manual for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED (station 256).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

IEC 61131-3

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The **SandFEnabled** input must be set to TRUE.
2. Configure the translation table by using the **setsf** function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when IEC 61131-3 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

IEC 61131-3 C Tools

- The IEC 61131-3 C language application program interface provides the following functions. Refer to the IEC 61131-3 C Tools Reference and User Manual for details.
- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

Related Topics

Store and Forward Configuration

SCADAPack Light Controller

An application program, written in Telepace Ladder Logic or Telepace C Tools and IEC 61131-3 IEC 61131-3 or IEC 61131-3 IEC 61131-3 C Tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

Telepace Ladder Logic

1. To enable the use of store and forward messaging on one or more serial ports the **Configuration Module - Serial Port Protocol Settings Method 1, 2 or 3** must be added to the register assignment. The store and forward enable register must be set to enable.
2. Add the **Configuration Module - Store and Forward** to the register assignment to configure the translation table.
3. Configure the translation table by writing the necessary translation table entries to the registers defined in the CNFG Store and Forward I/O module.

Telepace C Tools

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when Telepace programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

The Telepace C language application program interface provides the following functions. Refer to the Telepace C Tools Reference and User Manual for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED (station 256).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and

type of the first error if it is not valid.

IEC 61131-3

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The **SandFEnabled** input must be set to TRUE.

2. Configure the translation table by using the setsf function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when IEC 61131-3 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

IEC 61131-3 C Tools

- The IEC 61131-3 C language application program interface provides the following functions. Refer to the IEC 61131-3 C Tools Reference and User Manual for details.
- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

Related Topics

Store and Forward Configuration

SCADAPack Plus Controller

An application program, written in Telepace Ladder Logic or Telepace C Tools and IEC 61131-3 C Tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

Telepace Ladder Logic

1. To enable the use of store and forward messaging on one or more serial ports the **Configuration Module - Serial Port Protocol Settings Method 1, 2 or 3** must be added to the register assignment. The store and forward enable register must be set to enable.

2. Add the **Configuration Module - Store and Forward** to the register assignment to configure the translation table.

3. Configure the translation table by writing the necessary translation table entries to the registers defined in the CNFG Store and Forward I/O module.

Telepace C Tools

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when Telepace programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

The Telepace C language application program interface provides the following functions. Refer to the Telepace C Tools Reference and User Manual for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED (station 256).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

Related Topics

Store and Forward Configuration

IEC 61131-3

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The **SandFEnabled** input must be set to TRUE.

2. Configure the translation table by using the setsf function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when IEC 61131-3 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

IEC 61131-3 C Tools

- The IEC 61131-3 C language application program interface provides the following functions. Refer to the IEC 61131-3 C Tools Reference and User Manual for details.
- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

SCADAPack LP Controller

An application program, written in Telepace Ladder Logic or Telepace C Tools and IEC 61131-3 C Tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

Telepace Ladder Logic

1. To enable the use of store and forward messaging on one or more serial ports the **Configuration Module - Serial Port Protocol Settings Method 1, 2 or 3** must be added to the register assignment. The store and forward enable register must be set to enable.
2. Add the **Configuration Module - Store and Forward** to the register assignment to configure the translation table.
3. Configure the translation table by writing the necessary translation table entries to the registers defined in the CNFG Store and Forward I/O module.

Telepace C Tools

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when Telepace programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

The Telepace C language application program interface provides the following functions. Refer to the Telepace C Tools Reference and User Manual for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED (station 256).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

IEC 61131-3

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The **SandFEnabled** input must be set to TRUE.
2. Configure the translation table by using the setsf function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when IEC 61131-3 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

IEC 61131-3 C Tools

- The IEC 61131-3 C language application program interface provides the following functions. Refer to the IEC 61131-3 C Tools Reference and User Manual for details.
- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

Related Topics

Store and Forward Configuration

SCADAPack 100 Controller

An application program, written inTelepace Ladder Logic or Telepace C Tools and IEC 61131-3 IEC 61131-3 or IEC 61131-3 IEC 61131-3 C tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

Telepace Ladder Logic

1. To enable the use of store and forward messaging on one or more serial ports the **Configuration Module - Serial Port Protocol Settings Method 1, 2 or 3** must be added to the register assignment. The store and forward enable register must be set to enable.
2. Add the **Configuration Module - Store and Forward** to the register assignment to configure the translation table.
3. Configure the translation table by writing the necessary translation table entries to the registers defined in the CNFG Store and Forward I/O module.

Telepace C Tools

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when Telepace programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

The Telepace C language application program interface provides the following functions. Refer to the Telepace C Tools Reference and User Manual for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED (station 256).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

IEC 61131-3

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The **SandFEnabled** input must be set to TRUE.
2. Configure the translation table by using the **setsf** function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when IEC 61131-3 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

IEC 61131-3 C Tools

- The IEC 61131-3 C language application program interface provides the following functions. Refer to the IEC 61131-3 C Tools Reference and User Manual for details.
- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.

- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

Related Topics

Store and Forward Configuration

SCADAPack 3xx, SCADAPack 32 and 32P Controller

An application program, written in Telepace Ladder Logic or Telepace C Tools and IEC 61131-3 or IEC 61131-3 C Tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

Telepace Ladder Logic

1. When a SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P controllers are used the store and forward translation table is configured using an Element Configuration dialog. From the Controller menu select the Store and Forward command to access the element configuration. Refer to the Telepace Ladder Logic Program Reference Manual for complete information on using the Store and Forward element configuration.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when Telepace programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

Telepace C Tools

TheSCADAPack 32 C++ language application program interface provides the following functions. Refer to the SCADAPack 32 C++ Tools Reference and User Manual for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

IEC 61131-3

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The **SandFEnabled** input must be set to TRUE.

2. Configure the translation table by using the setsf function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when IEC 61131-3 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

IEC 61131-3 C Tools

- The IEC 61131-3 C language application program interface provides the following functions. Refer to the IEC 61131-3 C Tools Reference and User Manual for details.
- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

Related Topics

Store and Forward Configuration

Point-to-Point Protocol (PPP)

SCADAPack 32 and SCADAPack 32P controllers support Point-to-Point Protocol (PPP) on the serial ports. Any serial port may be configured for the PPP protocol. Once a PPP connection is established the serial port has access to all IP protocol servers enabled on the controller.

A serial port configured for PPP supports an auto answer mode when dialed up through a modem. After answering the modem the serial port performs the login steps according to the authentication option selected for the port.

PPP provides two authentication protocols, which automates logins - PAP (Password Authentication Protocol) and CHAP (Challenge-Handshake Authentication Protocol).

PPP settings are configurable for each serial port on the SCADAPack 32 or SCADAPack 32P controller.

An inactivity timeout closes the PPP connection and hangs up the modem when the connection becomes idle. The timeout may also be disabled. Timeout range is 1 to 65535 minutes (~1092 hour s maximum).

When the PPP protocol is selected for a serial port, the serial port must be assigned a unique IP address, different from the IP address assigned to Ethernet or any other active PPP connection.

The remote end of a PPP connection may request an IP address from the controller PPP Server. The PPP Server will provide this IP address if requested.

Only one default gateway may be assigned to the controller. A PPP connection may be configured as the gateway.

PPP Client Setup in Windows 2000

This section describes the procedure for setting up a PPP client from a Windows 2000 PC. Client setup for a dialup PPP connection and a direct serial PPP connection are presented.

Direct Serial PPP Connection using Windows 2000

Use this connection when an only serial cable is used to establish a PPP connection between a Windows 2000 PC and a SCADAPack 32, without a dialup modem.

Connection Setup

1. From the **Start** menu, right click **Network** and **Dial-up Connections** from the **Settings** group, and select **Open**. The **Network and Dial-up Connections** dialog is displayed.
2. Double click the item **Make New Connection** from the **Network and Dial-up Connections** dialog. The connection wizard dialog is displayed.
3. Select the **Next** button to display the connection type options dialog.
4. For Network Connection Type, select the type **Connect directly to another computer** and select the **Next** button. The Host or Guest options dialog is displayed.
5. Select the **Guest** option and the **Next** button. The **Select a Device** dialog opens.
6. From the menu select the serial port on your PC that will be used to connect to the SCADAPack 32. Select the **Next** button. The **Connection Availability** dialog opens.
7. Select either option and then select the **Next** button. The **Connection Name** opens.
8. Enter a name for the connection and select the **Finish** button. The username and password prompt is displayed.
9. Select the **Cancel** button. The **Network and Dial-up Connections** dialog reopens.
10. Right click your new **Direct Connection** icon from the **Network and Dial-up Connections** dialog and select **Properties** from the list. The Properties dialog opens.
11. Select the **Configure** button from the General page. The **Modem Configuration** dialog opens.
12. There is no modem in this direct connection is uncheck all items, including heardware flow control. Select the baud rate to use, e.g., 9000 bytes/s. Select **OK** to return to the **Properties** dialog opens.
13. From the **Properties** dialog select the **Networking** page.
14. Uncheck all components except the component **Internet Protocol (TCP/IP).** Select the component **Internet Protocol (TCP/IP)** and select the **Properties.** The **Internet Protocol (TCP/IP) Properties** dialog opens.
15. The SCADAPack 32 does not have a DHCP server to automatically provide an IP address. Instead the PC's serial port must be given a fixed IP address to use for PPP connections. Select the option Use the following IP address. Enter an IP address to assign to your PC's serial port. Obtain this IP address from your Network Administrator. Then select OK to return to the Properties dialog.
16. Select **OK** to close the dialog.

Making a PPP Connection to the SCADAPack 32

1. From the **Start** menu, right click **Network** and **Dial-up Connections** from the **Settings** group, and select **Open**. The **Network and Dial-up Connections** dialog is displayed.
2. Double click the item **Make New Connection** from the **Network and Dial-up Connections** dialog.
3. Right click your new **Direct Connection** icon that you created in the Connection Setup section.
4. Select **Connect** from the list. The **Connect Direct Connection** opens.
5. Enter a valid **PAP** Valid user names and passwords are configured on the PPP Login page of the Controller IP confuration dialog and must be downloaded to the SCADAPack 32. or CHAP username and password in the appropriate fields.
6. Select **Connect.**
7. A progress message is displayed.
8. You may now connect to the IP address assigned to SCADAPack 32 PPP serial port using an appropriate application and a supported protocol (e.g. Modbus/TCP). In the example below, **Firmware Loader** is used to connect over PPP to the SCADAPack 32. From the PC Communication Settings dialog, the IP address assigned to the SCADAPack 32 PPP serial port is selected as the **Connect to Host.**

Disconnecting a PPP Connection

To disconnect a PPP connection made using the Windows PPP Client, do the following:

1. From the **Start** menu, double click **Network and Dial-up Connections** from the **Settings** group. The **Network and Dial-up Connections** dialog is displayed.
2. Your **Direct Connection** icon should display the word **Connected** in the **Status** column. To disconnect, right click your **Direct Connection** icon and select **Disconnect** from the list.

Dial-up PPP Connection using Windows 2000

Connection Setup using Windows 2000

Use this connection when a dial-up modem is used to establish a PPP connection between a Windows 2000 PC and a SCADAPack 32.

1. From the Start menu, right click Network and Dial-up Connections from the Settings group, and select Open. The Network and Dial-up Connections dialog is displayed.
2. Double click the item Make New Connection from the Network and Dial-up Connections dialog. The connection wizard dialog is displayed.
3. Select the Next button to display the connection type options dialog.

4. For Network Connection Type select the type Dial-up to private network and select the Next button. If there is more than one modem installed on the PC, the Select a Device dialog is displayed. If not, proceed to the next step.
5. From the menu select the modem installed on your PC that will be used to connect to the SCADAPack 32. Select the Next button. The Phone Number to Dial dialog is displayed.
6. Enter the phone number to dial (this can be changed later) and select the Next button. The Connection Availability dialog is displayed.
7. Select either option and then select the Next button. The Connection Name dialog is displayed.
8. Enter a name for the connection and select the Finish button. The username and password prompt is displayed.
9. Select the Cancel button. The Network and Dial-up Connections dialog should be visible again.
10. Right click your new Dial-up Connection icon from the Network and Dial-up Connections dialog and select Properties from the list. The Properties dialog is displayed.
11. From the Properties dialog select the Networking page.
12. Uncheck all components except the component Internet Protocol (TCP/IP). Select the component Internet Protocol (TCP/IP) and select the Properties button. The Internet Protocol (TCP/IP) Properties dialog is displayed.
13. The SCADAPack 32 does not have a DHCP server to automatically provide an IP address. Instead the PC's serial port must be given a fixed IP address to use for PPP connections. Select the option Use the following IP address. Enter an IP address to assign to your PC's serial port. Obtain this IP address from your Network Administrator. Then select OK to return to the Properties dialog.
14. Select OK again to close the dialog.

Makinga PPP Dial-up Connection to the SCADAPack 32 using Windows 2000

A connection can only be made after successfully setting up a Dial-up Connection icon as described in the section Connection Setup above. Also, a serial port on the SCADAPack 32 must already be configured for the PPP protocol using the Controller IP Configuration dialog and must be downloaded to the SCADAPack 32.

1. From the Start menu, right click Network and Dial-up Connections from the Settings group, and select Open. The Network and Dial-up Connections dialog is displayed.
2. Right click your Dial-up Connection icon that was setup in the previous section and select Connect from the list. A prompt for username and password is displayed.
3. Enter a valid PAP or CHAP username and password. Valid usernames and passwords are configured on the PPP Login page of the Controller IP Configuration dialog and must be downloaded to the SCADAPack 32. Then select the Dial button. If neither PAP nor CHAP is being used, ignore the prompt and just select the Dial button.
4. A progress message is displayed. If the connection is successful your Dial-up Connection icon should display the word Connected in the Status column.
5. You may now connect to the IP address assigned to SCADAPack 32 PPP serial port using an appropriate application and a supported protocol (e.g. Modbus/TCP). In the example below, Firmware Loader is used to connect over PPP to the SCADAPack 32. From the PC Communication Settings dialog, the IP address assigned to the SCADAPack 32 PPP serial port is selected as the Connect to Host.

Disconnecting a PPP Connection using Windows 2000

To disconnect a PPP connection made using the Windows PPP Client, do the following:

1. From the **Start** menu, right click **Network and Dial-up Connections** from the ***Settings*** group, and select **Open**. The *Network and Dial-up Connections* dialog is displayed.
2. Your **Dial-up Connection** icon should display the word Connected in the Status column. To disconnect, right click your **Dial-up Connection** icon and select **Disconnect** from the list.

Telepace DF1 Protocols Overview

Telebus Protocol communication protocols provide a standard communication interface to the controller. The protocols operate on a wide variety of serial data links. These include RS-232 A standard for serial binary data signals connecting a data terminal equiment (DTE) and a data circuit-terminating equipment (DCE). serial ports, RS-485 serial ports, radios, leased line modems, and dial up modems. The protocols are generally independent of the communication parameters of the link, with a few exceptions.

Telebus Protocol protocol commands may be directed to a specific device, identified by its station number, or broadcast to all devices. You can connect up to 255 devices to one communication network.

The Telebus Protocol protocols provide full access to the I/O database in the controller. The I/O database contains user-assigned registers and general purpose registers. Assigned registers map directly to the I/O hardware or system parameter in the controller. General purpose registers can be used by ladder logic and C application programs to store processed information, and to receive information from a remote device.

Application programs can initiate communication with remote devices. A multiple port controller can be a data concentrator for remote devices, by polling remote devices on one port and responding as a slave on another port.

The protocol type, communication parameters and station address are configured separately for each serial port on a controller. One controller can appear as different stations on different communication networks. The port configuration can be set from an application program, from the Telepace programming software, or from another Modbus or DF1 compatible device.

Related Topics

Serial Port Configuration

Communication Parameters

I/O Database

Slave Mode

Master Mode

Sending Messages

Compatibility

Description

Compatibility refers to communication only. The protocol defines communication aspects such as commands, syntax, message framing, error handling and addressing. The controllers do not mimic the internal functioning of any programmable controller. Device specific functions that relate to the hardware or programming of a specific programmable controller are not implemented.

Related Topics

Telebus Protocol DF1 Protocols Overview

TeleBUS DF1 Protocols

Serial Port Configuration

Communication Parameters

TeleBUS DF1 Protocols

There are four Telebus Protocol DF1 protocols:

- The **Telebus Protocol DF1 Full-Duplex BCC** protocol is compatible with the DF1 Full-Duplex data link layer protocol with block check character (BCC) error checking.
- The **Telebus Protocol DF1 Full-Duplex CRC** protocol is compatible with the DF1 Full-Duplex data link layer protocol with 16-bit cyclic redundancy check (CRC-16) error checking.
- The **Telebus Protocol DF1 Half-Duplex BCC** protocol is compatible with the DF1 Half-Duplex data link layer protocol with block check character (BCC) error checking.
- The **Telebus Protocol DF1 Half-Duplex CRC** protocol is compatible with the DF1 Half-Duplex data link layer protocol with 16-bit cyclic redundancy check (CRC-16) error checking.

Related Topics

Telebus Protocol DF1 Protocols Overview

Compatibility

Serial Port Configuration

Communication Parameters

Serial Port Configuration

Description

The TeleBUS DF1 protocols are, in general, independent of the serial communication parameters. The baud rate and parity may be chosen to suit the host computer and the characteristics of the data link.

The port configuration can be set in four ways:

- using the Telepace program;
- using the set_port function from a C application program;
- writing to the I/O database from a C or ladder logic application program; or
- writing to the I/O database remotely from a Modbus or DF1 compatible device.

To configure a serial port through the I/O database, add the module and CNFG serial port settings to the Register Assignment.

Related Topics

Telebus Protocol DF1 Protocols Overview

TeleBUS DF1 Protocols

Communication Parameters

I/O Database

Configuring a Serial Port

Description

The port configuration can be set in four ways:

- using the Telepace program;
- using the set_port function from a C application program;
- writing to the I/O database from a C or ladder logic application program; or
- writing to the I/O database remotely from a Modbus or DF1 compatible device.

To configure a serial port through the I/O database, add the module, CNFG Serial port settings, to the Register Assignment.

Related Topics

Serial Port Configuration

Protocol Parameters

Default Parameters

Communication Parameters

Protocol Parameters

Baud Rate

Duplex Setting

Protocol Parameters

Description

The Telebus Protocol DF1 protocols operate independently on each serial port. Each port may set the protocol type, station number, protocol task priority and store-and-forward messaging options.

Protocol Type

The protocol type may be set to the DF1 or Modbus protocols, or it may be disabled. When the protocol is disabled, the port functions as a normal serial port.

Station Number

The Telebus Protocol DF1 protocols allow up to 255 devices on a network. Station numbers identify each device. A device responds to commands addressed to it, or to commands broadcast to all stations.

The station number is in the range 0 to 254. Address 255 indicates a command broadcast to all stations, and cannot be used as a station number. Each serial port may have a unique station number.

Task Priority

A task is responsible for monitoring each serial port for messages. The real time operating system (RTOS) schedules the tasks with the application program tasks according to the task priority. The priority can be changed only with the **set_protocol** function from a C application program.

The default task priority is 3. Changing the priority is not recommended.

Store and Forward Messaging

Store and forward messaging is not supported by theTelebus Protocol DF1 protocols.

Related Topics

Configuring a Serial Port

Default Parameters

Sending Messages

Default Parameters

Description

All ports are configured at reset with default parameters when the controller is started in SERVICE mode. The ports use user-defined parameters when the controller is reset in the RUN mode. The default parameters are listed below.

Parameter	com1	com2	com3	com4
Baud rate	9600	9600	9600	9600
Parity	none	none	none	none
Data bits	8	8	8	8
Stop bits	1	1	1	1
Duplex	full	full	half	half
Protocol	Modbus RTU	Modbus RTU	Modbus RTU	Modbus RTU
Rx flow control	none	none	Rx disabled	Rx disabled
Tx flow control	none	none	none	none
Serial time out	60 s	60 s	60 s	60 s
Type	RS-232	RS-232	RS-232	RS-232
Minimum protected DF1 address	0	0	0	0
Maximum protected DF1 address	11534	11534	11534	11534

Related Topics

Note 1:	com3 is supported only when the SCADAPack lower I/O module is installed. com4 is supported only when the SCADAPack upper I/O module is installed.

Note 2:	To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator display terminals, and for programming and diagnostics using the Telepace program.

Related Topics

Configuring a Serial Port

Protocol Parameters

Sending Messages

Communication Parameters

The TeleBUS DF1 protocols are, in general, independent of the serial communication parameters. The baud rate and parity may be chosen to suit the host computer and the characteristics of the data link.

Related Topics

Protocol Parameters

Baud Rate

Duplex Setting

Protocol Parameters

The TeleBUS DF1 protocols are eight bit character-oriented protocols. The table below shows possible and recommended communication parameters.

Parameter	Possible Settings	Recommended Setting
Baud rate	see Baud Rate	see Baud Rate
Data bits	8	8
Parity	none even odd	none
Stop bits	1 stop bit 2 stop bits[1]	1 stop bit
Flow control	disabled	disabled
Duplex	see Duplex mode	see Duplex mode

[1]Not applicable on the SCADAPack 314, SCADAPack 33x or SCADAPack 350.

Related Topics

Baud Rate

Description

The baud rate sets the communication speed. The possible settings are determined by the type of serial data link used. The table below shows the possible settings for the controller. Note that not all port types and baud rates are available on all controller ports.

Port Type	Possible baud Settings	Recommended Setting
RS-232 or RS-232 Dial-up modem	75 110 150 300 600 1200 2400 4800 9600 19200 38400 57600 115200	Use the highest baud setting supported by all devices on your network.
RS-485	75 110 150 300 600 1200 2400 4800 9600 19200 38400	Use the highest baud setting supported by all devices on your network.

Related Topics

Duplex Setting

Description

TheTelebus Protocol DF1 protocols communicate in one direction at a time. However the duplex setting is determined by the type of serial data link used. The table below shows the possible settings for the controller. Note that not all port types are available on all controllers.

Port Type	Possible baud Settings	Recommended Setting
RS-232 or RS-232 Dial-up modem	half duplex	Use full duplex wherever possible. Use half duplex for most external modems.

	full duplex	
RS-485	half duplex	Slave stations always use half duplex.
	full duplex	Master stations can use full duplex only on 4-wire systems.

Related Topics

I/O Database

The Telebus Protocol protocols read and write information from the I/O database. The I/O database is an area of memory that can be accessed by C programs, Ladder Logic programs,IEC 61131-3 programs and communication protocols. The I/O database allows data to be shared between these programs. The I/O database contains user-assigned registers and general purpose registers.

User-assigned registers map directly to the I/O hardware or system parameter in the controller. Assigned registers are initialized to the default hardware state or system parameter when the controller is reset. Assigned output registers do not maintain their values during power failures. However, output registers do retain their values during application program loading.

General purpose registers are used by ladder logic,IEC 61131-3 and C application programs to store processed information, and to receive information from remote devices. General purpose registers retain their values during power failures and application program loading. The values change only when written by an application program or a communication protocol.

Related Topics

Serial Port Configuration

Configuring a Serial Port

Coil and Status Registers

Input and Holding Registers

Accessing the I/O Database Using Telepace

Modbus Addressing

Accessing the I/O Database Using IEC 61131-3

DF1 Addressing

Converting Modbus to DF1 Addresses

Coil and Status Registers

Coil and status registers contain one bit of information, that is whether a signal is off or on.

For Telepace firmware coil registers are single bits, which the protocols can read and write. There are 4096 coil registers located in the digital output section of the I/O database. Coil registers are contained within the DF1 16-bit addresses 0 to 255.

For Telepace firmware status registers are single bits, which the protocols can read. There are 4096 status registers located in the digital input section of the I/O database. Status registers are contained within the DF1 16-bit addresses 256 to 511.

ForIEC 61131-3 firmware coil registers are single bits, which the protocols can read and write. There are 9999 coil registers located in the digital output section of the I/O database. Coil registers are contained within the DF1 16-bit addresses 0 to 624.

For IEC 61131-3 firmware status registers are single bits, which the protocols can read. There are 9999 status registers located in the digital input section of the I/O database. Status registers are contained within the DF1 16-bit addresses 625 to 1249.

Coil and status registers are accessed 16 at a time or individually (in some commands) using a bitmask. Writing a one to a bit within the 16-bit address turns the bit on. Writing zero to the bit turns the bit off. If the register is assigned to an I/O module, the bit status is written to the module output hardware or parameter.

Reading a coil or status register returns 1 if the bit is on, or 0 if the bit is off. The stored value is returned from general purpose registers. The I/O module point status is returned from assigned registers.

Related Topics

I/O Database

Input and Holding Registers

Accessing the I/O Database Using Telepace

Input and Holding Registers

Input and holding registers contain 16-bit values.

Input registers are 16-bit registers, which the protocol can read. For Telepace firmware, there are 1024 input registers located in the analog input section of the I/O database. Input registers are contained within the DF1 addresses 512 to 1535.

For IEC 61131-3 firmware there are 9999 input registers located in the analog input section of the I/O database. Input registers are contained within the DF1 addresses 1250 to 11247.

Holding registers are 16-bit registers that the protocol can read and write. For Telepace firmware there are 9999 holding registers located in the analog output section of the I/O database. Holding registers are contained within the DF1 addresses 1536 to 11534.

For IEC 61131-3 firmware there are 9999 holding registers located in the analog output section of the I/O database. Holding registers are contained within the DF1 addresses 11248 to 21247.

Writing any value to a general purpose register stores the value in the register. Writing a value to an assigned register, writes the value to the assigned I/O module.

Reading a general purpose register returns the value stored in the register. Reading an assigned register returns the value read from the I/O module.

Related Topics

I/O Database

Coil and Status Registers

Accessing the I/O Database Using Telepace

Accessing the I/O Database Using Telepace

Ladder logic programs access the I/O database through function blocks. All function blocks can access the I/O database. The function blocks in ladder logic use only the Modbus addressing scheme. Refer to the Telepace Ladder Logic Reference and User Manual for details.

C language programs access the I/O database with two functions. The dbase function reads a value from the I/O database. The setdbase function writes a value to the I/O database. These functions use either Modbus or DF1 physical addressing. Refer to the Telepace C Tools Reference and User Manual for full details on these functions.

Related Topics

I/O Database

Coil and Status Registers

Input and Holding Registers

Modbus Addressing

Modbus addressing is used in all ladder logic program functions. The controller's Register Assignment is also configured using Modbus addresses. The C functions **dbase** and **setdbase** support Modbus addressing. When the specified port is configured for one of the Modbus protocols, the function **master_message** uses Modbus addressing.The range of Modbus registers available depends on the controller being used. The following sections list the Modbus registers available for each controller type.

SCADAPack, SCADAPack 100: 1024K and SCADAPack 4202

Coil registers are single bit addresses ranging from 00001 to 04096.

Status registers are single bit addresses ranging from 10001 to 14096.

Input registers are 16-bit addresses in the range 30001 to 31024.

Holding registers are 16-bit addresses in the range 40001 to 49999.

SCADAPack 100: 256K

Coil registers are single bit addresses ranging from 00001 to 04096.

Status registers are single bit addresses ranging from 10001 to 14096.

Input registers are 16-bit addresses in the range 30001 to 30512.

Holding registers are 16-bit addresses in the range 40001 to 44000.

SCADAPack 3xx, SCADAPack 32 and SCADAPack 4203

Coil registers are single bit addresses ranging from 00001 to 04096.

Status registers are single bit addresses ranging from 10001 to 14096.

Input registers are 16-bit addresses in the range 30001 to 39999.

Holding registers are 16-bit addresses in the range 40001 to 49999.

DF1 Addressing

DF1 addressing is used by the **MSTR** ladder logic function when the specified port is configured for one of the DF1 protocols. Modbus addressing must be used in all other ladder logic program functions.

DF1 addressing is used by function **master_message** when the specified port is configured for one of the DF1 protocols. The functions **dbase**, **setdbase**, **setABConfiguration** and **getABConfiguration** also support DF1 addressing.

All DF1 addresses are absolute word addresses beginning with the first 16-bit register in the I/O database at address 0.

Converting Modbus to DF1 Addresses

I/O database registers are assigned within the controller's Register Assignment using Modbus addressing. When polling the controller from an DF1 device it is necessary to know the DF1 address corresponding to each assigned register.

In general, the cross-reference between Modbus and DF1 addressing is shown in the following table. DF1 addresses in this table are described in the format **word/bit** where **word** is the address of a 16-bit word and **bit** is the bit within that word.

Note:	The bit address is optional.

Address	Range	Description
Modbus	00001 to 09999	Digital output database (single bit registers)
DF1	0/0 to 624/15	
Modbus	10001 to 19999	Digital output database (single bit registers)
DF1	625/0 TO 1249/15	
Modbus	30001 to 39999	Analog input database (16-bit registers)
DF1	1250 to 11248	
Modbus	40001 to 49999	Analog input database (16-bit registers)
DF1	11249 to 21247	

NOTE:
The SLC-503 processor has a DF1 address space of 255 16-bit words for file integer N9. The N9 integer file is where the data is retrieved from or saved to in the SLC-503 processor during DF1 communication. When the SLC-503 is being used as the DF1 master when communicating with the SCADAPack controller it can only access

the Digital Output range in the SCADAPack controller.
Ladder Logic will need to be written in the SCADAPack controller to put data into this range when it is being read by the SLC-503 processor. The MOVE and OVER elements are useful for this logic. When getting data from this range after it has written to by the SLC-503 the MOVE and OVER elements can be used.

SCADAPack, SCADA Pack 100: 10024K and SCADAPack 4202

In general, the cross-reference between Modbus and DF1 addressing is shown in the following table. DF1 addresses in this table are described in the format **word/bit** where **word** is the address of a 16-bit word and bit is the **bit** within that word. The bit address is optional.

Address	Range	Description
Modbus DF1	00001 to 04096 0/0 to 255/15	Digital output database (single bit registers)
Modbus DF1	10001 to 14096 256/0 to 511/15	Digital input database (single bit registers)
Modbus DF1	30001 to 31024 512 to 1535	Analog input database (16-bit registers)
Modbus DF1	40001 to 49999 1536 to 11534	Analog output database (16-bit registers)

The SLC-503 processor has a DF1 address space of 255 16-bit words for file integer N9. The N9 integer file is where the data is retrieved from or saved to in the SLC-503 processor during DF1 communication. When the SLC-503 is being used as the DF1 master when communicating with the SCADAPack controller it can only access the Digital Output range in the SCADAPack controller.
Ladder Logic will need to be written in the SCADAPack controller to put data into this range when it is being read by the SLC-503 processor. The MOVE and OVER elements are useful for this logic. When getting data from this range after it has written to by the SLC-503 the MOVE and OVER elements can be used.

Coil registers

To convert a Modbus coil register, ModbusCoil, to a DF1 address word/bit:

word = (ModbusCoil - 00001) / 16

bit = remainder of { (ModbusCoil - 00001) / 16 }

Status registers

To convert a Modbus status register, ModbusStatus, to a DF1 address word/bit:

word = (ModbusStatus - 10001) / 16 + 256

bit = remainder of { (ModbusStatus - 10001) / 16 }

Input registers

To convert a Modbus input register, ModbusInput, to a DF1 address word:

word = (ModbusInput - 30001) + 512

Holding registers

To convert a Modbus holding register, ModbusHolding, to a DF1 address word:

word = (ModbusHolding - 40001) + 1536

Example

Description

In this example the equivalent DF1 addresses are shown next to a sample SCADAPack Register Assignment specified with Modbus addresses.

In this example the equivalent DF1 addresses are shown next to a sample SCADAPack Register Assignment specified with Modbus addresses.

Module	Start Register	End Register	DF1 Address Range
SCADAPack 5601 I/O module			
digital outputs	00001	00012	0/0 to 0/11
digital inputs	10001	10016	256/0 to 256/15
analog inputs	30001	30008	512 to 519
DIN Controller digital inputs	10017	10019	257/0 to 257/2
DIAG Force LED	10020	10020	257/3
SCADAPack AOUT module	40001	40002	1536 to 1537

SCADAPack 3xx and SCADAPack 32 Controllers

In general, the cross-reference between Modbus and DF1 addressing is shown in the following table. DF1 addresses in this table are described in the format **word/bit** where **word** is the address of a 16-bit word and **bit** is the bit within that word. The bit address is optional.

Address	Range	Description
Modbus DF1	00001 to 04096 0/0 to 255/15	Digital output database (single bit registers)

Modbus **DF1**	10001 to 14096 256/0 to 511/15	Digital input database (single bit registers)
Modbus **DF1**	30001 to 39999 512 to 10510	Analog input database (16-bit registers)
Modbus **DF1**	40001 to 49999 10511 to 20511	Analog output database (16-bit registers)

NOTE:
The SLC-503 processor has a DF1 address space of 255 16-bit words for file integer N9. The N9 integer file is where the data is retrieved from or saved to in the SLC-503 processor during DF1 communication. When the SLC-503 is being used as the DF1 master when communicating with the SCADAPack controller it can only access the Digital Output range in the SCADAPack controller.
Ladder Logic will need to be written in the SCADAPack controller to put data into this range when it is being read by the SLC-503 processor. The MOVE and OVER elements are useful for this logic. When getting data from this range after it has written to by the SLC-503 the MOVE and OVER elements can be used.

Coil registers

To convert a Modbus coil register, ModbusCoil, to a DF1 address word/bit:

word = (ModbusCoil - 00001) / 16

bit = remainder of { (ModbusCoil - 00001) / 16 }

Status registers

To convert a Modbus status register, ModbusStatus, to a DF1 address word/bit:

word = (ModbusStatus - 10001) / 16 + 256

bit = remainder of { (ModbusStatus - 10001) / 16 }

Input registers

To convert a Modbus input register, ModbusInput, to a DF1 address word:

word = (ModbusInput - 30001) + 512

Holding registers

To convert a Modbus holding register, ModbusHolding, to a DF1 address word:

word = (ModbusHolding - 40001) + 1536

Example

In this example the equivalent DF1 addresses are shown next to a sample SCADAPack Register Assignment specified with Modbus addresses.

Module	Start Register	End Register	DF1 Address Range
SCADAPack 5601 I/O module			
digital outputs	00001	00012	0/0 to 0/11
digital inputs	10001	10016	256/0 to 256/15
analog inputs	30001	30008	512 to 519
DIN Controller digital inputs	10017	10019	257/0 to 257/2
DIAG Force LED	10020	10020	257/3
SCADAPack AOUT module	40001	40002	1536 to 1537

Accessing the I/O Database Using IEC 61131-3

IEC 61131-3 programs access the I/O database through function blocks. The function blocks inIEC 61131-3 may use the Modbus addressing scheme when a Network Address is defined for a variable. Refer to the IEC 61131 User Manual for details.

C language programs access the I/O database with two functions. The **dbase** function reads a value from the I/O database. The **setdbase** function writes a value to the I/O database. These functions use either Modbus or DF1 physical addressing. Refer to the IEC 61131 User Manual for full details on these functions.

Modbus Addressing (COPY)

Description

Modbus addressing can be used inIEC 61131-3 program functions.

The C functions **dbase** and **setdbase** support Modbus addressing. When the specified port is configured for one of the Modbus protocols, the function **master_message** uses Modbus addressing.

Modbus addresses coil registers with a single bit address ranging from 00001 to 09999. Status registers are also addressed with single bit addresses ranging from 10001 to 19999. Input registers are addressed with 16-bit addresses in the range 30001 to 39999. And holding registers are addressed with a 16-bit address in the range 40001 to 49999.

DF1 Addressing

Description

DF1 addressing is used by the **master** function when the specified port is configured for one of the DF1 protocols. Modbus addressing must be used in all other ladder

logic program functions.

DF1 addressing is used by function **master_message** when the specified port is configured for one of the DF1 protocols. The functions **dbase, setdbase, setABConfiguration** and **getABConfiguration** also support DF1 addressing.

All DF1 addresses are absolute word addresses beginning with the first 16-bit register in the I/O database at address 0 and ending at address 21247.

Converting Modbus to DF1 Addresses

I/O database registers are assigned within the controller's Register Assignment using Modbus addressing. When polling the controller from an DF1 device it is necessary to know the DF1 address corresponding to each assigned register.

In general, the cross-reference between Modbus and DF1 addressing is shown in the following table. DF1 addresses in this table are described in the format **word/bit** where **word** is the address of a 16-bit word and **bit** is the bit within that word.

Note:	The bit address is optional.

Address	Range	Description
Modbus	00001 to 09999	Digital output database (single bit registers)
DF1	0/0 to 624/15	
Modbus	10001 to 19999	Digital output database (single bit registers)
DF1	625/0 TO 1249/15	
Modbus	30001 to 39999	Analog input database (16-bit registers)
DF1	1250 to 11248	
Modbus	40001 to 49999	Analog input database (16-bit registers)
DF1	11249 to 21247	

NOTE:
The SLC-503 processor has a DF1 address space of 255 16-bit words for file integer N9. The N9 integer file is where the data is retrieved from or saved to in the SLC-503 processor during DF1 communication. When the SLC-503 is being used as the DF1 master when communicating with the SCADAPack controller it can only access the Digital Output range in the SCADAPack controller.
Ladder Logic will need to be written in the SCADAPack controller to put data into this range when it is being read by the SLC-503 processor. The MOVE and OVER elements are useful for this logic. When getting data from this range after it has written to by the SLC-503 the MOVE and OVER elements can be used.

Coil Registers

To convert a Modbus coil register, ModbusCoil, to a DF1 address word/bit:

word = (ModbusCoil - 00001) / 16

bit = remainder of { (ModbusCoil - 00001) / 16 }

SCADAPack 3xx and SCADAPack 32 Controllers

To convert a Modbus status register, ModbusStatus, to a DF1 address word/bit:

word = (ModbusStatus - 10001) / 16 + 256

bit = remainder of { (ModbusStatus - 10001) / 16 }

Input Registers

Description

To convert a Modbus input register, *ModbusInput*, to a DF1 address word:

word = (*ModbusInput* - 30001) +1250

Holding Registers

Description

To convert a Modbus holding register, ModbusHolding, to a DF1 address word:

word = (*ModbusHolding* - 40001) + 11249

Example

Description

In this example the equivalent DF1 addresses are shown next to a sample SCADAPack Register Assignment specified with Modbus addresses.

In this example the equivalent DF1 addresses are shown next to a sample SCADAPack Register Assignment specified with Modbus addresses.

Module	Start Register	End Register	DF1 Address Range
SCADAPack 5601 I/O module			
digital outputs	00001	00012	0/0 to 0/11
digital inputs	10001	10016	256/0 to 256/15
analog inputs	30001	30008	512 to 519
DIN Controller digital inputs	10017	10019	257/0 to 257/2
DIAG Force LED	10020	10020	257/3

SCADAPack AOUT module	40001	40002	1536 to 1537

Slave Mode

Description

TheTelebus Protocol DF1 protocols operate in slave and master modes simultaneously. In slave mode the controller responds to commands sent by another device. Commands may be sent to a specific device or broadcast to all devices.

TheTelebus Protocol DF1 protocols provide the protocol functions required for communication with a host device which uses the Non-Privileged commands from the DF1 Basic Command Set. These functions are described below.

Consult the DF1 I/O driver documentation included with the SCADA package, or specific DF1 documentation, for details on these commands. In most cases a knowledge of the actual commands is not required to set up the host system.

Related Topics

Broadcast Messages

Function Codes

Broadcast Messages

Description

A broadcast message is sent to all devices on a network. Each device executes the command. No device responds to a broadcast command. The device sending the command must query each device to determine if the command was received and processed. Broadcast messages are supported for function codes that write information.

A broadcast message is sent to station number 255.

Function Codes

Description

The table summarizes the implemented function codes. Note that slave commands at the protocol layer access the I/O database in physical byte addresses. However, in master mode the interface to the TeleBUS DF1 protocol accesses the I/O database in physical 16-bit word addresses. The interface function block **MSTR** and C function **master_message** are described in the Master Mode section below.

The maximum number of 16-bit words that can be read or written with one slave command message is shown in the maximum column.

Function	Name	Description	Maximum
00	Protected Write	Writes words of data to limited areas of the database.	121
01	Unprotected Read	Reads words of data from any area of the database.	122
02	Protected Bit Write	Sets or resets individual bits within limited areas of the database.	30
05	Unprotected Bit Write	Sets or resets individual bits in any area of the database.	30
08	Unprotected Write	Writes words of data to any area of the database.	12

Functions 0, 2,5 and 8 support broadcast messages. The functions are described in the following sections:

Protected Write

Unprotected Read

Protected Bit Write

Unprotected Bit Write

Unprotected Write

Protected Write

Description

The **Protected Write** function writes 8-bit values into limited areas of the I/O database. Access to the I/O database is limited to the protected address range.

Any number of bytes may be written up to the maximum number. The write may start at any byte address, provided the entire block is within the protected address range.

The protected address range is set using the **setABConfiguration** function from a C application program.

Unprotected Read

Description

The **Unprotected Read** function reads 8-bit values from any area of the I/O database. Access to the I/O database is limited to the unprotected address range.

Any number of bytes may be read up to the maximum number. The read may start at any byte address, provided the entire block is within the unprotected address range.

Protected Bit Write

Description

The **Protected Bit Write** function sets or resets individual bits within limited areas of the I/O database. Access to the I/O database is limited to the protected address range.

Bits are accessed one byte at a time and may be written up to the maximum number of bytes. The write may start at any byte address, provided the entire block is within the protected address range.

Unprotected Bit Write

Description

The **Unprotected Bit Write** function sets or resets individual bits in any area of the I/O database. Access to the I/O database is limited to the unprotected address range.

Bits are accessed one byte at a time and may be written up to the maximum number of bytes. The write may start at any byte address, provided the entire block is within the unprotected address range.

Unprotected Write

Description

The **Unprotected Write** function writes 8-bit values into any area of the I/O database. Access to the I/O database is limited to the unprotected address range.

Any number of bytes may be written up to the maximum number. The write may start at any byte address, provided the entire block is within the unprotected address range.

Master Mode

Description

The Telebus Protocol DF1 protocols may act as a communication master on any serial port. In master mode, the controller sends commands to other devices on the network. Simultaneous master messages may be active on all ports.

Related Topics

Function Codes

Sending Messages

Function Codes

Description

The table shows the implemented function codes. Note that slave commands at the protocol layer access the I/O database in physical byte addresses. However, in master mode the interface to the TeleBUS DF1 protocol accesses the I/O database in physical 16-bit word registers. The interface function block **MSTR** and C function **master_message** are described in Sending Messages.

The maximum number of 16-bit registers that can be read or written with one message is shown in the maximum column. The slave device may support fewer registers than shown; consult the manual for the device for details.

Function	Name	Description	Maximum
00	Protected Write	Writes words of data to limited areas of the database.	121
01	Unprotected Read	Reads words of data from any area of the database.	122
02	Protected Bit Write	Sets or resets individual bits within limited areas of the database.	1
05	Unprotected Bit Write	Sets or resets individual bits in any area of the database.	1
08	Unprotected Write	Writes words of data to any area of the database.	121

Related Topics

Sending Messages

Protected Write

Description

The **Protected Write** function writes 16-bit values into the I/O database of the slave device. Access to the I/O database is limited to the protected address range of the slave device. The data may come from any area of the master I/O database within its unprotected address range.

Any number of 16-bit registers may be written up to the maximum number supported by the slave device or the maximum number above, which ever is less. The write may start at any 16-bit register, provided the entire block is within the protected address range of the slave device.

Related Topics

Master Mode

Function Codes

Unprotected Read

Protected Bit Write

Unprotected Bit Write

Unprotected Write

Unprotected Read

Description

The **Unprotected Read** function reads 16-bit values from any area of the I/O database of the slave device. Access to the I/O database is limited to the unprotected address range of the slave device. Data can be written into any area of the master I/O database within its unprotected address range.

Any number of 16-bit registers may be read up to the maximum number supported by the slave device or the maximum number above, which ever is less. The read may start at any 16-bit register, provided the entire block is within the unprotected address range of the slave device.

Related Topics

Master Mode

Function Codes

Protected Write

Protected Bit Write

Unprotected Bit Write

Unprotected Write

Protected Bit Write

Description

The **Protected Bit Write** function sets or resets individual bits in the I/O database of the slave device. Access to the I/O database is limited to the protected address range of the slave device. The data may come from any area of the master I/O database within the unprotected address range.

One 16-bit register with a bitmask is used to write up to 16 bits of data. The register must be within the protected address range of the slave device.

Related Topics

Master Mode

Function Codes

Protected Write

Unprotected Read

Unprotected Bit Write

Unprotected Write

Unprotected Bit Write

Description

The **Unprotected Bit Write** function sets or resets individual bits in any area of the I/O database of the slave device. Access to the I/O database is limited to the unprotected address range of the slave device. The data may come from any area of the master I/O database within its unprotected address range.

One 16-bit register with a bitmask is used to write up to 16 bits of data. The register must be within the unprotected address range of the slave device.

Related Topics

Master Mode

Function Codes

Protected Write

Unprotected Read

Protected Bit Write

Unprotected Write

Unprotected Write

Description

The **Unprotected Write** function writes 16-bit values into any area of the I/O database of the slave device. Access to the I/O database is limited to the unprotected address range of the slave device. The data may come from any area of the master I/O database within its unprotected address range.

Any number of 16-bit registers may be written up to the maximum number supported by the slave device or the maximum number above, which ever is less. The write may start at any 16-bit register, provided the entire block is within the unprotected address range of the slave device.

Related Topics

Master Mode

Function Codes

Protected Write

Unprotected Read

Protected Bit Write

Unprotected Bit Write

Sending Messages

Description

A master message is initiated in two ways:

- using the **master_message** function from a C application program; or
- using the **MSTR** function block from a ladder logic program.

These functions specify the port on which to issue the command, the function code, the slave station number, and the location and size of the data in the slave and master devices. The protocol driver, independent of the application program, receives the response to the command.

The application program detects the completion of the transaction by:

- calling the **get_protocol_status** function in a C application program; or
- using the output of the **MSTR** function block in a ladder logic program.

A communication error has occurred if the slave does not respond within the expected maximum time for the complete command and response. The application program is responsible for detecting this condition. When errors occur, it is recommended that the application program retry several times before indicating a communication failure.

The completion time depends on the length of the message, the length of the response, the number of transmitted bits per character, the transmission baud rate, and the maximum message turn-around time. One to three seconds is usually sufficient. Radio systems may require longer delays.

Related Topics

Master Mode

Function Codes

Polling DF1 PLCs

***

Polling DF1 PLCs

Description

All DF1 PLCs, except the PLC-5/VME, will support some portion of the basic commands implemented.

AB PLC	Unprotected Read	Unprotected Write	Protected Write	Protected Bit Write	Unprotected Bit Write
MicroLogix 1000[5]	√	√			
SLC500[1,3]	√	√			
SLC5/01[1,3]	√	√			
SLC5/02[1,2,3]	√	√			
SLC5/03[2,3]	√	√			
SLC5/04[2,3]	√	√			
1774-PLC	√	√	√	√	√
PLC-2	√	√	√	√	√
PLC-3[5]	√	√	√	√	√
PLC-5[5]	√	√	√	√	√
PLC-5/2505	√	√		√	√
PLC-5/VME					

Notes

[1] At the protocol level these commands convert and send the slave word address as a byte address. The SLC500, 5/01 (and 5/02, prior to Series C FRN 3) treat this byte address as if it were a word address in the SLC500. This means that the (desired word address)/2 must be specified for the Slave Register Address in MSTR or master_message. Note that this always results in an even starting word address in the slave SLC. (E.g. Slave word address of 7 specified in MSTR accesses SLC word address 14.)

[2] The SLC5/02, 5/03 and 5/04 have a status selection bit (S:2/8) which allows selection of either word or byte addressing when interpreting only these commands. (Setting SLC bit S:2/8 = 1 selects byte addressing so that, for example, a slave word address of 7 specified in MSTR accesses SLC word address 7.)

[3] These commands can access SLC memory only in the CIF or Common Interface File: N9. See further details in section 12-15 of the SLC500 Instruction Set manual.

[4] These commands can access MicroLogix memory only in the CIF or Common Interface File: N7. See further details in section 12-15 of the MicroLogix 1000 Instruction Set manual.

[5] These commands can access memory only in the CIF or “PLC-2 compatibility file”: N7. See further details in section 16-7 of the PLC-5 manual.

Related Topics

Master Mode

Function Codes

Sending Messages

DNP3 Reference

This section details implementation of the Distributed Network Protocol (DNP3) driver on SCADAPack controllers. While we continuously improve upon the contents of this section to simply the driver configuration tasks, we also assume that if you are attempting to configure the DNP3 on a SCADAPack controller, you have some preliminary understanding of DNP3.

DNP3 Overview

DNP, the Distributed Network Protocol, is a standards-based communications protocol developed to achieve interoperability among systems in the electric utility, oil & gas, water/waste water and security industries. This robust, flexible non-proprietary protocol is based on existing open standards to work within a variety of networks.

DNP offers flexibility and functionality that go far beyond conventional communications protocols. Among its robust and flexible features DNP 3.0 includes:

- Multiple data types (Data Objects) may be included in both request and response messages.
- Multiple master stations are supported for outstations.
- Unsolicited responses may be initiated from outstations to master stations.
- Data types (Objects) may be assigned priorities (Class) and be requested based on the priority.
- Addressing for over 65,000 devices on a single link.
- Time synchronization and time-stamped events.
- Broadcast messages.
- Data link and application layer confirmation
- Internal indications that report the health of a device and results of last request.
- Select-Before-Operate which is the ability to choose extra reliability when operating outputs.

DNP3 Architecture

DNP is a layered protocol that is based on the Open System Connection (OSI) 7-layer protocol. DNP supports the physical, data link and application layers only and terms this the Enhanced Performance Architecture (EPA). In addition to these three layers an additional layer, the pseudo-transport layer, is added to allow for larger application layer messages to be broken down into smaller frames for the data link layer to transmit.

Object Library	The data objects (binary inputs, binary outputs, and analog inputs) that reside in the master or outstations.
Application Layer	Application tasks for sending of solicited requests (master messages) to outstations or sending of unsolicited responses from outstations. These request and response messages are referred to as single- or multi-fragment messages in DNP3.
Pseudo-transport Layer	Breaks the application layer messages into smaller packets that can be handled by the data link layer. These packets are referred to as frames inDNP3.
Data Link Layer	Handles the transmission and reception of data frames across the physical layer.
Physical Layer	This is the physical communications media such as serial or Ethernet over which DNP3 communicates.

Object Library

The data types that are used in DNP3 are grouped together into object groups such as binary input objects and analog input objects. Individual data points, or objects within each group, which are further defined using object variations, for example, binary input change with time and 16-bit analog inputs..

There are two categories of data within each data type: static and event objects. Static objects contain the current value of the field point or software point. Event objects are generated as a result of the data changing.

In addition to the object group and variation data objects can be assigned to classes. In DNP3 there are four object classes:

- Class 0
- Class 1
- Class 2
- Class 3

Class 0 contains all static data. Classes 1, 2 and 3 provide a method to assign priority to event objects. While there is no fixed rule for assigning classes to data objects typically class 1 is assigned to the highest priority data and class 3 is assigned to the lowest priority data.

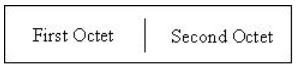
This object library structure enables the efficient transfer of data between master and outstations. The master can poll for high priority data (class 1) more often than it polls for low priority data (class 3). Since the data objects assigned to classes are event data, when the master polls for a class only the changed event data, is returned by the outstation. For data in an outstation that is not assigned a class, the master uses a class 0 poll to retrieve all static data from the outstation.

DNP allows outstations to report data to one or more master stations using unsolicited responses (report-by-exception) for event data objects. The outstation reports data based on the assigned class of the data. For example, the outstation can be configured to only report high priority class 1 data.

Internal Indications (IIN) Flags

An important data object is the Internal Indications (IIN) object. The Internal Indication (IIN) flags are set by a slave station to indicate internal states and diagnostic results. The following tables show the IIN flags supported by SCADAPack controllers. All bits except Device Restarted and Time Synchronization required are cleared when the slave station receives any poll or read data command.

The IIN is set as a 16 bit word divided into two octets of 8 bits. The order of the two octets is:



IIN First Octet

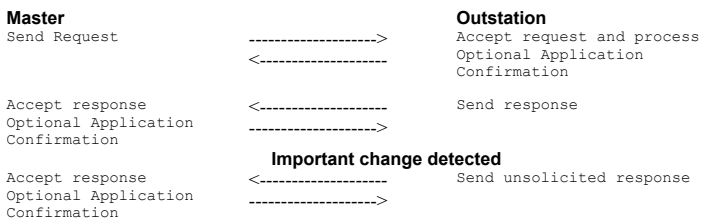
First Octet Bit	Description
0	last received message was a broadcast message
1	Class 1 data available
2	Class 2 data available
3	Class 3 data available
4	Time Synchronization required
5	not used (returns 0)
6	Device trouble • Indicates memory allocation error in the slave, or • For master in mimic mode indicates communication failure with the slave device.
7	Device restarted (set on a power cycle)

IIN Second Octet

Second Octet Bit	Description
0	Function Code not implemented
1	Requested object unknown or there were errors in the application data
2	Parameters out of range
3	Event buffer overflowed Indicates event buffer overflow in the slave or master. The slave will set this bit if the event buffer in the slave is overflowed. The master will set this bit if the event buffer in the master has overflowed with events read from the slave. Ensure the event buffer size, in the master and slave, is set to a value that will ensure the buffer does not overflow and events are lost.
4	not used (returns 0)
5	not used (returns 0)
6	not used (returns 0)
7	not used (returns 0)

Application Layer

The application layer in DNP3 processes complete messages for requesting or responding to requests for data. The following shows the sequence of application layer messages between one master and one outstation.



The complete messages are received from and passed to the pseudo-transport layer. Application layer messages are broken into fragments with each fragment size usually a maximum of 2048 bytes. An application layer message may be one or more fragments in size and it is the responsibility of the application layer to ensure the fragments are properly sequenced.

Application layer fragments are sent with or without a confirmation request. When a confirmation is requested the receiving device replies with a confirmation indicating the message was received and parsed without any errors.

Pseudo-transport Layer

The pseudo-transport layer formats the larger application layer messages into smaller packets that can be handled by the data link layer. These packets are referred to as frames in DNP3. The pseudo-transport layer inserts a single byte of information in the message header of each frame. This byte contains information such as whether the frame is the first or last frame of a message as well as a sequence number for the frame.

Data Link Layer

The data link layer handles the transmission and reception of data frames across the physical layer. Each data link frame contains a source and destination address to ensure the receiving device knows where to send the response. To ensure data integrity data link layer frames contain two **CRC** Cyclic redundancy check, a type of hash function used to produce a checksum, to detect errors in transmission or storage. bytes every 16 bytes.

Data link layer frames are sent with or without a confirmation request. When a confirmation is requested the receiving device replies with a confirmation indicating the message was received and the CRC checks passed.

Physical Layer

The physical layer handles the state of the communications media (for example, busy or clear) and synchronizes data flow across the media (starting and stopping). The more common physical media for DNP is over a simple serial layer such as RS-232 or RS-485. Telepace also supports implementing DNP3 over an Ethernet connection.

Modbus Database Mapping

In SCADAPack controllers static DNP objects such as binary input, analog input, binary counter and analog output are associated with Modbus registers. Whenever a DNP object is created an associated Modbus register(s) is also assigned. Application programs executing in the SCADAPack controller, C or logic, are able to assign physical I/O to Modbus registers using the Telepace Register Assignment or the IEC 61131-3 I/O Connection and these physical I/O points can then be assigned to DNP objects. User application data such as runtimes, flow totals etc. may be also be assigned to DNP objects.

This architecture enables DNP master stations and outstations to pass not only physical data points between them but also to monitor and control user applications executing in the SCADAPack controller. For example a master station can monitor a level in an outstation and then, based on the application program, send a set point value to another outstation to control the level.In SCADAPack controllers static DNP objects such as binary input, analog input, binary counter and analog output are associated with Modbus registers. Whenever a DNP object is created an associated Modbus register(s) is also assigned. Application programs executing in the SCADAPack controller, C or logic, are able to assign physical I/O to Modbus registers using the Telepace Register Assignment or the IEC 61131-3 I/O Connection and these physical I/O points can then be assigned to DNP objects. User application data such as runtimes, flow totals etc. may be also be assigned to DNP objects.

This architecture enables DNP master stations and outstations to pass not only physical data points between them but also to monitor and control user applications executing in the SCADAPack controller. For example a master station can monitor a level in an outstation and then, based on the application program, send a set point value to another outstation to control the level.

SCADAPack DNP3 Operation Modes

Within a DNP3 network, a SCADAPack controller can operate as a:

- DNP3 outstation (slave)
- DNP3 master
- DNP3 master mimic
- DNP3 router

DNP Master Mimic and DNP Router are incompatible and mutually-exclusive modes of operation.

A DNP outstation forms the basic class of any DNP node in a network. All other operational modes derive from a DNP Outstation. A DNP outstation responds to requests from one or more DNP master stations on a network. Also, a DNP Outstation is able to initiate unsolicited responses (messages) based on event data to a master station.

A DNP Master is capable of polling for data, accepting and processing unsolicited messages, and sending control commands to an outstation. Note that a DNP Master can also act perform all the duties of a DNP Outstation.

A SCADAPack controller acting as a DNP Router is simply acting a pass through, basically redirecting messages from one DNP node to another. Similarly to a DNP Master, a DNP Router can also perform all the duties of a DNP Outstation.

DNP Network topologies comprise several combinations of DNP Masters, DNP Routers, and DNP Outstations. Typical configurations possible with SCADAPack controllers are:

- DNP Master and single DNP Outstation
- DNP Master and multi-dropped DNP Outstations
- DNP SCADA Host, Data Concentrator (Mimic Master) and multi-dropped DNP Outstations
- DNP SCADA Host, DNP Router and multi-dropped DNP Outstations

The following topics describe SCADAPack DNP3 operation modes.

SCADAPack DNP Outstation

A DNP3 Outstation can be considered the base class of all terminal nodes on a DNP network. All other DNP3 configuration modes, such as Master, Mimic Master or Router, as implemented by the DNP driver, inherit their properties from the outstation base class. In other words, a SCADAPack controller can simultaneously take on any other operation mode, in addition to being a DNP outstation.

When configured as a DNP outstation a SCADAPack controller is able to:

- Map physical I/O data to DNP points.
- Define DNP points as Class 0 (Static or None), Class 1, Class 2 or Class 3 data types.
- Respond to requests from one or more master stations such as a SCADA hosts or other SCADAPack controllers capable of operating as DNP Masters.

- Initiate unsolicited responses to one or more master stations.

'Unsolicited responses' are also known as 'unsolicited messages'. 'Unsolicited messages' will be used predominantly in this document.

One distinguishing feature of a DNP outstation is this ability to trigger unsolicited messages to a master, upon event accumulation. Events are accumulated when the state of a DNP point changes or an analog values exceeds a threshold. Dead bands can be used to filter out noise from being reported as event data.

After accumulating a certain number of DNP events, or if a certain time period has expired, a DNP outstation will trigger an unsolicited message to all its configured master DNP stations, reporting event data. As defined by the DNP specification, an outstation that triggers an unsolicited message expects a confirmation from all the targeted masters (or peers). If an acknowledgement is not received with a configured Application Layer timeout, the outstation will retransmit the initial unsolicited message. If no response is received within the Application Layer timeout, the outstation will retransmit again. This process continues until the outstation has retransmitted the message a number of times as configured by its Application Layer Retries parameter.

If all retry attempts fail, this message is discarded from the transmit buffer. The SCADAPack DNP driver will re-attempt a failed DNP transaction after a random period of time has expired. Retransmissions will be attempted until the messages are eventually received by the master.

Application Layer messages that are larger than **249 bytes** are broken down into Data Link frames. The DNP protocol allows one to configure acknowledgements of individual Data Link frames, this enhancing network robustness, especially under noisy environments. When the underlying network structure is noise free (wired or networks for instance), enabling Application and Data Link confirmations are not necessary.

How to Configure a DNP Outstation

In this example a DNP outstation is configured with address 10 and 2 binary input points associated with Class 1 and Class 2 events. The station will be configured to trigger unsolicited messages to Master station 200, when Class 1 and Class 2 events occur on these digital inputs. The binary inputs are mapped to Modbus registers 10001 and 10002 with DNP Addresses of 0 and 1.

This example may be used with the example for How to Configure a DNP Master to gain a richer understanding of DNP configuration and communications.

After this example exercise, you should be able to:

- Enable the DNP protocol on a serial port (com2).
- Configure the DNP Application and Data Link Layers
- Configure Class Events Generation and Transmission
- Configure a DNP Routing table
- Configure DNP points.

Tasks to Complete

The following tasks are necessary to configure this simple DNP outstation. Each task is described throughout the rest of this topic.

- Enable the DNP protocol on a serial port.
- Configure the DNP Addresses for local station and master station.
- Configure the DNP Protocol Settings (Application and Data Link Layers).
- Configure a DNP Routing table
- Configure Event Transmission for DNP Class events.
- Configure DNP binary points and set them to trigger unsolicited messages to the master station.
- Write configuration to controller.
- Confirm successful configuration.

Enable DNP on a Serial Port

The first step recommended in configuration the DNP driver on a controller is to enable DNP on the communication interface.

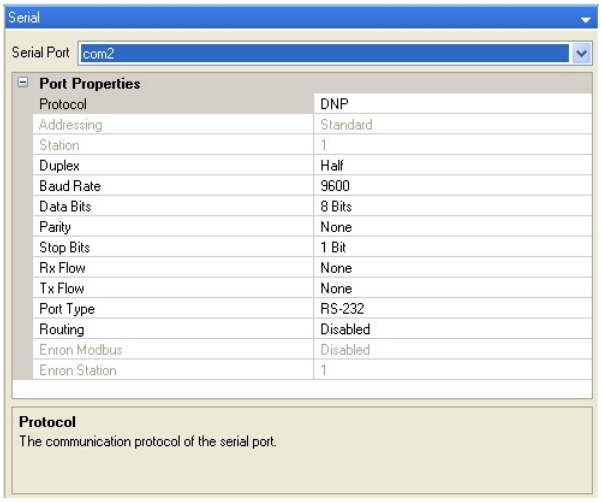
1. Open the Serial Port Settings pane.
2. Select **com2** from the Serial Port drop down list.
3. Select **DNP** from the Protocol drop down list.

Configure DNP Outstation Addresses

From the Edit Ribbon select DNP to open the DNP Settings pane.

1. Select the **DNP Addressing** from the tree list.
2. Set the **RTU Address** to **10**.
3. Set the **Master Address #1** to **200**.

These settings are shown below.



Configure DNP Protocol Settings

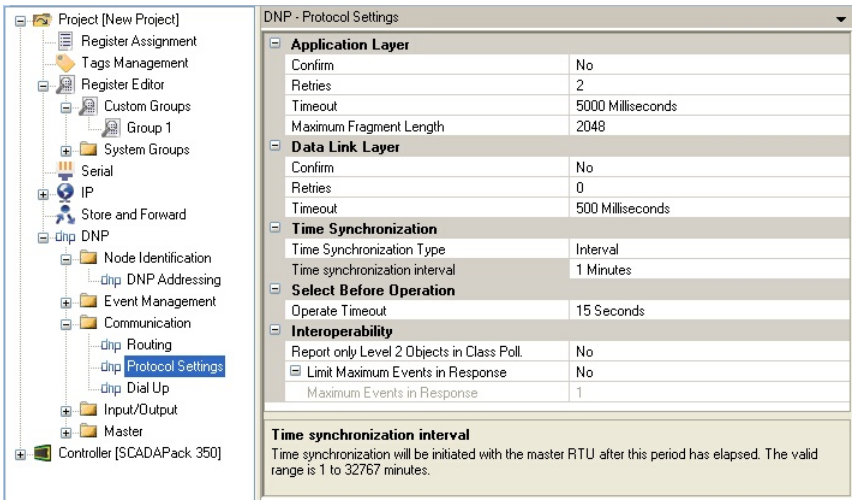
The DNP Protocol settings pane defines how this DNP station will communicate with other DNP devices.

1. Select the Protocol Setting branch from the DNP tree to open the Protocol Settings pane.
2. Set the Retries to 2 in the Application Layer section.
3. Leave all other settings at default values.

TIP:

- It is not necessary to enable the Application Layer confirmation as unsolicited events, by their nature, request for an Application Layer confirmation.
- Time Synchronization is left as None as it is recommended that the DNP master station initiate time synchronization.

These settings are shown below.



Configure DNP Routing Settings

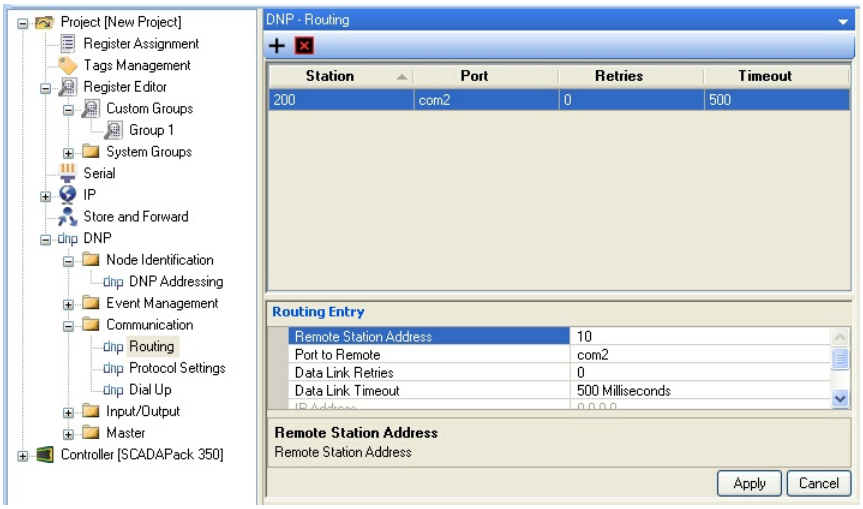
The Routing pane defines how this DNP station will route unsolicited messages between the local master station and the remote station.

1. Select the Routing branch from the DNP tree to open the DNP Routing pane.
2. Set the **Routing Entry Parameters**: **Remote Station Address** to **200** and the **Port to Remote** to **com2**.
3. Click the **ADD** button to add the entry to the DNP Routing Table.
4. Leave all other settings at default values.

TIP:

- Even though a controller outstation will respond successfully to master request, without this routing entry to the master, it is a good practice to always define such a routing entry from an outstation to its master. Moreover, without a routing entry defined to the master, the outstation will not know which port to send out unsolicited messages, if configured, to the master.

These settings are shown below.



Configure Event Transmission properties

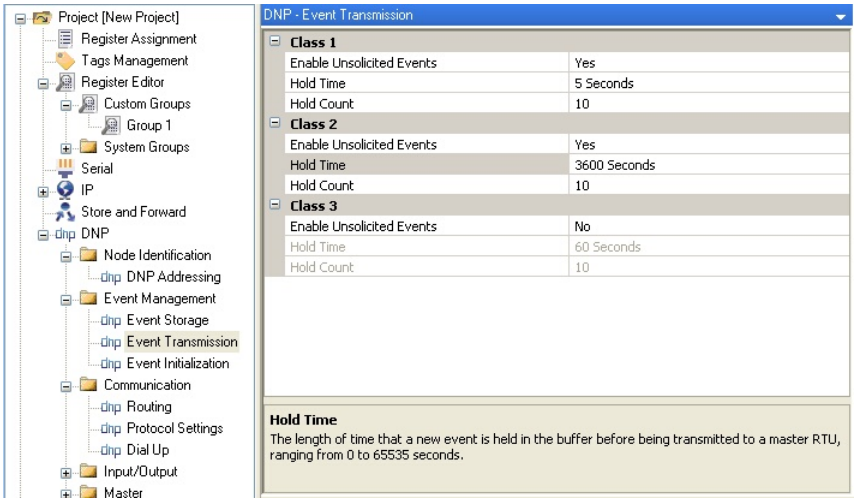
The Event Transmission pane defines how Class events are reported to the master station.

1. Select the Event Transmission branch from the DNP tree to open the Event Transmission pane.
2. Set the Class 1 properties: **Enabled Unsolicited Events** to **Yes**, **Hold Time** to **5 Seconds** and **Hold Count** to **100**.
3. Set the Class 2 properties: **Enabled Unsolicited Events** to **Yes**, **Hold Time** to **3600 Seconds** and **Hold Count** to **10**.
4. Leave all other settings at default values.

TIP:

- On systems with multiple outstations that could potentially transmit unsolicited messages to a master at the same time, it is recommended to use a combination of the Hold Time and Hold Count parameters to avoid multiple stations from transmitting at the same time.
- Class 2 events are typically of less importance than Class 1 events and may not need to be reported to the master as quickly as Class 1 events.
- It may be necessary to enable the Data Link confirmation on noisy networks. However, if the Maximum Application Fragment Length is reduced to 249 bytes, it is not necessary to enable the Data Link confirmation, as each data link packet is in essence an Application Layer fragment.

These settings are shown below.



Configure DNP Points

This step assumes a Register Assignment for your controller has been configured.

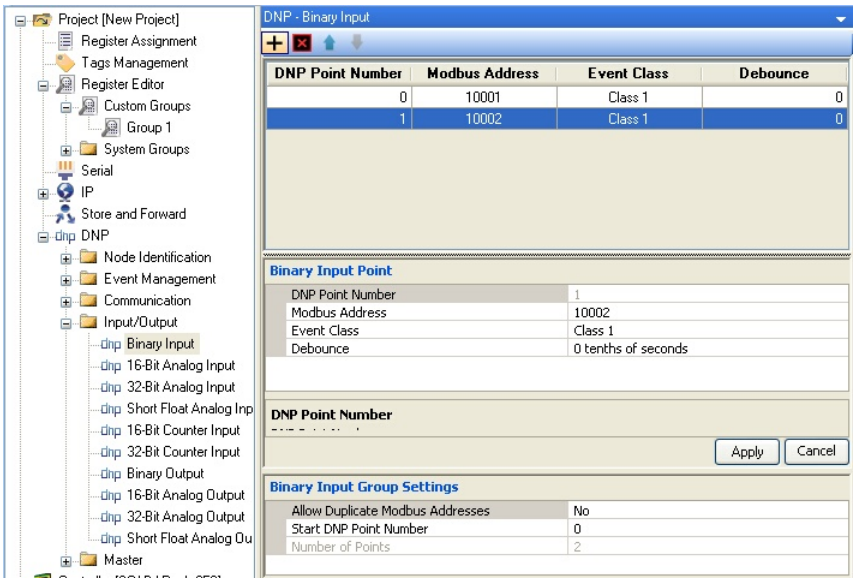
The Binary Input pane defines how DNP binary points are configured in the DNP outstation.

1. Select the Binary Input branch from the DNP tree to open the Input Point pane.
2. Click the **Add** button to open the Binary Input Point grid.
2. Set the Binary Input Properties properties: **Modbus Address** to **10001**, **Event Class** to **Class 1** and **Debounce** to **10 tenths of seconds**.
4. Click the **Apply** button to add the entry to the Binary Input table.
3. Set the Binary Input Properties properties: **Modbus Address to 10002**, **Event Class** to **Class 2** and **Debounce** to **10 tenths of seconds**.
4. Click the **Apply** button to add the entry to the Binary Input table.

TIP:

- It is a good idea to set a non- zero Debounce on unfiltered inputs, to avoid noise being collected as Class events. The same applies for analog inputs. A non-zero Deadband will prevent noise from being collected as Class events.

These settings are shown below.



Write Configuration to controller

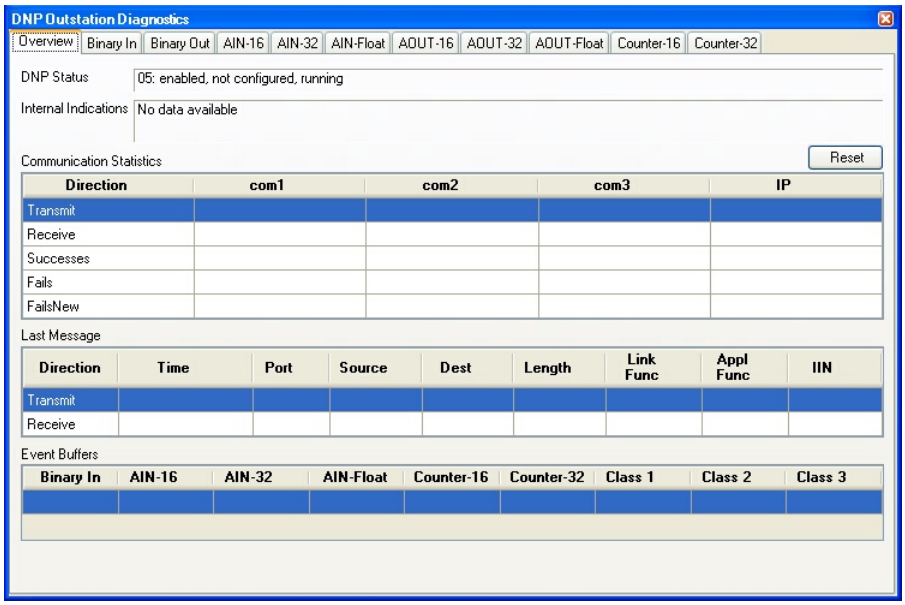
- Write the Telepace Studio project to the controller.
- in order to observe the outstation diagnostic statistics you will need to be connected to the target controller.

Confirm Successful Configuration

The DNP Outstation Diagnostics pane provides complete information on the operation of the DNP Outstation.

1. Select the DNP Outstation Diagnostics command from the Program ribbon.
2. Confirm that the **DNP Status** window displays: **07:enabled, configured, running**.
3. Toggle the state of digital input 1 configured earlier in this exercise and observe the event buffer for Binary Inputs increment on each change of state. After 5 seconds has elapsed, notice that an unsolicited DNP message is triggered to master station 200. Given that DNP master station 200 is not yet configured and connected, a response to the unsolicited message will not be received and the 5000ms Application layer timeout period will expire. The unsolicited message transmission will subsequently retransmitted and will be aborted after 3 retry attempts have been made. This confirms that your outstation is properly setup and unsolicited messages are being generated and sent.

The DNP Outstation Diagnostics window is shown below.



SCADAPack DNP3 Master

DNP master modes apply to the SCADAPack 32, SCADAPack 314, 33x, 350 and SCADAPack 4203 controllers. As a master, a SCADAPack controller can be a regular Master or Mimic master.

SCADAPack DNP3 Master Concepts

A DNP3 master station has all the characteristics of a DNP3 outstation and, in addition, a DNP3 master station is able to:

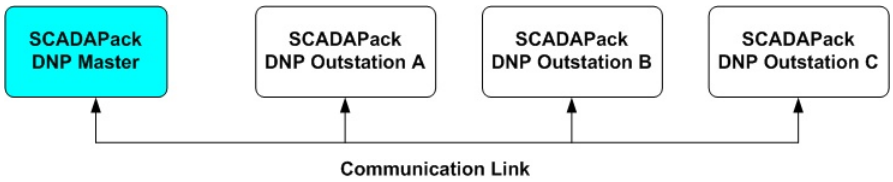
- Poll DNP3 outstations for static (Class 0) data and Class 1, 2 and 3 event data.
- Accept and process unsolicited response messages from polled outstations.

This configuration of a DNP3 master (Client) and DNP3 outstation (Server) forms the basis of a DNP3 Network. The SCADAPack DNP3 master may be configured to periodically poll a SCADAPack DNP3 outstation for Class 0, 1, 2, and 3 data objects and receive unsolicited responses from the outstation. The outstation may be configured to report change event data to the master station using unsolicited responses.

The arrowed line between the master and outstation in the diagram below represents a communication path connecting the two stations. This communication medium may be any type that is supported by both controllers, such as direct serial, leased line modem, dial-up modem and radio for example.

An extension of a simple DNP3 master and single outstation network, involves a SCADAPack DNP3 master connected to a number of outstations over a multi-drop communication channel. The DNP3 master may be configured to periodically poll each SCADAPack DNP3 outstation for Class 0, 1, 2, and 3 data objects and receive unsolicited responses from the outstations. The outstations may be configured to report change event data to the master station using unsolicited responses.

The arrowed line between the master and outstations, in the diagram below, represents the communication path connecting the stations. This communication path may be any multi-dropped type that is supported by the controllers, such as leased line modem, dial-up modem and radio for example.



The DNP3 master feature is limited to a SCADAPack32, SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADAPack 4203.

How to Configure a DNP Master

In this example a DNP master station is configured to poll a DNP outstation with address 10. The DNP master will be communicating to the outstation by requesting for Class event data and acknowledging receipt of unsolicited responses through com1.

This example may be used with the example for How to Configure a DNP Outstation to gain a richer understanding of DNP configuration and communications.

After this example exercise, you should be able to:

- Configure a DNP Master to poll for Static (Class 0) and Class 1 and Class 2 event data.
- Configure a DNP Master to accept and respond to unsolicited messages

Tasks to Complete

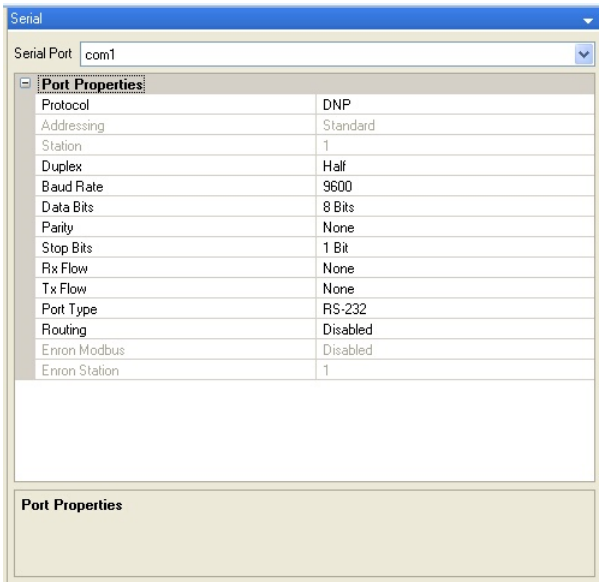
The following tasks are necessary to configure this simple DNP outstation. Each task is described throughout the rest of this topic.

- Enable the DNP protocol on a serial port (com1).
- Configure the DNP Addresses for local station and master station.
- Configure the DNP master to issue class polls to a DNP outstation (station 10 configured in the How to Configure a DNP Outstation topic).
- Configure the DNP master to accept unsolicited messages from a DNP outstation (station 10 configured in the How to Configure a DNP Outstation topic).
- Map DNP outstation DNP point to local DNP points.
- Write configuration to controller.
- Confirm successful configuration.

Enable DNP on a Serial Port

The first step recommended in configuration the DNP driver on a controller is to enable DNP on the communication interface.

1. Open the Serial Port Settings pane.
2. Select **com1** from the Serial Port drop down list.
3. Select **DNP** from the Protocol drop down list.

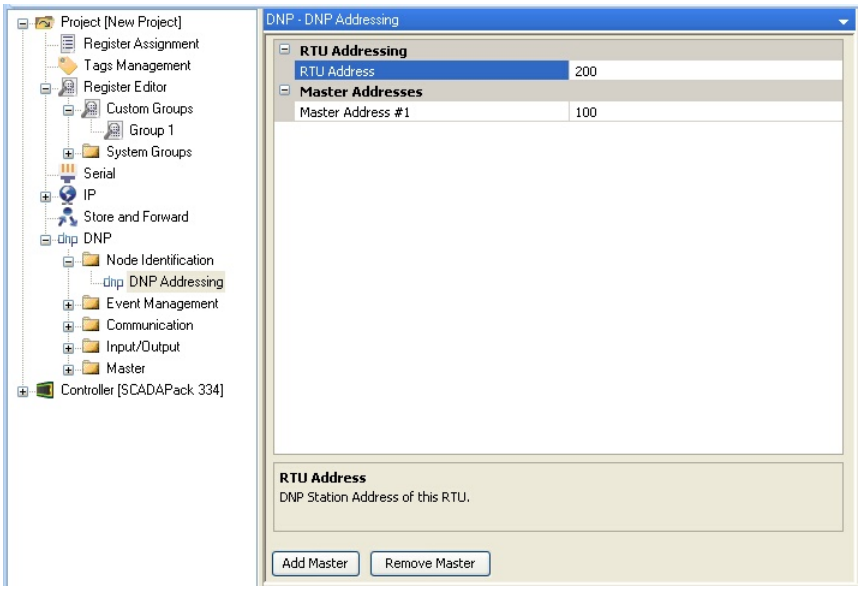


Configure DNP Outstation Addresses

From the Edit Ribbon select DNP to open the DNP Settings pane.

1. Select the **DNP Addressing** from the tree list.
2. Set the **RTU Address** to **200**.
3. Set the **Master Address #1** to **100**.

These settings are shown below.



Configure DNP Protocol Settings

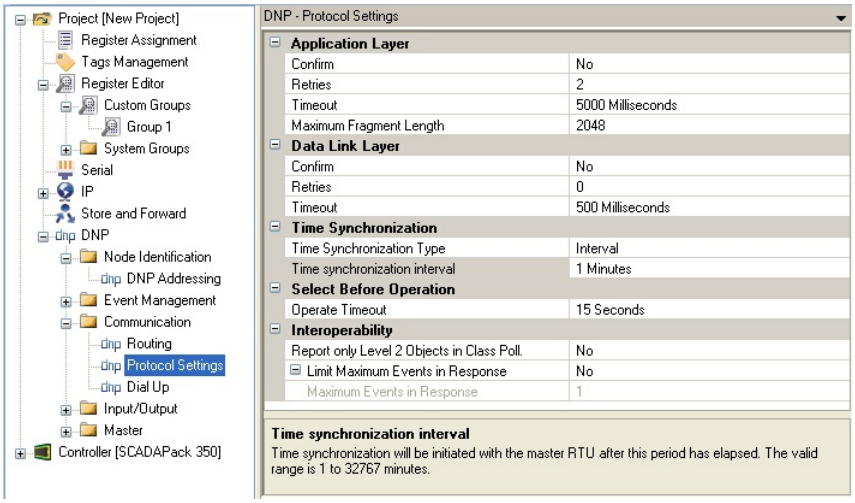
The DNP Protocol settings pane defines how this DNP station will communicate with other DNP devices.

1. Select the Protocol Setting branch from the DNP tree to open the Protocol Settings pane.
2. Set the Retries to 2 in the Application Layer section.
3. Leave all other settings at default values.

TIP:

- A master should not have to request for an Application Layer Confirmation, as an Application Layer response is implied in all master requests.

These settings are shown below.

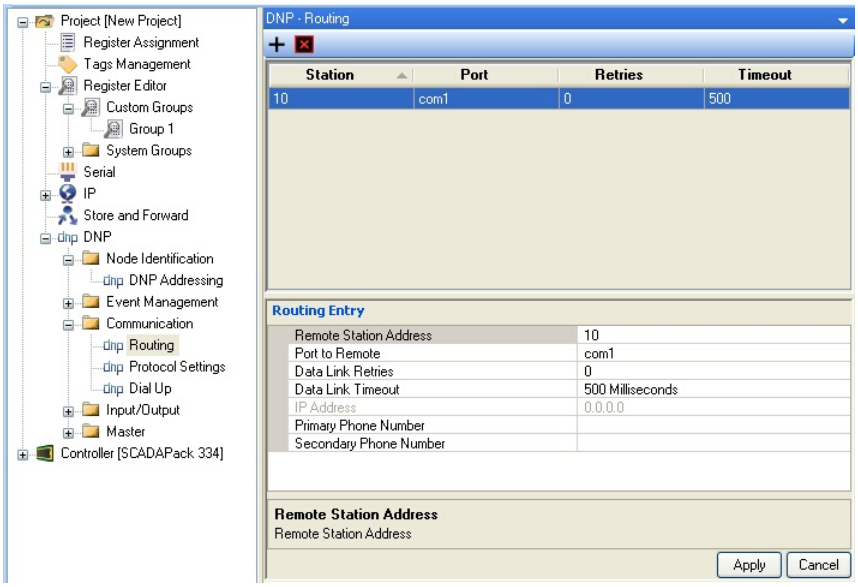


Configure DNP Routing Settings

The Routing pane defines how this DNP station will route unsolicited messages to the master station.

1. Select the Routing branch from the DNP tree to open the DNP Routing pane.
2. Set the **Routing Entry Parameters**: **Remote Station Address** to **10** and the **Port to Remote** to **com1**.
3. Click the **ADD** button to add the entry to the DNP Routing Table.
4. Leave all other settings at default values.

These settings are shown below.



Configure Master Poll Schedules

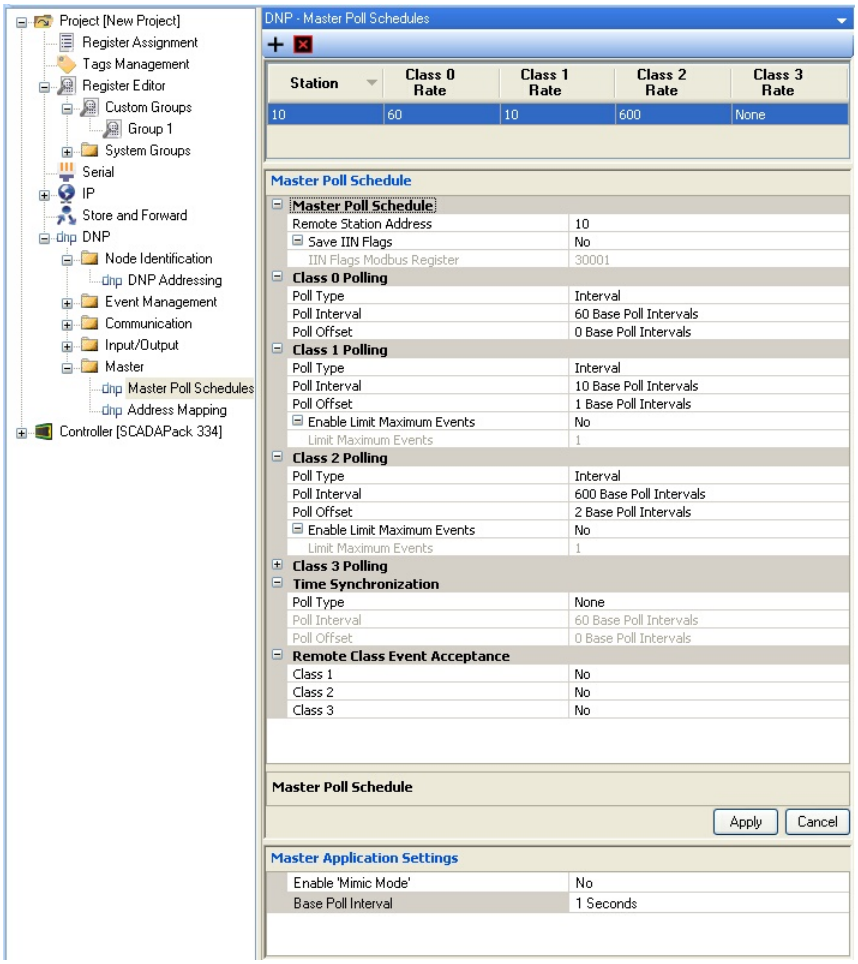
The Master Poll Schedules pane defines how the DNP master polls outstations and whether to accept outstation unsolicited messages.

1. Select the Master Poll Schedules branch from the DNP tree to open the Master Poll Schedule pane and click the Add button to start configuration.
2. Set the **Remote Station Address** to **10** in the Master Poll Schedule area.
3. Set the **Base Poll Interval** to **1 Second** in the Master Application Settings area.
4. Set the **Class 0 Polling** parameters; **Poll type** to **Interval**, **Poll Interval** to **3600 Base Poll Intervals** and **Poll Offset** to **0 Base Poll Intervals**.
5. Set the **Class 1 Polling** parameters; **Poll type** to **Interval**, **Poll Interval** to **10 Base Poll Intervals** and **Poll Offset** to **1 Base Poll Intervals**.
6. Set the **Class 2 Polling** parameters; **Poll type** to **Interval**, **Poll Interval** to **600 Base Poll Intervals** and **Poll Offset** to **2 Base Poll Intervals**.
7. Set the **Remote Class Acceptance** parameters; **Class 1** to **Yes**.
8. Set the **Remote Class Acceptance** parameters; **Class 2** to **Yes**.
9. Leave all other settings at default values.

TIPS:

- A small Base Poll interval provides better granularity.
- Static (Class 0) comprise current values of all DNP3 points in the I/O database. Due to the shear size of this data, it is recommended to reduce the frequency of static polls. Urgent data will be updated at the master via Class polls or unsolicited messages.
- Polling intervals on Master request for time synchronization are configured in this dialog. If possible, set this to a daily frequency. A small base poll interval limits that maximum

These settings are shown below.



Write Configuration to controller

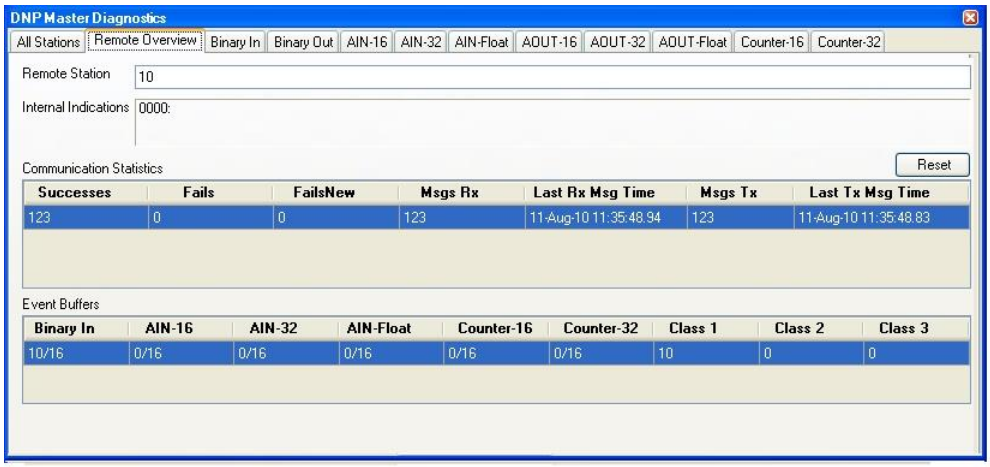
- Write the Telepace project to the controller.
- **Note** that in order to observe the outstation diagnostic statistics you will need to be connected to the target controller.

Confirm Successful Configuration

The DNP Master Diagnostics pane provides complete information on the operation of the DNP Outstation.

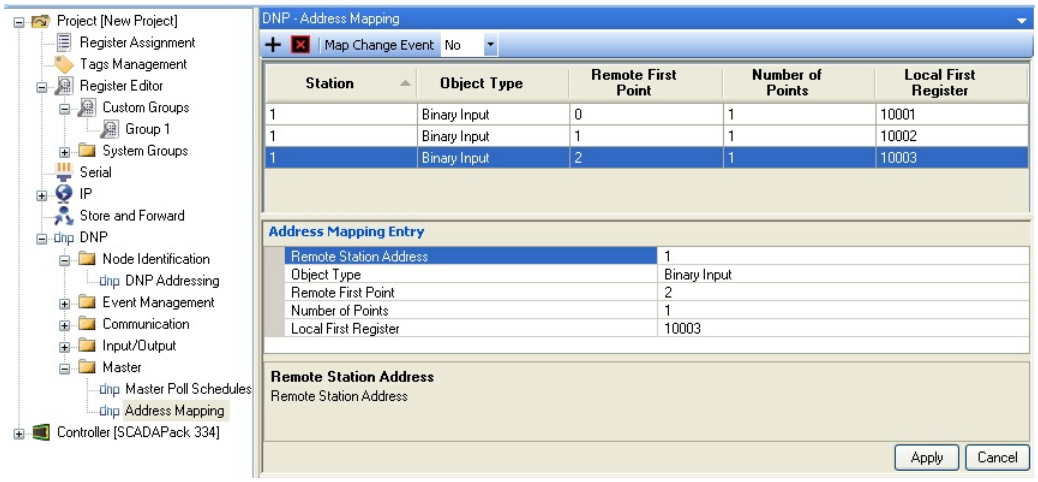
1. Select the DNP Master Diagnostics command from the Program ribbon.
2. Select the **Remote Overview** tab and enter **10** into the **Remote Station** window.
2. The **Communication Statistics** table displays the communication statistics between the master and connected outstation(s).

The DNP Master Diagnostics window is shown below.



SCADAPack DNP3 Address Mapping

Address mapping provides a direct link between DNP3 points on an outstation and local Modbus registers within the SCADAPack DNP3 master. These remote DNP3 points are mapped into specific regions of the Modbus database of the DNP3 master.

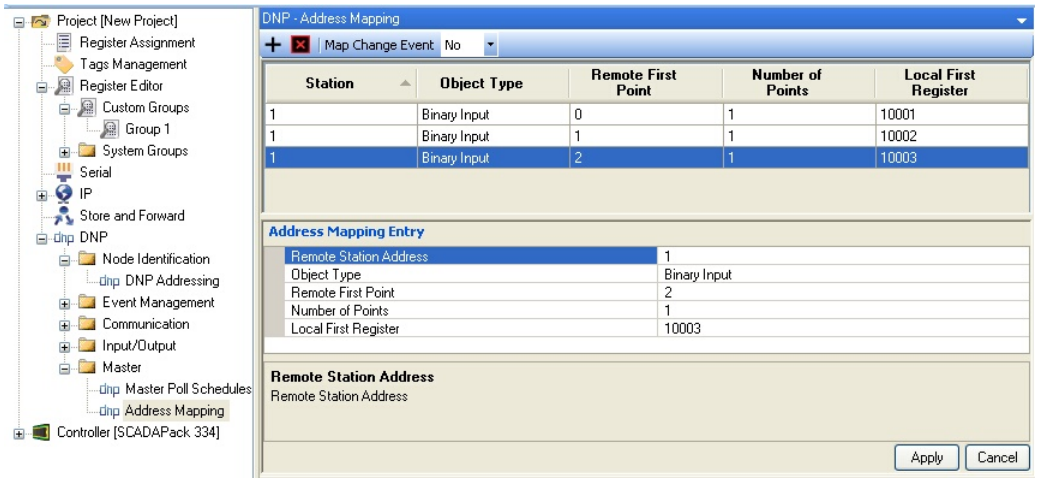


When DNP3 data points are received from an outstation, a cross reference to the address mapping table is made, and if a match is found, the DNP3 data will be written to the corresponding local Modbus register. 'Input' DNP3 object types from the outstation are mapped to the master's local input Modbus register space 1xxxx or 3xxxx. These local Modbus registers are updated after the corresponding DNP3 point gets updated; usually by a class poll to the outstation, or if the outstation issues an unsolicited response based on a change of value or state on these points.

'Output' DNP3 object types from the outstation are mapped to the master's local output Modbus register space 0xxxx or 4xxxx. Changes made to the local Modbus register will trigger a DNP3 Write message, with the current point value, to the outstation. DNP3 Write implemented in SCADAPack controllers requires an Application Layer confirmation from the target outstation.

By configuring the Address Mapping table, outstation DNP3 points are therefore mapped to local Modbus registers. As mapped local Modbus points, the data is available for use in application programs such as Telepace and IEC 61131-3. In addition a Modbus SCADA host can poll the SCADAPack master for these points.

The following diagram shows a simple DNP3 Address Mapping network.



In this network the SCADAPack master updates is local database with mapped outstation data. The manner and frequency with which the SCADAPack master updates the local Modbus registers, depends on the number and type of I/O object types the registers are mapped to.

This feature is limited to the SCADAPack 32, SCADAPack 330/334, SCADAPack 350 and SCADAPack 4203 controllers.

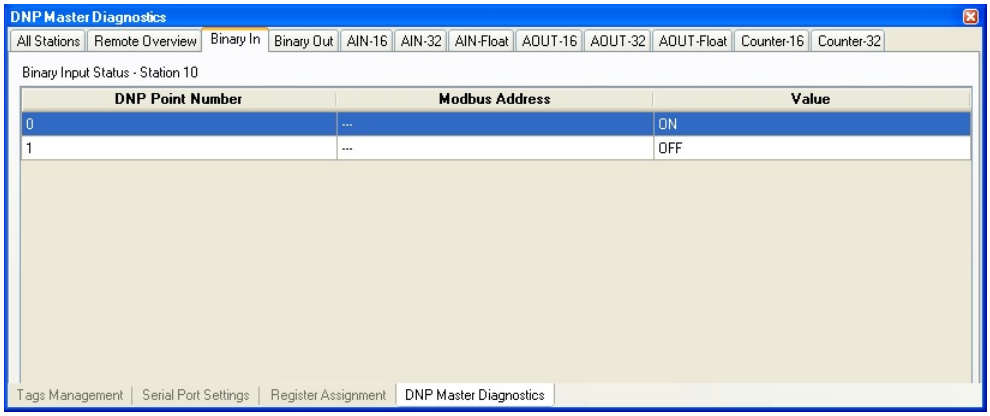
Mapping numerous local Modbus output registers (0xxxx and 4xxxx), to a remote DNP3 device may cause frequent communications between the master and the slave, if the associated registers are being changed frequently in the master. On limited bandwidth or radio networks, care must be taken to ensure that your network capacity can handle all the traffic that will be generated from these local changes.

How to Configure DNP Address Mapping

Using the example configurations for DNP Outstation and DNP Master it has been seen that the DNP Master is able to poll for all DNP Outstation points. After a

successful poll the current status of the DNP Outstation points can be viewed using the tabs for the data point types.

In the example configurations DNP binary input points 0 and 1 are polled from the DNP Outstation form the DNP Master station. Viewing the Binary In tab, shown below, the DNP Point Number and Value fields contain data but the Modbus Address field is blank. This indicates that the remote DNP points have not been mapped to local Modbus registers in the Master.



While this data is available in the DNP Address space of the master, it is not available for use within a local application (Ladder Logic or C++) program. To maker DNP data available to a local application program, Address Mapping between DNP points and Modbus registers is needed.

The following section describes how to map DNP binary input data from DNP Outstation 10 to the DNP Master's local Modbus database.

Configure Address Mapping

The Address Mapping pane defines how the DNP points from a remote outstation are mapped to local Modbus registers in the DNP Master.

1. Select the Address Mapping branch from the DNP tree to open the Master Poll Schedule pane and click the **Add** button to start configuration.
2. In the Address Mapping Entry grid:

   Set the **Remote Station Address** to **10.**

   Set the **Object Type** to **Binary Input.**

   Set the **Remote First Point** to **0**.

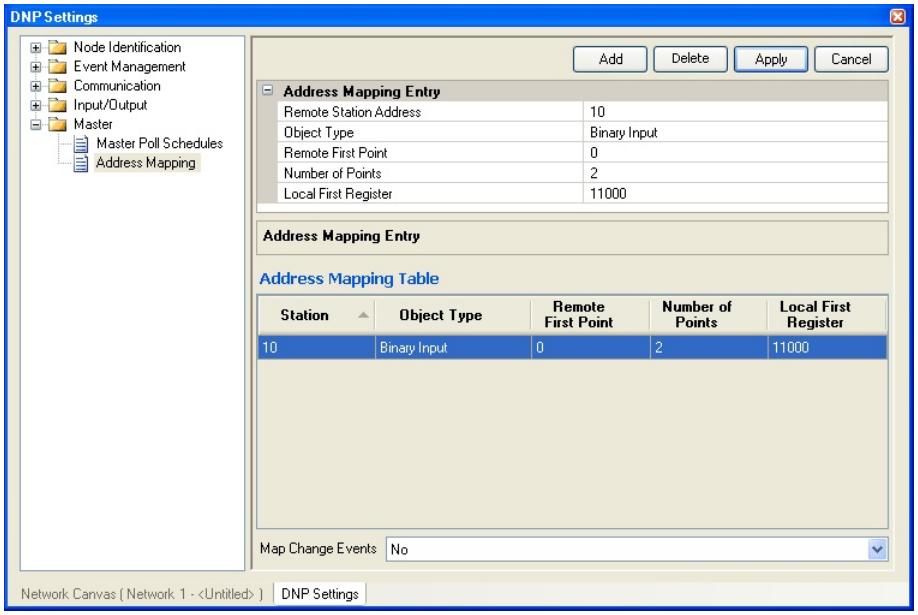
   Set the **Number of Points** to **2**.

   Set the **Local First Register** to **11000**.
3. Click the Apply button to add the entry to the Address Mapping Table.

TIPS:

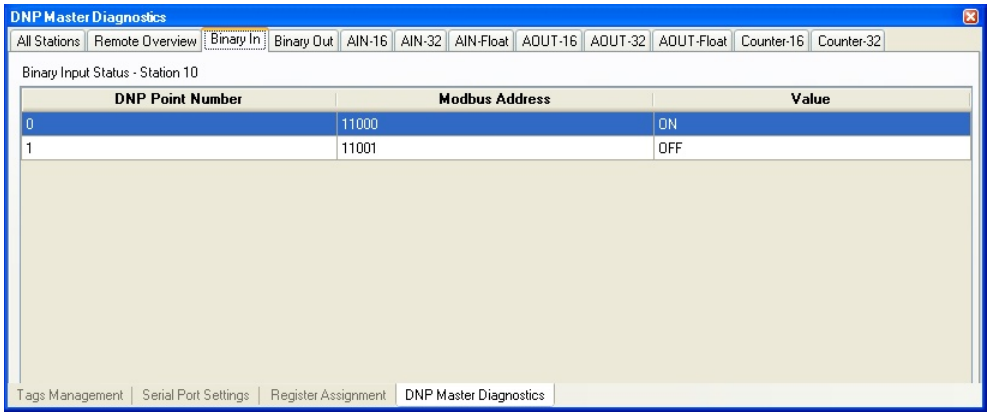
- In a practical setting, a DNP Master may have local I/O mapped also mapped to points within its DNP database. Ensure that you are not mapping outstation DNP points to local address being used by local I/O.

These settings are shown below.



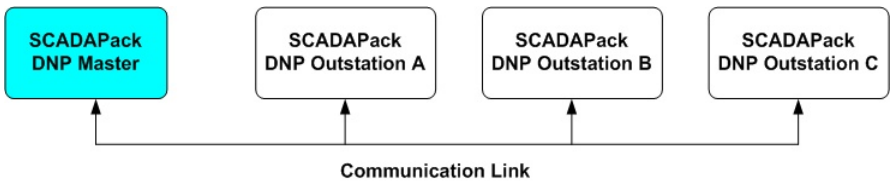
- Write the Telepace project to the controller.

The DNP Master Diagnostics pane view when DNP Address Mapping is used is shown below.



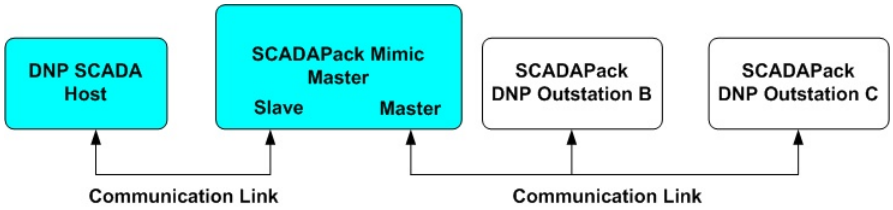
SCADAPack DNP3 Mimic Master

In a typical DNP network a SCADA host master communicates with a number of outstations. The SCADA host will poll each outstation for data and may receive change event data in the form of unsolicited responses from the outstations. This type of DNP3 network is shown in the following diagram.



In the above configuration the SCADA host manages the communication path with each outstation. When the communication path is slow, such as with dial-up communication, or subject to high error rates, such as with some radio communication, the data update rate at the SCADA host can become very slow.

Adding a SCADAPack controller configured for Master Mimic Mode, allows for the SCADA host to poll the SCADAPack (master mimic) for all outstation data instead. In essence, the SCADAPack master mimic is acting as a Data Concentrator, reporting on behalf of all the outstations currently configured in its routing table. The following diagram shows the addition of the SCADAPack master mimic.



In this configuration the outstation side of the network has been decoupled from the host side of the network, as the SCADAPack mimic master now manages all the communication with the outstations.

The SCADA host and all outstations will typically be connected to different communication ports of the SCADAPack Master Mimic. The mimic will respond to the following DNP3 messages on behalf of the targeted station:

- Read messages (this includes class polls as well as individual point reads) from SCADA host
- Write messages from SCADA host
- Unsolicited messages from an outstation
- Direct operate messages from SCADA host

The following DNP3 messages cannot be mimicked (Mimic does not respond on behalf of target DNP3 station), and are routed directly to the target outstation by the Mimic:

- Select and Operate messages
- Data Link Layer messages (e.g. get link status, reset link status, etc)
- Enable/Disable Unsolicited Message commands (FC 20 and 21)
- Other control messages

To provide current outstation data to the SCADA host, the SCADAPack mimicking master independently communicates with each outstation to update a local copy of its database with data from the outstations. This communication may be initiated by the SCADAPack mimicking master, either by polling each outstation in turn using solicited messages; or the outstations could initiate unsolicited messages back to the mimicking master. There could also be a combination of solicited and unsolicited messages between the mimicking master and the outstations.

In the Mimic mode diagram above the SCADAPack mimic master polls each outstation, A and B, for data and holds images of this data in its memory. When the SCADA host poll outstations A and B for data, the mimic master replies from its own images of the outstations. The SCADA host can also poll the SCADAPack master for its own local data.

Typically the messaging strategy chosen will depend on the relative importance of the data, and the required maximum end-to-end delays for data being transferred through the network. If the requirement is for a reasonably short end-to-end delay for all data points, a round-robin polling scheme is best, without any unsolicited messages. If there are some data points, which are higher priority and must be transferred as fast as possible, unsolicited messages should be used.

The advantage of having the SCADA system communicating with the SCADAPack 32 mimic, instead of direct communication to the outstations is that communication

delays and high error rates are effectively removed. The physical connection between the SCADA system and mimic master SCADAPack is typically a direct high-speed reliable connection and all message transactions are fast. outstations may often be connected via slow PSTN or radio links, and therefore message transactions are subject to substantial delays. They may also be unreliable communication links subject to high error rates.

By having a multiple-level network the communication between the SCADAPack master and outstations is separated from communication between SCADA system and the SCADAPack master. The delays and error rates, which may be inherent in the outstation communication paths, can be isolated from communications with the SCADA system, thereby increasing overall system performance.

One particular advantage of Mimic Mode is that the master SCADAPack does not need to know, or be configured with, any details of the DNP3 points configured in the outstations. This makes it relatively simple to insert such a SCADAPack master into any existing DNP3 network. The SCADAPack master in Mimic Mode behaves transparently to the higher-level SCADA system, and can easily be configured with communication paths and polling instructions for each connected outstation.

This feature is available in the SCADAPack 32, SCADAPack 330/334, SCADAPack 350 and SCADAPack 4203 controllers.

How to Configure DNP Mimic Master

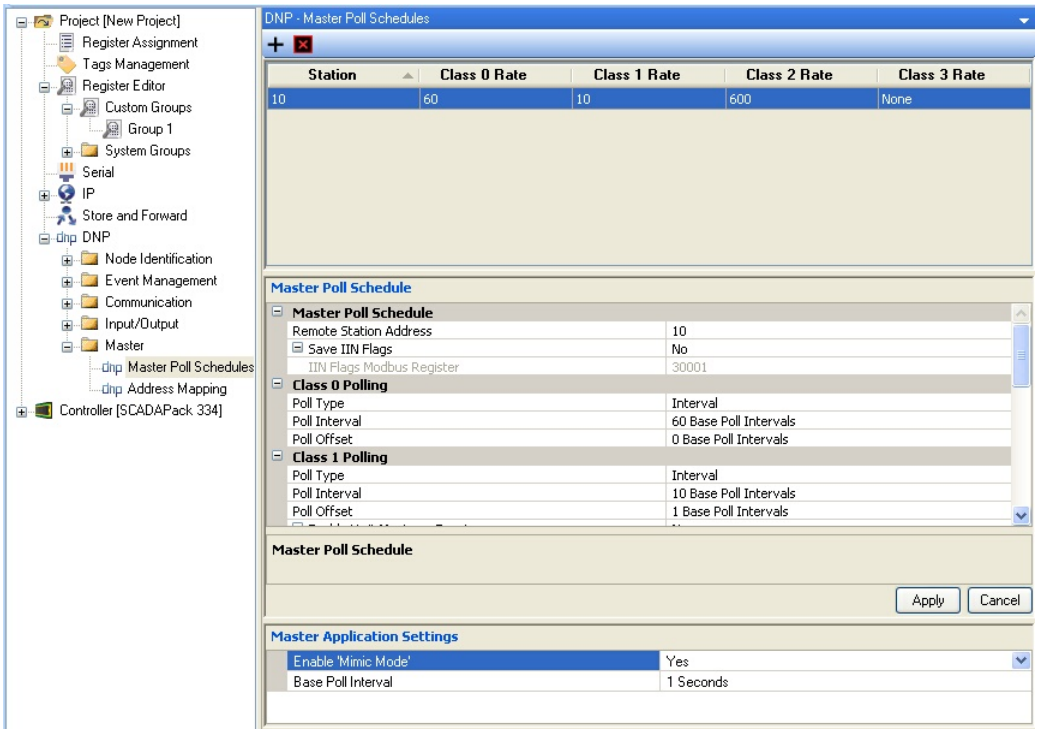
In addition to the configuration parameters set for a DNP Master the following step enables the DNP Mimic Master functionality.

Configure DNP Mimic Master

The Master Poll Schedules pane defines how the DNP master polls outstations and whether to Mimic Mode is enabled.

1. Select the Master Poll Schedules branch from the DNP tree to open the Master Poll Schedule pane and click the Add button to start configuration.
2. Set the Enable **'Mimic Mode'** to **Yes**.

The setting is shown below.



SCADAPack DNP3 Router

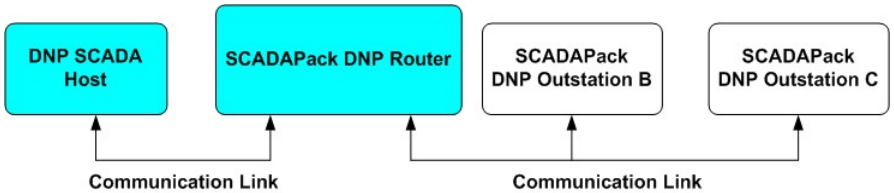
SCADAPack controllers can be configured as a DNP Router. A unique characteristic of a SCADAPack DNP router is the ability to:

- Route (or forward) DNP messages not destined to this station, using rules defined within a routing table.

Otherwise, a SCADAPack controller not configured for DNP routing will simply discard a message whose DNP destination address does not match that of the controller.

A DNP router is typically used when a direct communication link between the DNP master and outstation cannot be established, typically due to different physical layers on the two network segments. For instance, the physical network between the DNP SCADA Host and the router could be an Ethernet connection, while the physical layer between the router and all outstations could be a multi-drop serial RS-485 or even an RS-232 radio connection. Given that messages are routed directly from the DNP SCADA Host to the outstations, bandwidth limitations are dictated by the speed of the serial multi-drop connection. On the contrary, there is no bandwidth limitation within a DNP Mimic architecture, as the Mimic Master immediately responds to the DNP SCADA Host on behalf of the targeted outstation. Of course, the side effect of the DNP Mimic architecture is that polled data obtained by the DNP SCADA Host may not be very current. In either case, careful design considerations based on these tradeoffs should be exercised.

As mentioned above, the SCADA Host has only one connection to a SCADAPack DNP Router. All target outstations of the SCADA Host are connected down stream of the DNP Router as illustrated in the figure below.



In the above configuration the SCADAPack DNP Router (Outstation A above) manages all the communication with the outstations. The SCADAPack DNP router receives messages from the SCADA Host for each outstation and route or forwards the messages to the outstations, based on routing rules established with the DNP Routing table.

DNP Messages are routed based on the following logic:

```
if (a message is received which needs to be retransmitted to someone else)
    if (the message target is configured in our routing table)
        if (the destination port is different from the incoming port)
        or (routing is enabled on the incoming port)
        then retransmit the message
```

Change event data in the form of unsolicited responses from the outstations are routed directly to the DNP SCADA Host, by the SCADAPack DNP router.

A DNP Router is different from a Mimic in that a router forwards all messages are directly to the outstations, whereas the mimic responds to some messages on behalf of the outstations. Therefore, both operation modes have the advantage of delegating the task of DNP Routing of multiple outstations to this intermediate unit. The SCADAPack DNP router handles all communications paths to outstations, including such tasks as dial-up radio communication. In contrast to Mimic mode, however, the SCADA Host system still has to handle the long delays and high error rates that may be present on the communications links to the outstations.

Note:	Mimic Master and Routing are incompatible modes that should never be used together.

How to Configure DNP Router

In this example a DNP master station is configured to route DNP messages received from DNP Master 32001 on its Ethernet port, out through com2. This message is destined for outstation 20. .This exercise assumes a valid Ethernet connection between your PC or laptop and the controller2.

After this example exercise, you should be able to:

- Configure DNP/TCP on an Ethernet port.
- Configure a DNP Router to route DNP messages from a SCADA Host application, such as ClearSCADA, to a DNP Outstation.

Tasks to Complete

The following tasks are necessary to configure this simple DNP outstation. Each task is described throughout the rest of this topic.

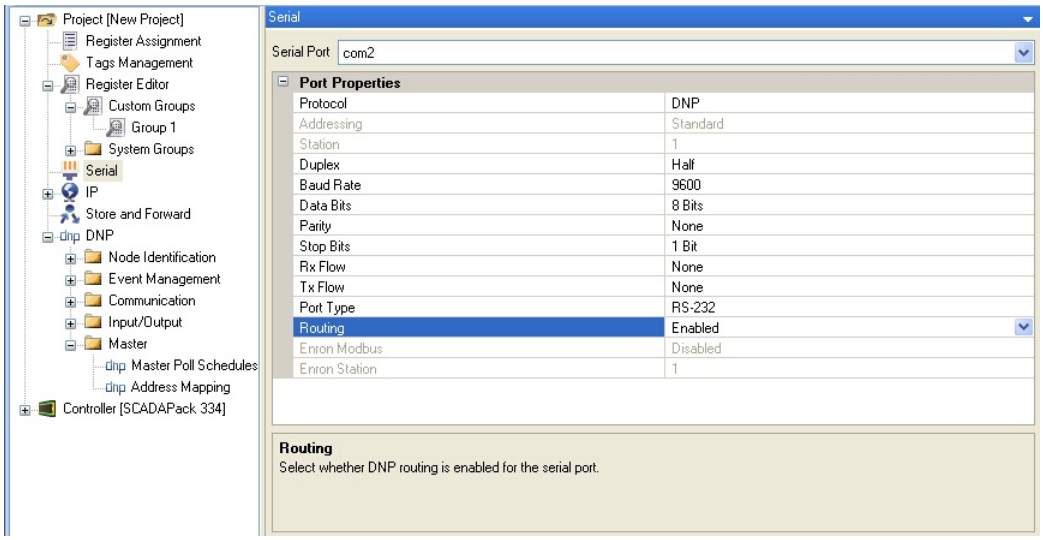
- Enable the DNP protocol on a serial port (com1).
- Enable DNP protocol on the Ethernet port.
- Configure DNP Routing for the serial port and Ethernet port.

Enable DNP and Routing on a Serial Port

These steps configure the serial port to use DNP and to enable Routing on the serial port.

1. Open the Serial Port Settings pane.
2. Select **com2** from the **Serial Port** drop down list.
3. Select **DNP** from the **Protocol** drop down list.
4. Select **Enabled** from the **Routing** drop down list.

These settings are shown below.



Enable DNP on the Ethernet Port

These steps configure the ethernet port to use DNP.

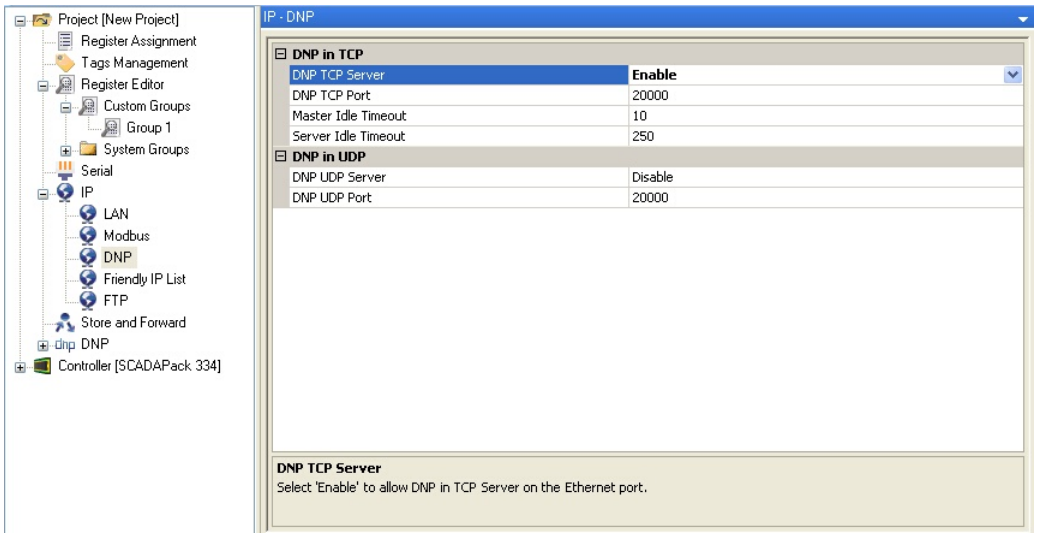
1. From the Edit Ribbon select IP DNP to open the DNP Settings pane. Open the Serial Port Settings pane.
2. Select **com2** from the **Serial Port** drop down list.
3. Select **DNP** from the **Protocol** drop down list.
4. Select **Enabled** from the **Routing** drop down list.

TIPS:

- In this configuration, the controller DNP Router is acting as a DNP Server on the Ethernet port. The **Server Idle Timeout** parameter will be used to determine how long this connection will be kept open from time of last communication activity. For a **Server Idle Timeout** default value of 4 minutes, and an **Application Layer Timeout** default value of 5 minutes, there is the possibility that the IP port will be closed, if the router is experiencing communication problems with the

outstations. In this case, it is a good idea to increase the Server Idle Timeout to at least 2x the DNP configuration Application Layer Timeout. Or, simply reduce the Application Layer timeout to a value less than 2x the Server Idle Timeout.

These settings are shown below.



Configure DNP Routing Settings

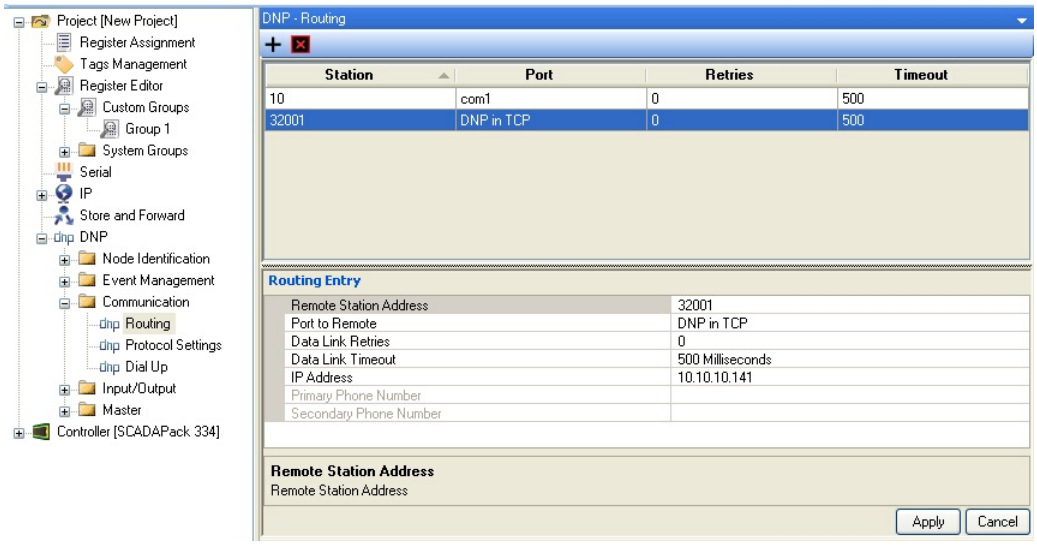
The Routing pane defines how this DNP station will route unsolicited messages between the local master station and the remote station.

1. Select the Routing branch from the DNP tree to open the DNP Routing pane.
2. Click the **ADD** button to open the **DNP Routing Entry** grid.
3. Set the **Routing Entry Parameters**: **Remote Station Address** to **20** and the **Port to Remote** to **com2**.
4. Click the **Apply** button to add the entry to the **DNP Routing Table**.
2. Click the **ADD** clear the **DNP Routing Entry** grid.
3. Set the **Routing Entry Parameters**: **Remote Station Address** to **32001,** the **Port to Remote** to **DNP in TCP** and the **IP Address** to the address of the SCADA Host computer. In this case ClearSCADA is running on a PC with an IP address of **10.10.10.141**.
4. Click the **Apply** button to add the entry to the **DNP Routing Table**.
4. Leave all other settings at default values.

TIP:

- For proper operation of the router, there must be two routing entries in the routing table for each outstation; An entry specifying how the communication path from this router to the outstation and another communication path from the router to the SCADA DNP Master

These settings are shown below.



Design Considerations

The strength of DNP3 lies in its ability to offer time-stamped data, scheduled polling of data from multiple outstations and time synchronization, event data buffering and reporting by exception.

DNP3 was originally design to be used over a serial point-to-point (RS-232) link. As such, the protocol implements certain measures against data corruption and data loss in its Application and Data Link layers. Such measures include timeouts, retries, and checksums.

These data recovery mechanisms provided by the protocol, can be counter productive when not properly configured over an underlying communication medium, such as Ethernet, that already provides robust measures. In almost all of such cases, the recovery mechanisms offered by DNP3 need to be turned off. Such considerations together with good engineering judgment, therefore, must be practiced before one embarks on the design of a large DNP3 network.

This topic outlines special considerations of the DNP3 protocol and implications within the SCADAPack DNP3 driver that should be considered when designing large networks. We also list common malpractices and a list of Frequently Asked Questions (FAQs) that arise during the course of network design.

Considerations of DNP3 Protocol and SCADAPack DNP3 Driver

To ensure consistent network performance, even under worse case scenarios, the following DNP3 specification rules should be considered when designing a DNP3 network using SCADAPack as the main nodes.

Unsolicited Messages always request a Confirmation

An outstation will always request for an Application Layer confirmation when it sends an unsolicited message, even if the Application Layer confirmation field is not enabled. If no response is received within an Application Layer timeout, the outstation will retry the message a number of times as determined by the Application Layer Retry parameter.

Master should never request for Application Layer Confirmation

As implemented in the SCADAPack DNP3 driver, a DNP3 Write request (FC 02) requires an Application Layer response from the outstation. If an acknowledgement is not received within the configured Application Layer timeout interval, the message is retried a number of times as determined by the Application Layer retry parameter.

DNP Write Messages always request for a confirmation

As implemented in the SCADAPack DNP3 driver, a DNP3 Write request (FC 02) requires an Application Layer response from the outstation. If an acknowledgement is not received within the configured Application Layer timeout interval, the message is retried a number of times as determined by the Application Layer retry parameter.

Only one DNP transaction can be pending at a time

A SCADAPack DNP3 station will not initiate or process another DNP3 transaction, as long as one is outstanding. Thus, once a SCADAPack has initiated a DNP3 transaction, all subsequent <%DNP33%> messages received but not related to the original transaction are buffered.

SCADAPack controllers buffer 3 DNP messages

A SCADAPack serial port receive buffer can hold a maximum of 3 DNP3 messages or Data Link frames. If an additional DNP3 message is received when the buffer is full, the oldest message in the buffer is replaced with the newest one.

Output points in DNP Address Mapping issue DNP write

Digital and analog output points contained within the DNP3 Address Mapping of a SCADAPack controller automatically issue DNP3 Write messages when their value or state changes.

Typical Configuration Malpractices and Recommendations

DNP3 is a capable protocol that effectively transfers some of the system engineering effort from designing a sophisticated logic program, to configuring and tuning the system using parameters. However, DNP3 does not eliminate the need to properly evaluate and engineer the communication media to support the performance expectations of the system, especially under worse case scenarios.

DNP3 networks can be designed around polling or report-by-exception. In a polling environment, each master request can be viewed as an invitation for an outstation device to transmit data on the shared communications medium. The master controls which device can transmit, thereby preventing collisions from occurring, as the timing of responses is predictable under all situations. In addition, masters can ask again if a response is not received, thus providing an opportunity for the outstation to re-send lost data. Using this strategy, a master effectively manages media access thereby preventing contention with those outstations unexpectedly transmitting on their own.

DNP3 networks can also be designed around unsolicited communications. In this case, the outstations transmit events to the master as they occur. When using this strategy, the communications media must be evaluated carefully in regards to the need for collision detection and prevention, if consistent network performance is to be expected.

Given that typical systems are designed using a combination of both strategies, is a good idea to start by configuring the network for poll mode, as it can be easily tuned to cater for unsolicited messaging, when system characteristics under worst case conditions become known. As with any communications system, the designer should pay careful attention to bandwidth allocation and management for a successful system implementation.

Below are several requirements of DNP3 system architecture that need careful engineering judgment.

1. Multiple high priority unsolicited messages configured in outstation.
2. System with multiple outstations, each containing numerous Class 1 events, configured with a Class 1 Hold Count of 1.
3. Relying on unsolicited messaging to get event data to master. System not designed around master polling for events.
4. Multiple masters with poor communication link.
5. Insufficient use of Deadband and debounce to curb event generation.
6. Master RTU has Application Layer confirmation enabled.
7. Enabling both the Application and Data Link Layer confirmations.
8. Setting very high Application Layer timeout values over high speed networks.
9. DNP3 Address mapping contains multiple analog and digital output points that change rapidly.

The aforementioned statements and recommendations are provided in the linked topics. Note that these recommendations are to ensure, consistent performance under

worse case situations, and are based on the special considerations provided in the previous section.

Multiple High Priority Unsolicited Messages

A common configuration malpractice is to enable numerous high priority events objects within an outstation, and configure the outstation to trigger an unsolicited message to the master each time a new event occurs. In a SCADAPack controller, this is accomplished by configuring numerous Class 1 event objects, and enabling Class 1 Unsolicited Responses (Messaging) with a Hold Count or Hold Time of 1.

A Hold Count of 1 and Hold Time of 60 seconds specified for Class 1 events, imply that the controller will immediately trigger an unsolicited event as one occurs. If this outstation and others have a multitude of Class 1 event objects, visualize worse case scenario as a burst of messages being transmitted to the master at the same time. Given that a SCADAPack serial port buffer can only handle three DNP3 Data Link frames at any given time, some messages might get lost, especially if the master is required to immediately retransmit this message to some other node in return.

Such a system is designed around unsolicited messaging and is, therefore, far more susceptible to network collisions if proper management of bandwidth it not exercised. Given that a SCADAPack controller can only process one DNP3 transaction at a time, there is also a good chance that the serial port receive buffer will overflow, adding to the cost of lost messages.

Recommendations:

In general, bandwidth is used more efficiently in a large DNP3 system if the master is designed to poll for event data more frequently and static data less regularly.

Recommended practice is also to reserve unsolicited messaging for a small number of critical data. If possible, it may be best to ensure that no more than 3 messages are sent to the master at exactly the same time, under worse case scenario, as some event data may be lost if the master is currently busy processing another transaction, unless random retry intervals are put in place.

If unsolicited messaging is the predominant data transfer method, an approach to manage network usage, could be to configure a group of three or less outstations with a Hold Time that is unique within the group.

The table below shows an example configuration for Hold Time and Hold Counts for Class 1 events across six outstations.

DNP3 Outstation Address	Hold Time (seconds)	Hold Count
11	1	100
12	1	100
13	1	100
14	2	100
15	2	100
16	2	100

Master not polling frequently causing event buffer overflows

An outstation does not discard the events within its buffer until all its configured masters have acknowledged receipt of these events. This means that an outstation event buffer may eventually fill up and overflow leading to loss of events. Buffer overflows typically indicate a poorly configured system.

When the system is designed around unsolicited messaging, there is a good likelihood of media contention causing buffer overflows. On the contrary, if the system is designed around frequent master poll for event data, there will be fewer chances of buffers overflowing causing loss of event data.

As stated earlier, immediate reporting of events using unsolicited messaging should be reserved for those critical, yet absolutely rare occurring events. This is because unsoliciting these messages back to the master will be reliable only if there is a substantial amount of unused bandwidth on the communication media. A good rule of thumb is to have 50% or more of unused bandwidth available, evenly distributed over a time frame.

Recommendation:

Design the system around frequent master poll of class events and less regular integrity polls. Reserve unsolicited messaging for infrequent high priority events. If network traffic is predominated by unsolicited messaging, allocate 50% or more unused bandwidth as quiet time.

Outstation Reports to Multiple Masters with Poor Communications

A poor communication link to one of an outstation's multiple masters will prevent the outstation's event buffer to be emptied, as events cannot be reported to the master. This could lead to buffer overflow situations and loss of event data.

Recommendation:

Ensure that the communication path to all masters of an outstation is robust.

Insufficient Use of Input Deadband or Debounce

Event generation on a DNP3 analog input is controlled by a Deadband parameter. On a digital input, event generation is controlled by a debounce parameter. Default settings of zero for these parameters are typically overly aggressive and may lead to events being generated due to noise.

Recommendation:

Set the analog Deadband and debounce parameters appropriately to non-zero values.

Master Confirmation and Retries

Application or Data Link Layer confirmations should never be enabled on a master as:

1. Master requests typically will fit within a single Application Layer fragment hence there is need for Data Link Layer confirmations.
2. Master request typically require a response, hence no need for Application Layer confirmations.

Thus, enabling the Application Layer Confirmation on a DNP3 master is obsolete practice and may instead degrade system performance.

Recommendation:

Disable the Application Layer Confirmation in a master SCADAPack controller. Typical retry values for Application Layer retries lie between 1 and 3. Lengthy retries may instead burden the communication medium

Outstation Confirmation and Retries

Confirmations on an outstation serve two useful purposes:

1. Ensure that a master received unsolicited responses from the outstation.
2. To ensure that a master correctly received responses to its request

Unsolicited messages will always request for an Application Layer confirmation, whether or not the Application Layer Confirmation is enabled on the outstation. If network traffic is predominantly unsolicited messaging, the Application Layer confirmation does not need to be enabled.

When the master is configured, as recommended, to frequently poll the outstation for event data using read request, while imposing a limit on the number of events the outstation should include in its response, the outstation still needs to know if the master received its replies so that it can:

- Remove these events from its buffer
- Know what to transmit next.

To cater for confirmations to read responses, Application Layer Confirmation in the outstation typically needs to be enabled.

The Data Link Layer breaks down Application Layer fragments into smaller frames. Smaller packet sizes reduce bit error in noisy environments. While it is better to accept the overhead of confirming each Data Link Layer frame of a multi-frame message, and re-transmit corrupted frames, than to re-send an entire Application Layer fragment, a viable alternative is to reduce the Application Layer fragment size and use only Application Layer confirmations. When fragments are reduced to the size of a Data Link Layer frame, the overhead of Application Layer confirmations, and the probability of noise corrupting those confirmation messages, is nearly the same as for Data Link Layer confirmations.

Enabling the Data Link layer confirmation on the outstation, therefore, is not required when the communication medium is not reliable. For example, certain data radios, e.g. FreeWave 9000 MHz spread spectrum radios, implement a robust mechanism to ensure that a data packet make it to their desired destination; TCP/IP incorporates robust mechanisms to prevent data loss; a local serial link between stations is also very reliable. In these cases, it is not necessary to enable the Data Link Layer confirmations.

If, however, physical medium quality if below par, such as in the case of noisy radio networks, or a shaky PSTN connections, then one should enable the Data Link Layer confirmation only, or as mentioned earlier, reduce the Application Layer maximum fragment length below 249 bytes.

If either the Application or Data Link Layer Confirmation is enabled, retries should be configured to a low non-zero value. Typical retry values lie between 1 and 3. Lengthy retries may instead burden the communication medium

Recommendation:

Application and Data Link Layer confirmations in an outstation can be set according to the following table:		Communication Medium Reliability	
		High	Low
Data Acquisition Configuration	master polls outstation frequently for event data (also limits number of events in read response)	Enable Application Layer Confirmation Disable Data Link Layer Confirmation	Disable Application Layer Confirmation Enable Data Link Layer Confirmation
	master does not poll frequently enough and outstation generates lot of unsolicited messages	Enable Application Layer Confirmation Disable Data Link Layer Confirmation	Disable Application Layer Confirmation Enable Data Link Layer Confirmation
	Regardless of the data acquisition strategy, if the Max Application Layer fragment is set to a values less than 249	Enable Application Layer Confirmation Disable Data Link Layer Confirmation	Enable Application Layer Confirmation Disable Data Link Layer Confirmation

Note: It is never required to enable BOTH the Application and Data Link Layer Confirmations.

Setting relatively large Application Layer timeouts

On a high speed link, such as Ethernet, configuring a high Application Layer timeout does not increase network reliability. Instead this reduces system performance, as there will be a significant portion of time within the timeout period, after which the IP transaction may have been terminated.

Typically, an Ethernet transaction is completed in the order of a millisecond and a DNP3 master SCADAPack controller, by default, closes its DNP3 TCP port within 10 seconds of no activity. A DNP3 SCADAPack controller acting as an outstation closes its port by default in about 4 minutes.

Under these default conditions, if the application layer timeout on a SCADAPack DNP3 master is set for 15 seconds, for instance, the port may have closed 10 seconds after last activity, but the application may still be waiting for a timeout.

If a message is somehow lost, and the timeout is set for 5 seconds, for instance, the application will still be waiting for a response even though the IP transaction has terminated. This results to wasted bandwidth.

Recommendation:

When operating over high speed links, make Application Layer timeouts as small as possible.

DNP3 Address Mapping contains multiple output points

The DNP3 Address Mapping table allows local Modbus registers in the SCADAPack DNP3 master to be mapped to DNP3 points in an outstation. Each time an output register defined within the DNP3 Address Mapping table changes, a DNP3 Write message (FC 2) is immediately issued to update the corresponding DNP3 point in the outstation.

If numerous output registers that change frequently, are listed in the Address Mapping table, the network will be overburdened with a multitude of DNP3 Write messages.

Recommendation:

Reserve Address Mapping only for mapping of outstation DNP3 data that needs to be used by the master Modbus database, or to segregate points from different outstations in the master. If numerous points are being mapped from the outstation to the master, the system is not designed properly. In this case, it may be worthwhile to consider transferring application logic from the master to the outstation.

Configuration FAQ

Some frequently asked questions and answers:

Question When configuring a routing entry in the DNP3 Routing table, the Data Link Layer Timeout and Retries must be specified. Do these fields take precedence over the same fields found under the Protocol Settings configuration view?
Answer Yes.

Question In a master-outstation architecture, what is the recommend method to configure time synchronization?
Answer Recommended practice is to configure the master to initiate time synchronization to the outstations.

Question DNP3 provides 4 data classes; Class 0 (Static or None), Class 1, Class 2 and Class 3. How does I decide which class to assign any given I/O point?
Answer In a SCADAPack controller, all configured DNP3 points by nature, are members of the Class 0 type. Class 0 data is the current value or state of a DNP3 point. So, when a master does a Class 0 poll to an outstation, the most current value or state of all DNP3 points within the database are returned. Value or state changes on a point are captured as Class 1, Class 2 or Class 3 event data. Typically, highest priority events are assigned to Class 1 and the lowest priority event to Class 3.

Question What does Class of 'None' mean?
Answer Class None is Class 0 or Static.

Question Why does the Enable Unsolicited Responses On Startup setting do?
Answer This setting enables unsolicited response (or unsolicited message) transmission, when power to an RTU is cycled or when its configuration is changed. In this case, the RTU does not have to wait for Function Code 20 or 0x14 (Enable Unsolicited Responses) from the master before is starts sending any collected events. This field should be set to No, to allow a master control when an outstation is able to send unsolicited messages. Recommended practice is to allow a master to enable unsolicited message transmission on all outstations.

Question Why would I ever need to change the Application Layer Maximum Fragment Length?
Answer The Application Layer Maximum Fragment Length determines the maximum amount of memory that is reserved for each application layer fragment. The default is 2048 bytes on SCADAPack controllers although outstations must be prepared to receive fragments sizes of at least 249 bytes. When communicating with those devices with insufficient memory it is necessary to limit the maximum application layer fragment length to what the outstation can handle. In addition, limiting the application layer fragment size beyond 249 also reduces the maximum Data Link layer frame length.

Certain data radios may give better efficiency when transmitting data packets less than the maximum data link fragment size of 249 bytes. With these radios, it is necessary to reduce the application layer's maximum fragment size below 249 bytes as required by the radio.

In addition, when the communication medium is noisy, it is typically more efficient to transmit smaller packets than larger packets. In this case, setting small Application Layer fragments would force smaller data link frames, which is a better strategy in a noisy environment.

Limiting the application layer fragment size reduces the rate at which event data is retrieved from the buffers, thus increasing the possibility of event buffer overflows, if the event data is not being retrieved in a timely fashion. Reducing the maximum application layer fragment size, increases network traffic and also reduced data throughput as an Application Layer Confirmation is required for each fragment of a multi-fragment message.

Question Why would users ever want to 'Limit that maximum number of events in a read response?"
Answer This is another strategy that can be used to limit the Application Layer fragment of an outstations' response message. This strategy could be used under noisy environments. Also, this could be used to prevent an outstation with a large collection of event data to hold the communication media captive while transmitting all its events.

Question What behavior should we expect from a SCADAPack when the event logs are full?
Answer When a new event is collected and the SCADAPack DNP3 event buffer is full, the oldest event is deleted and the newest event added into the buffer.

What is the main difference between SCADAPack DNP3 driver configuration modes?

	DNP3 Master	DNP3 Mimic Master	Address Mapping	Router	Outstation
Define DNP3 I/O points	Not necessary	Not necessary	Not necessary	Not necessary	Yes
Enable Application Layer Confirmation	No	No	No	No.	Not necessary.
Should Time Synchronization be used?	Yes	No.	No.	No.	No.
Poll for Class DNP3 static and Class data?	Yes	Yes	Yes	No.	No
Initiates Unsolicited messages?	No	No	No	No	Yes
Router Messages not destined to this station?	No	Some	No	Yes	No
When best to use	Master in a Point to Multipoint network	Data Concentrator with many outstations that will take a while to configure. When outstation data does not need to be	When remote DNP3 data is needed by local program	Strictly Repeater	Forms basic node in a DNP3 network.

		available to logic in this node.			

.