



ProTalk

PTQ-DNP

Quantum Platform

Distributed Network Protocol Interface Module

Application Reference Guide

June 30, 2008



Please Read This Notice

Successful application of this module requires a reasonable working knowledge of the Schneider Electric Quantum hardware, the PTQ-DNP Module and the application in which the combination is to be used. For this reason, it is important that those responsible for implementation satisfy themselves that the combination will meet the needs of the application without exposing personnel or equipment to unsafe or inappropriate working conditions.

This manual is provided to assist the user. Every attempt has been made to ensure that the information provided is accurate and a true reflection of the product's installation requirements. In order to ensure a complete understanding of the operation of the product, the user should read all applicable Schneider Electric documentation on the operation of the Schneider Electric hardware.

Under no conditions will ProSoft Technology be responsible or liable for indirect or consequential damages resulting from the use or application of the product.

Reproduction of the contents of this manual, in whole or in part, without written permission from ProSoft Technology is prohibited.

Information in this manual is subject to change without notice and does not represent a commitment on the part of ProSoft Technology Improvements and/or changes in this manual or the product may be made at any time. These changes will be made periodically to correct technical inaccuracies or typographical errors.

PTQ Installation and Operating Instructions

The statement "power, input and output (I/O) wiring must be in accordance with Class I, Division 2 wiring methods Article 501-10(b) of the National Electrical Code, NFPA 70 for installations in the U.S., or as specified in section 18-1J2 of the Canadian Electrical Code for installations within Canada and in accordance with the authority having jurisdiction".

The following or equivalent warnings shall be included:

- A** Warning - Explosion Hazard - Substitution of components may Impair Suitability for Class I, Division 2;
- B** Warning - Explosion Hazard - When in Hazardous Locations, Turn off Power before replacing Wiring Modules, and
- C** Warning - Explosion Hazard - Do not Disconnect Equipment unless Power has been switched Off or the Area is known to be Nonhazardous.
- D** Caution: The Cell used in this Device may Present a Fire or Chemical Burn Hazard if Mistreated. Do not Disassemble, Heat above 100°C (212°F) or Incinerate.

Important Notice:



CAUTION: THE CELL USED IN THIS DEVICE MAY PRESENT A FIRE OR CHEMICAL BURN HAZARD IF MISTREATED. DO NOT DISASSEMBLE, HEAT ABOVE 100°C (212°F) OR INCINERATE.

Maximum battery load = 200 μ A.

Maximum battery charge voltage = 3.4 VDC.

Maximum battery charge current = 500 μ A.

Maximum battery discharge current = 30 μ A.

Your Feedback Please

We always want you to feel that you made the right decision to use our products. If you have suggestions, comments, compliments or complaints about the product, documentation or support, please write or call us.

ProSoft Technology

1675 Chester Avenue, Fourth Floor

Bakersfield, CA 93301

+1 (661) 716-5100

+1 (661) 716-5101 (Fax)

<http://www.prosoft-technology.com>

Copyright © ProSoft Technology, Inc. 2000 - 2008. All Rights Reserved.

PTQ-DNP Application Reference Guide

June 30, 2008

PSFT.DNP.PTQ.UM.08.06.30

ProSoft Technology®, ProLinX®, inRAx®, ProTalk® and RadioLinX® are Registered Trademarks of ProSoft Technology, Inc.

Contents

PLEASE READ THIS NOTICE.....	2
PTQ Installation and Operating Instructions.....	2
Important Notice:	2
Your Feedback Please	3
GUIDE TO THE PTQ-DNP APPLICATION REFERENCE GUIDE.....	7
1 START HERE.....	9
1.1 Hardware and Software Requirements	9
2 CONFIGURING THE PROCESSOR WITH CONCEPT.....	11
2.1 Information for Concept Version 2.6 Users	12
2.2 Create a New Project	13
2.3 Add the PTQ Module to the Project.....	16
2.4 Set up Data Memory in Project.....	18
2.5 Download the Project to the Processor	21
2.6 Verify Successful Download	24
3 CONFIGURING THE PROCESSOR WITH PROWORX	29
4 CONFIGURING THE PROCESSOR WITH UNITYPRO XL	33
4.1 Create a New Project	33
4.2 Add the PTQ Module to the Project.....	35
4.3 Build the Project	37
4.4 Connect Your PC to the Processor	38
4.5 Download the Project to the Processor	40
5 SETTING UP THE PROTALK MODULE.....	41
5.1 Install the ProTalk Module in the Quantum Rack.....	41
5.2 Connect the PC to the ProTalk Configuration/Debug Port.....	43
5.3 Cable Connections	44
5.4 Collision Avoidance (DNP modules only).....	48
6 CONFIGURING THE MODULE	51
6.1 Obtain the Sample Configuration Files.....	51
6.2 Edit the Configuration File	51
6.3 Uploading and Downloading the Configuration File	104
6.4 Verification and Troubleshooting.....	110
7 DIAGNOSTICS AND TROUBLESHOOTING	111
7.1 The Configuration/Debug Menu	111
7.2 Required Hardware	112
7.3 Required Software.....	112

- 7.4 Using the Configuration/Debug Port 113
- 7.5 LED Status Indicators 121
- 8 REFERENCE 123**
- 8.1 Product Specifications 123
- 8.2 Functional Overview..... 125
- 8.3 PTQ-DNP Error Status Table..... 137
- 8.4 PTQ-DNP Module Internal Indication Bits (IIN Bits) for DNP Server..... 144
- 8.5 DNP Subset Definition 145
- 8.6 Event Size Computation 157
- 8.7 DNP Collision Avoidance 158
- 8.8 Frequently Asked Questions..... 160
- 9 SUPPORT, SERVICE & WARRANTY 163**
- 9.1 Return Material Authorization (RMA) Policies and Conditions 163
- 9.2 LIMITED WARRANTY 165
- 9.3 How to Contact Us: Technical Support..... 169
- INDEX..... 171**

Guide to the PTQ-DNP Application Reference Guide

Function		Section to Read	Details
Introduction (Must Do)	→	Start Here (page 9)	This Section introduces the customer to the module. Included are: package contents, system requirements, hardware installation, and basic configuration.
Verify Communication, Diagnostic and Troubleshooting	→	Verifying Communication (page 110) Diagnostics and Troubleshooting (page 111)	This section describes how to verify communications with the network. Diagnostic and Troubleshooting procedures.
Reference Product Specifications Functional Overview Glossary	→	Reference (page 123) Functional Overview (page 125) Product Specifications (page 123)	These sections contain general references associated with this product, Specifications, and the Functional Overview.
Support, Service, and Warranty Index	→	Support, Service and Warranty (page 163)	This section contains Support, Service and Warranty information. Index of chapters.

1 Start Here

In This Chapter

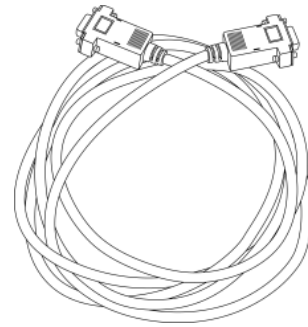
- ❖ Hardware and Software Requirements9

This guide is intended to guide you through the ProTalk module setup process, from removing the module from the box to exchanging data with the processor. In doing this, you will learn how to:

- Set up the processor environment for the PTQ module
- View how the PTQ module exchanges data with the processor
- Edit and download configuration files from your PC to the PTQ module
- Monitor the operation of the PTQ module

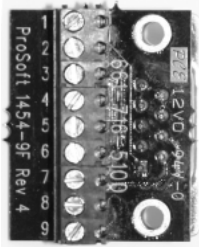
1.1 Hardware and Software Requirements

1.1.1 ProTalk Module Carton Contents



ProTalk Module

Null Modem Serial Cable



1454-9F DB-9 Female to 9 Pos Screw Terminal adapter (Serial protocol modules only)

ProSoft Solutions CD

Note: The DB-9 Female to 5 Pos Screw Terminal adapter is not required on Ethernet modules and is therefore not included in the carton with these types of modules.

1.1.2 Quantum / Unity Hardware

This guide assumes that you are familiar with the installation and setup of the Quantum / Unity hardware. The following should be installed, configured and powered up before proceeding:

- Quantum or Unity Processor
- Quantum rack
- Quantum power supply
- Quantum Modbus Plus Network Option Module (NOM Module) (optional)
- Quantum to PC programming hardware
- NOM Ethernet or Serial connection to PC

1.1.3 PC and PC Software

- Windows-based PC with at least one COM port
- Quantum programming software installed on machine
- or
- Concept™ PLC Programming Software version 2.6
- or
- ProWORX PLC Programming Software
- or
- UnityPro XL PLC Programming Software
- HyperTerminal (used in this guide) This is a communication program that is included with Microsoft Windows. You can normally find it in **Start / Programs / accessories / Communications**.

Note: ProTalk modules are compatible with common Quantum / Unity programming applications, including Concept and UnityPro XL. For all other programming applications, please contact technical support.

2 Configuring the Processor with Concept

In This Chapter

- ❖ Information for Concept Version 2.6 Users..... 12
- ❖ Create a New Project 13
- ❖ Add the PTQ Module to the Project..... 16
- ❖ Set up Data Memory in Project..... 18
- ❖ Download the Project to the Processor 21
- ❖ Verify Successful Download 24

The following steps are designed to ensure that the processor is able to transfer data successfully with the PTQ module. As part of this procedure, you will use Concept configuration software from Schneider Electric to create a project, add the PTQ module to the project, set up data memory for the project, and then download the project to the processor.

Important Note: Concept software does not report whether the PTQ module is present in the rack, and therefore is not able to report the health status of the module when the module is online with the Quantum processor. Please take this into account when monitoring the status of the PTQ module.

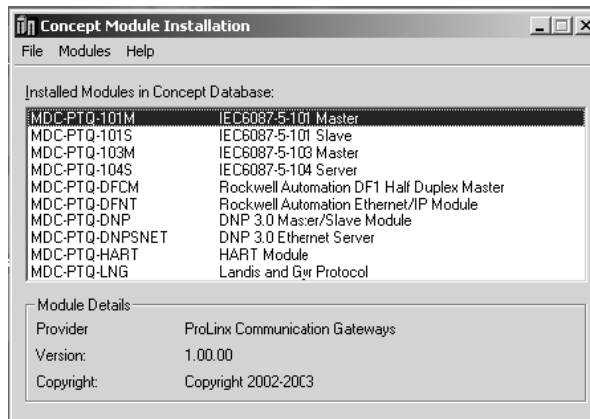
2.1 Information for Concept Version 2.6 Users

This guide uses Concept PLC Programming Software version 2.6 to configure the Quantum PLC. The ProTalk installation CD includes MDC module configuration files that help document the PTQ installation. Although not required, these files should be installed before proceeding to the next section.

2.1.1 Installing MDC Configuration Files

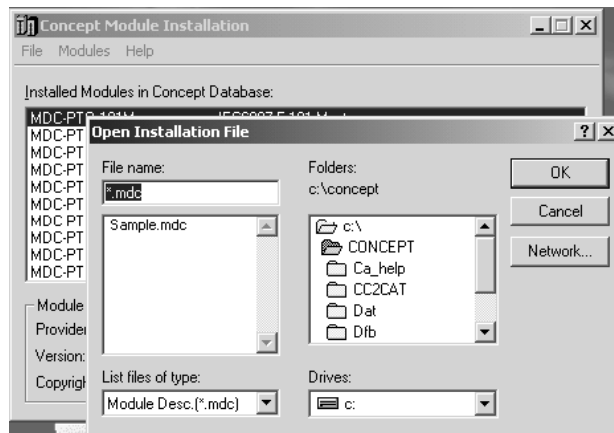
- 1 From a PC with Concept 2.6 installed, choose **Start / Programs / Concept / ModConnect Tool**.

This action opens the Concept Module Installation dialog box.



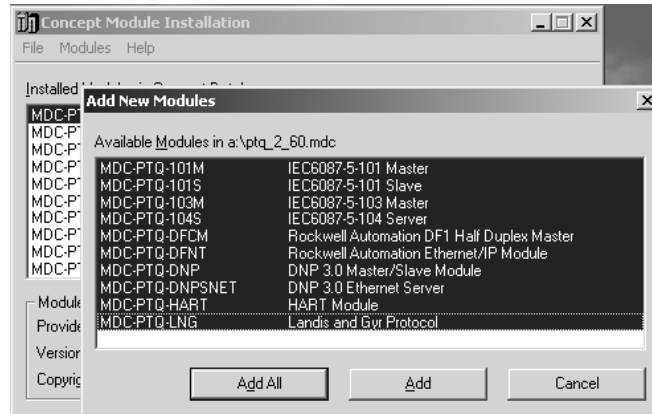
- 2 Choose **File / Open Installation File**.

This action opens the Open Installation File dialog box:



- 3 If you are using a Quantum processor, you will need the MDC files. In the Open Installation File dialog box, navigate to the **MDC Files** directory on the ProTalk CD.
- 4 Choose the MDC file and help file for your version of Concept:
 - o Concept 2.6 users: select PTQ_2_60.mdc and PTQMDC.hlp
 - o Concept 2.5 users: select PTQ_2_50.mdc and PTQMDC.hlp.

Select the files that go with the Concept version you are using, and then click **OK**. This action opens the add New Modules dialog box.

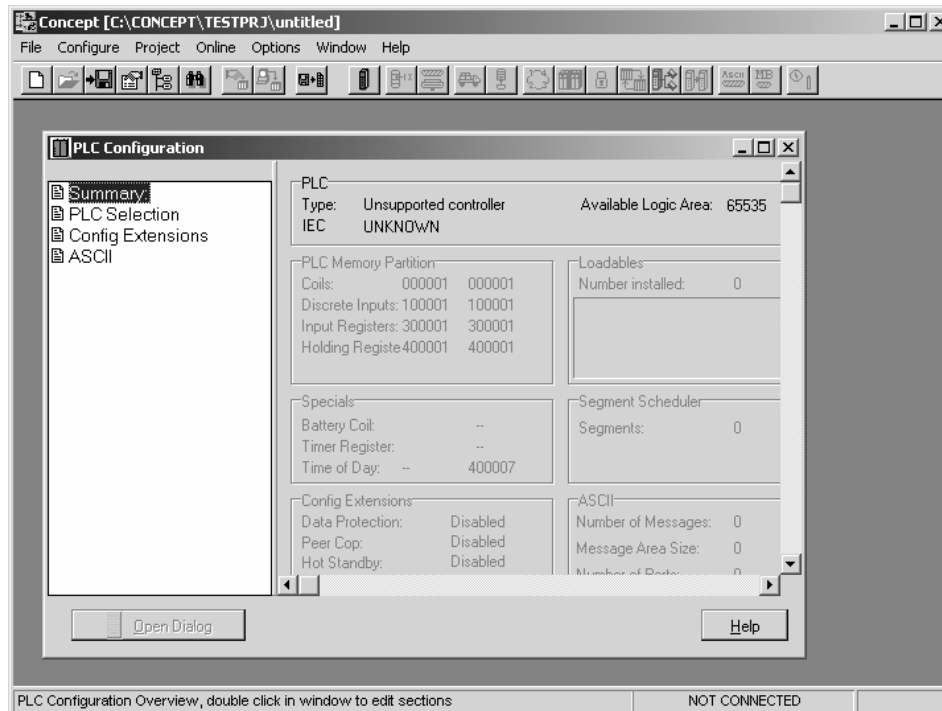


- 5 Click the **add all** button. A series of message boxes may appear during this process. Click **Yes** or **OK** for each message that appears.
- 6 When the process is complete, open the File menu and choose Exit to save your changes.

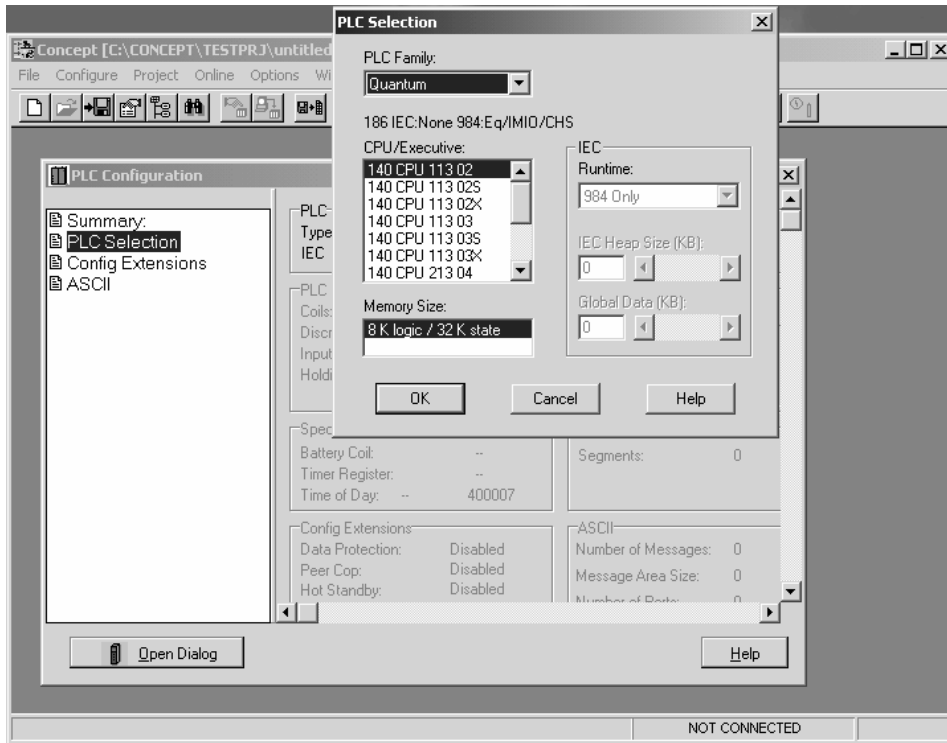
2.2 Create a New Project

This phase of the setup procedure must be performed on a computer that has the Concept configuration software installed.

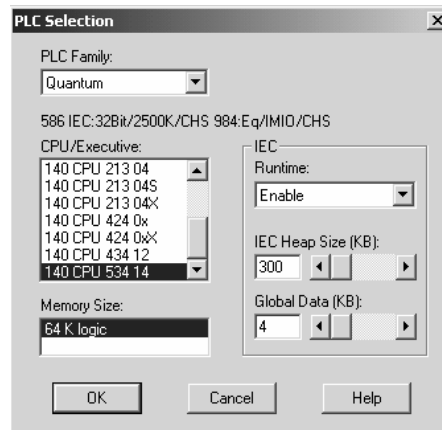
- 1 From your computer, choose **Start / Programs / Concept V2.6 XL.EN / Concept**. This action opens the Concept window.
- 2 Open the File menu, and then choose **New Project**. This action opens the PLC Configuration dialog box.



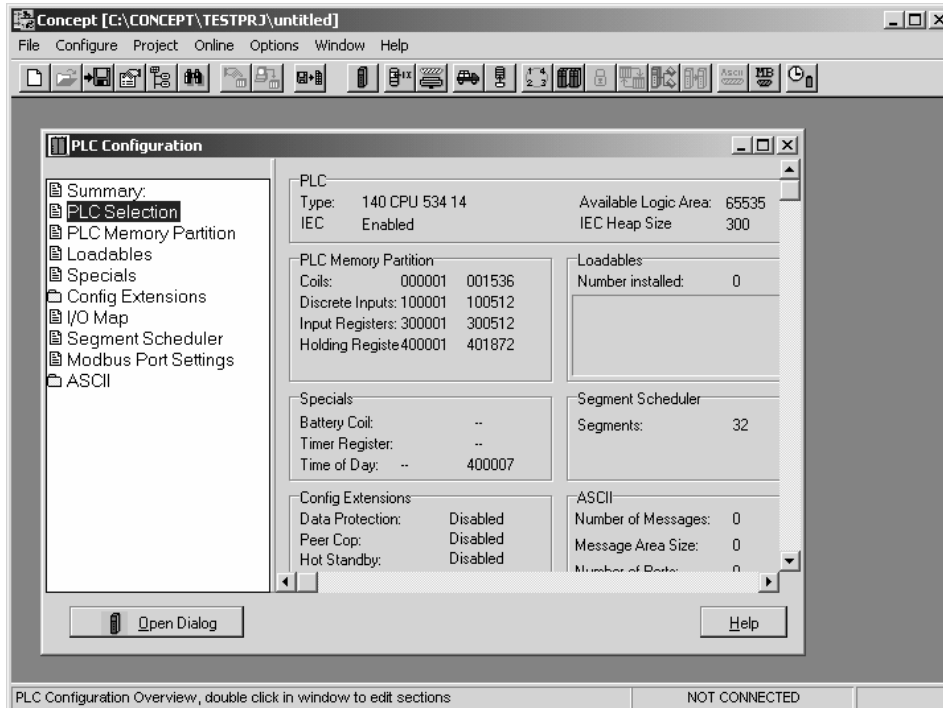
- 3 In the list of options on the left side of this dialog box, double-click the *PLC Selection* folder. This action opens the PLC Selection dialog box.



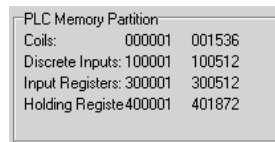
- 4 In the *CPU/Executive* pane, use the scroll bar to locate and select the PLC to configure.



- Click **OK**. This action opens the *PLC Configuration* dialog box, populated with the correct values for the PLC you selected.



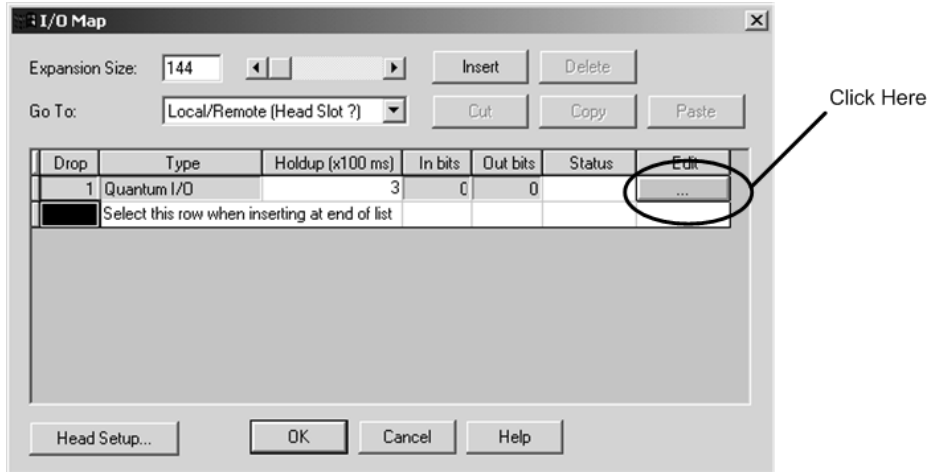
- Make a note of the holding registers for the module. You will need this information when you modify your application as outlined in the ProTalk application Reference Guides. The Holding Registers are displayed in the PLC Memory Partition pane of the PLC Configuration dialog box.



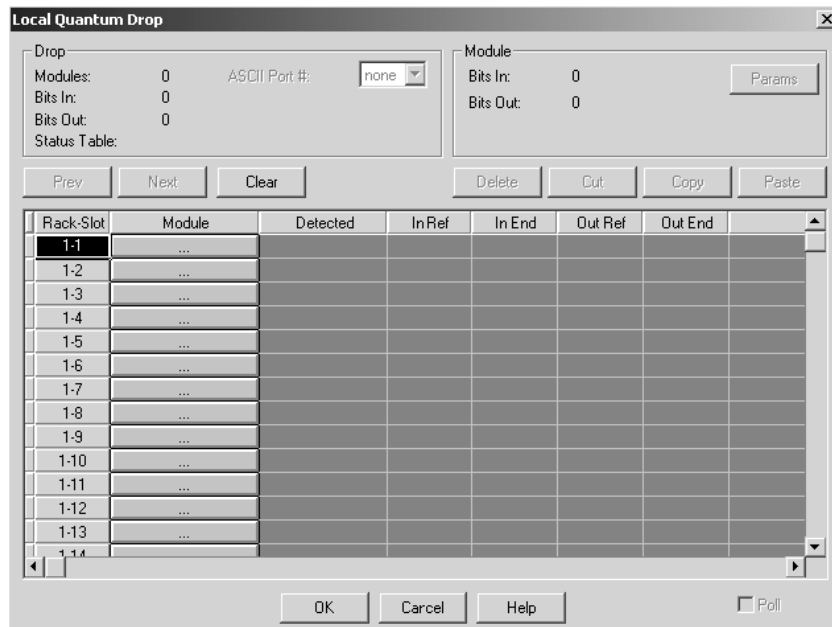
2.3 Add the PTQ Module to the Project

The next step is to add one or more of the PTQ modules to the Project. To add modules:

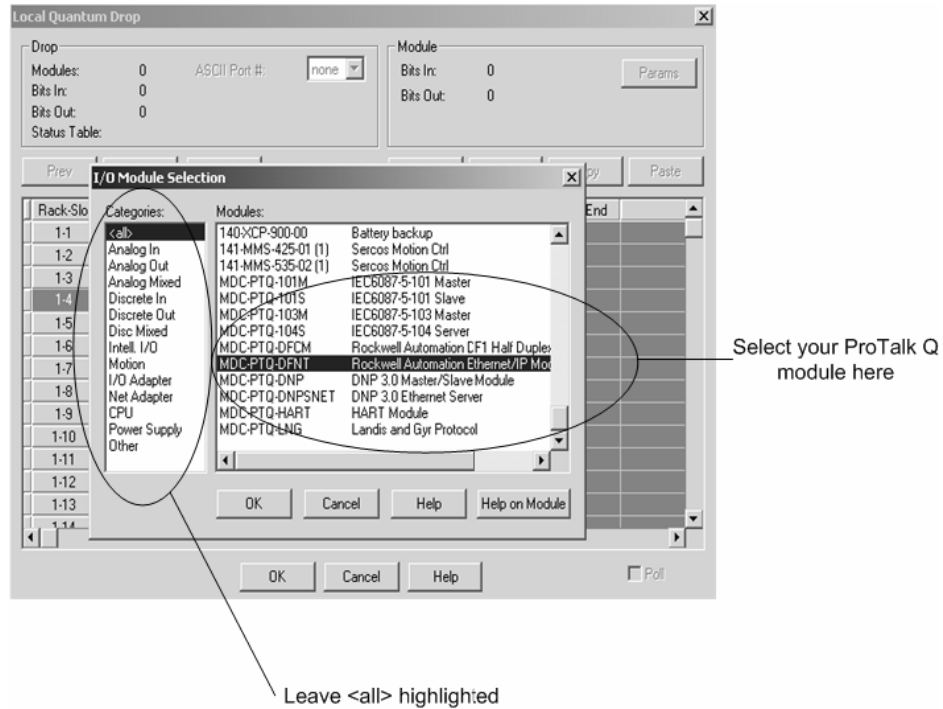
- 1 In the list of options on the left side of the *PLC Configuration* dialog box, double-click *I/O Map*. This action opens the *I/O Map* dialog box.



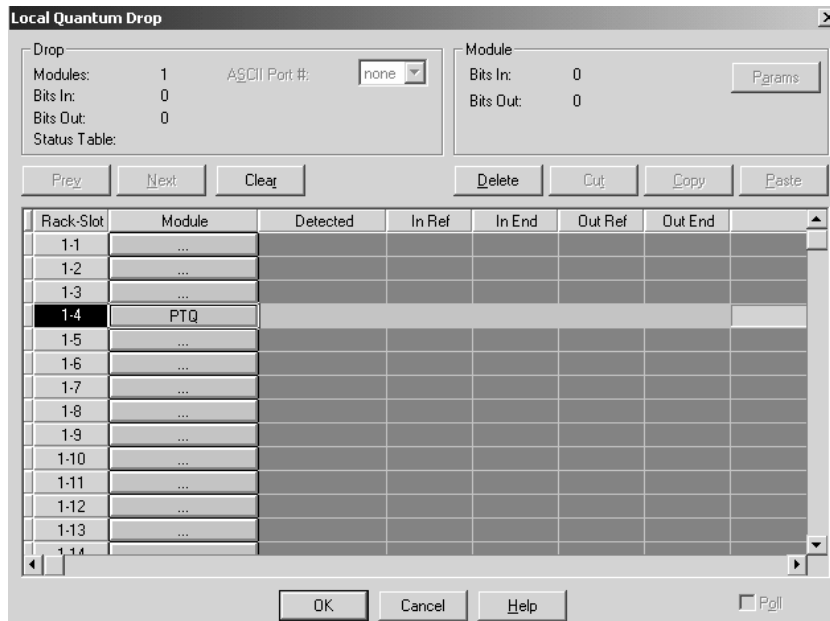
- 2 Click the **Edit** button to open the *Local Quantum Drop* dialog box. This dialog box is where you identify rack and slot locations.



- Click the Module button next to the rack/slot position where the ProTalk module will be installed. This action opens the I/O Module Selection dialog box.

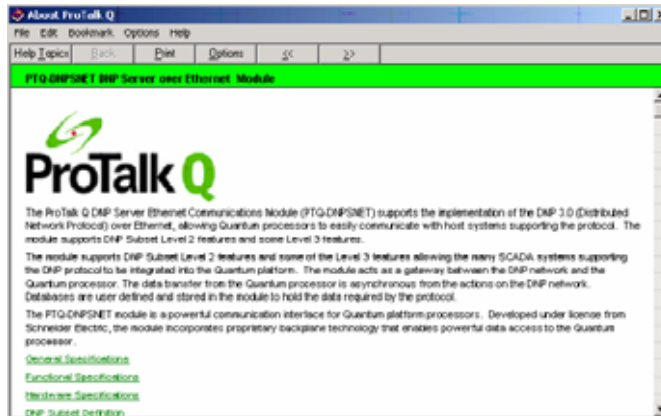


- In the Modules pane, use the scroll bar to locate and select the ProTalk module, and then click OK. This action copies the description of the ProTalk module next to the assigned rack and slot number of the Local Quantum Drop dialog box.



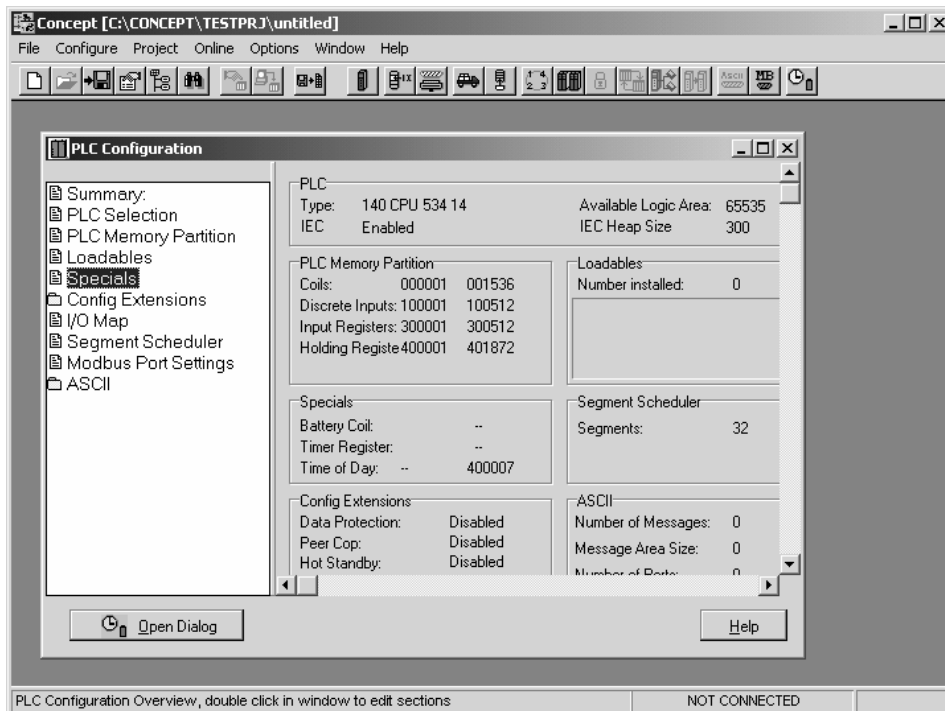
- Repeat steps 3 through 5 for each ProTalk module you plan to install. When you have finished installing your ProTalk modules, click OK to save your settings. Click Yes to confirm your settings.

Tip: Select a module, and then click the Help on Module button for help pages.

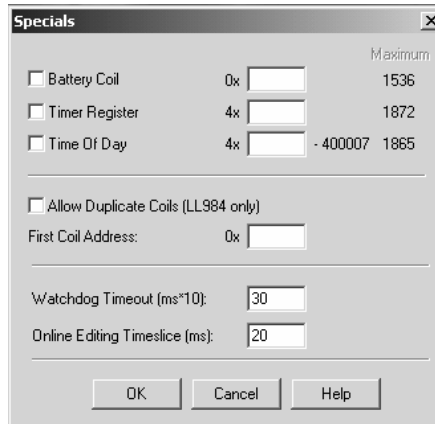


2.4 Set up Data Memory in Project

- In the list of options on the left side of the PLC Configuration dialog box, double-click Specials.

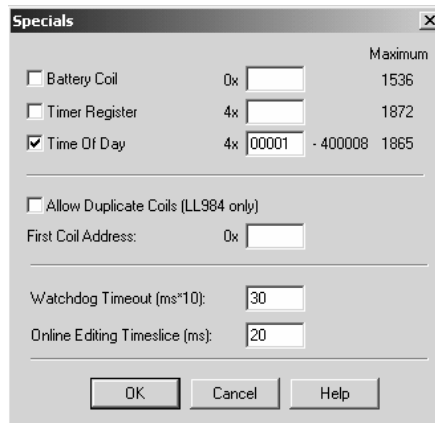


- This action opens the Specials dialog box.



Selecting the Time of Day

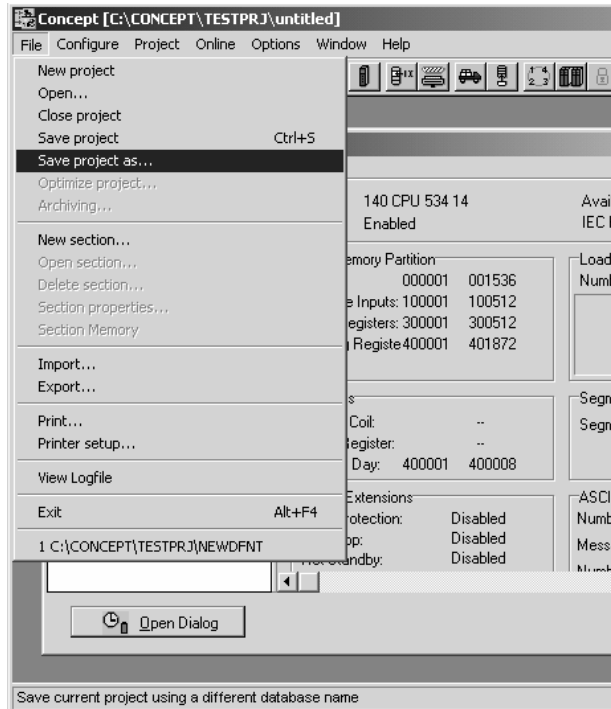
- Select (check) the Time of Day box, and then enter the value 00001 as shown in the following example. This value sets the first time of day register to 400001.



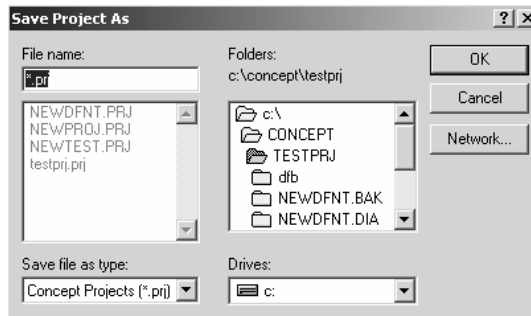
- Click OK to save your settings and close the Specials dialog box.

Saving your project

- 1 In the PLC Configuration dialog box, choose File / Save project as.



- 2 This action opens the Save Project as dialog box.



- 3 Name the project, and then click OK to save the project to a file.

2.5 Download the Project to the Processor

The next step is to download (copy) the project file to the Quantum Processor.

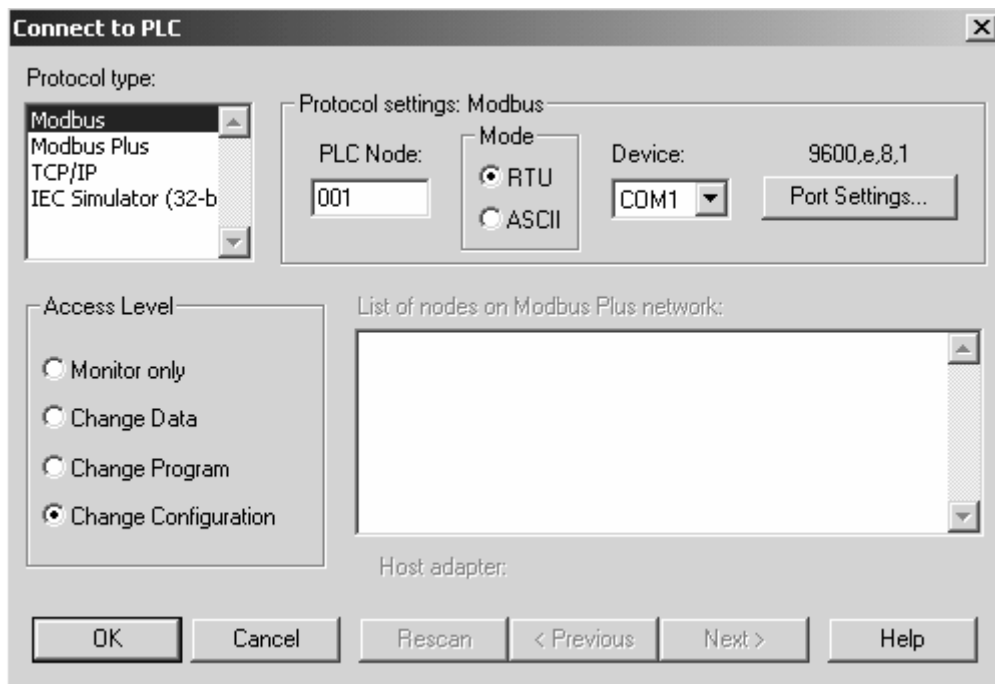
- 1 Use the null modem cable to connect your PC's serial port to the Quantum processor, as shown in the following illustration.



Note: You can use a Modbus Plus Network Option Module (NOM Module) module in place of the serial port if necessary.

- 2 Open the PLC menu, and then choose Connect.

- 3 In the PLC Configuration dialog box, open the Online menu, and then choose Connect. This action opens the Connect to PLC dialog box.



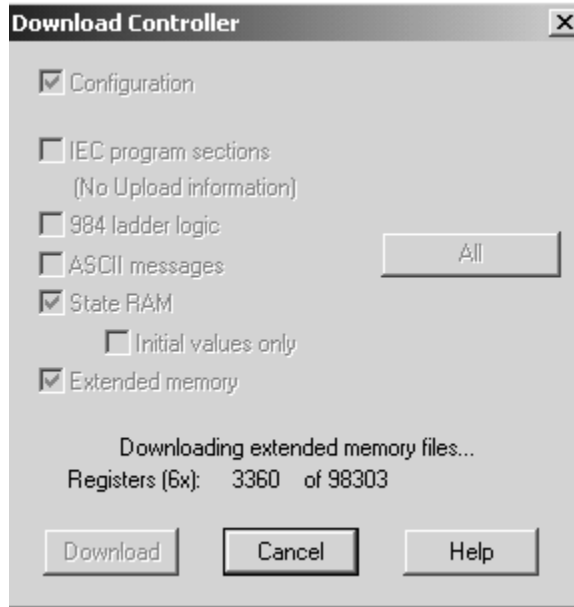
- 4 Leave the default settings as shown and click OK.

Note: Click OK to dismiss any message boxes that appear during the connection process.

- 5 In the PLC Configuration window, open the Online menu, and then choose Download. This action opens the Download Controller dialog box.



- Click all, and then click Download. If a message box appears indicating that the controller is running, click Yes to shut down the controller. The Download Controller dialog box displays the status of the download as shown in the following illustration.

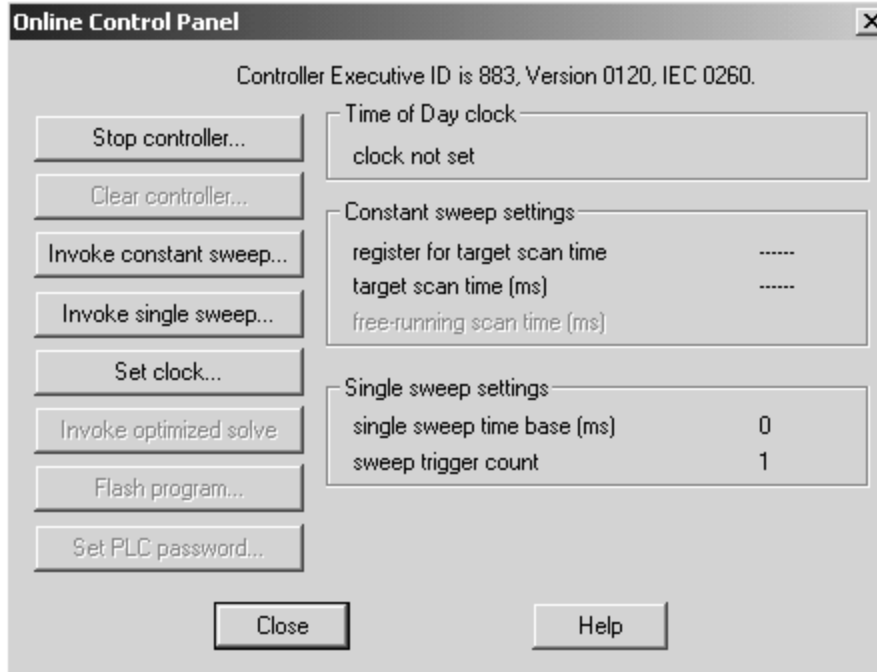


- When the download is complete, you will be prompted to restart the controller. Click Yes to restart the controller.

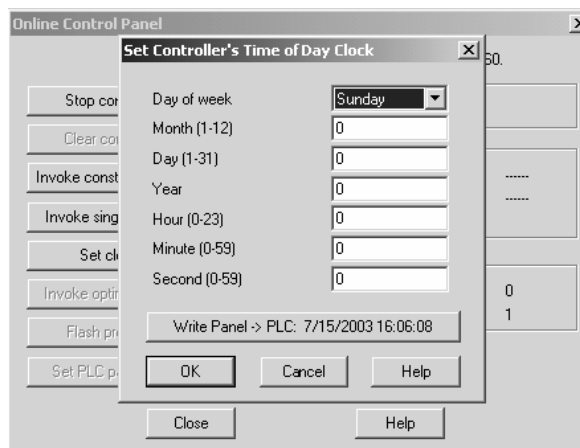
2.6 Verify Successful Download

The final step is to verify that the configuration changes you made were received successfully by the module, and to make some adjustments to your settings.

- 1 In the PLC Configuration window, open the Online menu, and then choose Online Control Panel. This action opens the Online Control Panel dialog box.

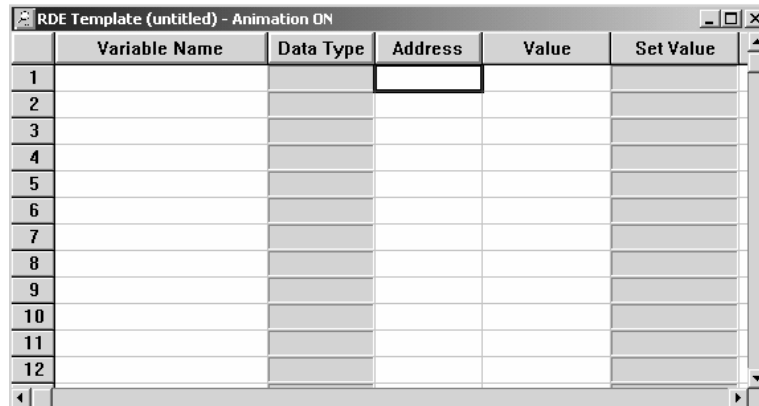


- 2 Click the Set Clock button to open the Set Controller's Time of Day Clock dialog box.



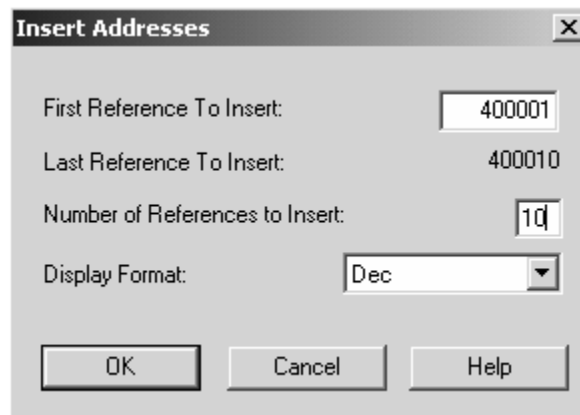
- 3 Click the Write Panel button. This action updates the date and time fields in this dialog box. Click OK to close this dialog box and return to the previous window.
- 4 Click Close to close the Online Control Panel dialog box.

- 5 In the PLC Configuration window, open the Online menu, and then choose Reference Data Editor. This action opens the Reference Data Editor dialog box. On this dialog box, you will add preset values to data registers that will later be monitored in the ProTalk module.
- 6 Place the cursor over the first address field, as shown in the following illustration.



	Variable Name	Data Type	Address	Value	Set Value
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					

- 7 In the PLC Configuration window, open the Templates menu, and then choose Insert addresses. This action opens the Insert addresses dialog box.
- 8 On the Insert addresses dialog box, enter the values shown in the following illustration, and then click OK.



Insert Addresses

First Reference To Insert: 400001

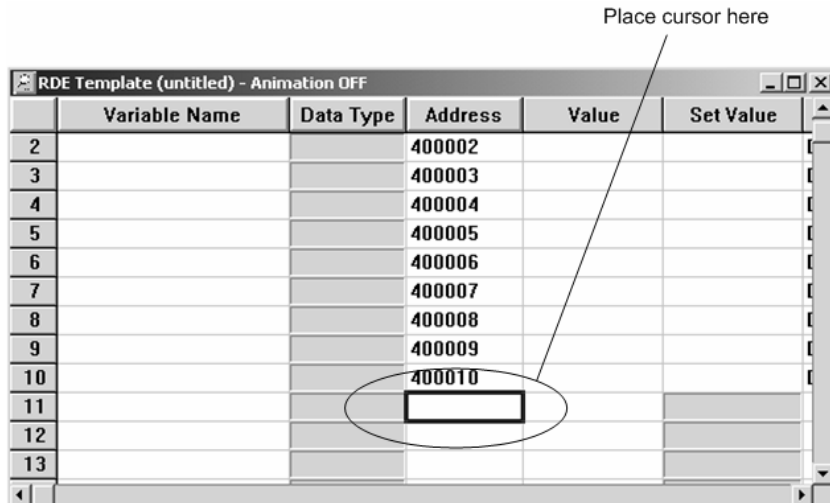
Last Reference To Insert: 400010

Number of References to Insert: 10

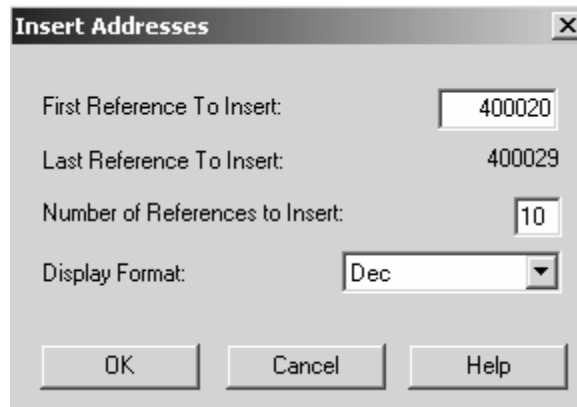
Display Format: Dec

OK Cancel Help

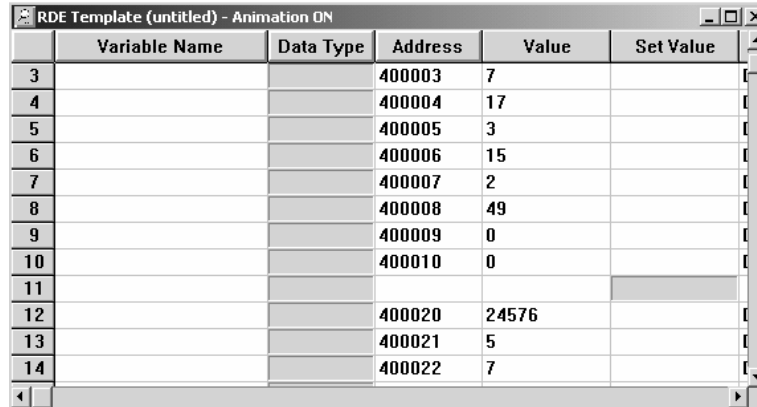
- 9 Notice that the template populates the address range, as shown in the following illustration. Place your cursor as shown in the first blank address field below the addresses you just entered.



- 10 Repeat steps 6 through 9, using the values in the following illustration:



- 11** In the PLC Configuration window, open the Online menu, and then choose animate. This action opens the RDE Template dialog box, with animated values in the Value field.



	Variable Name	Data Type	Address	Value	Set Value
3			400003	7	
4			400004	17	
5			400005	3	
6			400006	15	
7			400007	2	
8			400008	49	
9			400009	0	
10			400010	0	
11					
12			400020	24576	
13			400021	5	
14			400022	7	

- 12** Verify that values shown are cycling, starting from address 400065 on up.
- 13** In the PLC Configuration window, open the Templates menu, and then choose Save Template as. Name the template ptqclock, and then click OK to save the template.
- 14** In the PLC Configuration window, open the Online menu, and then choose Disconnect. At the disconnect message, click Yes to confirm your choice.

At this point, you have successfully

- Created and downloaded a Quantum project to the PLC
- Preset values in data registers that will later be monitored in the ProTalk module.

You are now ready to complete the installation and setup of the ProTalk module.

3 Configuring the Processor with ProWORX

When you use ProWORX 32 software to configure the processor, use the example SaF file provided on the ProTalk Solutions CD-ROM.

Important Note: Proworx software does not report whether the PTQ module is present in the rack, and therefore is not able to report the health status of the module when the module is online with the Quantum processor. Please take this into account when monitoring the status of the PTQ module.

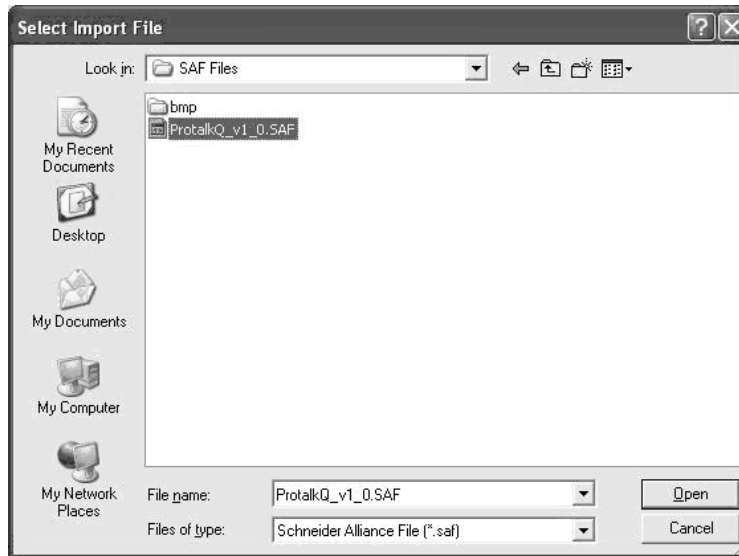
- 1 Run the Schneider_alliances.exe application that is installed with the Proworx 32 software:



- 2 Click on Import...



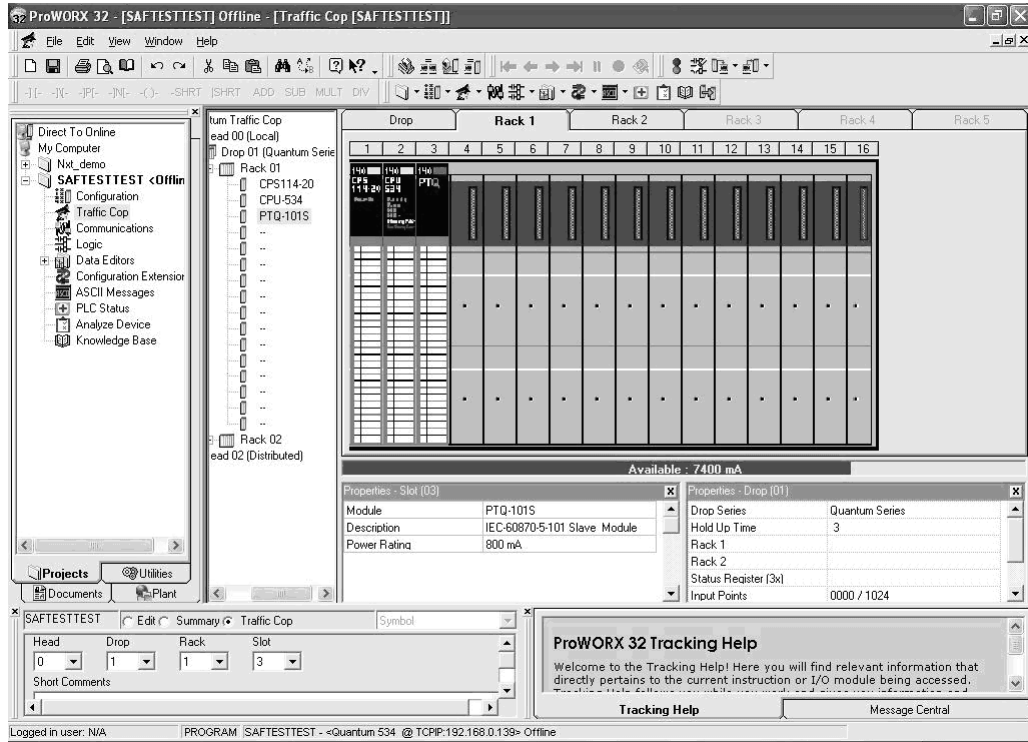
- 3 Select the .SaF File that is located at the CD-ROM shipped with the PTQ module.



- 4 After you click on Open you should see the PTQ modules imported (select I/O series as Quantum):



Now you can close the Schneider alliances application and run the Proworx 32 software. At the Traffic Cop section, select the PTQ module to be inserted at the slot:



4 Configuring the Processor with UnityPro XL

In This Chapter

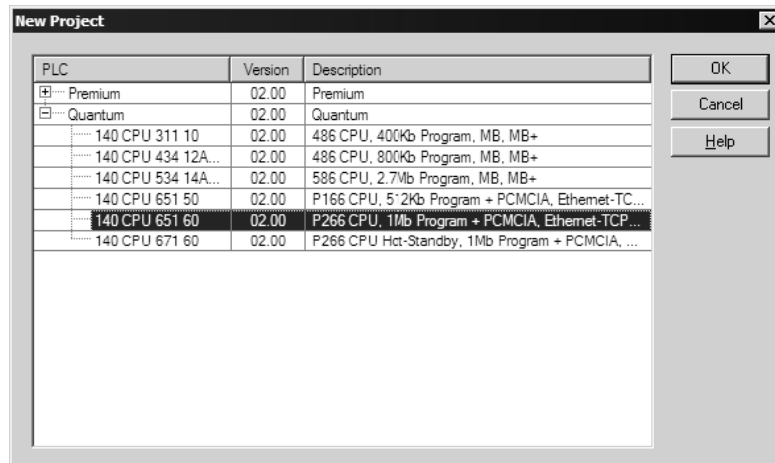
- ❖ Create a New Project 33
- ❖ Add the PTQ Module to the Project..... 35
- ❖ Build the Project 37
- ❖ Connect Your PC to the Processor 38
- ❖ Download the Project to the Processor 40

The following steps are designed to ensure that the processor (Quantum or Unity) is able to transfer data successfully with the PTQ module. As part of this procedure, you will use UnityPro XL to create a project, add the PTQ module to the project, set up data memory for the project, and then download the project to the processor.

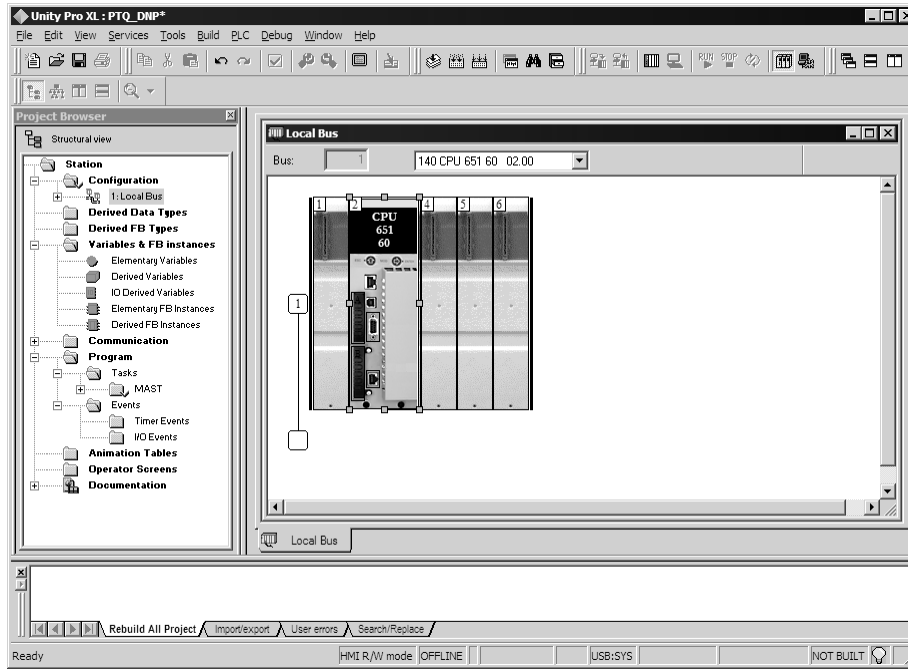
4.1 Create a New Project

The first step is to open UnityPro XL and create a new project.

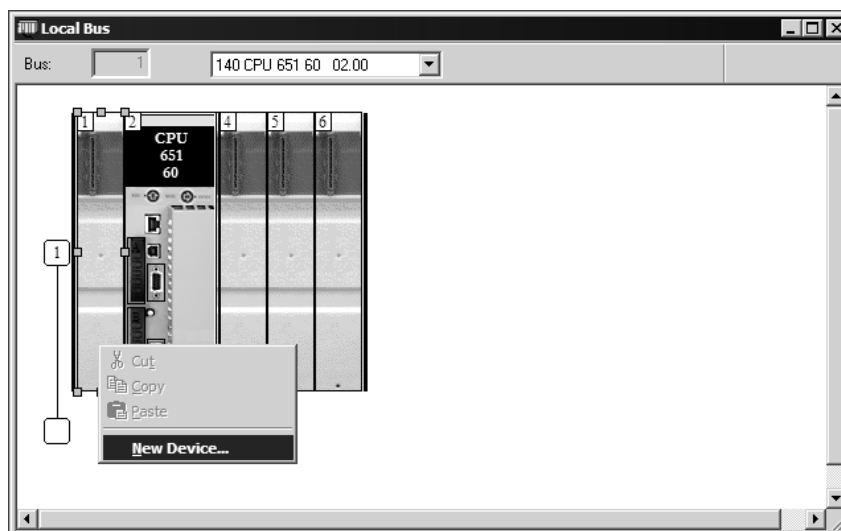
- 1 In the New Project dialog box, choose the CPU type. In the following illustration, the CPU is 140 CPU 651 60. Choose the processor type that matches your own hardware configuration, if it differs from the example. Click OK to continue.



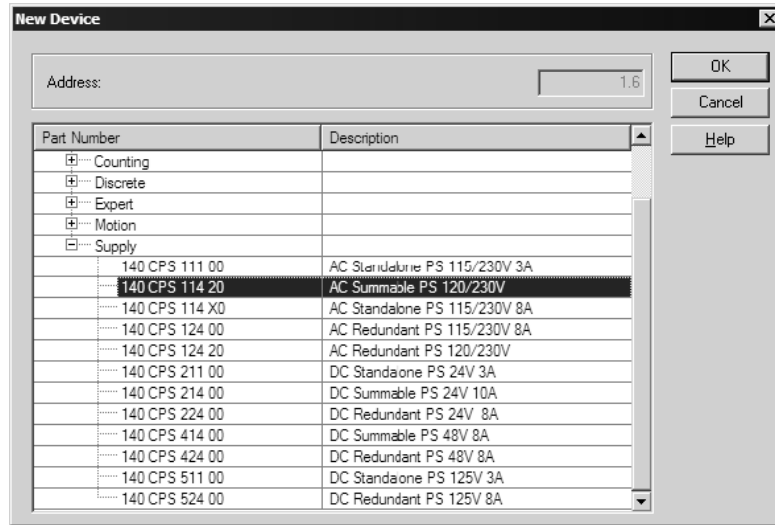
- The next step is to add a power supply to the project. In the Project Browser, expand the Configuration folder, and then double-click the 1:LocalBus icon. This action opens a graphical window showing the arrangement of devices in your Quantum rack.



- Select the rack position for the power supply, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose New Device..



- Expand the Supply folder, and then select your power supply from the list. Click OK to continue.

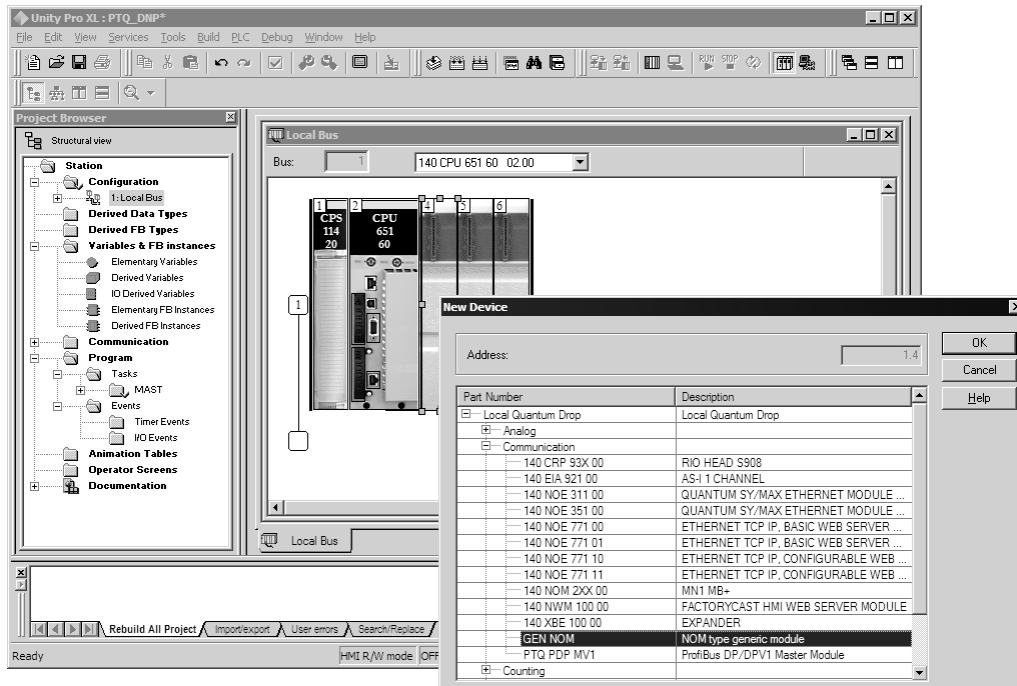


- Repeat these steps to add any additional devices to your Quantum Rack.

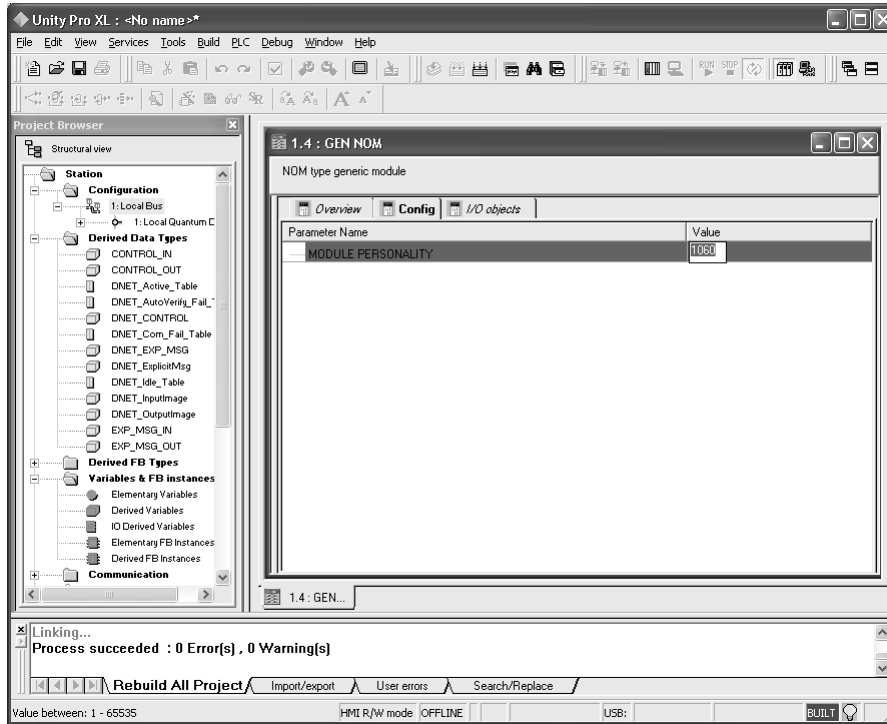
4.2 Add the PTQ Module to the Project

The next step is to add the PTQ module.

- Expand the Communication tree, and select GEN NOM. This module type provides extended communication capabilities for the Quantum system, and allows communication between the PLC and the PTQ module without requiring additional programming.



- Next, enter the module personality value. The correct value for ProTalk modules is 1060 decimal (0424 hex).



- Before you can save the project in UnityProXL, you must validate the modifications. Open the Edit menu, and then choose Validate. If no errors are reported, you can save the project.
- Save the project.

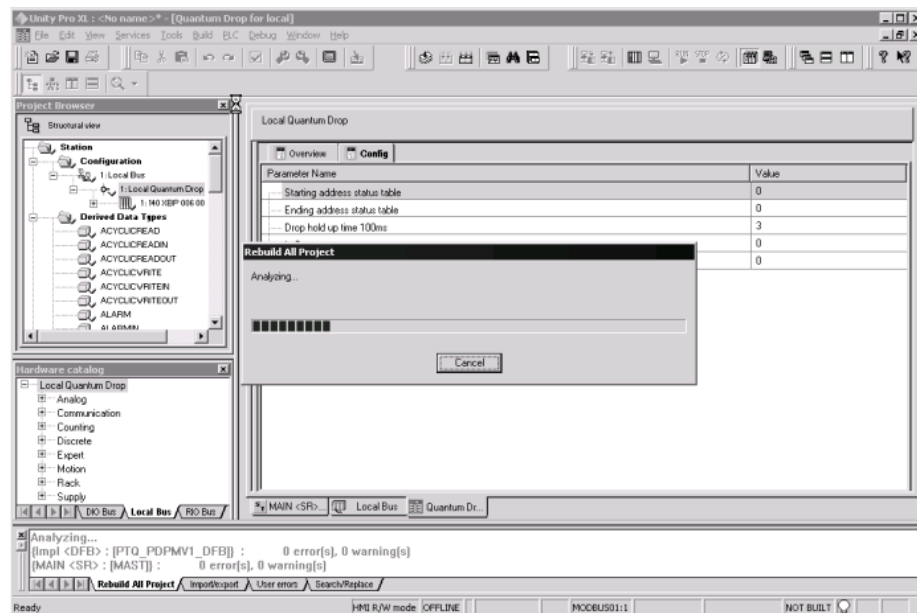
4.3 Build the Project

Whenever you update the configuration of your PTQ module or the processor, you must import the changed configuration from the module, and then build (compile) the project before downloading it to the processor.

Note: The following steps show you how to build the project in Unity Pro XL. This is not intended to provide detailed information on using Unity Pro XL, or debugging your programs. Refer to the documentation for your processor and for Unity Pro XL for specialized information.

To build (compile) the project:

- 1 Review the elements of the project in the Project Browser.
- 2 When you are satisfied that you are ready to download the project, open the Build menu, and then choose Rebuild all Project. This action builds (compiles) the project into a form that the processor can use to execute the instructions in the project file. This task may take several minutes, depending on the complexity of the project and the resources available on your PC.
- 3 As the project is built, Unity Pro XL reports its process in a Progress dialog box, with details appearing in a pane at the bottom of the window. The following illustration shows the build process under way.



After the build process is completed successfully, the next step is to download the compiled project to the processor.

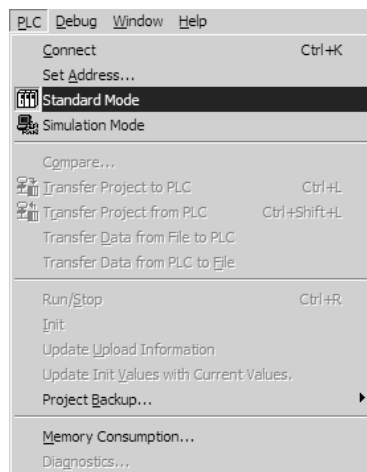
4.4 Connect Your PC to the Processor

The next step is to connect to the processor so that you can download the project file. The processor uses this project file to communicate over the backplane to modules identified in the project file.

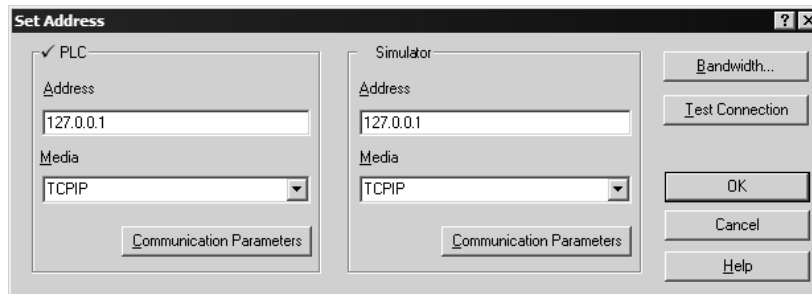
Note: If you have never connected from the PC to your processor before, you must verify that the necessary port drivers are installed and available to UnityPro XL.

To verify address and driver settings in UnityPro XL:

- 1 Open the PLC menu, and choose Standard Mode. This action turns off the PLC Simulator, and allows you to communicate directly with the Quantum or Unity hardware.



- 2 Open the PLC menu, and choose Set address... This action opens the Set address dialog box. Open the Media dropdown list and choose the connection type to use (TCPIP or USB).



- If the Media dropdown list does not contain the connection method you wish to use, click the Communication Parameters button in the PLC area of the dialog box. This action opens the PLC Communication Parameters dialog box.



- Click the Driver Settings button to open the SCHNEIDER Drivers management Properties dialog box.



- Click the Install/update button to specify the location of the Setup.exe file containing the drivers to use. You will need your UnityPro XL installation disks for this step.

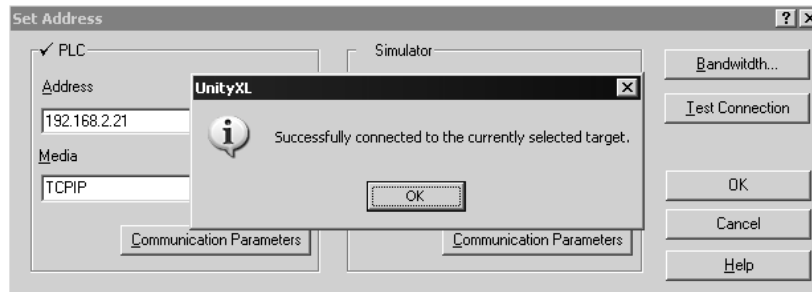


- Click the Browse button to locate the Setup.exe file to execute, and then execute the setup program. After the installation, restart your PC if you are prompted to do so. Refer to your Schneider Electric documentation for more information on installing drivers for UnityPro XL.

4.4.1 Connecting to the Processor with TCP/IP

The next step is to download (copy) the project file to the processor. The following steps demonstrate how to use an Ethernet cable connected from the Processor to your PC through an Ethernet hub or switch. Other connection methods may also be available, depending on the hardware configuration of your processor, and the communication drivers installed in UnityPro XL.

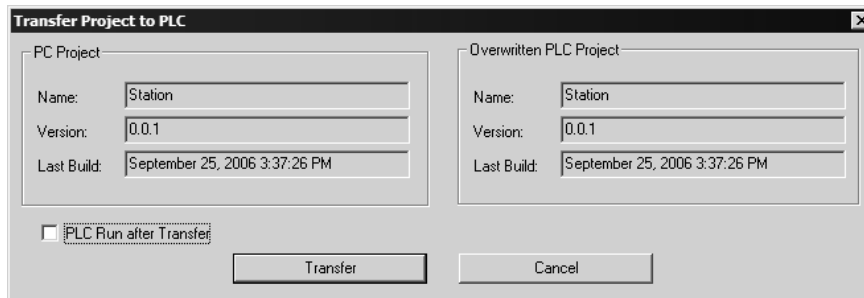
- 1 If you have not already done so, connect your PC and the processor to an Ethernet hub.
- 2 Open the PLC menu, and then choose Set address.
 - **Important:** Notice that the Set address dialog box is divided into two areas. Enter the address and media type in the PLC area of the dialog box, not the Simulator area.
- 3 Enter the IP address in the address field. In the Media dropdown list, choose TCP/IP.
- 4 Click the Test Connection button to verify that your settings are correct.



The next step is to download the Project to the Processor.

4.5 Download the Project to the Processor

- 1 Open the PLC menu and then choose Connect. This action opens a connection between the Unity Pro XL software and the processor, using the address and media type settings you configured in the previous step.
- 2 On the PLC menu, choose Transfer Project to PLC. This action opens the Transfer Project to PLC dialog box. If you would like the PLC to go to "Run" mode immediately after the transfer is complete, select (check) the PLC Run after Transfer after Transfer check box.



- 3 Click the Transfer button to download the project to the processor. As the project is transferred, Unity Pro XL reports its process in a Progress dialog box, with details appearing in a pane at the bottom of the window.

When the transfer is complete, place the processor in Run mode.

5 Setting Up the ProTalk Module

In This Chapter

- ❖ Install the ProTalk Module in the Quantum Rack 41
- ❖ Connect the PC to the ProTalk Configuration/Debug Port 43
- ❖ Cable Connections 44
- ❖ Collision Avoidance (DNP modules only) 48

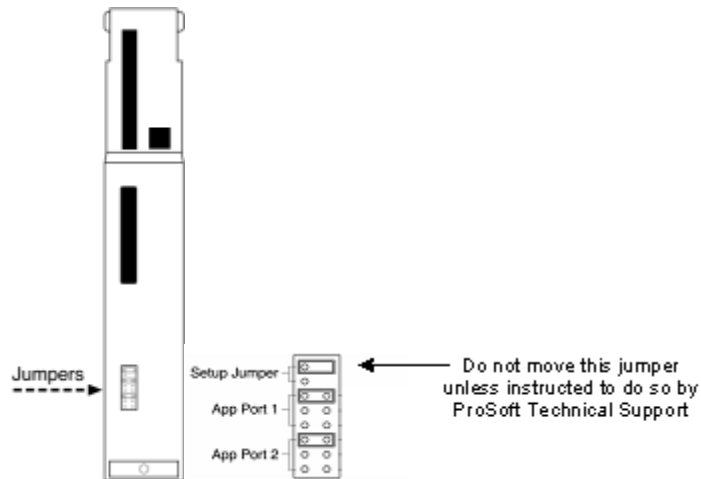
After you complete the following procedures, the ProTalk module will actively be transferring data bi-directionally with the processor.

5.1 Install the ProTalk Module in the Quantum Rack

5.1.1 Verify Jumper Settings

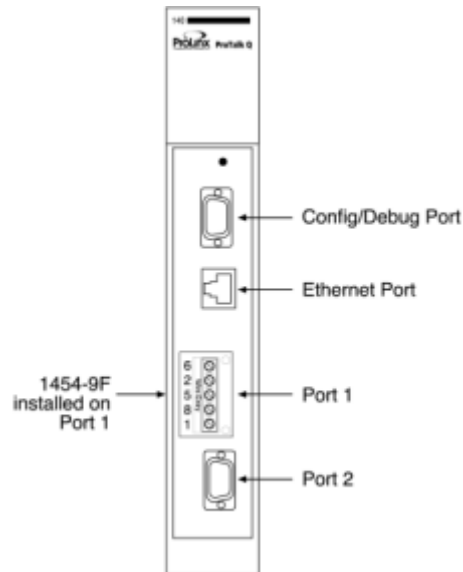
ProTalk modules are configured for RS-232 serial communications by default. To use RS-422 or RS-485, you must change the jumpers.

The jumpers are located on the back of the module as shown in the following illustration:



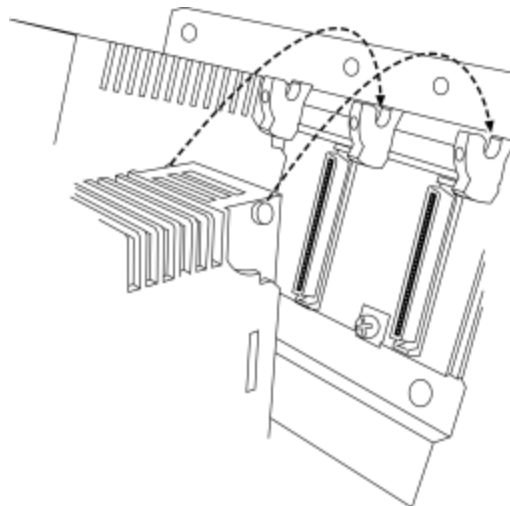
5.1.2 Inserting the 1454-9F connector

Insert the 1454-9F connector as shown. Wiring locations are shown in the table:

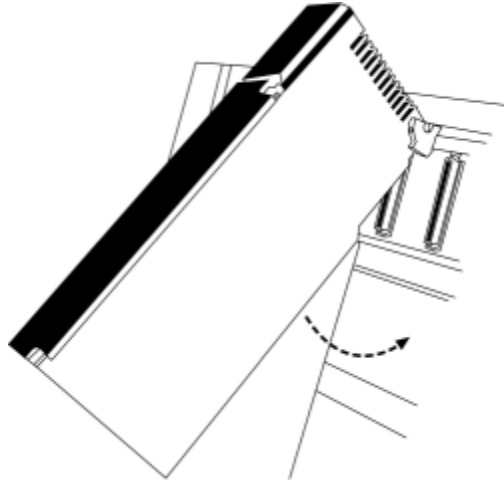


5.1.3 Install the ProTalk Module in the Quantum Rack

- 1 Place the Module in the Quantum Rack. The ProTalk module must be placed in the same rack as the processor.
- 2 Tilt the module at a 45° angle and align the pegs at the top of the module with slots on the backplane.



- 3 Push the module into place until it seats firmly in the backplane.

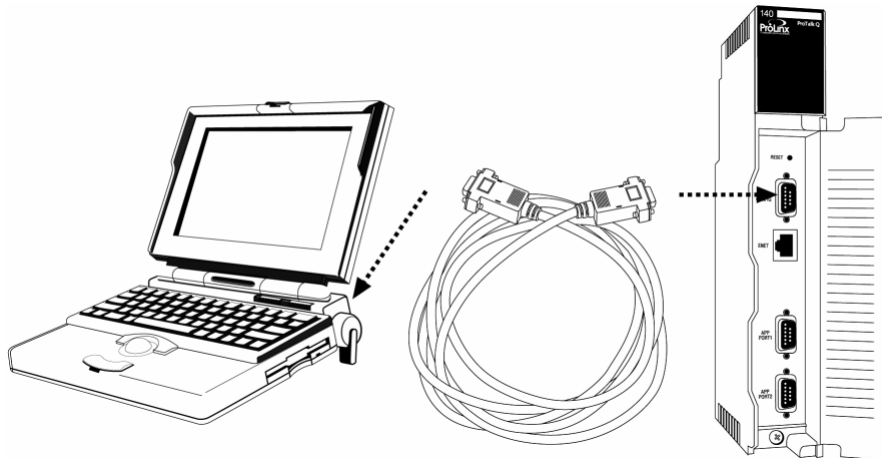


CaUTION: The PTQ module is hot-swappable, meaning that you can install and remove it while the rack is powered up. You should not assume that this is the case for all types of modules unless the user manual for the product explicitly states that the module is hot-swappable. Failure to observe this precaution could result in damage to the module and any equipment connected to it.

5.2 Connect the PC to the ProTalk Configuration/Debug Port

Make sure you have exited the Quantum programming software before performing these steps. This action will avoid serial port conflict.

- 1 Using the supplied Null Modem cable, connect your PC or Laptop to the Configuration/Debug port on the ProTalk module as shown



- 2 Click the Windows Start button, then choose Programs / accessories / Communications / HyperTerminal.

- 3 In the HyperTerminal window, enter a connection name, for example **Test**, and then click OK. This action opens the Connect To dialog box.



- 4 In the Connect Using field, ensure that the com port matches the port on your PC to which you connected the Null Modem cable, and then click OK. This action opens the COMx Properties dialog box.



- 5 Verify that the settings match those shown in the example above, and then click OK. If your port settings are configured correctly, you will return to the HyperTerminal window.
- 6 In the HyperTerminal window, press [?]. This action opens the module's Configuration/Debug menu.

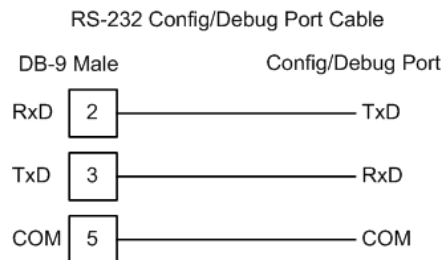
5.3 Cable Connections

The application ports on the PTQ-DNP module support RS-232, RS-422, and RS-485 interfaces. Please inspect the module to ensure that the jumpers are set correctly to correspond with the type of interface you are using.

Note: When using RS-232 with radio modem applications, some radios or modems require hardware handshaking (control and monitoring of modem signal lines). Enable this in the configuration of the module by setting the UseCTS parameter to 1.

5.3.1 RS-232 Configuration/Debug Port

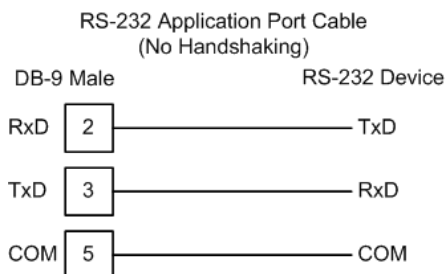
This port is physically a DB-9 connection. This port permits a PC based terminal emulation program to view configuration and status data in the module and to control the module. The cable for communications on this port is shown in the following diagram:



The Ethernet port on this module (if present) is inactive.

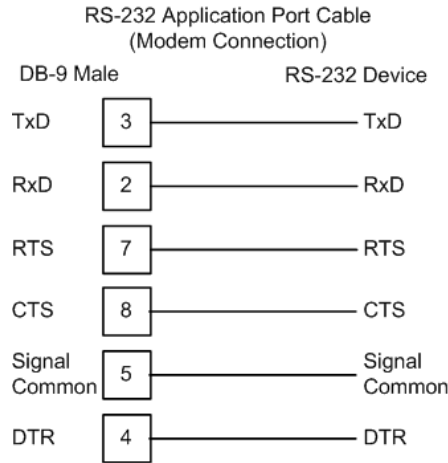
5.3.2 RS-232

When the RS-232 interface is selected, the use of hardware handshaking (control and monitoring of modem signal lines) is user definable. If no hardware handshaking will be used, the cable to connect to the port is as shown below:



RS-232: Modem Connection

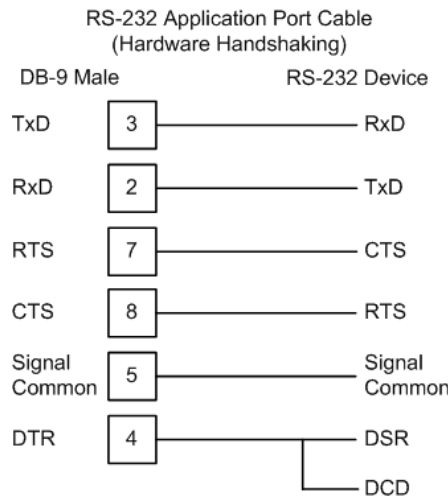
This type of connection is required between the module and a modem or other communication device.



The "Use CTS Line" parameter for the port configuration should be set to 'Y' for most modem applications.

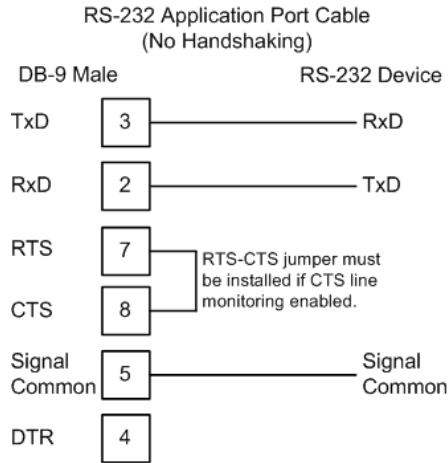
RS-232: Null Modem Connection (Hardware Handshaking)

This type of connection is used when the device connected to the module requires hardware handshaking (control and monitoring of modem signal lines).



RS-232: Null Modem Connection (No Hardware Handshaking)

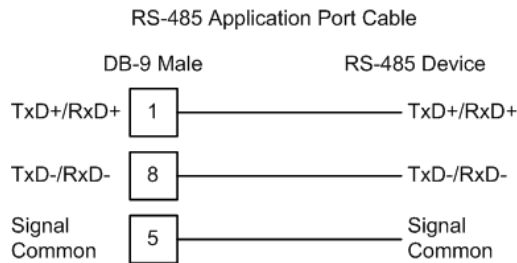
This type of connection can be used to connect the module to a computer or field device communication port.



Note: If the port is configured with the "Use CTS Line" set to 'Y', then a jumper is required between the RTS and the CTS line on the module connection.

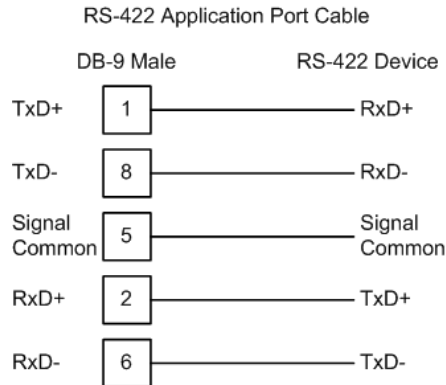
5.3.3 RS-485

The RS-485 interface requires a single two or three wire cable. The Common connection is optional and dependent on the RS-485 network. The cable required for this interface is shown below:



Note: Terminating resistors are generally not required on the RS-485 network, unless you are experiencing communication problems that can be attributed to signal echoes or reflections. In this case, install a 120 ohm terminating resistor on the RS-485 line.

5.3.4 RS-422

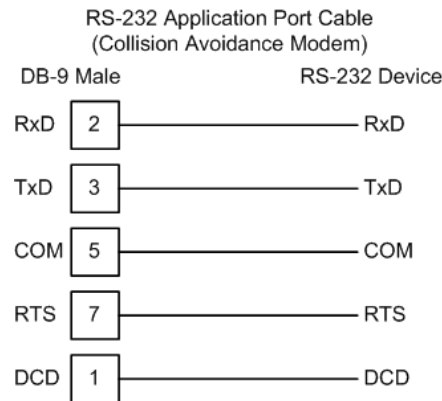


RS-485 and RS-422 Tip

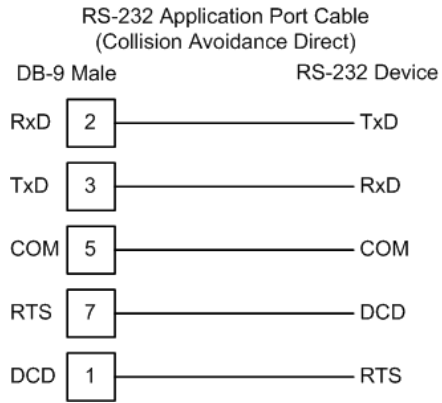
If communication in the RS-422/RS-485 mode does not work at first, despite all attempts, try switching termination polarities. Some manufacturers interpret +/- and A/B polarities differently.

5.4 Collision Avoidance (DNP modules only)

The RTS line is controlled by the RTS on and off parameters set for the port. If the CTS line is used (usually only required for half-duplex modems and not defined for use in the DNPS specification), the RTS and CTS lines must either be connected together or connected to the modem. The following illustration shows the cable required when connecting the port to a modem.



If collision avoidance is used in a point-to-point connection on the RS-232 interface, the following cable should be used.



6 Configuring the Module

In This Chapter

- ❖ Obtain the Sample Configuration Files 51
- ❖ Edit the Configuration File 51
- ❖ Uploading and Downloading the Configuration File..... 104
- ❖ Verification and Troubleshooting 110

6.1 Obtain the Sample Configuration Files

The ProSoft Solutions CD is organized in folders by module name. In the folder for the module you are using, you will find sample configuration files and other information.

- 1 Use Windows Explorer to locate the sample configuration files for your PTQ module on the PTQ CD.
- 2 When you have located the correct configuration files, use the Copy and Paste commands to move the files to a location on your PC's hard drive. We recommend C:\temp.
- 3 Files copied from a CD-ROM are read-only. The next step is to make the files writable. Navigate to the directory where you copied the files, then select the files and click the right mouse button to open a shortcut menu. On the shortcut menu, select Properties, and clear (uncheck) the Read Only check box.
- 4 The next step is to open the configuration files in a text editor such as Notepad, which comes with Windows. To start Notepad, click the Start button, and then choose **Programs / accessories / Notepad**.
- 5 When Notepad starts, open the File menu, and then choose **Open**. Navigate to the folder where you copied the configuration file on your PC and select the file. Click **Open**. The configuration file will open in Notepad, ready for editing.

Note: We do not recommend opening the configuration file in a word processor such as Microsoft Word, because the file may be saved in a format that cannot be read by the module.

6.2 Edit the Configuration File

Note: It is important that you plan your configuration before modifying the configuration files. The remainder of this step provides the information you must make the appropriate modifications to the configuration files.

Important: This module supports a maximum configuration file size of 128 kilobytes (131072 bytes). If the configuration file is larger than this size, the module will not accept the download. You can reduce the size of the configuration file by opening the file in a text editor and removing comment lines (lines preceded with the # character).

The DNP.CFG file has four main sections:

- [Module]
- [Backplane Data Exchange]
- [DNP Slave]
 - [DNP Slave Database]
 - [DNP Slave Binary Inputs]
 - [DNP Slave Analog Inputs]
 - [DNP Slave Float Inputs]
 - [DNP Slave Double Inputs]
 - [Secondary Port]
- [DNP Master]
 - [IED Database]
 - [DNP Master Slave List]
 - [DNP Master Commands]

Important notes to consider when editing the sample configuration file:

- Comments within the file are preceded by the pound (#) sign. Any text on a line that occurs after the # character will be ignored.
- Do not use tabs or other non-printing characters instead of spaces to separate parameters (spacebar).
- Parameter names must begin in the first column of a line, and may not be preceded with a space (spacebar) or other non-printing character.

6.2.1 [Module]

This section provides the module with a unique name, identifies the method of failure for the communications for the module if the PLC is not in run, and describes how to initialize the module upon startup.

The following example shows a sample [Module] section:

```
[Module]
Module Name: PTQ-DNP Module
```

Modify each of the parameters based on the needs of your application.

Module Name

0 to 80 characters

This parameter assigns a name to the module that can be viewed using the configuration/debug port. Use this parameter to identify the module and the configuration file.

6.2.2 [Backplane Data Exchange]

The [Backplane Data Exchange] section of the configuration file transfers to transfer information to and from the module. One command from the list will execute during each I/O service interval at the end of the PLC ladder logic evaluation.

Note that the PTQ module is not able to schedule the execution of the data exchange. The programmer normally performs scheduling in the PLC environment, however scheduling is not included in the PTQ module in order to allow the commands to run as fast as possible to maintain the synchronization of the two databases. For example, example if your configuration contains 10 "Backplane Data Exchange" commands, it will require 10 PLC scans to process the entire list.

The following is an *example* of a typical [Backplane Data Exchange] section:

```
[Backplane Data Exchange]
# Data Start PT      Point      Point      Word
# Type  Address      Type       Address     Count
START
  101      0          4          101        10 #Move DNP BI data to module
  102      0          4          111        10 #Move DNP BO data to quantum
  103      0          4          121        20 #Move DNP Cntr data to module
  104      0          4          201        10 #Move DNP AI data to module
  105      0          4          301        10 #Move DNP AO data to quantum
  106      0          4          401        20 #Move DNP FI data to module
  108      0          4          501         8 #Move DNP FO data to quantum
  201      0          4          601        10 #Move IED BI data to quantum
  202      0          4          611        10 #Move IED BO data to module
  203      0          4          621        20 #Move IED Cntr data to quantum
  204      0          4          650        50 #Move IED AI data to quantum
  205      0          4          700         8 #Move IED AO data to module
    3      0          4          801        64 #command control data area
END
```

This section may contain up to 100 individual commands used in any combination to transfer data to/from the processor. The following topics provide information on the use of the commands as well as a simple example.

Setting up the Commands

The commands take five parameters:

- *Data Type:* (See Data Type table)
- *Start PT Address:* The destination for the data retrieved from the processor.
- *Point Type:* The type of register within the processor.
 - Quantum addresses:(0:x = 0, 10:x = 1 30:x = 3 or 40:x = 4)
 - Unity addresses (0=%M, 1=%I, 3=%IW, and 4=%MW)
- *Point Address:* The source of the data within the processor or the module.
- *Word Count:* The number of words to copy. The length of this copy may be any length of 1 to 130 inclusive. If your application requires the movement of additional data, you may enter additional commands.

Example 0:x or 10:x Register Transfer

The transfer of Coils and Input bits require some forethought as the command transfers **words** and not bits. This means that if you want to transfer bits 000005 to 000007 from the processor to word 21 in the module you would have to transfer the **word** within the processor containing bits 000001 to 000016 to a word within the module's memory.

Note: When you are planning the application, carefully consider the transfer of bits so as to optimize the usage of available bits, and to preserve the integrity of your information.

Data Types Table

Data Type Code	Description
Data Moves From processor to Module	
101	Move DNP Binary Input Data to Module
103	Move DNP Counter Data to Module
104	Move DNP Analog Input Data to Module
106	Move DNP Floating Point Input Data to Module
202	Move IED Binary Output Data to Module
205	Move IED Analog Output Data to Module
Data Moves from Module to processor	
102	Move DNP Binary Output Data to processor
105	Move DNP Analog Output Data to processor
108	Move DNP Floating Point Output Data to processor
201	Move IED Binary Input Data to processor
203	Move IED Counter Data to processor
204	Move IED Analog Input Data to processor

You may be asked to provide access to 10 words (160 bits) of Binary Input information for other devices on the network. This information resides in the PLC at addresses 400101 to 400110 (Quantum) or %MW101 to %MW110 (Unity), and must be available from the module's Binary Input database inside the module. This would require the use of a single command to move the data.

The highlighted command in the following section of code shows an example of how to accomplish this:

```
# This section is used by the ProTalk module to define the data transferred
# between the module and processor.
#
# Data Type      --> 101=DNP BI, 102=DNP BO, 103=DNP Cntr, 104=DNP AI,
#                  105=DNP AO, 106=DNP FI, 107=RESERVED,
#                  108=DNP FO, 109= RESERVED,
#                  201=IED BI, 202=IED BO, 203=IED Cntr, 204=IED AI,
#                  205=IED AO,
#                  2=Read from Status Database  3=Command Control
#
# Start Point Address --> Index of first point in data type
#
# Point Type      --> 0=0x, 1=1x, 3=3x,  and 4=4x (Quantum data type)
#                  0=%M, 1=%I, 3=%IW, and 4=%MW (Unity data type)
```

```

#
# Point Address --> point address (1 based)(0x and 1x must be at start of word
                    (i.e., 1, 17, 33, ...))
# Word Count     --> number of words to transfer (1 to 130)

[Backplane Data Exchange]
# Data Start PT      Point      Point      Word
# Type  Address     Type       Address    Count
START
    101      0        4         101      10
#Move DNP BI data to module
END

```

Field	Value	Meaning
Data Type	101	The type of operation to perform Transfer Binary Input information to the module
Start PTQ Address	0	The destination address within the Binary Input database within the PTQ
Point Type	4	The range of registers to read from the processor 4=4x (Quantum data type) 4=%MW (Unity data type)
Point Address	101	The starting address of the data within the processor This would be Point Type + offset Example: 400000 + 101 = 400101 (Quantum) %MW0 + 101 = %MW101 (Unity)
Word Count	10	The number of registers to transfer

Please note that with the exception of the Function 3 commands all [Backplane Data Exchange] functions work in a similar fashion. The only real difference between the commands is the type of data and the direction of transfer. When you understand the basic principle of the transfer command, it becomes relatively easy to add new commands. The following table shows the various types of data and the associated direction of transfer.

Data Block Capacities

The following table shows the maximum database capacity for each supported data type. The block codes identify the data type. Each command can move a maximum of 130 words. If additional data is required, you must enter additional commands in the [Backplane Data Exchange] section of the configuration file.

Data Type	Block Code	Max Pnts/Blk	Max # of Points
Digital Input	101	2080	8000
Digital Output	102	2080	2000
Counters	103	65	250
Analog Input	104	130	500
Analog Output	105	130	500
IED Digital Input	201	2080	8000
IED Digital Output	202	2080	2000
IED Counters	203	65	250

Data Type	Block Code	Max Pnts/Blk	Max # of Points
IED Analog Input	204	130	500
IED Analog Output	205	130	500
DNP Float Input	106	65	250
DNP Float Output	108	65	250

Note: Refer to the specifications for more information on the maximum number of points supported by the module.

Defining Special Data Transfer Functions

Your application may perform what might be considered a special function such as setting/retrieving the time and date or issuing an event to the module. This section will discuss the requirements for the command and offer an example of how it could be used.

Assuming that you have chosen registers 400801 to 400864 (Quantum) or %MW801 to %MW864 (Unity) as the target for your Command Function 3, you could enter the following command into the Backplane Data Exchange section of your configuration file.

```
[Backplane Data Exchange]
# Data Start PT      Point      Point      Word
# Type  Address      Type      Address     Count
START
    3          0          4          801        64    #command control data area
END
```

The fourth command states:

Field	Value	Meaning
Data Type	3	The type of operation to perform 3 = Read/Write special function to the processor
Start PT Address	0	This is ALWAYS 0 and will not overwrite your database.
Point Type	4	The range of registers to read from the processor 4=4x (Quantum data type) 4=%MW (Unity data type)
Point Address	801	The starting address of the data within the processor This would be Point Type + processor Address Example: 40000 + 801 = 40801 (%MW801)
Word Count	64	This is ALWAYS 64 words in length.

Note: This command requires two PLC scans to complete. When you issue a Function 3 we will examine the "Processor Address" registers, process the information, clear the registers and post the status if applicable.

The following table shows additional functionality that may be implemented using the Function 3 services.

Special Control Block Codes

Code	Function/Description
9901	CROB Control Block for Digital Outputs
9902	Command Control Block (Add command to Command List Queue)
9903	Pass events to processor
9958	PLC Binary Input Event data
9959	PLC Analog Input Event Data
9970	Set processor time using module's DNP time
9971	Set module's time using processor time
9998	Warm Boot Request from processor (Block contains no data)
9999	Cold Boot Request from processor (Block contains no data)

Special Control Block Codes Example

For example, if the following backplane command lists are configured:

```
[Backplane Data Exchange]
# Data Start PT Point Point Word
# Type Address Type Address Count
START
101 0 4 101 10 #Move DNP BI data to module
102 0 4 201 10 #Move DNP BO data to quantum
103 0 4 301 40 #Move DNP Cntr data to module
104 0 4 341 50 #Move DNP AI data to module
105 0 4 401 28 #Move DNP AO data to quantum
106 0 4 501 10 #Move DNP FI data to module
108 0 4 511 8 #Move DNP FO data to quantum
201 0 4 601 10 #Move IED BI data to quantum
202 0 4 611 3 #Move IED BO data to module
203 0 4 614 20 #Move IED Cntr data to quantum
204 0 4 650 50 #Move IED AI data to quantum
205 0 4 700 8 #Move IED AO data to module
3 0 4 801 64 #command control data area
END
```

This means that Binary Input data will occupy 10 words (160 points) as follows:

Quantum Address	Unity Address	PTQ-DNP
400101	%MW101 →	Binary Input Word 0 (contains 16 BI points)
400102	%MW102 →	Binary Input Word 1 (contains 16 BI points)
400103	%MW103 →	Binary Input Word 2 (contains 16 BI points)
400104	%MW104 →	Binary Input Word 3 (contains 16 BI points)
400105	%MW105 →	Binary Input Word 4 (contains 16 BI points)
400106	%MW106 →	Binary Input Word 5 (contains 16 BI points)
400107	%MW107 →	Binary Input Word 6 (contains 16 BI points)
400108	%MW108 →	Binary Input Word 7 (contains 16 BI points)
400109	%MW109 →	Binary Input Word 8 (contains 16 BI points)
400110	%MW110 →	Binary Input Word 9 (contains 16 BI points)

For example, if these Quantum addresses have the following values:

Address	Value
400101	1
400102	2
400103	3
400104	4
400105	5
400106	6
400107	7
400108	8
400109	9
400110	10

The debug port would show the current values in the PTQ-DNP database for the Binary Input points (click on U>Show DNP Databases → 1-Binary Inputs):

```

P=Previous Page
+=Skip 5 Pages
N=Next Page
D=Word Decimal Display
H=Word Hexadecimal Display
L=Double Word Decimal Display
X=Double Word Hexadecimal Display
F=Float Display
E=Double Float Display (only for double databases)
A=ASCII Display
1=DNP Binary Inputs      2=DNP Binary Outputs
3=DNP Counters          4=DNP Analog Inputs
5=DNP Analog Outputs    6=DNP Frozen Counters
7=DNP Float Inputs      8=DNP Double Inputs
9=DNP Float Outputs     0=DNP Double Outputs
B=IED Binary Inputs     C=IED Binary Outputs
G=IED Counters          I=IED Analog Inputs
J=IED Analog Outputs
M=Main Menu
DNP BINARY INPUT DATABASE DISPLAY 0 TO 159 (DECIMAL)

      1      2      3      4      5      6      7      8      9      10
    
```

Defining Error/Status Functions - Function 2

Your application may require certain status and error data to copy from the module to the processor. In order to use this feature, a command function 2 has to be configured to copy the data from the module's status database to the processor.

The PTQ-DNP status database is defined as follows:

Address Range	Description
0 to 137	Module Status
138 to 499	Spare
510 to 899	Slave Status
900 to 999	Spare
1000 to 1299	Command error list

For example, to copy the first 50 command error codes from the module to the processor at holding register address 400501, the following command function 2 would be used:

```

203      0      4      621      20      #Move IED Cntr data to quantum
204      0      4      650      50      #Move IED AI data to quantum
205      0      4      700      8       #Move IED AO data to module
2       1000    4      501      50      #command control data area
END

```

Each parameter for this example is discussed below:

Field	Value	Meaning
Data Type	2	The type of operation to perform 2 = Read Error/Status Data from the PTQ-DNP
Start PT Address	1000	This is the starting address in the module's internal status database
Point Type	4	The range of registers to read from the processor 4 = (holding registers)
Point Address	501	The starting address where the data will be copied to the processor database. Example: 40000 + 501 = 40501 (Quantum) %MW0 + 501 = %MW501 (Unity)
Word Count	50	This is the number of words (16-bit) to be transferred from the module to the processor. In this example 50 words will be transferred.

Implementing Ladder to Support Special Functions.

The previous discussions about Command Function 2 and Command Function 3 have not required that you implement any form of logic within the PLC, however if you are required to use the Command Function 3, you must implement some form of control logic. The following section uses structured text language to illustrate how a typical function might be implemented.

Example: Rebooting the module (All modules)

```

(*)
MyTrigger is an alias for register 401000 (%MW1000)
MyFunction3 is an alias for register 400500 (%MW500)
MyData1-MyData63 are aliases for 400501-400563 (%MW501 - %MW563)

```

```

The premise for this logic is:
IF MyTrigger = SOMEVALUE THEN
  Fill the buffer;
  set MyFunction3 to the appropriate value;
  Clear MyTrigger with a 0;
END_IF;

*)

IF MyTrigger = 9999 THEN
  MyFunction3 := MyTrigger;
  MyTrigger := 0;
END_IF;

```

Example: Setting / Retrieving the time of day (DNP and IEC protocol modules only)

```
( *
Block ID 9971 - Set Modules Time using the PLC's Time
Assumption:
The MyYear, MyMonth etc. values for time and date represent aliases for your
time source.
MyTrigger is an alias for register 401000 (%MW1000).
*)

IF MyTrigger = 9971 THEN;
    MyData1 := MyYear;
    MyData2 := MyMonth;
    MyData3 := MyDay;
    MyData4 := MyHour;
    MyData5 := MyMinute;
    MyData6 := MySeconds;
    MyData7 := MyMillisec;
    MyFunction3 := 9971;
    MyTrigger := 0;
END_IF;

( *
Block ID 9970 - Set PLC's time using the modules time
Assumption:
The MyYear, MyMonth etc. values for time and date are representative of your
aliases for your time source.
MyTrigger is an alias for register 400010 (%MW10).
*)

IF MyTrigger = 9970 THEN;
    MyFunction3 := MyTrigger;
    IF MyFunction3 = 0 AND MyData1 = 9970 THEN;
        MyYear := MyData2;
        MyMonth := MyData3;
        MyDay := MyData4;
        MyHour := MyData5;
        MyMinute := MyData6;
        MySeconds := MyData7;
        MyTrigger := 0;
    END_IF;
END_IF;
```

The previous examples all utilize structured text for the process control logic but follow the same basic program flow.

- 1 Copy the data related to the block function into registers 400801 to 400864 (Quantum) or %MW801 to %MW864 (Unity) as required.
- 2 As your last step, copy the BLOCK ID number of the special function into register 400801 (%MW801).
- 3 Clear your permissive condition.

Block Definition

The PTQ-DNP module uses new blocks to transfer the online/offline status of monitored points from the processor to the module. These blocks are transferred using the normal backplane transfer facility provided on the module using a type code of 3 (control block). The following table describes the block numbers utilized:

Block Number(s)	Data Types	Word Count
9000 to 9008	Binary Input	500 (8000 points)
9010	Analog Input	32
9030	Floating-Point Input	16
9040	Double Input	8

Each bit in the words present in the blocks represent the online/offline state of a specific point. For example, to set the first analog input point to the online state, set bit 0 in word 3 of block 9010 to 1. To set the same point to offline, clear the same bit (value of 0). Each word contains the status values for 16 points. The module is set up to handle all the words possible for each data type even if the points are not utilized in the user application.

The binary input online/offline status data requires several blocks because of the large number of points that can be defined. The following table contains a listing of the points considered in each block:

Block Number	Points	Word Count
9000	0 to 959	60
9001	960 to 1919	60
9002	1920 to 2879	60
9003	2880 to 3839	60
9004	3840 to 4799	60
9005	4800 to 5759	60
9006	5760 to 6719	60
9007	6720 to 7679	60
9008	7680 to 7999	20

Block Transfer Structure

The structure of the new control blocks to transfer the online/offline status is discussed in this section. The following structure informs the module that a new block of data is available to be read:

Block Offset	Description
0	Block ID to consider (that is, 9000)
1	Ignored for the read operation
2	Reserved for future use (not used)
3 to 62	Words of online/offline status data
63	Reserved for future use (not used)

After the module has completed processing the block, it will set the first word in the block to zero and will set the second word to the block identification code received. All other data in the block will remain unchanged.

In order to simplify transfer of the online/offline data from the processor to the module, a separate data area should be defined in the processor for each block. This way the processor logic only needs to alter the data in the memory area to change the online/offline state, and then, insert the block number in word zero of the block in order to have the data transferred.

Block Transfer Configuration

In order to transfer the online/offline status data from the processor to the module, block transfer commands must be set up in the [Backplane Data Exchange] section of the configuration file. An example section is shown:

```
[Backplane Data Exchange]
# Data Start PT      Point      Point      Word
# Type  Address      Type       Address     Count
START
  101      0          4          101         10 #Move DNP BI data to module
  102      0          4          111         10 #Move DNP BO data to quantum
  103      0          4          121         20 #Move DNP Cntr data to module
  104      0          4          201         10 #Move DNP AI data to module
  105      0          4          301         10 #Move DNP AO data to quantum
  106      0          4          401         20 #Move DNP FI data to module
  108      0          4          501          8 #Move DNP FO data to quantum
  201      0          4          601         10 #Move IED BI data to quantum
  202      0          4          611         10 #Move IED BO data to module
  203      0          4          621         20 #Move IED Cntr data to quantum
  204      0          4          650         50 #Move IED AI data to quantum
  205      0          4          700          8 #Move IED AO data to module
   3       0          4          801         64 #command control data area
   3       0          4          901         64 #BI online/offline
   3       0          4         1001         64 #AI online/offline
   3       0          4         1101         64 #FI online/offline
   3       0          4         1201         64 #DI online/offline
END
```

In this example, the binary input status data (only block 9000 is used in this example) is transferred from the processor memory area 40901 (Quantum) or %MW901 (Unity) to the module. The analog input status data is transferred from 41001 (Quantum) or %MW1001 (Unity). Note that because the command control code of 3 is utilized for the transfer operation, the point type must always be 4 and the word count must be 64.

Set Up Command Function 2 (Read Status Error / Status)

This section provides information on how to request the module to transfer error/status data to the processor.

The error/status data is stored in the status database which is defined as follows:

Address Range	Description
0 to 137	Module Status
138 to 499	Spare
510 to 899	Slave Status
900 to 999	Spare
1000 to 1299	Command error list

Module Status

For more information about the Module Status region refer to the PTQ-DNP **Error Status Table** section.

Slave Status

The Slave Status data shows the current status for each one of the configured slaves. The data is structured into groups of 8 words for each slave described as follows:

0	1	2	3	4	5	6	7
Index	Slave Addr	Bad CRC	Buff Ovflw	Tran. Seq#	Conf Retry	Conf Fail	No App Rsp

Where:

Index	This value corresponds to the index in the device array for the slave.
Slave Addr	This value corresponds to the DNP slave address for the device.
Bad CRC	This value represents the number of bad CRC values received from the slave device.
Buff Ovflw	This value represents the number of buffer overflow messages received from the slave device.
Tran Seq#	This value represents the number of incorrect transport layer sequence number errors.
Conf Retry	This value represents the number of data link layer confirm request retries.
Conf Fail	This value represents the number of data link layer confirm request failures.
No App Rsp	This value represents the number of application layer no responses to requests.

Command Error List

Each word in this region represents an error code for each configured command. (starting for command 0 at address 1000). For information about the command error codes refer to the PTQ-DNP **Error Status Table** section.

In order to transfer data from the PTQ-DNP Status Database to the processor you must configure a backplane command function 2.

This command takes the following parameters:

- **Data Type:** 2 (Read Status Data from the PTQ-DNP)
- **Start PT Address:** This value is the starting address in the module's Status Database
- **Point Type:** The type of register within the processor (use a value of 4 for function 2)
- **Point Address:** The destination address within the processor. The address is expressed without the use of the register range, for example 400001 would be entered as 1. (400001 to 40000 =1)

For example, to copy the first 50 command error codes from the module to the processor at holding register address 400501, the following command function 2 would be used:

```

203      0      4      614      20      #Move IED Cntr data to quantum
204      0      4      650      50      #Move IED AI data to quantum
205      0      4      700      8       #Move IED AO data to module
  2     1000    4      501      50      #command control data area
END
    
```

Set Up Command Function 3 (Special Functions)

This section provides information on how to request the module to perform *special non-typical* functions that may be required by an application.

Command Function 3 (three) if required should be the first item entered in the [BACKPLANE DATA EXCHANGE] section of your configuration file.

This may be used to implement the following functionality:

- Force a reboot of the PTQ module (Special Function 9998 or 9999)
- Set / Retrieve Time and Date
- Register events with the protocol

This command takes the following parameters:

- *Data Type*: 3 (Write data to the processor)
- *Start PT Address*: This value is **ALWAYS** 0. Note: This will **NOT** overwrite your application database in the PTQ but merely serves as an additional flag to notify the module of the unique nature of the command.
- *Point Type*: The type of register within the processor
 0=0x, 1=1x, 3=3x, and 4=4x (Quantum data type)
 0=%M, 1=%I, 3=%IW, and 4=%MW (Unity data type)
- *Point Address*: The source register within the processor. The address is expressed without the use of the register range, for example 400001 would be entered as 1. (400001 - 400000 = 1 or 40001 - 40000 = 1)
- *Word Count*: This value is **ALWAYS** 64. Make sure that 64 words of memory are available within the processor.

Example 30:x or 40:x Register Transfer

The following *example* shows a typical command used to retrieve a special function command from the processor. In this *example*, registers 400500 to 400563 (Quantum) or %MW500 to %MW563 (Unity) from the processor will be used to provide the information required by the module.

```
# Data   Start PT      Point      Point      Word
# Type   Address      Type      Address      Count
START
      3         0         4         500         64
END
```

Special Functions

Block ID 9901 - CROB Control Block for Digital Outputs

This block issues one or more command control requests to slave(s) attached to the DNP Master port for Object 12 data. When the module receives a block 9901 identification code, it will place the included commands into the command queue.

Word Offset in Block	Data Field(s)	Description
0	Block ID	This field contains the block identification code of 9901 for the block.
1	Command Count	This field defines the number of CROB blocks to generate. The valid range for the field is 1 to 6.

Word Offset in Block	Data Field(s)	Description
2 to 11	Command #1	Data for the command relay block (CROB) to be generated.
12 to 21	Command #2	Data for the command relay block (CROB) to be generated.
22 to 31	Command #3	Data for the command relay block (CROB) to be generated.
32 to 41	Command #4	Data for the command relay block (CROB) to be generated.
42 to 51	Command #5	Data for the command relay block (CROB) to be generated.
52 to 61	Command #6	Data for the command relay block (CROB) to be generated.

The following fields are used for each 10-word record in the command list:

Word Offset	Definitions	Description
0	Port/Flags	This field is currently ignored as all 9901 blocks are sent immediately out the master port.
1	Slave Address	This is the IED node address for the slave to consider on the network.
2	Object	Object type always 12
3	Variation	Variation always 1
4	Function	Function codes 3, 5 and 6 supported. Function code 4 is automatically sent after a successful function 3.
5	Address in Slave	Point in IED to consider with the CROB.
6	Control Code	This is a standard DNP protocol control code byte (see description below).
7	Pulse Count	This parameter specifies the number of pulses to generate for pulse output control. This parameter has a range of 0 to 255 as the value is a byte parameter in the CROB. If a value of zero is entered, the operation will not execute.
8	Pulse On Time	This parameter specifies the on-time interval for pulse control.
9	Pulse Off Time	This parameter specifies the off-time interval for pulse control.

The control code in the command is a bit coded byte value with the following definition:

Bits	Definitions	Description
0 to 3	Code	These bits determine the control operation to be performed by the command: 0=No operation, 1=Pulse on, 2=Pulse off, 3=Latch on and 4=Latch off. All other values are undefined in the DNP protocol.
4	Queue	0=Normal (execute once), 1=Requeue (place at end of queue after operation).
5	Clear	This parameter clears the queue. If the value is set to zero, the queue is not affected. If the value is set to 1, the queue will be cleared.
6 to 7	Trip/Close	These two bits select the trip or close relay. For close relay control, set the bits to 01. For trip relay control, set the bits to 10. A value of 00 for the bits is used for single point control of normal digital output points.

Block Format from Module:

Word Offset in Block	Data Field(s)	Description
0	Complete	This word will contain a value of zero when the module has completed the operation.
1	Block ID	This field contains the block identification code of 9901 for the block.

Block ID 9902 - Command Control Block (Add command to Command List Queue)

This block code is used by the PLC to send a list of commands to be placed in the command queue. Command placed in the queue with this method need not have their enable bit set.

Word Offset in Block	Data Field(s)	Description
0	Block ID	This field contains the value of 9902 identifying the enable command to the module.
1	Command count	This field contains the number of commands to enable in the command list. Valid values for this field are 1 to 60.
2 to 61	Command Numbers to enable	These 60 words of data contain the command numbers in the command list to enable. The commands in the list will be placed in the command queue for immediate processing by the module. The first command in the list has an index of 0.

There is no response to this block by the module. The module will place the selected commands into the command queue. If the command references a IED unit that is not in the slave list, the command will not be placed in the command queue. Normal processing of the command list will continue after the commands specified in the block are processed.

Block Format from Module:

Word Offset in Block	Data Field(s)	Description
0	Complete	This word will contain a value of zero when the module has completed the operation.
1	Block ID	This field contains the value of 9902 identifying the enable command to the module.

Block ID 9903 - Event Pass-Through Block

The event pass-through functionality allows the module to pass events to the processor after these are received from the DNP slave devices.

Note: The event pass-through functionality is only available for PTQ-DNP version 2.27 or later and PTQ-DNPQ version 2.30 or later.

To verify the firmware revision of your module, at the main menu refer to the REVISION value located at the bottom of the menu (for the example below the revision version is 2.30). If necessary please contact the ProSoft Technology tech support team for information on how to upgrade your module.

```

***** DNP DEBUG PORT HELP *****
KEY      FUNCTION                               | KEY FUNCTION
-----|-----
0-9,A-F  Sets debug level                          | Y   Class/Deadband Assignments
L        Display error list                  | U   Show DNP Databases
P        Display setup & pointers            | <  Receive Configuration
O        Operating parameters                | >  Send Configuration
R        Reboot module                       |
S        Display Comm Stats                  | H   Backplane Transfer Commands
W        Clear error list                    | N   Display Blk X-fer Stats
V        List COM States                     | X   Master Port Commands
T        Master Port Slave Setup             | Z   Master Port Slave Errs
G        Version Information                 | ?   Display this screen

PRODUCT = DNPQ   REVISION = 2.30   OP SYS REV = 1205   PROD RUN # = 0401
_

```

Connected 5:11:14 Auto detect 57600 8-N-1 SCROLL CAPS NUM Capture Print echo

The event pass-through functionality must be initially enabled by the user through the following configuration parameter:

```
Event Messages to PLC: Y #Pass event messages to processor
```

The next step is to configure a backplane data exchange for block 9903 (event pass-through block). Here you will configure the Quantum memory location where the block will be copied to after the module receives an event. Follows a valid example for this application:

```
[Backplane Data Exchange]
# Cmd  DB  Point  Point  Word
# Type Address  Type  Address Count
START
9903  3000   4    3001  121
END
```

Where:

Cmd Type: always use 9903 for event pass-through

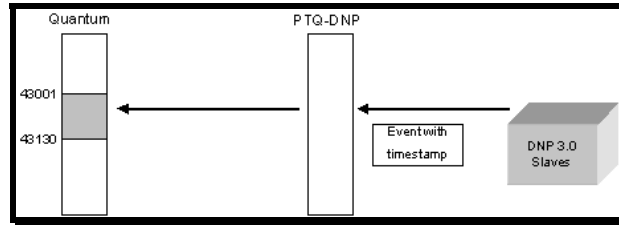
DB Address: this is the database address in the module associated to this block. Because the block will be handled in processor logic (not in the module) this value will not generally be used. Just make sure that this range is located in a database area that is not being used for any other purpose (121 words total)

Point Type: always select 4 (holding register) for event pass-through

Point Address: this is the starting Quantum memory address where the event pass-through block will be copied to (according to the selected point type). For this procedure we will be considering a value of 43001.

Word Count: always use a value of 121 for the event pass-through block. Therefore for this example all registers between 43001 and 43121 will be reserved for the event pass-through block. Make sure that no other processor applications are overwriting to this area.

The following diagram shows the basic idea of the event pass-through functionality. So after the module receives the event from the remote device it will build block 9903 that will be copied to the processor at the configured memory address:



Event Pass-Through Block Format

The block that is copied from the module to the processor has the following format. Each block can contain up to 10 events. The number of events per block will typically depend on the rate between how fast the module receives the events and how fast these can be passed to the processor (typically depends on the processor scan rate).

Block Format for Read

Word Offset in Block	Data Field(s)	Description
0	Event Count	This field contains the number of events present in the block. Values of 1 to 20 are valid.
1 to 12	Event 1	Event message
13 to 24	Event 2	Event message
25 to 36	Event 3	Event message
37 to 48	Event 4	Event message
49 to 60	Event 5	Event message
61 to 72	Event 6	Event message
73 to 84	Event 7	Event message
85 to 96	Event 8	Event message
97 to 108	Event 9	Event message
109 to 120	Event 10	Event message

The format of each 12 word data region in the block is as follows:

Word Offset	Definitions	Description
0	Device Index	Logical slave device index in module
1	IED Point	Logical point address in IED database
2	DNP Point	Logical point address in DNP database
3	Slave Address	Remote slave address that generated event

Word Offset	Definitions	Description
4	Point Number	Point address in remote device
5	Object	DNP object number for point
6	Variation	DNP variation for event
7 to 9	Timestamp	48-bit DNP time: Unix style timestamp in ms
10 to 11	Value	Event value

The processor logic should recognize the event count value greater than zero and read all events in the block. After that this value should be reset to zero to prepare the logic for the next incoming block. Refer to the following topic that shows a sample function block for the event pass-thru functionality.

[How to Set up and Use the Sample Function Block for Concept](#)

[EVENTFB Function Block Overview](#)

The purpose of the EVENTFB sample function block is to transfer the events into a buffer that consists of an array of elements that stores all data in a convenient format for the user application. The block 9903 passes data into a compacted format thus occupying the minimum amount of registers. The EVENTFB sample function block already extracts each event value into a separate register.

Follows below the structure of each element of the buffer (extracted from the data type definition file):

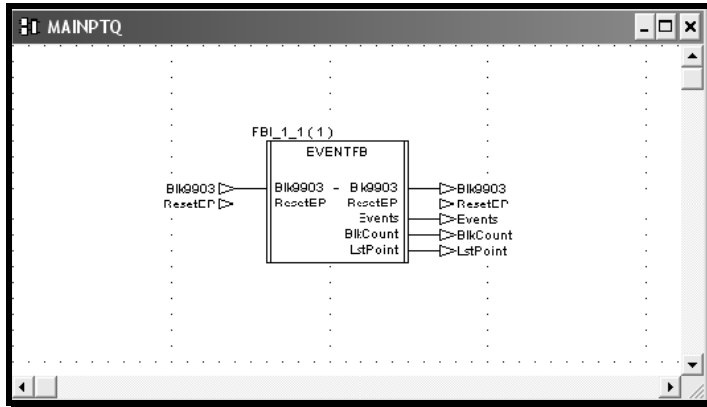
```

TYPE EVENTDNP:
STRUCT
  DeviceIndex: INT;    (* Logical slave device index in module *)
  IEDPoint: INT;      (* Logical point address in IED database *)
  DNPPoint: INT;      (* Logical point address in DNP database *)
  SlaveAddress: INT;  (* Remote slave address that generated event *)
  PointNumber: INT;   (* Point address in remote device *)
  Object: INT;        (* DNP object number for point *)
  Variation: INT;     (* DNP variation for event *)
  TimeStamp: ARRAY[0..2] OF INT; (* 48-bit DNP time *)
  Value: ARRAY[0..1] OF INT;  (* Value for event *)
END_STRUCT;
END_TYPE

```

The data structure that stores the incoming events consists on a circular buffer that can store up to 60 events. So the buffer consists on an array of 60 "EVENTDNP" elements presented previously. The element index can vary from 0 to 59. If the last event updated was located at index 59 then the next event will be copied to index 0.

Follows an instance example of the EVENTFB function block:



The EVENTFB function block contains the following PINs:

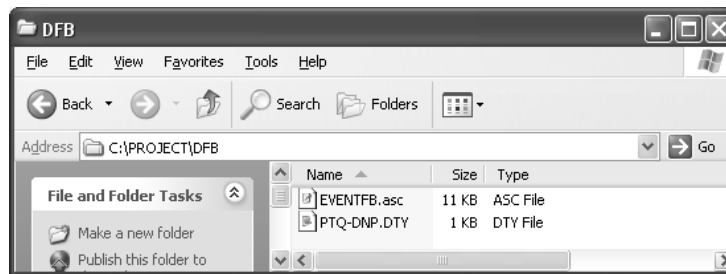
PIN	PIN Type	Data Type	Description
Blk9903	input/output	WORD121	Stores the memory area updated by block 9903. The start address must point to the same start address defined for block 9903 backplane data exchange (Point Address parameter).
ResetEP	input/output	INT	Move a value of one to reset the event pointer. This will cause the next event to be written to index 0 at the circular buffer. The register will be automatically reset to zero after the request was processed. This register should be only used for very specific applications (because the circular buffer automatically changes the element pointer from 59 to 0 after the maximum index was reached)
Events	output	EVENTSTRUCT	Circular buffer that stores all received events in a convenient format for the user application. It can store up to 60 events (index varies from 0 to 59). After event 59 is updated the next event to be received will be automatically updated at index 0.
BlkCount	output	INT	Incremented after a block is received (and after the events in that block have been read into the circular buffer). The maximum value for this counter is 10000 (then it is automatically reset to 0)
LstPoint	output	INT	Pointer to the last event index read from the module. For example, if last event was updated at index 5 then this value will have the same value.

Before You Begin

This setup procedure requires the following pre-requisites:

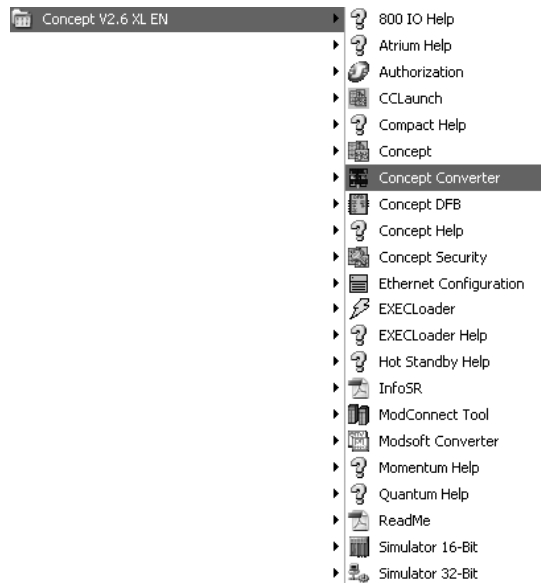
- 1 Make sure that your computer has the Concept Programming Unit installed.
- 2 The PTQ-DNP or PTQ-DNPQ firmware revision must support the event pass-thru functionality. This feature is available for version 2.27 (PTQ-DNP) or version 2.30 (PTQ-DNPQ) or later. Refer to the "REVISION" value at the debug main menu.

- Using Windows Explorer create a folder for your Concept project with a "DFB" subfolder. This procedure will consider as an example the folder C:\PROJECT\DFB, where:
C:\PROJECT- will store the main Concept project (.PRJ)
C:\PROJECT\DFB - will store the data type definition file (PTQ-DNP.DTY) and the function block that will be presented later at this document.
- Refer to the CD-ROM or to the web site for the PTQDNPConcept_Block9903.zip file and extract the two files below:
EVENTFB.asc (function block)
PTQ-DNP.DTY (data type definition)
Use Windows Explorer to move these files to C:\PROJECT\DFB as follows:



Convert the EVENTFB Function Block

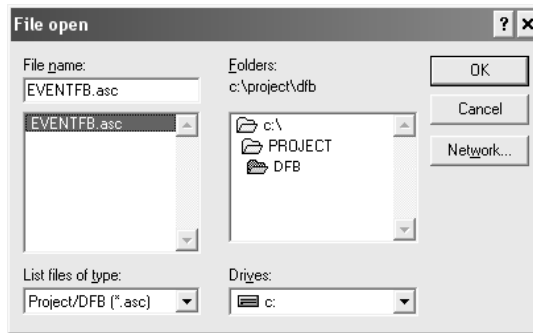
- Select Concept v2.6 XL EN - Concept Converter as follows:



- 2 When the Concept Converter windows is displayed select File-Import



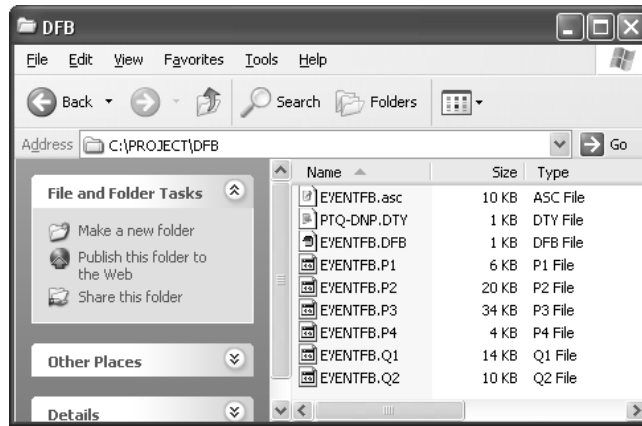
- 3 Select the EVENTFB.asc file located at C:\PROJECT\DFB as follows:



- 4 When the importing procedure is completed you will observe the following confirmation screen:

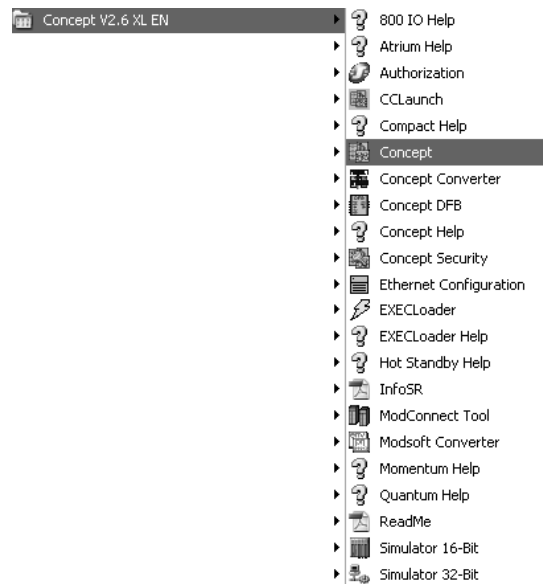


- 5 Close the Concept Converter tool. Now you can refer to C:\PROJECT\DFB to check that the function block (.DFB) was exported and is ready to be used.

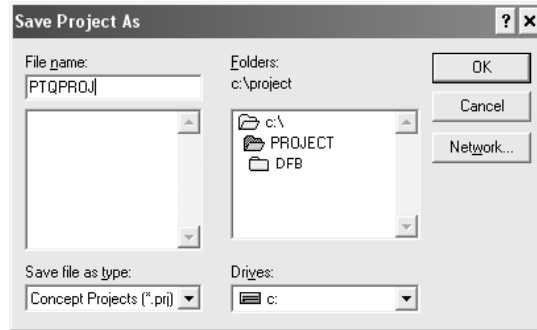


Setup the Concept Project

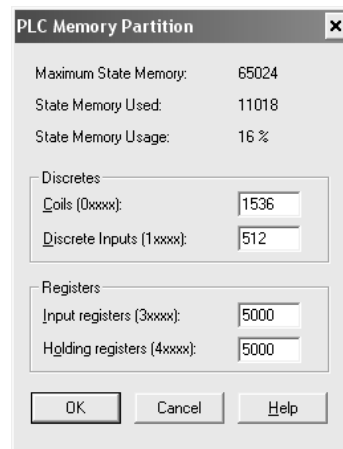
- 1 Run the Concept software as follows



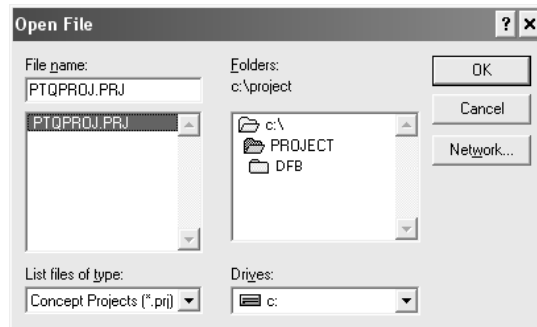
- 2 Create a new project and save it at the C:\PROJECT folder. For this example we will consider the project name as PTQPROJ.



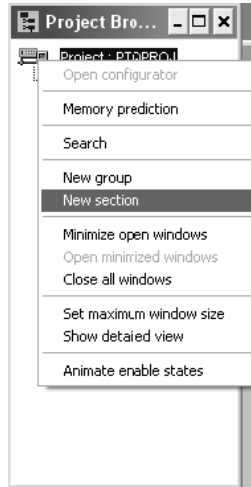
- 3 At PLC Memory Partition make sure that the processor memory range is configured large enough for the PTQ-DNP backplane usage.



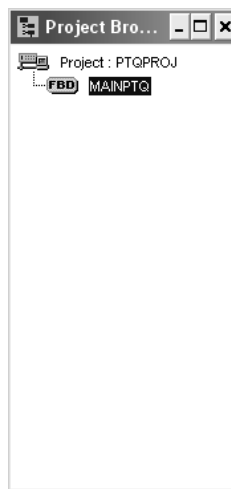
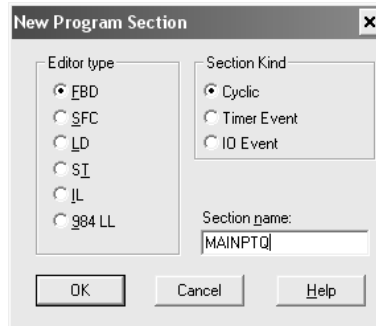
- 4 Select File-Close Project and then File-Open... to select the PTQPROJ file again. This step allows the Concept application to recognize the new data types defined at the PTQ-DNP.DTY file.



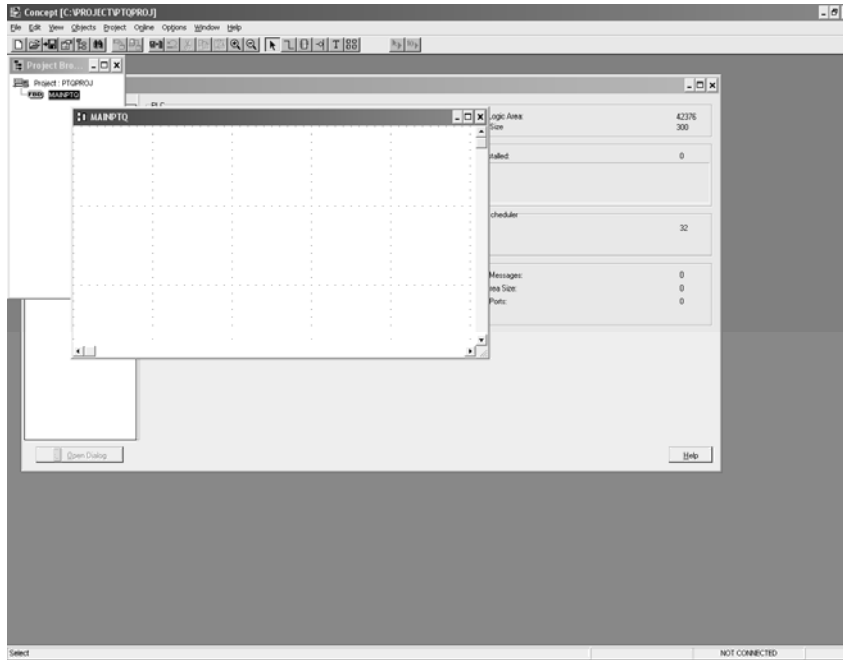
- 5 Select Project Browser. Right-click at Project: PTQPROJ and select New Section



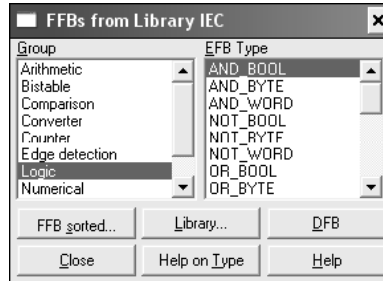
- 6 Select FBD. The procedure will refer to this section as MAINPTQ. Select OK



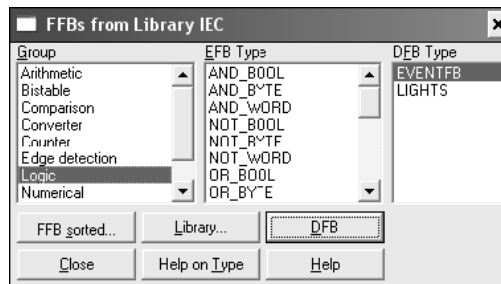
7 Double-Click at the section to display the FBD section:



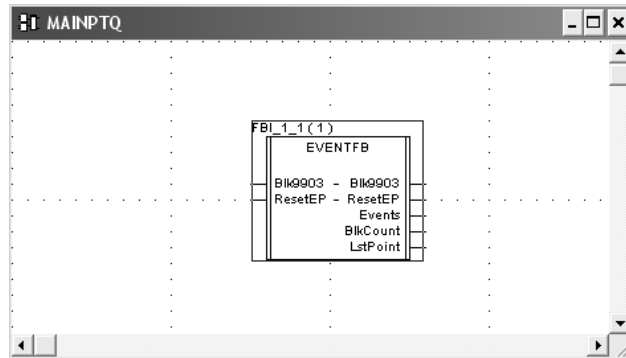
8 Select Objects-FFB Selection...



9 Click at the DFB button and select the EVENTFB function block as follows. Then close the window.



Now you should see the EVENTFB function block at the FBD section:



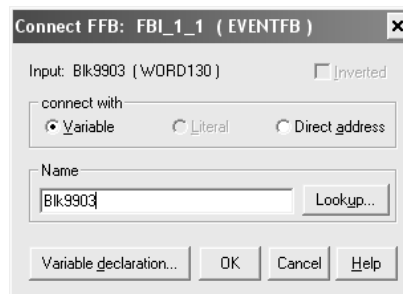
- 10** This step will create variables to be associated to the function block PINs. We will start with the Blk9903 PIN. The variable for this PIN must point to the same start address where block 9903 will be copied to. For this example we are considering the following configuration for block 9903:

```
[Backplane Data Exchange]
# Cmd      DB   Point   Point   Word
# Type  Address   Type   Address  Count
START
  9903     3000     4     3001   121 #only used if Pass-Through Events = Y
END
```

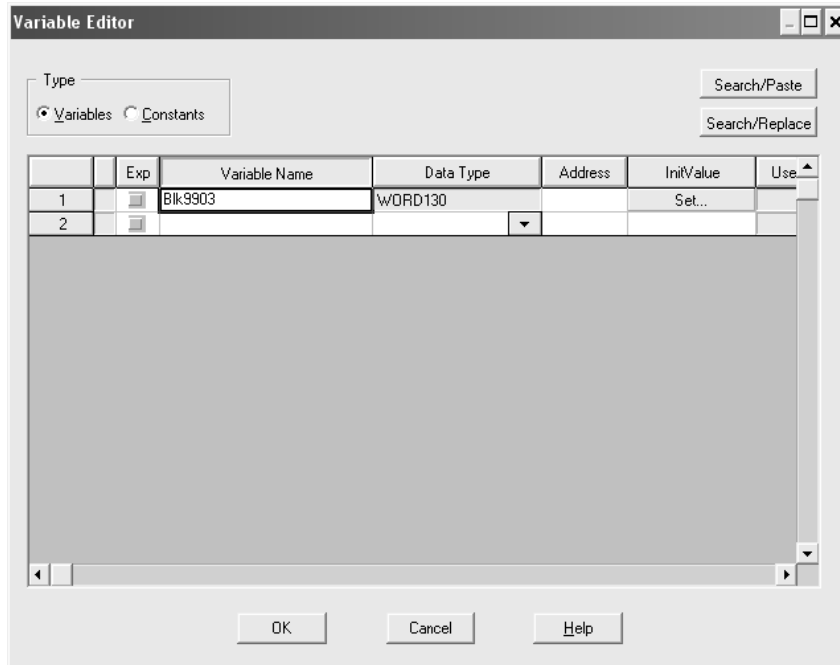
This implies that the variable associated to PIN Blk9903 must also start at the same register address (43001 for this example).

Note: this is the only variable to be associated to a PIN that requires the specification of a Quantum memory address.

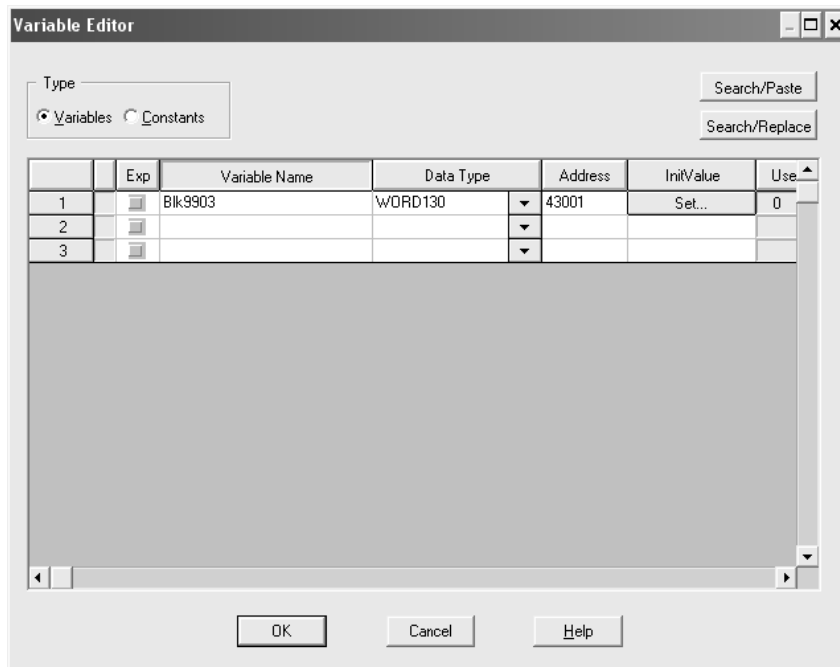
Double-click at the Blk9903 input/output PIN and create a name for the variable to be associated. This example will use the same name as the PIN.



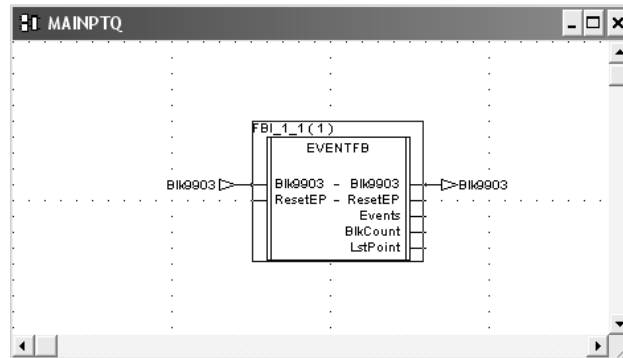
Click at the Variable declaration... button:



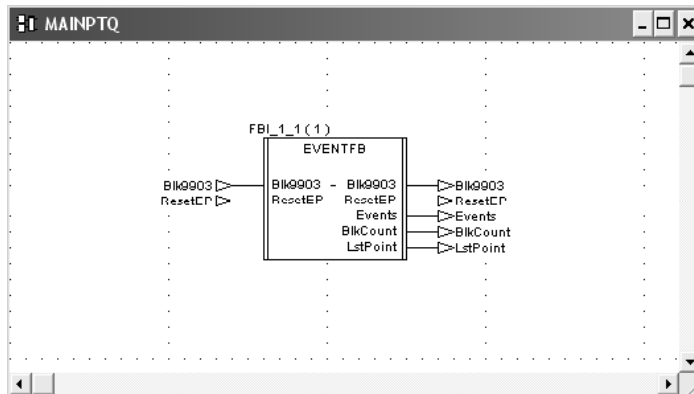
Select the memory address that you have previously configured for Block 9903 (Backplane Exchange - Point Address) . For this example we will consider a value of 43001:



Select OK and you should see the new variable associated to the Blk9903 PIN:



- Repeat the last item for the other PINS (it is not necessary to associate any memory address to the other variables).

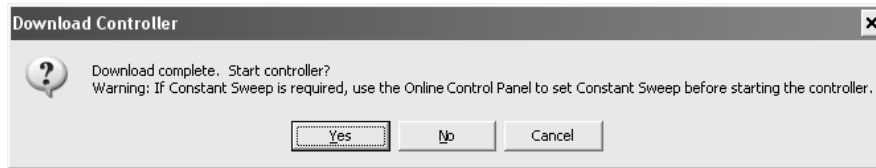


Download the Concept Project

- Select Online-Download to download the Concept Project. Make sure that the IEC program sections checkbox is selected:



- When the download is completed you should see the following window. Select Yes.



Using the EVENTFB Function Block

- In order to show how the function block can be used we will create the following Template. This template shows the BlkCount, LstPoint, ResetEP variables and also the first two event elements (Events.Event[0] and Events.Event[1]):

	Variable Name	Data Type	Address	Value	Set Value	Format	
1							
2	ResetEP	INT		0		Dec	
3	BlkCount	INT		0		Dec	
4	LstPoint	INT		0		Dec	
5							
6	Events.Event[0].DeviceIndex	INT		0		Dec	
7	Events.Event[0].IEDPoint	INT		0		Dec	
8	Events.Event[0].DNPPPoint	INT		0		Dec	
9	Events.Event[0].SlaveAddress	INT		0		Dec	
10	Events.Event[0].PointNumber	INT		0		Dec	
11	Events.Event[0].Object	INT		0		Dec	
12	Events.Event[0].Variation	INT		0		Dec	
13	Events.Event[0].TimeStamp[0]	INT		0	0	Dec	
14	Events.Event[0].TimeStamp[1]	INT		0	0	Dec	
15	Events.Event[0].TimeStamp[2]	INT		0	0	Dec	
16	Events.Event[0].Value[0]	INT		0		Dec	
17	Events.Event[0].Value[1]	INT		0		Dec	
18							
19	Events.Event[1].DeviceIndex	INT		0		Dec	
20	Events.Event[1].IEDPoint	INT		0		Dec	
21	Events.Event[1].DNPPPoint	INT		0		Dec	
22	Events.Event[1].SlaveAddress	INT		0		Dec	
23	Events.Event[1].PointNumber	INT		0		Dec	
24	Events.Event[1].Object	INT		0		Dec	
25	Events.Event[1].Variation	INT		0		Dec	
26	Events.Event[1].TimeStamp[0]	INT		0		Dec	
27	Events.Event[1].TimeStamp[1]	INT		0		Dec	
28	Events.Event[1].TimeStamp[2]	INT		0		Dec	
29	Events.Event[1].Value[0]	INT		0		Dec	
30	Events.Event[1].Value[1]	INT		0	0	Dec	

- In this example, the remote device has sent two events with timestamp to the module (in different blocks 9903). The following shows an example of how the variables associated to the EVENTFB function block would be updated.

BlkCount: shows a value of 2 because the processor has received two blocks 9903

LstPoint: shows a value of 1 because the last element that was updated has an index of 1 (Events.Event[1]).

Events.Event[0]: shows the first event received from the module

Events.Event[1]: shows the second event received from the module

	Variable Name	Data Type	Address	Value	Set Value	Format
1						
2	ResetEP	INT		0		Dec
3	BlkCount	INT		2		Dec
4	LstPoint	INT		1		Dec
5						
6	Events.Event[0].DeviceIndex	INT		1		Dec
7	Events.Event[0].IEDPoint	INT		0		Dec
8	Events.Event[0].DNPPoint	INT		-1		Dec
9	Events.Event[0].SlaveAddress	INT		32		Dec
10	Events.Event[0].PointNumber	INT		0		Dec
11	Events.Event[0].Object	INT		2		Dec
12	Events.Event[0].Variation	INT		2		Dec
13	Events.Event[0].TimeStamp[0]	INT		E403	0	Hex
14	Events.Event[0].TimeStamp[1]	INT		82A7	0	Hex
15	Events.Event[0].TimeStamp[2]	INT		49	0	Hex
16	Events.Event[0].Value[0]	INT		1		Dec
17	Events.Event[0].Value[1]	INT		0		Dec
18						
19	Events.Event[1].DeviceIndex	INT		1		Dec
20	Events.Event[1].IEDPoint	INT		0		Dec
21	Events.Event[1].DNPPoint	INT		-1		Dec
22	Events.Event[1].SlaveAddress	INT		32		Dec
23	Events.Event[1].PointNumber	INT		0		Dec
24	Events.Event[1].Object	INT		2		Dec
25	Events.Event[1].Variation	INT		2		Dec
26	Events.Event[1].TimeStamp[0]	INT		EA99	0	Hex
27	Events.Event[1].TimeStamp[1]	INT		82A7	0	Hex
28	Events.Event[1].TimeStamp[2]	INT		49	0	Hex
29	Events.Event[1].Value[0]	INT		0		Dec
30	Events.Event[1].Value[1]	INT		0	0	Dec

Block ID 9958 - Binary Input Event

If the module retrieves a BLOCK ID of 9958 from the PLC when it issues the Command Function 3, it will place the binary input event data contained within the block into the event buffer and alter the data values for the points in the DNP binary input database.

Using the example data buffer of 400500 to 563 (Quantum) or %MW500 to %MW563 (Unity), the contents of the block would look as follows:

Word Offset In Block	Quantum Address	Unity Address	Data	Description
0	400500	%MW500	Block ID	This field contains the value of 9958 identifying the event block to the module
1	400501	%MW501	Event Count	This field contains the number of events in the block. Valid values for this field are 1 to 12

Word Offset In Block	Quantum Address	Unity Address	Data	Description
2	400502	%MW502	Sequence Counter	This field holds the sequence counter for each 9958 block transfer. This synchronizes and confirms receipt of the block by the module.
3	400503	%MW503	DNP Binary Input Data Point	This is the data point in the DNP binary input database represented by the event.
4	400504	%MW504	Month/Day/State	Formatted: bits 0 to 4 = Day, bits 8 to 11 = Month, bit 15 = digital state for point. All other bits are ignored.
5	400505	%MW505	Hour/Minute	Formatted: bits 0 to 5 = minutes, bits 8 to 12 = hour, All other bits are ignored.
6	400506	%MW506	Sec/Millisecond	Formatted: bits 0 to 9 = milliseconds, bits 10 to 15 = seconds
7	400507	%MW507	Year	This is the four digit year for the event
8 to 12	400508 to 400512	%MW508 to %MW512		Same Five word data structure repeated for Event #2
13 to 17				Same Five word data structure repeated for Event #3
...				...
58 to 62	400558 to 400562	%MW558 to %MW562		Same Five word data structure repeated for Event #12

Up to 12 events can be passed from the PLC to the module in each block. To insure that the block reached the module and was processed, the module will return a response in the following format:

Word Offset in Block	Quantum Address	Unity Address	Data	Description
0	400500	%MW500	0	If it completed successfully
1	400501	%MW501	Block Id	9958
2	400502	%MW502	Event Count	This field contains the number of events processed by the module.
3	400503	%MW503	Sequence Counter	This field contains the sequence counter of the last successful block id 9958 received. (This should match the sequence number in word 2 above if the command was successful)

In your table, word zero will contain a value of zero, word one will contain the BLOCK ID code, and word two will contain the event count.

Block ID 9959 - Analog Input Event

If the module retrieves a BLOCK ID of 9959 from the PLC when it issues the Command Function 3, it will place the analog input event data in the block into the event buffer and alter the data values for the points in the DNP analog input database. Using the example data buffer of 400500 to 563 (Quantum) or %MW500 to %MW563 (Unity), the contents of the block would look as follows:

Word Offset in Block	Quantum Address	Unity Address	Data	Description
0	400500	%MW500	Block ID	This field contains the value of 9959 identifying the event block to the module
1	400501	%MW501	Event Count	This field contains the number of events in the block. Valid values for this field are 1 to 12
2	400502	%MW502	Sequence Counter	This field holds the sequence counter for each 9959 block transfer. This synchronizes and confirms receipt of the block by the module.
3	400503	%MW503	DNP Analog Input Data Point	This is the data point in the DNP Analog Input database represented by the event.
4	400504	%MW504	Date Value	This is the value of the date
5	400505	%MW505	Month/Day	Formatted: bits 0 to 4 = day, bits 8 to 11 = Month. All other bits are ignored.
6	400506	%MW506	Hour/Minute	Formatted: bits 0 to 5 = minutes, bits 8 to 12 = hour, All other bits are ignored.
7	400507	%MW507	Sec/ Millisecond	Formatted: bits 0 to 9 = milliseconds, bits 10 to 15 = seconds
8	400508	%MW508	Year	This is the four digit year for the event
9 to 13	400509 to 400513	%MW509 to %MW513		Same Five word data structure repeated for Event #2
14 to 18				Same Five word data structure repeated for Event #3
...				...
59 to 63	400558 to 400562	%MW558 to %MW562		Same Five word data structure repeated for Event #12

Up to 12 events can be passed from the processor to the module in each block. To insure that the block reached the module and was processed, the module will return a response in the following format:

Word Offset in Block	Quantum Address	Unity Address	Data	Description
0	400500	%MW500	0	If completed successfully
1	400501	%MW501	Block Id	9959
2	400502	%MW502	Event Count	This field contains the number of events processed by the module.
3	400503	%MW503	Sequence Counter	This field contains the sequence counter of the last successful block 9959 received. (This should match the sequence number in word 2 above if the command was successful)

In your table, word zero will contain a value of zero, word one will contain the BLOCK ID code, word two will contain the event count and word 3 the sequence number that matching the one sent.

Block ID 9970 - Set Processor's Time using the Module

If the module retrieves a BLOCK ID of 9970 from the processor when it issues the Command Function 3, it will return the time and date as known by the module into the buffer in the PLC. This data can then be used to set the Time/Date within the processor. Using the example data buffer of 400500 to 563 (Quantum) or %MW500 to %MW563 (Unity), the contents of the block would look as follows:

Word Offset in Block	Quantum Address	Unity Address	Data	Description
0	400500	%MW500	Block ID (9970)	This field contains the value of 9970 identifying the block id type to the module.

The module responds to the request with a read block 9970 with the following format:

Word Offset in Block	Quantum Address	Unity Address	Data Field(s)	Description
0	400500	%MW500	0	If completed successfully
1	400501	%MW501	Block ID	9970.
2	400502	%MW502	Year	This field contains the four-digit value to be used with the new time value.
3	400503	%MW503	Month	This field contains the month value for the new time. Valid entry for this field is in the range of 1 to 12.
4	400504	%MW504	Day	This field contains the day value for the new time. Valid entry for this field is in the range of 1 to 31.
5	400505	%MW505	Hour	This field contains the hour value for the new time. Valid entry for this field is in the range of 0 to 23.
6	400506	%MW506	Minute	This field contains the minute value for the new time. Valid entry for this field is in the range of 0 to 59.
7	400507	%MW507	Seconds	This field contains the second value for the new time. Valid entry for this field is in the range of 0 to 59.
8	400508	%MW508	Milliseconds	This field contains the millisecond value for the new time. Valid entry for this field is in the range of 0 to 999.
9	400509	%MW509	Remote Time Synchronization	This field informs the PLC if the data and time passed has been synchronized with a remote DNP master device on the module's slave port.

Block ID 9971 - Set Module's Time Using Processor's Time

If the module retrieves a BLOCK ID of 9971 from the processor when it issues the Command Function 3, it will set the time and date in the module to that known by the module. Using the example data buffer of 400500 to 563 (Quantum) or %MW500 to %MW563 (Unity), the contents of the block would look as follows:

Word Offset in Block	Quantum Address	Unity Address	Data Field(s)	Description
0	400500	%MW500	Block ID	9971.
1	400501	%MW501	Year	This field contains the four-digit value to be used with the new time value.
2	400502	%MW502	Month	This field contains the month value for the new time. Valid entry for this field is in the range of 1 to 12.
3	400503	%MW503	Day	This field contains the day value for the new time. Valid entry for this field is in the range of 1 to 31.
4	400504	%MW504	Hour	This field contains the hour value for the new time. Valid entry for this field is in the range of 0 to 23.
5	400505	%MW505	Minute	This field contains the minute value for the new time. Valid entry for this field is in the range of 0 to 59.
6	400506	%MW506	Seconds	This field contains the second value for the new time. Valid entry for this field is in the range of 0 to 59.
7	400507	%MW507	Milliseconds	This field contains the millisecond value for the new time. Valid entry for this field is in the range of 0 to 999.

The module will respond to a valid 9971 Block ID by returning the following data in the buffer:

Word Offset in Block	Quantum Address	Unity Address	Data Field(s)	Description
0	400500	%MW500	0	If completed successfully
1	400501	%MW501	Block ID	9971

Block ID 9998 or 9999 - Reboot Module

If the processor sends a block number 9998 or 9999, the module will reset the contents of the data block to zero and perform a complete reboot operation.

Word Offset in Block	Quantum Address	Unity Address	Data Field(s)	Description
0	400500	%MW500	9998 or 9999	Block ID to reboot module

6.2.3 [DNP Slave]

This section provides information required to configure a slave application with the module. Most entries contained within this section are self explanatory with the possible exception of the Use IP List directive. This directive instructs the module to verify the address of the received message and ignore the message if it is not on our list of acceptable clients. Another item of concern is the maximum size of the total database, although it is possible to configure a database of considerable size, this would not work, as the maximum Class 0 request may not exceed 2048 bytes in size.

The following example shows a sample [DNP Slave] section:

```
# This section is used to define the configuration for the Module.
# port. This port will receive requests from a remote DNP master unit.
#
[DNP Slave]
Internal Slave ID      : 6          #0-65534 slave identification code for this unit

# DNP slave communication port configuration
Baud Rate              : 19200      #Baud rate for port 110-115200
RTS On                 : 0          #0-32000 mSec before message
RTS Off                : 0          #0-32000 mSec after message
Min Response Delay     : 0          #0-32000 mSec before response sent from slave

# DNP slave modem configuration
Modem                  : No         #Use a dial-up modem on this port (Yes or No)
Connect Timeout        : 20000     #0-65535 milliseconds before connect timeout
First Character Delay  : 1000      #0-65535 milliseconds before 1st char after connect
Redial Delay Time     : 100        #0-65535 1/10 seconds min before redial attempt
Redial Random Delay    : 150       #0-65535 1/10 seconds random before redial attempt
Idle Timeout           : 200       #0-65535 1/10 seconds inactive timeout
Phone Number           : ATDT18001234567

# Collision Avoidance parameters
Collision Avoidance    : No         #Use Collision Avoidance (Yes or No)
CD Idle Time           : 10         #0-32000 mSec min idle time before transmit
CD Random Time         : 15         #0-32000 mSec random idle time before transmit
CD Time Before Receive : 5         #0-65535 milliseconds before receive

#Default Class Settings
BI Class               : 1         #Default class for binary input events
AI Class               : 2         #Default class for analog input events
Float Class            : 3         #Default class for float input events
Double Class           : 0         #(Not Used)

# DNP specific parameters
AI Deadband            : 10         #0-32767 analog deadband value for events
Float Deadband         : 10.0      #Single float deadband
Double Deadband        : 0         #(Not Used)
Select/Operate Arm Time : 2000     #1-65535 milliseconds arm timeout for select/op outputs
Write Time Interval    : 60        #0-1440 minutes for time sync from master

Data Link Confirm Mode : Never     #DL confirm mode (N=Never,S=Sometimes,A=Always)
Data Link Confirm Tout : 1000     #1-65535 milliseconds DL confirm timeout
Data Link Max Retry    : 2         #0-255 maximum DL confirm retry count
App Layer Confirm Tout : 2000     #1-65535 milliseconds App Layer confirm timeout
```

```

Unsolicited Response      : No      #Generate Unsolicited responses (Yes or No)
Class 1 Unsol Resp Min   : 2        #1-255 min number of events before send
Class 2 Unsol Resp Min   : 3        #1-255 min number of events before send
Class 3 Unsol Resp Min   : 4        #1-255 min number of events before send
Unsol Resp Delay         : 10000    #0-65535 milliseconds before events sent
UResp Master Address     : 2        #DNP address of master to send UResp data
UResp Retry Count        : 0        #0-255 Number of retries before switching ports

AI Events with time      : No      #timestamp AI Event data default (Yes or No)
Time Sync Before Events : No      #timesync module before events gen (Yes or No)
Initialize DNP Database : No      #Initialize the DNP Slave output database areas
(Y/N)

```

Modify each parameter based on the needs of your application:

Internal Slave ID

```
Internal Slave ID      : 6      #0-65534 slave identification code for this unit
```

This is the DNP address for the module. All messages with this address received from the master will be processed by the module. This example shows the slave identification code of 6.

Baud Rate

```
Baud Rate              : 19200  #Baud rate for port 110-115200
```

Primary DNP Port Baud Rate: 300, 600, 1200, 2400, 4800, 9600, 19200, 384 (38400), 576 (57600), 115 (115200). The module has been tested for baud rates up to 19200.

RTS On

```
RTS On                 : 0      #0-65535 milliseconds before message
```

This value represents the number of 1 ms increments to be inserted between asserting the RTS modem line and the actual transmission of the data.

RTS Off

```
RTS Off                : 0      #0-65535 milliseconds after message
```

This value represents the number of 1 ms increments to be inserted after the last character of data is transmitted before the RTS modem line is dropped.

Minimum Response Delay

```
Min Response Delay     : 0      #0-65535 milliseconds before response sent from slave
```

Minimum time between receiving a request and transmitting a response. Allows master time to disable transmitter on an RS-485 network.

Modem

Modem : No #Use a dial-up modem on this port (Yes or No)

This parameter defines if a dial-up modem is used on the secondary DNP slave port. A modem cannot be used if the port is configured as a master.

Connect Timeout

Connect Timeout : 20000 #0-65535 milliseconds before connect timeout

Defines the number of milliseconds to wait for the CD signal to be set high. The CD signal indicates a connection is made using a dial-up modem.

First Character Delay

First Character Delay: 1000 #0-65535 milliseconds before 1st char after connect

Defines the number of milliseconds to wait before sending the first message after the connection is first made. This delay only applies to the first packet sent to the modem.

Redial Delay Time

Redial Delay Time : 100 #0-65535 1/10 seconds min before redial attempt

Defines the minimum number of milliseconds to wait before a redial attempt is made by the slave.

Redial Random Delay

Redial Random Delay: 150 #0-65535 1/10 seconds random before redial attempt

Defines a random millisecond time range to be added to the redial delay time before the modem is accessed.

Idle Timeout

Idle Timeout : 200 #0-65535 1/10 seconds inactive timeout

Defines the number of milliseconds the modem is inactive before it will disconnect.

Phone Number

Phone Number : ATDT18001234567

This field contain a null-terminated, ASCII character string used by the dial-up modem. The string must contain all characters required by the modem. An example string is ATDT18001234567. Maximum length is 34 bytes including the terminating 0.

Collision Avoidance

Collision Avoidance : No #Use Collision Avoidance (Yes or No)

This parameter defines if collision avoidance will be utilized on the primary DNP slave port.

CD Idle Time

CD Idle Time : 10 #0-32000 mSec min idle time before transmit

Defines the minimum number of milliseconds to wait before transmitting a message after the CD signal is recognized as low.

CD Random Time

CD Random Time : 15 #0-32000 mSec random idle time before transmit

Defines the range of random time to be added to the CD Idle Time before a message will be transmitted from the slave.

CD Time Before Receive

CD Time Before Receive : 5 #0-65535 milliseconds before receive

Defines the number of milliseconds to wait before receiving characters after the CD signal is recognized as high.

BI Class

0=disable, else 1 to 3

This parameter specifies the default class to be utilized for all the binary input points in the DNP database that are not defined in the override list section.

AI Class

0=disable, else 1 to 3

This parameter specifies the default class to be utilized for all the analog input points in the DNP database that are not defined in the override list section.

Float Class

0=disable, else 1 to 3

This parameter specifies the default class to be utilized for all the floating-point input points in the DNP database that are not defined in the override list section.

AI Deadband

AI Deadband : 1 #0-32767 analog deadband value for events

This parameter specifies the default deadband value assigned to all points not defined in the override list for the analog input point type in the DNP database.

Float Deadband

Float Deadband : 1000.0 #Single float deadband

This parameter specifies the default deadband value assigned to all points not defined in the override list for the floating-point input point type in the DNP database.

Double Deadband

Double Deadband : 4000.0 #Double float deadband (Not Used)

This parameter specifies the default deadband value assigned to all points not defined in the override list for the double floating-point input point type in the DNP database.

Select/Operate Arm Time

Select/Operate Arm Time: 2000 #1-65535 milliseconds arm timeout for select/op outputs

Time period after select command received in which operate command will be performed. After the select command is received, the operate command will only be honored if it arrives within this period of time.

Write Time Interval

Write Time Interval : 60 #0-1440 minutes for time sync from master

Time interval to set the need time IIN bit (0=never), which will cause the master to write the time. For example, if this parameter is configured for 60 minutes, it would mean 60 minutes after the last write date and time request. The module would set the "Need Time" bit again.

Data Link Confirm Mode

Data Link Confirm Mode: Never #DL confirm mode (N=Never, S=Sometimes, A=Always)

IED can request acknowledgement from master station when sending data. The codes are as follows: 0=Never, 1=Sometimes, 2=Always.

Data Link Confirm Tout

Data Link Confirm Tout : 1000 #1-65535 milliseconds DL confirm timeout

Time period to wait for Master Data Link confirmation of last frame sent. This time is in milliseconds. This parameter is only used if the frame is sent with confirmation requested.

Data Link Max Retry

Data Link Max Retry : 2 #0-255 maximum DL confirm retry count

Maximum number of retries at the Data Link level to obtain a confirmation. If this value is set to 0, retries are disabled at the data link level of the protocol. This parameter is only used if the frame is sent with confirmation requested.

App Layer Confirm Tout

App Layer Confirm Tout : 2000 #1-65535 milliseconds App Layer confirm timeout

Event data contained in the last response may be sent again if not confirmed within the millisecond time period set. If application layer confirms are used with data link confirms, ensure that the application layer confirm timeout is set long enough.

Unsolicited Response

Unsolicited Response : No #Generate Unsolicited responses (Yes or No)

Set if the slave unit will send unsolicited response messages. If set to No, the slave will not send unsolicited responses. If set to Yes, the slave will send unsolicited responses. The module will send the event when one of the following conditions are satisfied:

- 1 Minimum number of events is reached
- 2 Delay time is reached

Class 1 Unsol Resp Min

Class 1 Unsol Resp Min : 10 #1-255 min number of events before send

Minimum number of events in Class 1 required before an unsolicited response will be generated.

Class 2 Unsol Resp Min

Class 2 Unsol Resp Min : 10 #1-255 min number of events before send

Minimum number of events in Class 2 required before an unsolicited response will be generated.

Class 3 Unsol Resp Min

Class 3 Unsol Resp Min : 10 #1-255 min number of events before send

Minimum number of events in Class 3 required before an unsolicited response will be generated.

Unsol Resp Delay

Unsol Resp Delay : 2000 #0-65535 milliseconds before events sent

Maximum number of 1 millisecond intervals to wait after an event occurs before sending an unsolicited response message. If set to 0, only use minimum number of events.

UResp Master Address

UResp Master Address : 1 #DNP address of master to send UResp data

DNP destination address where unsolicited response messages are sent.

UResp Retry Count

UResp Retry Count : 0 #0-255 Number of retries before switching ports

Determines the number of unsolicited message retries sent on primary DNP port before changing to secondary port. If the value is 0, port switching will be disabled.

AI Events with Time

AI Events with time : No #timestamp AI Event data default (Yes or No)

This parameter sets if the analog input events generated by the module will include the date and time of the event. If the parameter is set to No, the default is set to no time data. If the parameter is set to Yes, the default object will include the time of the event.

Time Sync Before Events

Time Sync Before Events: No #timesync module before events gen (Yes or No)

This parameter determines if events are to be generated by the module before the time synchronization from the master unit. If the parameter is set to Yes, no events will be generated until the module's time has been synchronized. If the parameter is set to No, events will always be generated.

Initialize DNP Database

Initialize DNP Database: No #Initialize the DNP Slave output database areas (Y/N)

This parameter determines if the module will request data from the processor to initialize the DNP database output data areas. During the first scan, the module will read all output points from the processor to initialize its internal database.

6.2.4 [DNP Slave Database]

The following shows an example [DNP Slave Database] section:

```
[DNP Slave Database]
Binary Inputs      : 160 #0-8000 point count to hold BI data
PLC Binary Inputs  : 160 #0-8000 BI point count from PLC
Analog Inputs      : 10  #0-500 points of analog input data
PLC Analog Inputs  : 10  #0-500 analog input points from PLC
Float Inputs       : 10  #0-250 points of floating-point format data
PLC Float Inputs   : 10  #0-250 points of floating-point format data
Double Inputs      : 0   #(Not Used)
PLC Double Inputs  : 0   #(Not Used)
Counters           : 10  #0-250 points of counter data
PLC Counters       : 10  #0-250 counter points from PLC
Binary Outputs     : 160 #0-2000 point count to hold BO data
PLC Binary Outputs : 160 #0-2000 BO point count from PLC
Analog Outputs     : 10  #0-500 points of analog output data
PLC Analog Outputs : 10  #0-500 analog output points from PLC
Float Outputs      : 10  #0-250 points of floating-point format data
PLC Float Outputs  : 10  #0-250 points of floating-point format data
Double Outputs     : 0   #(Not Used)
PLC Double Outputs : 0   #(Not Used)
```

Edit each parameter as required for your application. The following topics describe each parameter.

Binary Inputs

Binary Inputs : 160 #0-8000 point count to hold BI data

Number of digital input points to configure in the DNP slave device. Each point will be stored as a single bit in the module memory.

PLC Binary Inputs

PLC Binary Inputs : 160 #0-8000 BI point count from PLC

Number of digital input points configured above that are to be obtained from the processor. All other binary input points must come from the attached IED units.

Analog Inputs

Analog Inputs : 50 #0-500 points of analog input data

Number of analog input points to configure in the DNP slave device. Each point will occupy a one word area in the module memory.

PLC Analog Inputs

PLC Analog Inputs : 50 #0-500 analog input points from PLC

Number of analog input points configured above that are to be obtained from the processor. All other analog input points must come from the attached IED units.

Float Inputs

Float Inputs : 5 #0 to 250 points of floating-point format data

Number of floating-point input points to configure in the DNP slave device. Each point will occupy a two-word area in the module memory.

PLC Float Inputs

PLC Float Inputs : 5 #0-250 points of floating-point format data

Number of floating-point input points configured above that are to be obtained from the PLC.

Counters

Counters : 20 #0-250 points of counter data

Number of counter points to configure in the DNP slave device. Each point will occupy a two word area in the module memory. This number corresponds to the number of frozen counters. The application maps the counters to the frozen counters directly.

PLC Counters

PLC Counters : 20 #0-250 counter points from PLC

Number of counter points configured above that are to be obtained from the processor. All other counter points must come from the attached IED units.

Binary Outputs

Binary Outputs : 160 #0-2000 point count to hold BO data

Number of digital output points to configure in the DNP slave device. Each point will be stored as a single bit in the module memory.

PLC Binary Outputs

PLC Binary Outputs : 160 #0-2000 BO point count from PLC

Number of digital output points configured above that are to be sent to the processor. All other binary output points will be sent to the attached IED units.

Analog Outputs

Analog Outputs : 28 #0-500 points of analog output data

Number of analog output points to configure in the DNP slave device. Each point will occupy a one word area in the module memory.

PLC Analog Outputs

PLC Analog Outputs : 28 #0-500 analog output points from PLC

Number of analog output points configured above that are to be sent to the processor. All other analog output points will be sent to the attached IED units.

Float Outputs

Float Outputs : 4 #0-250 points of floating-point format data

Number of floating-point output points to configure in the DNP slave device. Each point will occupy a two- word area in the module memory.

PLC Float Outputs

PLC Float Outputs : 4 #0-250 points of floating-point format data

Number of floating-point output points configured above that are to be sent to the processor.

6.2.5 [DNP Slave Binary Inputs]

This section of the configuration file overrides the Class 2 binary database points. Enter the list of points between the start and end labels:

```
[DNP Slave Binary Inputs]
# This area is to override the class (2) binary input database points.
#
# Point#   Class
Start
#   0       1
#   1       2
#   2       3
#   3       0   #Events will never be generated for this point
End
```

This section takes the following parameters:

Parameter Number	Parameter Name	Parameter Description
1	Point #	This is the information object address of the point.
2	Class	Class 1 - Highest priority Class 2 - Middle priority Class 3 - Lowest priority 0 - Disable.

6.2.6 [DNP Slave Analog Inputs]

This section of the configuration file overrides the Class 3 and deadband for the integer analog input database. The point number is the offset from the start of the analog input database.

```
[DNP Slave Analog Inputs]
# This area is to override the class (3) and deadband for the integer analog
# input database. The point # is the offset from the start of the analog
# input database.
#
# Point#   Class   Deadband
Start
#   6       1       2000 #points 0-5=class 1, deadband = 1000
#   7       1       2000
#   8       2       1000
End
```

This section takes the following parameters:

Parameter Number	Parameter Name	Parameter Description
1	Point #	This is the information object address of the point.
2	Class	Class 1 - Highest priority Class 2 - Middle priority Class 3 - Lowest priority 0 - Disable
3	Deadband	A range of values within which the module will avoid generating events.

6.2.7 [DNP Slave Float Inputs]

This area overrides the Class 3 and deadband for the single float database. The point number is not the address in the analog database, but rather the offset from the start of the single floating-point database.

```
[DNP Slave Float Inputs]
# This area is to override the class (3) and deadband for the single float
# database. The point # is not the address in the analog database, but is
# the offset from the start of the single floating-point database.
#
# Point#   Class   Deadband
Start
  0         1       100.
  1         2       12.34
  3         0       13.45 #Events will never be generated for this point
  4         2       3000.0 #points 5 to 11=class 1, deadband = 1000.00
End
```

This section takes the following parameters:

Parameter Number	Parameter Name	Parameter Description
1	Point #	This is the information object address of the point.
2	Class	Class 1 - Highest priority Class 2 - Middle priority Class 3 - Lowest priority 0 - Disable.
3	Deadband	A range of values within which the module will avoid generating events.

6.2.8 [Secondary Port]

The following is an example of the [Secondary Port] section:

```
[Secondary Port]
Type           : M      #' '=Disabled, M=Master, S=Slave
Baud Rate      : 19200  #Baud rate for port 110-115200
RTS On        : 10     #0-65535 milliseconds before message
RTS Off        : 0      #0-65535 milliseconds after message
Min Response Delay : 0   #0-65535 milliseconds before response sent from slave

# Collision Avoidance parameters
Collision Avoidance : No  #Use Collision Avoidance (N=No, Y=Yes)
CD Idle Time       : 10  #0-32000 mSec min idle time before transmit
CD Random Time     : 20  #0-32000 mSec random idle time before transmit
CD Time Before Receive : 6  #0-65535 milliseconds before receive
```

Configure each parameter to work with your application.

Type

Type : M #' '=Disabled, M=Master, S=Slave

This parameter defines the functionality of the secondary port on the module.

M = emulate a DNP master port

S = back-up DNP slave port to the primary port.

Any other value will disable the port.

Baud Rate

Baud Rate : 19200 #Baud rate for port 110-115200

Secondary DNP Port Baud Rate: 300, 600, 1200, 2400, 4800, 9600, 19200, 384 (38400) , 576 (57600), 115 (115200).

RTS On

RTS On : 10 #0-65535 milliseconds before message

This value represents the number of 1 ms increments to be inserted between asserting the RTS modem line and the actual transmission of the data.

RTS Off

RTS Off : 0 #0-65535 milliseconds after message

This value represents the number of 1 ms increments to be inserted after the last character of data is transmitted before the RTS modem line is dropped.

Min Response Delay

Min Response Delay : 0 #0-65535 milliseconds before response sent from slave

Minimum time between receiving a request and transmitting a response. Allows master time to disable transmitter on an RS-485 network.

Collision Avoidance

Collision Avoidance : No #Use Collision Avoidance (N=No, Y=Yes)

This parameter defines if collision avoidance will be utilized on the primary DNP slave port.

CD Idle Time

CD Idle Time : 10 #0-32000 mSec min idle time before transmit

Defines the minimum number of milliseconds to wait before transmitting a message after the CD signal is recognized as low.

CD Random Time

CD Random Time : 20 #0-32000 mSec random idle time before transmit

Defines the range of random time to be added to the CD Idle Time before a message will be transmitted from the slave.

CD Time Before Receive

CD Time Before Receive : 6 #0-65535 milliseconds before receive

Defines the number of milliseconds to wait before receiving characters after the CD signal is recognized as high.

6.2.9 [DNP Master]

The following shows an example of the [DNP Master] section:

```
[DNP Master]
Internal ID          : 1      #0-65534 identification code for this unit
Initialize IED Database: Yes  #Initialize the IED input database areas (Y/N)
Event Messages to PLC : Yes   #Pass received events to processor (Y/N)
```

Configure each parameter to suit the needs of your application:

Internal ID

Internal ID : 1 #0-65534 identification code for this unit

This is the DNP address for the module. All messages with this address from the master will be processed by the module.

Initialize IED Database

Initialize IED Database: Yes #Initialize the IED input database areas (Y/N)

This parameter determines if the module will request data from the processor to initialize the IED database input data areas. If this option is utilized, ladder logic is required to send the requested block from the processor to the module.

Event Messages to PLC

Event Messages to PLC : Yes #Pass received events to processor (Y/N)

Enables the pass-through functionality that allows the module to pass received timestamp events from the remote slave device to the processor. It requires the configuration of block 9903 in the backplane command section. Refer to the Block 9903 section of this User Manual for further information.

6.2.10 [IED Database]

The following shows an example of an [IED Database] section:

```
[IED Database]
Binary Inputs      : 160   #0-2048 point count to hold BI data
Analog Inputs      : 50    #0-256 points of analog input data
Counters           : 10    #0-64 points of counter data
Binary Outputs     : 48    #0-2048 point count to hold BO data
Analog Outputs     : 8     #0-128 points of analog output data
```

Binary Inputs

Binary Inputs : 160 #0-2048 point count to hold BI data

Number of binary input points contained in the IED database to transfer to the processor and obtained from the attached IED units.

Analog Inputs

Analog Inputs : 50 #0-256 points of analog input data

Number of analog input points contained in the IED database to transfer to the processor and obtained from the attached IED units.

Counters

Counters : 10 #0-64 points of counter data

Number of counter points contained in the IED database to transfer to the processor and obtained from the attached IED units.

Binary Outputs

Binary Outputs : 48 #0-2048 point count to hold BO data

Number of binary output points contained in the IED database which are transferred from the processor and used by the attached IED units.

Analog Outputs

Analog Outputs : 8 #0-128 points of analog output data

Number of analog output points contained in the IED database, which are transferred from the processor and used by the attached IED units.

6.2.11 [DNP Master Slave List]

The [DNP Master Slave List] section stores information about each slave being used by the master port. There must be an entry in this table for each node to be used in the command list. Two of the parameters in this list are coded values and are described in the following two sections.

```
[DNP Master Slave List]
# This section is used to store information about each slave to be
# used by the master port. There must be an entry in this table for each
# node to be used in the command list. Two of the parameters in this list
# are coded values:
#   Conf Mode ==> 0=Never, 1=Sometimes and 2=Always (select 0).
#   Flags is bit coded as follows:
#     Bit 0 (decimal 1) ==> Enable the slave
#     Bit 1 (decimal 2) ==> Use Unsolicited messaging with this slave
#     Bit 2 (decimal 4) ==> Use delay measurement with this slave
#     Bit 3 (decimal 8) ==> Auto time synchronization enabled
#
START
#   Node DL Conf      Conf      Conf  App Rsp
# Address  Mode  Timeout  Retry  Timeout  Flags
#         32      0      1000      0      7000      9
END
```

6.2.12 [DNP Master Commands]

The [DNP Master Commands] section contains the list of commands to process on the master port. Node addresses present in the command list must have an entry in the [DNP Slave List]. Commands with nodes not present in the list will not be executed. The module supports up to 100 commands.

The following shows an example of a [DNP Master Commands] section:

```
[DNP Master Commands]
# This section contains the list of commands to process on the master port.
# Node addresses present in the command list must have an entry in the
# [DNP Slave List]. Commands with nodes not present in the list will not be
# executed.
#
START
#      1          2          3          4      5          6          7          8          9          10
#Flags/   Node   Data      Data  Cmd  Device Point  DNP DB  IED DB  Poll
#Enable  Address Object Variation Func Address Count Address Address Interval
      6      32      60          5    1      0      60      -1      -1      2
      6      32       1          1    1      0      60      -1       0      5
      6      32      30          1    1      0      50      -1       0      6
END
```

This section takes the following parameters:

Parameter Number	Parameter Name	Parameter Description
1	Flags/Enable	See following topics for descriptions
2	Node Address	
3	Data Object	
4	Data Variation	
5	Cmd Func	
6	Device Address	
7	Point Count	
8	DNP DB Address	
9	IED DB Address	
10	Poll Interval	

The definition of each parameter required for each command is provided in the following table.

Bits in the Port/Flags parameter are dependent on the data type. The following table defines the Port/Flags bits for binary input, analog input and counter data points.

Port/Flags Bits	Description	Decimal Equivalent
0 to 1	Communication port (0=Internal, 1=Port 1, 2=Port 2, 3=Port 3)	0 to 3
2	Enable/Disable Command (1=Enable, 0=Disable)	4
3	RBE Flag (0=Events from IED, 1=Events by module)	8
4 to 7	Not Used	

For these data types the qualifier used in the data request is dependent on the Point Count and Address in Slave fields in the command as follows:

- If Point Count < 0, then use Qualifier 06h (All points, packed & -Point Count = # of points to consider)
- If Address in Slave = 0 & Point Count > 0, then use Qualifier 00h or 01h (points 0 to Point Count -1)
- If Address in Slave > 0 & Point Count > 0, then use Qualifier 00h or 01h (Address in Slave to Address in Slave + Point Count -1)

The following table defines the Port/Flags bits for binary output and analog output points.

Port/Flags Bits	Description	Decimal Equivalent
0 to 1	Communication port (0=Internal, 1=Port 1, 2=Port 2, 3=Port 3)	0 to 3
2	Enable/Disable Command (1=Enable, 0=Disable)	4
3	Poll Type (0=Poll, 1=Exception)	8
4	Data Source (0=DNP Database, 1=IED Database)	16
5 to 7	Not Used	

For these data types the qualifier used in the data request is dependent on the Point Count and Address in Slave fields in the command as follows:

- If Address in Slave = 0 & Point Count > 0, then use Qualifier 17h or 28h (Point Count specified starting at point 0)
- If Address in Slave > 0 & Point Count > 0, then use Qualifier 17h or 28h (points from Address in Slave to Address in Slave + Point Count -1)
- If Point Count ≤ 0, then ignore because this is illegal for outputs.

Node Address

This parameter specifies the IED unit address on the DNP network to consider with the command. The parameter has a range of 0 to 65535. The value of 65535 is reserved for broadcast messages. Verify that the slave configuration information is set up in the module for each slave defined in the command list.

Data Object

This parameter specifies the DNP object type in the command. Valid objects for the module are 1, 2, 12, 20, 21, 30, 32, 41, 50, 60 and 80. A value of 0 is permitted in this field for a set of special commands.

Data Variation

This parameter is specific to the object type selected.

Cmd Function

This parameter specifies the DNP function for the command list object. The object type determines the value of the functions permitted. For example, the only function permitted for binary input data points is the read function (Function Code 1). For counter and output objects, more functions are available.

Device Address

This value must be greater-than or equal to zero. If it is set to a value less-than zero, the command will be ignored. This parameter specifies the starting point address to consider in the IED unit.

Point Count

This parameter defines the number of points in the IED unit. Refer to the discussion above for the interpretation of this parameter's values for the different object types.

DNP DB Address

This parameter defines the starting location in the DNP database to be used with the command. If the parameter has a value of -1, the DNP database is not used with the point.

IED DB Address

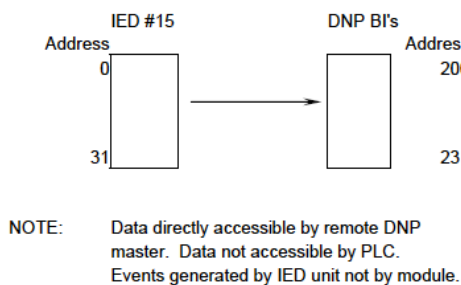
This parameter defines the starting location in the IED database to be used with the command. If the parameter has a value of -1, the IED database is not used with the point.

Poll Interval

This parameter specifies the minimum frequency at which the module should execute the command. The value is entered in units of seconds. For example, to execute a command every 10 seconds, enter a value of 10 in the field. A value of 0 for the parameter implies that the command should be executed every scan of the list.

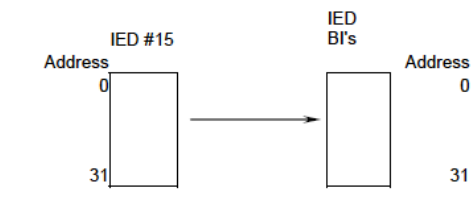
Binary Input Command Examples

	Port/ Fig	Slave	Object	Var	Func	Addr	Pnt Cnt	DNP DB	IED DB	Poll Int
Word	0	1	2	3	4	5	6	7	8	9
Value	6	15	1	0	1	0	-32	200	-1	2



Command for Port 2, Enabled, RBE flag not set.
 IED Unit 15 is to be polled.
 Object type is 1 (Binary Input).
 Variation of 0 (default variation).
 Function 1 is for a read.
 Slave address ignored (Qual 6, all points).
 Point count of -32 indicates only first 32 points are to be used.
 DNP DB address of 200 is where first data point will be placed.
 IED DB is not used (-1).
 Poll command every 2-seconds.

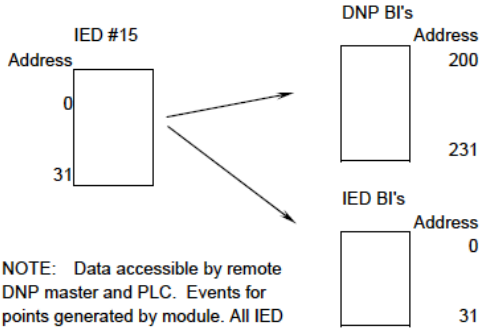
	Port/ Flg	Slave	Object	Var	Func	Addr	Pnt Cnt	DNP DB	IED DB	Poll Int
Word	0	1	2	3	4	5	6	7	8	9
Value	6	15	1	0	1	0	-32	-1	0	2



NOTE: Data not accessible by remote DNP master. Data accessible by PLC. No DNP data so RBE flag ignored.

Command for Port 2, Enabled.
IED Unit 15 is to be polled.
Object type is 1 (Binary Input).
Variation of 0 (default variation).
Function 1 is for a read.
Slave address ignored (Qual 6, all points).
Point count of -32 indicates only first 32 points are to be used.
DNP DB is not used (-1).
IED DB address of 0 is where first data point is placed.
Poll command every 2-seconds.

	Port/ Flg	Slave	Object	Var	Func	Addr	Pnt Cnt	DNP DB	IED DB	Poll Int
Word	0	1	2	3	4	5	6	7	8	9
Value	14	15	1	0	1	0	-32	200	0	2



NOTE: Data accessible by remote DNP master and PLC. Events for points generated by module. All IED generated events are ignored.

Command for Port 2, Enabled, RBE Flag Set.
IED Unit 15 is to be polled.
Object type is 1 (Binary Input).
Variation of 0 (default variation).
Function 1 is for a read.
Slave address ignored (Qual 6, all points).
Point count of -32 indicates only first 32 points are to be used.
DNP DB address of 200 is where first data point is placed.
IED DB address of 0 is where first data point is placed.
Poll command every 2-seconds.

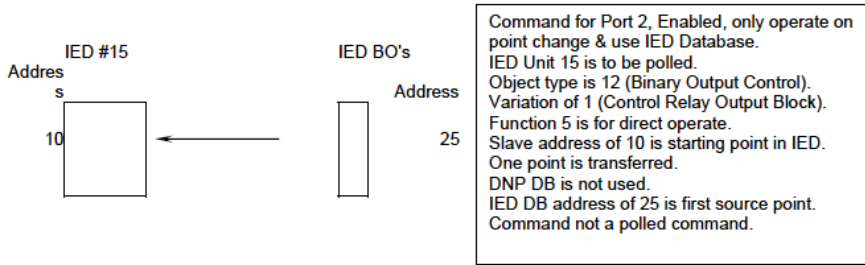
Binary Output Command Examples

	Port/ Flg	Slave	Object	Var	Func	Addr	Pnt Cnt	DNP DB	IED DB	Poll Int
Word	0	1	2	3	4	5	6	7	8	9
Value	14	15	12	1	5	10	2	200	0	0

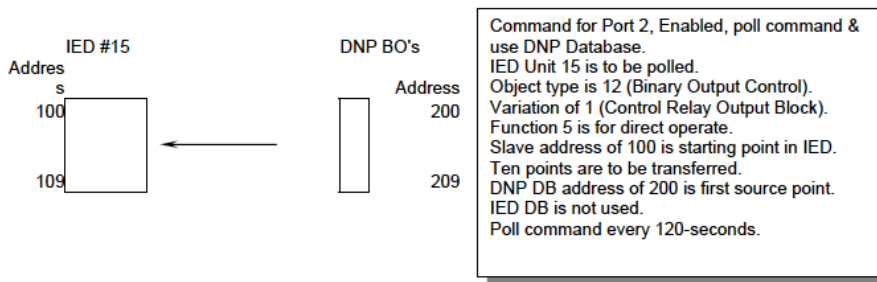


Command for Port 2, Enabled, only operate on point change & use DNP Database.
IED Unit 15 is to be polled.
Object type is 12 (Binary Output Control).
Variation of 1 (Control Relay Output Block).
Function 5 is for direct operate.
Slave address of 10 is starting point in IED.
Two points are to be transferred.
DNP DB address of 200 is first source point.
IED DB is not used.
Command not a polled command.

	Port/ Flg	Slave	Object	Var	Func	Addr	Pnt Cnt	DNP DB	IED DB	Poll Int
Word	0	1	2	3	4	5	6	7	8	9
Value	30	15	12	1	5	10	1	0	25	0



	Port/ Flg	Slave	Object	Var	Func	Addr	Pnt Cnt	DNP DB	IED DB	Poll Int
Word	0	1	2	3	4	5	6	7	8	9
Value	6	15	12	1	5	100	10	200	0	120



6.3 Uploading and Downloading the Configuration File

ProSoft modules are shipped with a pre-loaded configuration file. In order to edit this file, you must transfer the file from the module to your PC. After editing, you must transfer the file back to the module.

This section describes these procedures.

Important: The illustrations of configuration/debug menus in this section are intended as a general guide, and may not exactly match the configuration/debug menus in your own module. For specific information about the configuration/debug menus in your module, refer to The Configuration/Debug Menu (page 111).

6.3.1 Required Hardware

You can connect directly from your computer's serial port to the serial port on the module to view configuration information, perform maintenance, and send (upload) or receive (download) configuration files.

ProSoft Technology recommends the following minimum hardware to connect your computer to the module:

- 80486 based processor (Pentium preferred)
- 1 megabyte of memory
- At least one UART hardware-based serial communications port available. USB-based virtual UART systems (USB to serial port adapters) often do not function reliably, especially during binary file transfers, such as when uploading/downloading configuration files or module firmware upgrades.
- A null modem serial cable.

6.3.2 Required Software

In order to send and receive data over the serial port (COM port) on your computer to the module, you must use a communication program (terminal emulator).

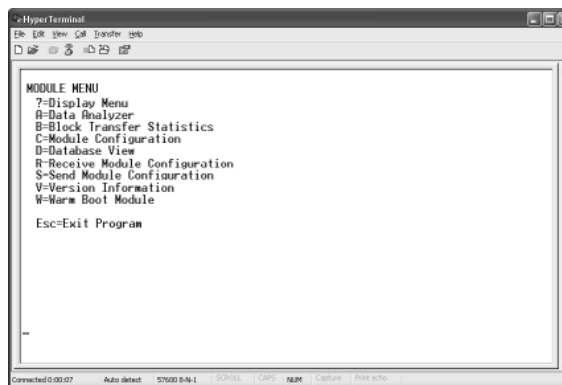
A simple communication program called HyperTerminal is pre-installed with recent versions of Microsoft Windows operating systems. If you are connecting from a machine running DOS, you must obtain and install a compatible communication program. The following table lists communication programs that have been tested by ProSoft Technology.

DOS	ProComm, as well as several other terminal emulation programs
Windows 3.1	Terminal
Windows 95/98	HyperTerminal
Windows NT/2000/XP	HyperTerminal

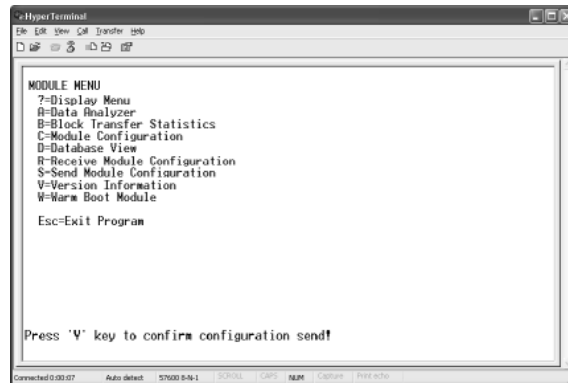
The module uses the Ymodem file transfer protocol to send (upload) and receive (download) configuration files from your module. If you use a communication program that is not on the list above, please be sure that it supports Ymodem file transfers.

6.3.3 Transferring the Configuration File to Your PC

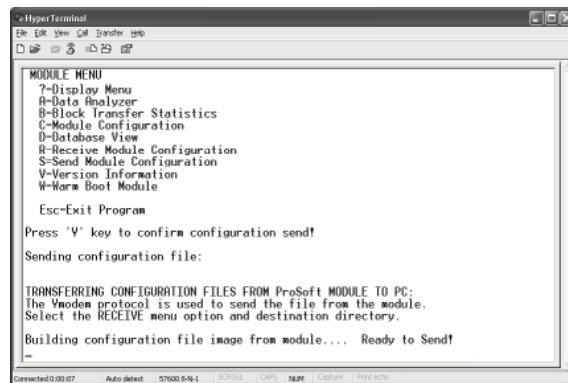
- 1 Connect your PC to the Configuration/Debug port of the module using a terminal program such as HyperTerminal. Press **[?]** to display the main menu.



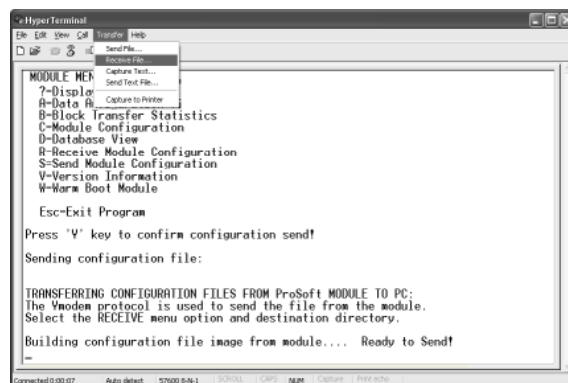
- 2 Press **[S]** (Send Module Configuration). The message "Press Y key to confirm configuration send!" is displayed at the bottom of the screen.



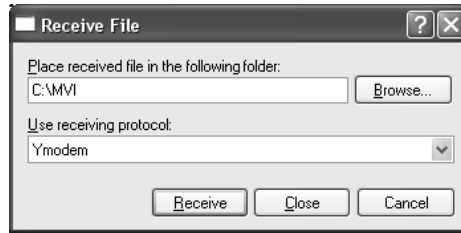
- 3 Press **[Y]**. The screen now indicates that the module is ready to send.



- 4 From the **Transfer** menu in HyperTerminal, select **Receive File**. This action opens the Receive File dialog box.

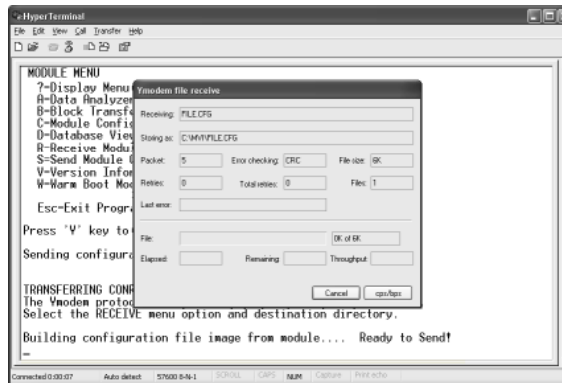


- Use the Browse button to choose a folder on your computer to save the file.

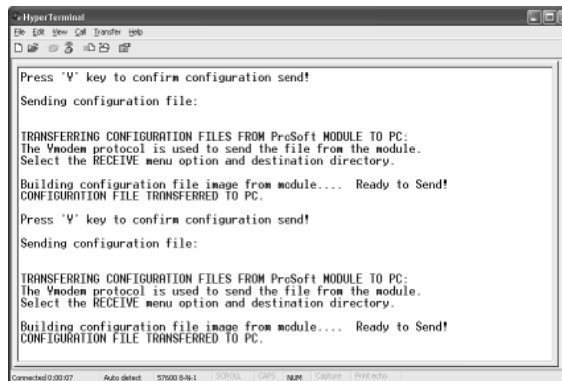


- Note:** ProSoft Technology suggests that you upload the configuration file pre-loaded on your module. However, configuration files are also available on the ProSoft CD as well as the ProSoft Technology web site at <http://www.prosoft-technology.com>.

- Select Ymodem as the receiving protocol.
- Click the Receive button. This action opens the Ymodem File Receive dialog box, showing the progress of your file transfer.



When the configuration file has been transferred to your PC, the dialog box will indicate that the transfer is complete.



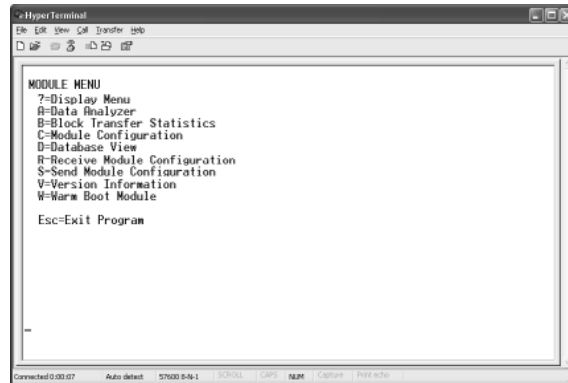
The configuration file is now on your PC at the location you specified.

- You can now open and edit the file in a text editor such as Notepad. When you have finished editing the file, save it and close Notepad.

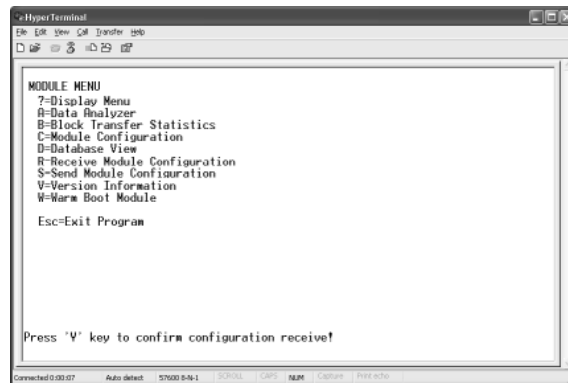
6.3.4 Transferring the Configuration File to the Module

Perform the following steps to transfer a configuration file from your PC to the module.

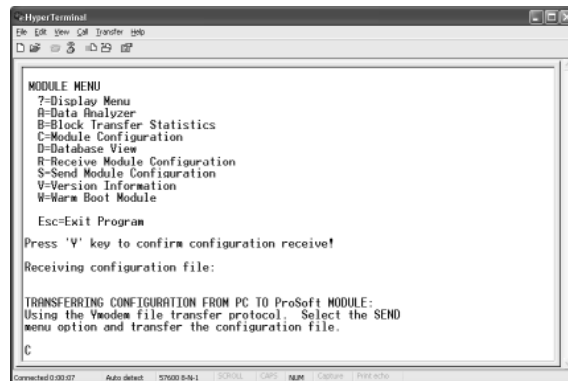
- 1 Connect your PC to the Configuration/Debug port of the module using a terminal program such as HyperTerminal. Press [?] to display the main menu.



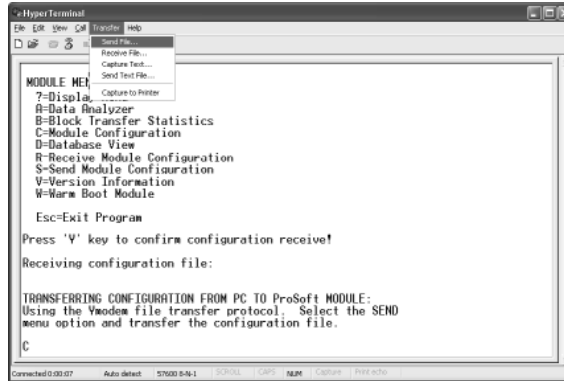
- 2 Press [R] (Receive Module Configuration). The message "Press Y key to confirm configuration receive!" is displayed at the bottom of the screen.



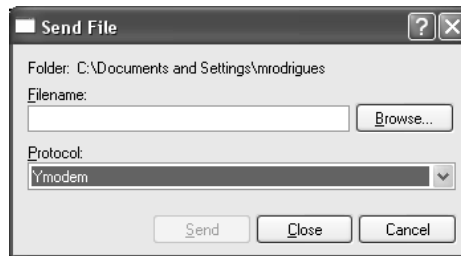
- 3 Press [Y]. The screen now indicates that the PC is ready to send.



- From the **Transfer** menu in HyperTerminal, select **Send File**.



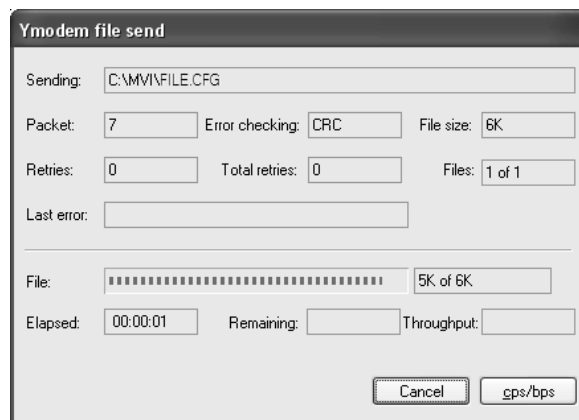
The Send File dialog appears.



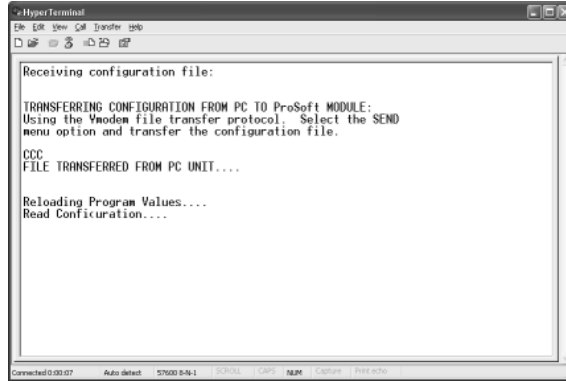
- Use the Browse button to locate the configuration file your computer.

Note: This procedure assumes that you are uploading a newly edited configuration file from your PC to the module. However, configuration files are also available on the ProSoft CD as well as the ProSoft Technology web site at <http://www.prosoft-technology.com>.

- Select **Ymodem** as the protocol.
- Click the Send button. This action opens the Ymodem File Send dialog box.



When the file transfer is complete, the module's configuration/debug screen indicates that the module has reloaded program values, and displays information about the module.



8 Your module now contains the new configuration.

6.4 Verification and Troubleshooting

You can now verify that the module is configured properly by viewing parameters that you specified in the configuration file. This is done using the module's Main Menu. If you are not already at the Main menu, press **[Shift][/]**.

Use the database menu to verify that data appears in registers that you've mapped. Refer to Diagnostics and Troubleshooting (page 111) for information on accessing information on the operation of the module.

7 Diagnostics and Troubleshooting

In This Chapter

❖ The Configuration/Debug Menu	111
❖ Required Hardware	112
❖ Required Software.....	112
❖ Using the Configuration/Debug Port.....	113
❖ LED Status Indicators.....	121

The module provides information on diagnostics and troubleshooting in the following forms:

- Status data values are transferred from the module to the processor.
- Data contained in the module can be viewed through the Configuration/Debug port attached to a terminal emulator.
- LED status indicators on the front of the module provide information on the module's status.

7.1 The Configuration/Debug Menu

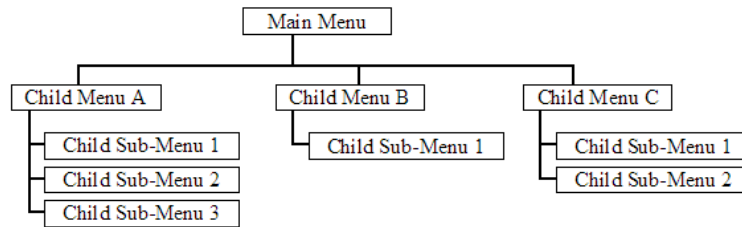
The Configuration and Debug menu for this module is arranged as a tree structure, with the Main Menu at the top of the tree, and one or more sub-menus for each menu command. The first menu you see when you connect to the module is the Main menu.

Because this is a text-based menu system, you enter commands by typing the command letter from your computer keyboard in the terminal application (for example, HyperTerminal). The module does not respond to mouse movements or clicks. The command executes as soon as you press the command letter — you do not need to press **[Enter]**. When you type a command letter, a new screen will be displayed in your terminal application.

7.1.1 Navigation

All of the sub-menus for this module contain commands to redisplay the menu or return to the previous menu. You can always return from a sub-menu to the next higher menu by pressing **[M]** on your keyboard.

The organization of the menu structure is represented in simplified form in the following illustration:



The remainder of this section shows you the menus available for this module, and briefly discusses the commands available to you.

7.1.2 Keystrokes

The keyboard commands on these menus are almost always non-case sensitive. You can enter most commands in lower case or capital letters.

The menus use a few special characters ([?], [-], [+], [@]) that must be entered exactly as shown. Some of these characters will require you to use the **[Shift]**, **[Ctrl]** or **[Alt]** keys to enter them correctly. For example, on US English keyboards, enter the [?] command as **[Shift][/]**.

Also, take care to distinguish capital letter **[I]** from lower case letter **[i]** (L) and number **[1]**; likewise for capital letter **[O]** and number **[0]**. Although these characters look nearly the same on the screen, they perform different actions on the module.

7.2 Required Hardware

You can connect directly from your computer's serial port to the serial port on the module to view configuration information, perform maintenance, and send (upload) or receive (download) configuration files.

ProSoft Technology recommends the following minimum hardware to connect your computer to the module:

- 80486 based processor (Pentium preferred)
- 1 megabyte of memory
- At least one UART hardware-based serial communications port available. USB-based virtual UART systems (USB to serial port adapters) often do not function reliably, especially during binary file transfers, such as when uploading/downloading configuration files or module firmware upgrades.
- A null modem serial cable.

7.3 Required Software

In order to send and receive data over the serial port (COM port) on your computer to the module, you must use a communication program (terminal emulator).

A simple communication program called HyperTerminal is pre-installed with recent versions of Microsoft Windows operating systems. If you are connecting from a machine running DOS, you must obtain and install a compatible communication program. The following table lists communication programs that have been tested by ProSoft Technology.

DOS	ProComm, as well as several other terminal emulation programs
Windows 3.1	Terminal
Windows 95/98	HyperTerminal
Windows NT/2000/XP	HyperTerminal

The module uses the Ymodem file transfer protocol to send (upload) and receive (download) configuration files from your module. If you use a communication program that is not on the list above, please be sure that it supports Ymodem file transfers.

7.4 Using the Configuration/Debug Port

To connect to the module's Configuration/Debug port:

- 1 Connect your computer to the module's port using a null modem cable.
- 2 Start the communication program on your computer and configure the communication parameters with the following settings:

Baud Rate	57,600
Parity	None
Data Bits	8
Stop Bits	1
Software Handshaking	None

- 3 Open the connection. When you are connected, press the [?] key on your keyboard. If the system is set up properly, you will see a menu with the module name followed by a list of letters and the commands associated with them.

If there is no response from the module, follow these steps:

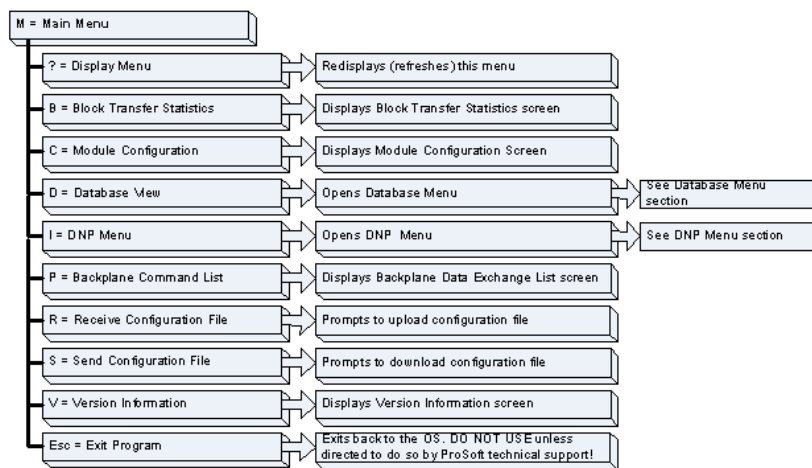
- 1 Verify that the null modem cable is connected properly between your computer's serial port and the module. A regular serial cable will not work.
- 2 Verify that another program is not controlling the COM port.
- 3 Verify that your communication software is using the correct settings for baud rate, parity and handshaking.
- 4 On computers with more than one serial port, verify that your communication program is connected to the same port that is connected to the module.

If you are still not able to establish a connection, you can contact ProSoft Technology Technical Support for further assistance.

7.4.1 Main Menu

When you first connect to the module from your computer, your terminal screen will be blank. To activate the main menu, press the **[?]** key on your computer's keyboard. If the module is connected properly, the following menu will appear on your terminal screen:

Caution: Some of the commands available to you from this menu are designed for advanced debugging and system testing only, and can cause the module to stop communicating with the processor or with other devices, resulting in potential data loss or other failures. Only use these commands if you are specifically directed to do so by ProSoft Technology Technical Support staff. Some of these command keys are not listed on the menu, but are active nevertheless. Please be careful when pressing keys so that you do not accidentally execute an unwanted command.



Viewing Module Configuration

Press **[C]** to view the Module Configuration screen.

Use this command to display the current configuration and statistics for the module.

Viewing Block Transfer Statistics

Press **[B]** from the Main Menu to view the Block Transfer Statistics screen.

Use this command to display the configuration and statistics of the backplane data transfer operations between the module and the processor. The information on this screen can help determine if there are communication problems between the processor and the module.

Tip: To determine the number of blocks transferred each second, mark the numbers displayed at a specific time. Then some seconds later activate the command again. Subtract the previous numbers from the current numbers and divide by the quantity of seconds passed between the two readings.

Opening the Database Menu

Press **[D]** to open the Database View menu. Use this menu command to view the current contents of the module's database.

Opening the DNP Menu

Press **[I]** from the Main Menu to open the DNP Menu. This menu allows you to view all data associated with the DNP Server driver. For more information about the commands on this menu, refer to DNP Menu.

Viewing the Backplane Command List

Press **[P]** from the Main Menu to view the Backplane Data Exchange List. Use this command to display the configuration and statistics of the backplane data transfer operations.

```

BACKPLANE DATA EXCHANGE LIST -- COMMANDS 0 TO 9
TYPE  DBREG  DBTYPE  ADDRESS  COUNT  LASTERR
0      0      0        0        0      0X0000
0      0      0        0        0      0X0000
0      0      0        0        0      0X0000
0      0      0        0        0      0X0000
0      0      0        0        0      0X0000
0      0      0        0        0      0X0000
0      0      0        0        0      0X0000
0      0      0        0        0      0X0000
0      0      0        0        0      0X0000
0      0      0        0        0      0X0000

```

Tip: Repeat this command at one-second intervals to determine the number of blocks transferred each second.

Receiving the Configuration File

Press **[R]** to download (receive) the current configuration file from the module. For more information on receiving and sending configuration files, please see Uploading and Downloading the Configuration File (page 104).

Sending the Configuration File

Press **[S]** to upload (send) an updated configuration file to the module. For more information on receiving and sending configuration files, please see Uploading and Downloading the Configuration File (page 104).

Viewing Version Information

Press **[V]** to view Version information for the module.

Use this command to view the current version of the software for the module, as well as other important values. You may be asked to provide this information when calling for technical support on the product.

Values at the bottom of the display are important in determining module operation. The Program Scan Counter value is incremented each time a module's program cycle is complete.

Tip: Repeat this command at one-second intervals to determine the frequency of program execution.

Warm Booting the Module

Caution: Some of the commands available to you from this menu are designed for advanced debugging and system testing only, and can cause the module to stop communicating with the processor or with other devices, resulting in potential data loss or other failures. Only use these commands if you are specifically directed to do so by ProSoft Technology Technical Support staff. Some of these command keys are not listed on the menu, but are active nevertheless. Please be careful when pressing keys so that you do not accidentally execute an unwanted command.

Press **[W]** from the Main Menu to warm boot (restart) the module. This command will cause the program to exit and reload, refreshing configuration parameters that must be set on program initialization. Only use this command if you must force the module to re-boot.

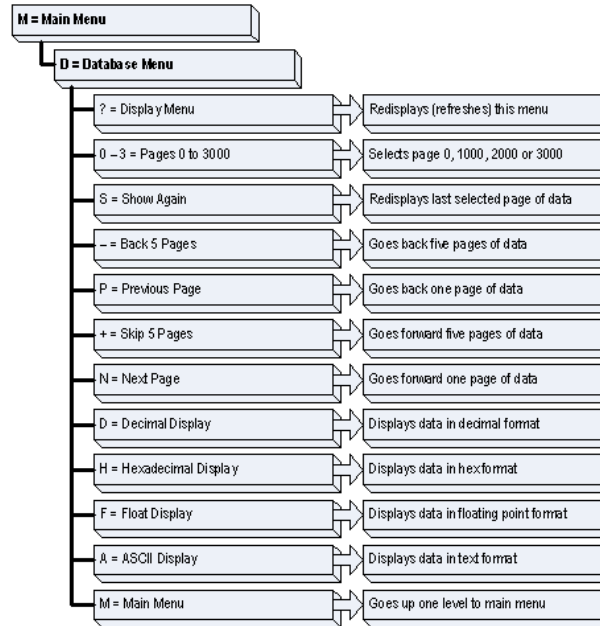
Exiting the Program

Caution: Some of the commands available to you from this menu are designed for advanced debugging and system testing only, and can cause the module to stop communicating with the processor or with other devices, resulting in potential data loss or other failures. Only use these commands if you are specifically directed to do so by ProSoft Technology Technical Support staff. Some of these command keys are not listed on the menu, but are active nevertheless. Please be careful when pressing keys so that you do not accidentally execute an unwanted command.

Press **[Esc]** to restart the module and force all drivers to be loaded. The module will use the configuration stored in the module's Flash ROM to configure the module.

7.4.2 Database View Menu

Press **[D]** from the Main Menu to open the Database View menu. Use this menu command to view the current contents of the module's database. Press **[?]** to view a list of commands available on this menu.



Viewing Register Pages

To view sets of register pages, use the keys described below:

Command	Description
[0]	Display registers 0 to 99
[1]	Display registers 1000 to 1099
[2]	Display registers 2000 to 2099

And so on. The total number of register pages available to view depends on your module's configuration.

Displaying the Current Page of Registers Again

```

DATABASE DISPLAY 0 TO 99 <DECIMAL>
100  101  102  4  5  6  7  8  9  10
 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  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  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0
  
```

This screen displays the current page of 100 registers in the database.

Moving Back Through 5 Pages of Registers

Press **[-]** from the Database View menu to skip back to the previous 500 registers of data.

Viewing the Previous 100 Registers of Data

Press **[P]** from the Database View menu to display the previous 100 registers of data.

Skipping 500 Registers of Data

Hold down **[Shift]** and press **[=]** to skip forward to the next 500 registers of data.

Viewing the Next 100 Registers of Data

Press **[N]** from the Database View menu to select and display the next 100 registers of data.

Viewing Data in Decimal Format

Press **[D]** to display the data on the current page in decimal format.

Viewing Data in Hexadecimal Format

Press **[H]** to display the data on the current page in hexadecimal format.

Viewing Data in Floating Point Format

Press **[F]** from the Database View menu. Use this command to display the data on the current page in floating point format. The program assumes that the values are aligned on even register boundaries. If floating-point values are not aligned as such, they are not displayed properly.

Viewing Data in ASCII (Text) Format

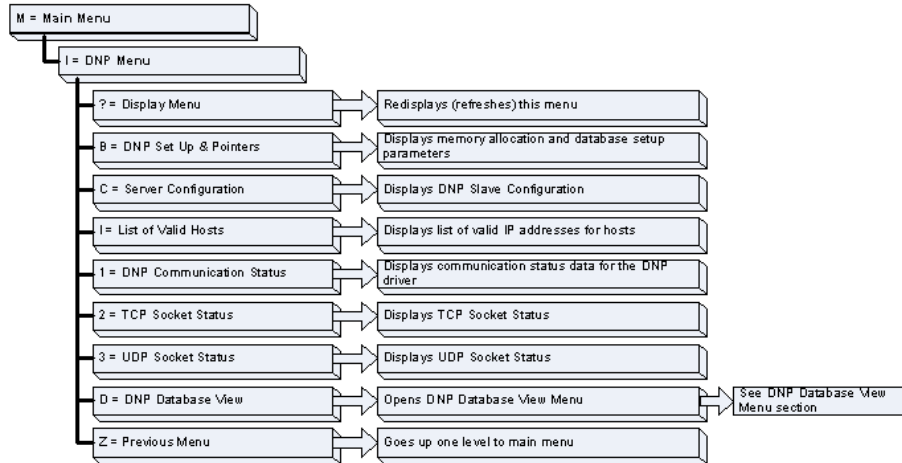
Press **[A]** to display the data on the current page in ASCII format. This is useful for regions of the database that contain ASCII data.

Returning to the Main Menu

Press **[M]** to return to the Main Menu.

7.4.3 DNP Menu

Press **[I]** from the Main Menu to open the DNP Menu. This menu allows you to view all data associated with the DNP Server driver.



Viewing Memory Allocation and Database Setup Parameters

Press **[B]** from the DNP Menu to view memory allocation and database setup parameters.

```

***** DNP SET UP & POINTERS *****
BIPtr   = B51C:0004  COUNTI = 2
AIPtr   = B51C:0008  COUNTI = 50  DEADBAND = 1000
CPtr    = B51C:006C  COUNTI = 29
BOPtr   = B51C:00E0  COUNTI = 2
AOPtr   = B51C:00E4  COUNTI = 40
FCPtr   = B51C:0134  COUNTI = 29
BIEventPtr = B51C:0242  COUNTI = 100 <WITH TIME><TIME SYNC>
AIEventPtr = B51C:068E  COUNTI = 100 <WITH TIME><TIME SYNC>
BILastPtr = B51C:01A8  AILastPtr = B51C:01A0  EndEventPtr = B51C:0C6A
DATA MEMORY REQUIRED = 3174 BYTES
COMMUNICATION PARAMETERS:
DNP SLAVE ADDRESS = 2
  
```

Viewing Server Configuration Information

Press **[C]** from the DNP Menu.

```

***** DNP SLAVE CONFIGURATION *****
DNP SLAVE ADDRESS = 2
MIN TRANS DELAY = 0          DL CONFRIM MODE = 0          DL CONFIRM TOUT = 1000
MAX DL RETRIES = 2          SEL/OP TIME DUR = 2000      APP CONFRM TOUT = 2000
WRITE TIME INTR = 3600000
UNSOL RESP MODE = 0          UNSOL CLASS #1 = 10        UNSOL CLASS #2 = 10
UNSOL CLASS #3 = 10          UNSOL RESP DELAY= 5000     UNSOL RESP ADDR = 5
ERR/STAT POINTER= 8000       ERR/STAT FREQ = 300        USE IP LIST = 1
  
```

Use this option to determine if the module is set up as desired. If any parameter is not setup correctly, change the configuration file and upload the altered file to the module.

Viewing a List of Valid Hosts

Press [I] to view the list of IP addresses from which the module will accept connections This list is only used if the module configuration parameter, Use IP List, is set to a value other than 0.

Viewing DNP Communication Status

Press [1] from the DNP Menu.

```

IP ADDRESS VALUE      (VALUE)
***** DNP STATISTICS *****
PROGRAM CYCLE COUNT   : 27757      TIME: 00C2A9B20D35
COMMUNICATION STATISTICS:
DNP ERRORS:
DNP PHYSICAL LAYER:  SYNC = 0      OVERRUN = 0      LENGTH = 0
DNP DATA LINK LAYER:  BAD CRC = 0
DNP TRANSPORT LAYER:
USER DATA OVER FLOW = 0      SEQUENCE = 0      ADDRESS = 0
DNP APPLICATION LAYER:
BAD FUNC = 0      OBJECT UNKNOWN = 0      OUT OF RANGE = 0
OVERFLOW = 0      MASTER MULTI-FRAME = 0
DNP MESSAGE STATS:
TOTAL FRAMES RECEIVED = 0
TCP/IP FRAMES RECEIVED = 0      FRAMES TRANSMITTED = 0
UDP/IP FRAMES RECEIVED = 0      FRAMES TRANSMITTED = 0
EVENT STATISTICS: (UNSOLED ENABLED: CLASS 1: NO CLASS 2: NO CLASS 3: NO)
BINARY INPUT:  TOTAL COUNT = 0      EVENT COUNT = 0      MAXIMUM = 200
ANALOG INPUT:  TOTAL COUNT = 0      EVENT COUNT = 0      MAXIMUM = 200
DNP INTERNAL INDICATION BITS:
| RESTART || TIME |
    
```

Should be all zeros
 Check ObjectList in appendix
 Message stats should match
 Check database

Use this command to view the communication status data for the DNP driver.

Viewing TCP Socket Status

Press [2] from the DNP Menu.

```

TCP SOCKET STATUS
Rx Count      : 0
Tx Count      : 0
Tx State      : 0
TCP State     : 0
Busy Flag     : 0
App Frame     : 0
Tx Frame     : 1
Packet Length : 0
    
```

This screen shows the following parameters:

- **Rx Count:** Number of messages received on TCP socket
- **Tx Count:** Number of messages transmitted on TCP socket
- **Tx State:** 0 = Not Transmitting, 1 = Transmitting
- **TCP State:** Value used for TCP/IP socket state machine
- **Busy Flag:** 0 = Not Busy, 1 = TCP has control of DNP Server, 2 = UDP has control of DNP Server, 3 = Unsolicited message being sent
- **App Frame:** 0 = No application data from frame, 1 = Application data available
- **Tx Frame:** 0 = Data link level frame ready to send, 1 = Data link level message not ready to send
- **Packet Length:** Length of message left to process

Viewing UDP Socket Status

Press **[3]** from the DNP Menu.

```

UDP SOCKET STATUS
Rx Count      : 0
Tx Count      : 0
Tx State      : 0
UDP State     : 0
Busy Flag     : 0
App Frame     : 0
Tx Frame      : 1
Packet Length : 0

```

This screen shows the following parameters:

- **Rx Count:** Number of messages received on UDP socket
- **Tx Count:** Number of messages transmitted on UDP socket
- **Tx State:** 0 = Not Transmitting, 1 = Transmitting
- **TCP State:** Value used for UDP/IP socket state machine
- **Busy Flag:** 0 = Not Busy, 1 = TCP has control of DNP Server, 2 = UDP has control of DNP Server, 3 = Unsolicited message being sent
- **App Frame:** 0 = No application data from frame, 1 = Application data available
- **Tx Frame:** 0 = Data link level frame ready to send, 1 = Data link level message not ready to send
- **Packet Length:** Length of message left to process

Opening the DNP Database View Menu

Press **[D]** to open the DNP Database View menu. Use this command to display the database associated with each data type.

7.5 LED Status Indicators

The LEDs indicate the module's operating status as follows:

ProSoft Module	Color	Status	Indication
PRT1	Green	On	Data is being transferred between the module and a remote terminal using the Configuration/Debug port.
		Off	No data is being transferred on the Configuration/Debug port.
PRT2	Green	On	Data is being transferred
		Off	No data
PRT3	Green	On	Data is being transferred
		Off	No data
ERR1	Red	Off	The PTQ-DNP is working normally.
		On	The PTQ-DNP module program has recognized an application error.
ERR2	N/A		Not used in application

ProSoft Module	Color	Status	Indication
ERR3	Red	On	Configuration Error
Active	Green	On	The LED is on when the module recognizes a processor and is able to communicate if the [Backplane Data Movement] section specifies data transfer commands.
		Off	The LED is off when the module is unable to speak with the processor. The processor either absent or not running.
BAT	Red	Off	The battery voltage is OK and functioning.
		On	The battery voltage is low or the battery is not present. The battery LED will illuminate briefly upon the first installation of the module or if the unit has been un-powered for an extended period of time. This behavior is normal, however should the LED come on in a working installation please contact ProSoft Technology.

If your module is not operating, and the status LEDs are not illustrated in the table above, please call ProSoft Technology for technical assistance.

8 Reference

In This Chapter

❖ Product Specifications	123
❖ Functional Overview	125
❖ PTQ-DNP Error Status Table	137
❖ PTQ-DNP Module Internal Indication Bits (IIN Bits) for DNP Server ...	144
❖ DNP Subset Definition	145
❖ Event Size Computation	157
❖ DNP Collision Avoidance	158
❖ Frequently Asked Questions	160

8.1 Product Specifications

The DNP 3.0 Master/Slave Communication Module is a single slot, backplane compatible DNP 3.0 interface solution for the Schneider Electric Quantum or Unity platform. This module provides highly configurable support of both DNP 3.0 Master and Slave implementations.

8.1.1 Features and Benefits

The module supports DNP Subset Level 2 features and some of the Level 3 features allowing the many SCADA and field devices supporting the DNP protocol to be integrated into the Quantum or Unity platform. The module acts as an input/output module between the DNP network and the Modicon backplane. The data transfer from the Quantum or Unity processor is asynchronous from the actions on the DNP network. Databases are user defined and stored in the module to hold the data required by the protocol.

8.1.2 General Specifications

- Single Slot - Quantum backplane compatible
- The module is recognized as an Options module and has access to PLC memory for data transfer
- Configuration data is stored in non-volatile memory in the ProTalk module
- Up to six modules can be placed in a rack
- Local rack - The module must be placed in the same rack as processor.
- Compatible with common Quantum / Unity programming tools.
- Quantum data types supported: 0x, 1x, 3x, 4x
- High speed data transfer across backplane provides quick data update times.
- Sample function blocks available.

8.1.3 Hardware Specifications

Specification	Value
Backplane Current Load	800 mA @ 5 V
Operating Temperature	0 to 60°C (32 to 140°F)
Storage Temperature	-40 to 85°C (-40 to 185°F)
Relative Humidity	5% to 95% (non-condensing)
Vibration	Sine vibration 4-100 Hz in each of the 3 orthogonal axes
Shock	30G, 11 mSec. in each of the 3 orthogonal axes
LED Indicators	Module Status Backplane Transfer Status Serial Port Activity LED Serial Activity and Error LED Status
Configuration Serial Port (Debug)	DB-9M PC Compatible RS-232 only No hardware handshaking
Application Serial Ports (PRT1, PRT2)	DB-9M PC Compatible RS-232/422/485 jumper selectable RS-422/485 screw termination included RS-232 handshaking configurable 500V Optical isolation from backplane
Certifications	cULus, ATEX, CE

8.1.4 Functional Specifications

The module has two DNP protocol ports that can be configured to operate in a Master/Slave or Slave/Slave redundant port configuration. User defined internal register space is accessible to the protocol driver and to the Quantum processor memory.

Redundant Slave Port Operation

When configured in the Slave/Slave port configuration, the module's slave ports operate in a primary and secondary fashion. In this mode, a single host polls the module via redundant physical layer connections.

DNP 3.0 Slave Protocol Specifications

The DNP Slave port accepts DNP commands to control and monitor data stored in the module's DNP Slave databases. If a DNP Master port is configured, a portion of the slave databases can be derived from or can control IED devices connected to the DNP master port.

- Report-by-Exception data is logged to the module's database
- Supports unsolicited messaging
- Each DNP point type is user configurable by point
- Class assignments are completely user-definable on a Type and point basis (BI, AI, FI, DI point types)

- Supports clock synchronization from a master or from the processor
- Support for four octet-strings are supported (object type 110) in the slave driver to return version and other module information
- Up to 400 events are stored for Floats, Binary In, Analog In and Double Inputs
- Configurable event buffer transmission threshold based on count and/or time since last event transmission
- Collision avoidance for redundant port switching
- Special modem AT command string and timing support for dialing out on redundant port

DNP 3.0 Master Protocol Specifications

The DNP 3.0 Master port can be configured as a virtual DNP Master device that actively issues user-defined DNP commands to nodes on the network.

- Supports 300 user defined commands
- Master port logically supports up to 40 slave devices
- Individual command configuration includes conditional or continuous polling, Poll Delay Time
- Slave status and Command status available for transfer to the processor
- Event data received from the slave devices updates the module database (Date and Time stamping is not stored or used by module)
- Special command handling for Digital Output CROB under processor control for pulse output control
- Supports Report-by-Exception and Unsolicited Responses on a Time Interval basis or on a user determined Event Count basis. Analog and Binary input points are supported

DNP 3.0 ports (PRT1 & PRT2)

- User-definable module memory usage
- Full radio, modem and multi-drop support
- Support for the storage and transfer of all DNP data types across the backplane

8.2 Functional Overview

This documentation describes the PTQ-DNP module configuration and setup as it applies to application design. Before attempting to implement this module with a DNP network, verify that the whole design of the system is complete. This includes definition of all the data types and point counts required for each type, all communication parameters required for the network including media type and the use of advanced features such as unsolicited messaging. These must be defined for all master and slave devices on the network. Additionally, the DNP Device Profiles and DNP Subset Definition documents for each device must be reviewed to make sure all the devices will interact on the network as expected. Failure to fully understand these important documents for all devices on the network will usually lead to many problems when implementing the design.

It is important to fully understand the DNP specification as outlined in the Basic Four Documents. These are available to users of the DNP users group. It is recommended that all users of the module have access to these important documents as they define the DNP data types, functions and variations. It will be very difficult to implement the module without an understanding of the protocol and the rules that are defined in the specification. Additionally, potential users should review the DNP Subset and Conformance Test documents and the document that discusses DNP protocol support on Ethernet using the UDP and TCP protocols. These documents provide auxiliary information on the protocol. All of these documents are available to members of the DNP User Group at <http://www.dnp.org> (<http://www.dnp.org>). Please check this site for other important information regarding the DNP protocol.

8.2.1 Design

In order to implement a solution using the module, the processor must be set up using predefined user data structures. This program will interact with the module by sending and receiving data and issuing special control commands.

An internal database in the processor contains the data to be used by the module and the configuration information is stored in the text file, DNP.CFG, stored on the module's non-volatile memory. Before you generate the program or layout the data files, you must first design your system. Time spent doing system design at the outset of the project will greatly enhance the success and ease of development of the project.

Designing the system

System design defines the data requirements of the system, communication parameters, and module functionality. The application developer should refer to the person responsible for the DNP master and slave device configurations to verify that the functionality and data types required for the whole system are consistent. Review the DNP Device Profile and DNP Subset documentation for a definition of the level of DNP support offered by the module.

The following topics describe each element of system design.

Data Requirements

This phase of design defines what data elements are to be interfaced in the processor with the DNP master. The module provides the following data types: digital input, digital output, counter, analog input and analog output. All communications between the DNP master and the PLC is through these data types. Therefore, all data to be used by the system must be contained and configured in one of these data types.

The following illustration shows the databases maintained by the module for the DNP data.

PTQ-DNP Module Databases

Data Area	Code		
DNP DATA	BINARY INPUTS	PLC DATA	101
		IED DATA	
	BINARY OUTPUTS	PLC DATA	102
		IED DATA	
	COUNTER DATA	PLC DATA	103
		IED DATA	
	ANALOG INPUTS	PLC DATA	104
		IED DATA	
	ANALOG OUTPUTS	PLC DATA	105
		IED DATA	
	FLOAT INPUTS	PLC DATA	106
	Reserved	Reserved	107
	FLOAT OUTPUTS	PLC DATA	108
	Reserved	Reserved	109
	FROZEN COUNTER DATA		
	BINARY INPUT EVENTS		
	ANALOG INPUT EVENTS		
	FLOAT INPUT EVENTS		
LAST VALUE DATA	BINARY INPUTS		
	ANALOG INPUTS		
	FLOAT INPUTS		
	DNP BINARY OUTPUTS		
	DNP ANALOG OUTPUTS		
	IED BINARY OUTPUTS		
	IED ANALOG OUTPUTS		
IED DATA	BINARY INPUTS		201
	BINARY OUTPUTS		202
	COUNTER DATA		203
	ANALOG INPUTS		204
	ANALOG OUTPUTS		205
RBE FLAGS	BINARY INPUT		
	ANALOG INPUT		

Command Database

List of Commands with the following parameters for each node in list:

- Port/Flags
- Slave Address
- Object

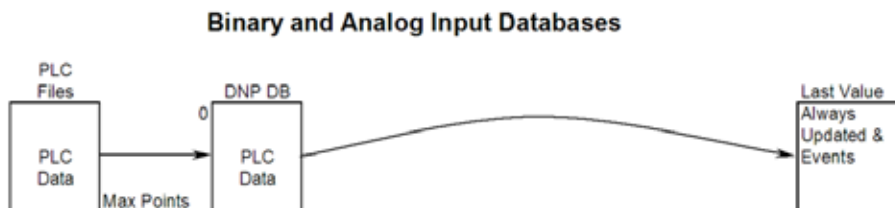
- Variation
- Function
- Address in Slave
- Point Count
- DNP DB Address
- IED DB Address
- Poll Interval
- Polling Time
- Last Error Code
- Next Pointer

Commands are issued based on the following criteria:

- Commands Issued Each Scan
 - Enabled
 - Poll Interval = 0
 - BO & AO have Exception bit = 0
- Commands Issued at Poll Time
 - Enabled
 - Poll Interval > 0
 - BO & AO have Exception bit = 0
- Commands Issued on Data Change (BO & AO)
 - Enabled
 - BO & AO have Exception bit = 1
- Commands Issued by PLC

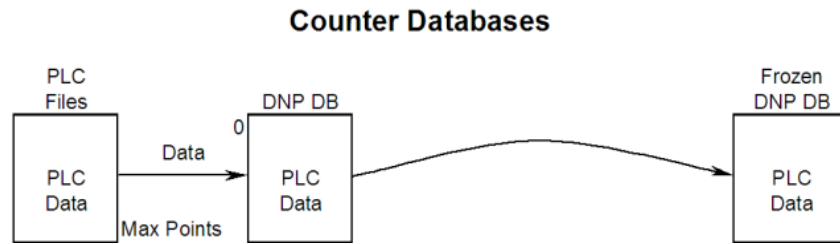
The module maintains the databases using data acquired from the PLC and DNP master attached network port.

The following illustration shows the interaction of the binary and analog input points with the databases.



All data for these data types is derived from the processor and is passed to the module over the backplane. The module will constantly monitor for changes in this data and generate event messages when point values change. For binary input points, events will be generated on any state change. For analog input points, events will be generated for points that have a current value outside of the user-set deadband based on the last value used for an event.

The following illustration shows the interaction of the counter points with the databases.

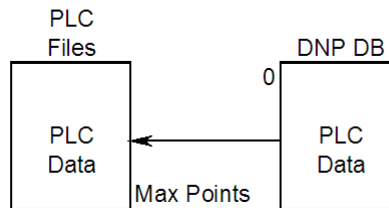


This data is constantly sourced from the processor and placed in the module's internal database. This information is available to the remote master for monitoring. When the module receives a freeze command from the master unit, it will copy the current counter values into the frozen counter database area. The remote master can then monitor this information. If the module receives a counter freeze with reset command, the current counter values will be passed to the frozen counter database and only the module's values will be set to 0.

Note: This data is not sent to the controller, and the zero data be overwritten by the counter data contained in the controller. Therefore, the freeze with reset should not be used with this module. The results will not be as expected. There is no way to guarantee that counts will not be lost during the reset step in the module and controller. As a result, this feature was not implemented in the module.

The following illustration shows the interaction of the binary and analog output points with the databases.

Binary and Analog Output Databases

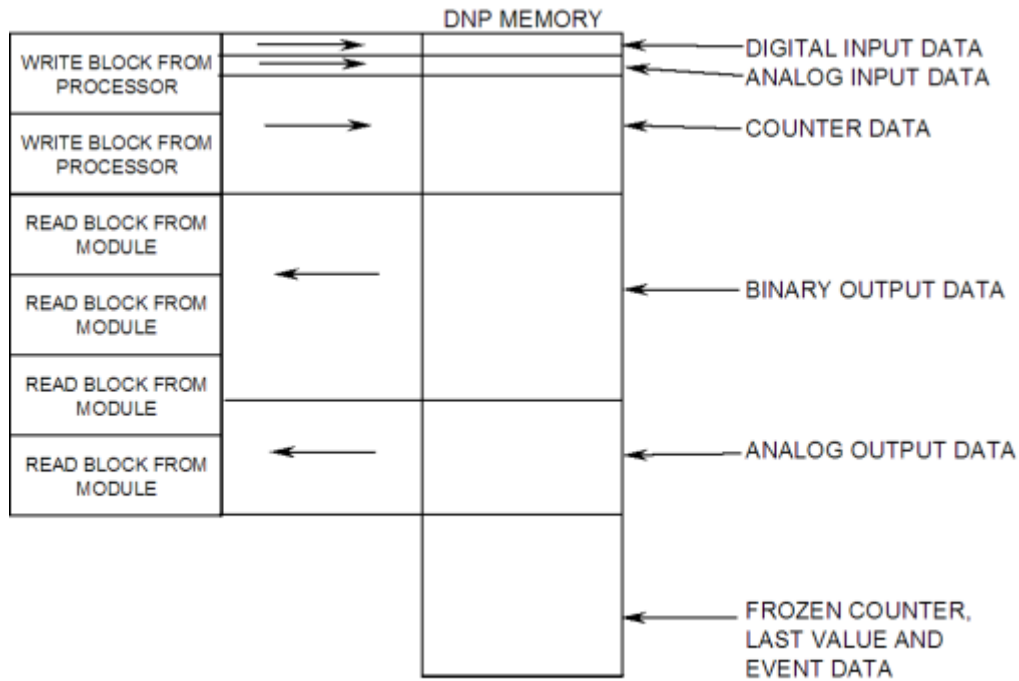


Output data is sourced from the controlling master station and passed to the processor over backplane from the module. These data are used in the ladder logic to control operations and I/O in the processor.

Data Transfer Interface

The following figure displays the direction of movement of the DNP database data between the module and the processor.

Backplane Commands

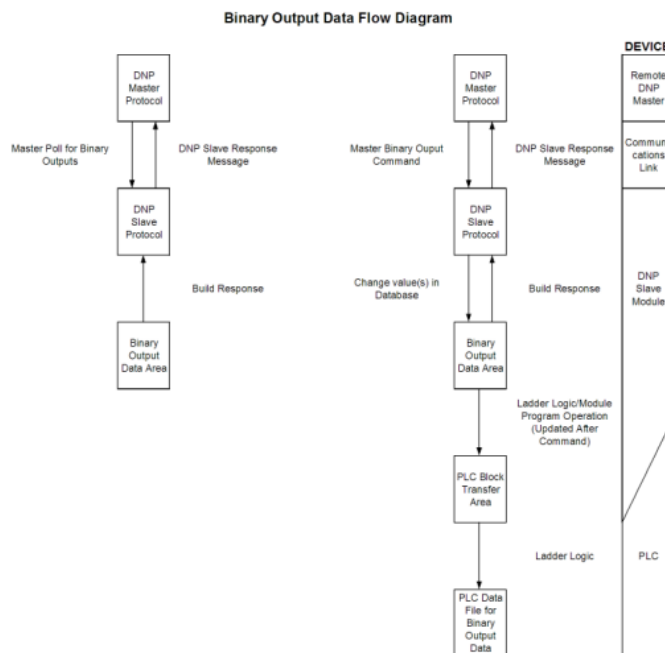


DNP Digital Input Data

This data type stores the binary value of 1 or 0. The size of this data area is determined from the configuration parameter Binary Inputs (number of words, each containing 16 binary input points). These data are transferred to the module from the PLC using the read operation. Therefore, these data are read-only for the module and the DNP master unit communicating with the module. When the module receives a new block of this data from the PLC, it compares the new values to those currently in the database. If there is a change in any of the data, the module will generate an event message for the points that change.

DNP Digital Output Data

This data type stores digital control and command state data received from the DNP master unit with a value of 1 or 0. The size of this data area is determined from the configuration parameter Binary Outputs (defines number of words, each containing 16 binary output points). These data are transferred from the module to the PLC using the write operation. Therefore, these data are read-only for the PLC, as the PLC cannot directly alter these values in module. It is the responsibility of the DNP master unit to maintain this data. For example, if the DNP master sets a digital point on, it will remain on until the master resets the point. A data flow diagram for the digital output data is shown in the following figure.

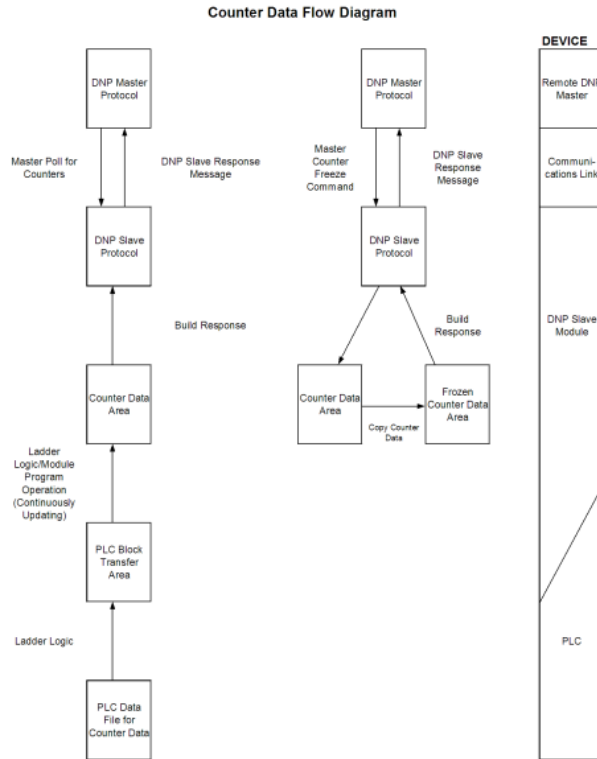


DNP Counter Data

This data type stores accumulated count data. These data are stored in the module in a double word value and have a data range of 0 to 4,294,967,296. The size of this data area is determined from the configuration parameter Counters. The PLC transfers data of this type to the module using the read operation. The module maintains two values for each counter point: a current running value and a frozen value. The DNP master must send the freeze command to the module in order to transfer the current running values to the frozen area.

Note: The freeze-reset command is not supported in the data transfer operation. There is no way to guarantee counts will not be lost using the freeze-reset operation, therefore, this feature is not implemented.

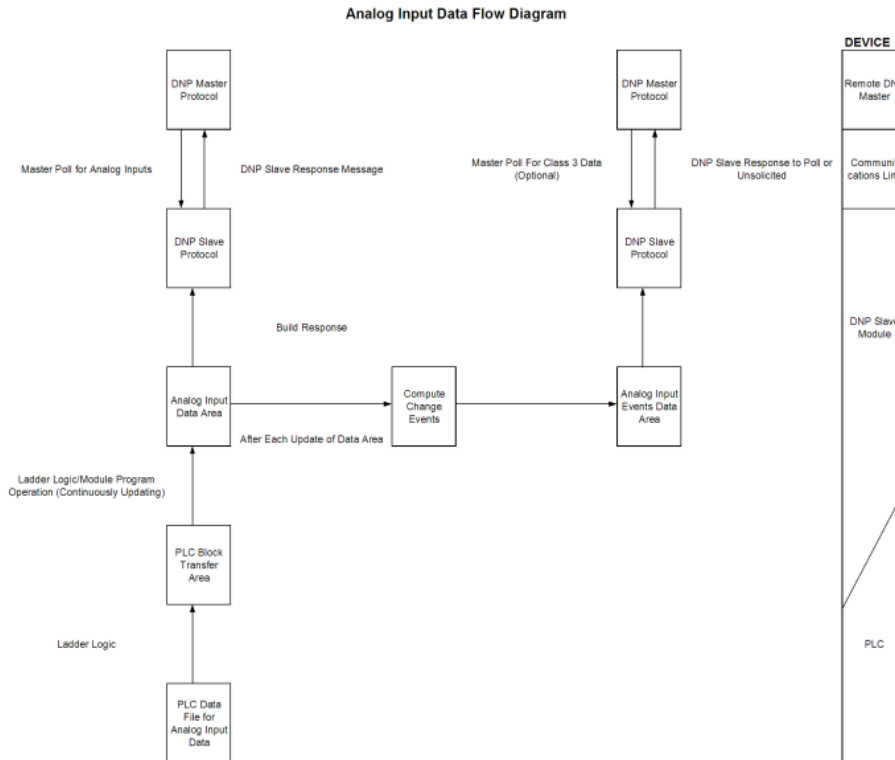
A data flow diagram for the counter data is shown in the following figure.



DNP Analog Input Data

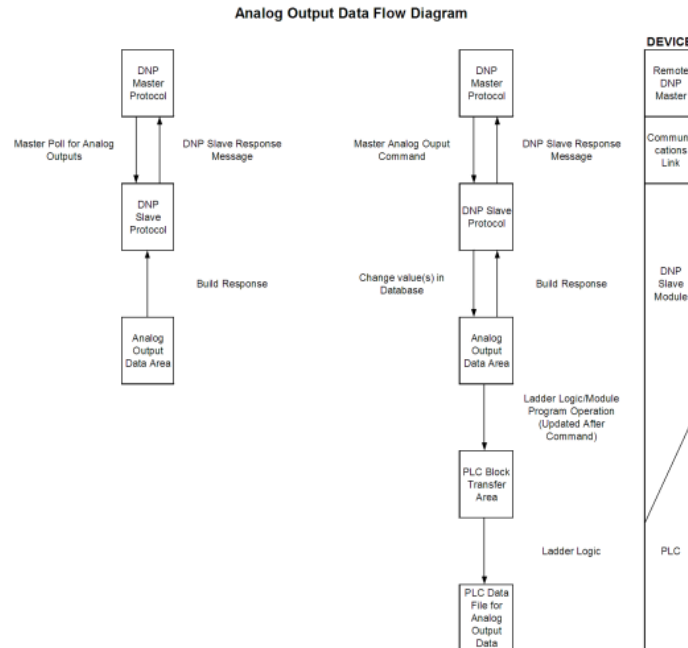
This data type stores analog data with a data range of 0 to 65535 or -32768 to 32767. The size of this data area is determined from the configuration parameter Analog Inputs. These data are transferred to the module from the processor using the read operation. Therefore, these data are read-only for the module and the DNP master unit. When the module receives a new block of this data from the processor, it compares the new values to those currently in the database. If there is a change in any of the data, the module will generate an event message for the points that change. The dead-band parameter configured for the module determines the variance required for the event message.

The DNP master unit can read the current value data and the event data from the module. Event messages generated by the module can be retrieved using a poll for Class 3 data, as all analog input events are considered a Class 3 data type. If unsolicited message generation is enabled in the application, the events will automatically be sent by the module to the DNP master unit when the minimum event count for Class 3 data is reached or when the timeout for unsolicited messages is exceeded. A data flow diagram for the analog input data is shown in the following figure.



DNP Analog Output Data

This data type stores analog values sent from the DNP master unit to the module and processor with a data range of 0 to 65535 or -32768 to 32767. The size of this data area is determined from the configuration parameter Analog Outputs. These data are transferred from the module to the processor using the write operation. Therefore, these data are read-only for the processor, as the processor cannot directly alter these values in the module. It is the responsibility of the DNP master unit to maintain this data. For example, if the DNP master sends a value of 3405 to the module for a specific point, the value will be stored in the module until changed by the master. A data flow diagram for the analog output data is shown in the following figure.



8.2.2 Functionality

This phase of design defines the features of the DNP Level 2 Subset supported by the module and to be utilized in the specific application. For example, will the unit use unsolicited messaging? Coordination with the DNP master developer is required to verify that the host will support the functionality you select. The features that must be defined in this design step are as follows:

- Will analog events be returned with or without a time value?
- Will events be logged before time synchronization has occurred?
- Will the module start with database values initialized by the processor?

For a complete description of the module configuration, refer to *Configuring the Module* (page 51).

8.2.3 Data Transfer at Startup

The module can be configured to have the internal databases initialized with data contained in the processor. Data to be initialized are as follows: Binary and Analog Output data. This feature can be used to bring the module to a known state (last state set in controller) when the module is first initialized. For example, in order to have the module startup using the last set of binary output values and setpoint values (analog outputs), enable this feature. Refer to the "Initialize DNP Database" parameter in the configuration file to enable this feature.

8.2.4 Module Operation

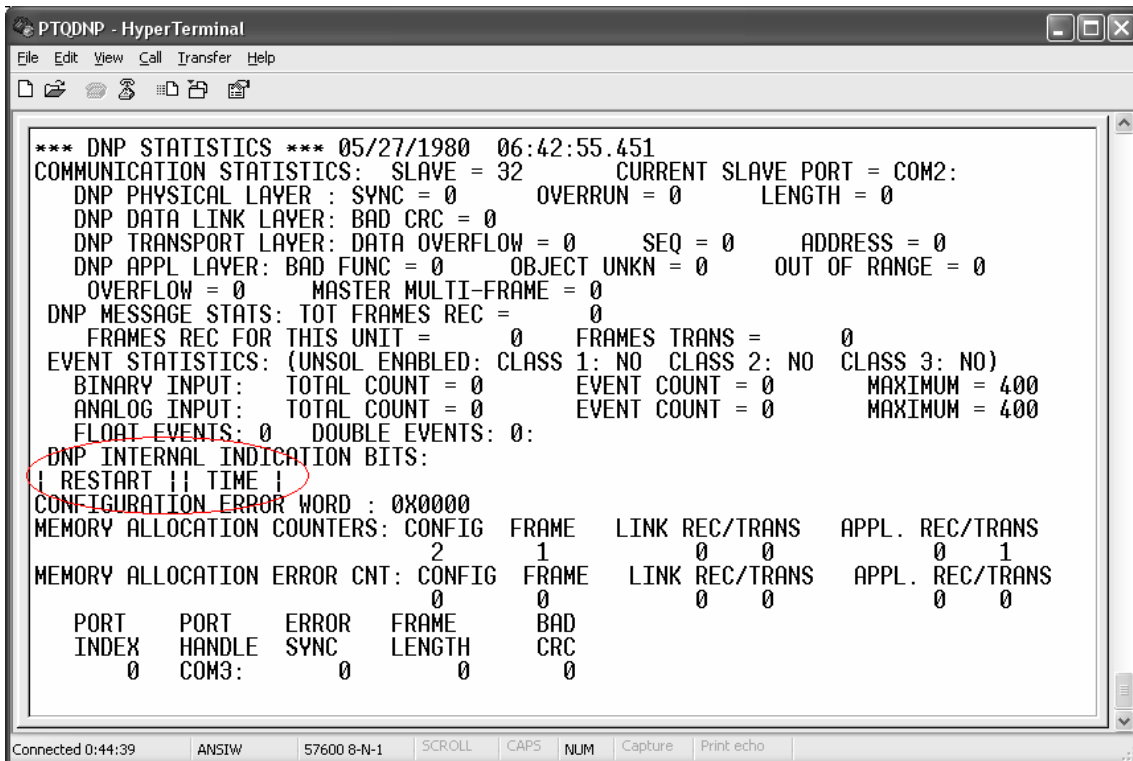
After the system has been designed and the system is set up, the module will be ready to operate. When the module is first initialized, it will read the configuration file. After the file is processed, the module will use the data to set up the data structures of the application. If any errors are encountered during the initialization process, the default value for the parameter will be assigned and used.

The module will next check if the output initialization feature is utilized. The option permits the processor to set these read-only data at startup. There is no static memory available on the module to remember the last values for these data types. In order to prevent a "shock" to the system at boot time, this option can be used to set the module's database to the last transferred set of data.

If the module is configured for unsolicited messaging, the module will immediately send an unsolicited response after the remote master connects to the module, informing the master of a module restart. The module will not log events or process any data read operations from the master until the master clears the restart IIN data bit. The master must also synchronize the time with the module before events will be generated if the module is so configured. The master also enables the unsolicited message facility in the module by sending the Enable Unsolicited Messaging command to the module.

If the module is not configured for unsolicited messaging, the DNP master must clear the restart IIN bit before the module will start logging events. The master must also synchronize the time with the module before events will be generated if the module is so configured.

The user can verify the IIN Bits by choosing the "S" key (Display Comm Status) from the Main menu.



```
PTQDNP - HyperTerminal
File Edit View Call Transfer Help
*** DNP STATISTICS *** 05/27/1980 06:42:55.451
COMMUNICATION STATISTICS: SLAVE = 32 CURRENT SLAVE PORT = COM2:
DNP PHYSICAL LAYER : SYNC = 0   OVERRUN = 0   LENGTH = 0
DNP DATA LINK LAYER: BAD CRC = 0
DNP TRANSPORT LAYER: DATA OVERFLOW = 0   SEQ = 0   ADDRESS = 0
DNP APPL LAYER: BAD FUNC = 0   OBJECT UNKN = 0   OUT OF RANGE = 0
OVERFLOW = 0   MASTER MULTI-FRAME = 0
DNP MESSAGE STATS: TOT FRAMES REC = 0
FRAMES REC FOR THIS UNIT = 0   FRAMES TRANS = 0
EVENT STATISTICS: (UNSOLED ENABLED: CLASS 1: NO CLASS 2: NO CLASS 3: NO)
BINARY INPUT: TOTAL COUNT = 0   EVENT COUNT = 0   MAXIMUM = 400
ANALOG INPUT: TOTAL COUNT = 0   EVENT COUNT = 0   MAXIMUM = 400
FLOAT EVENTS: 0   DOUBLE EVENTS: 0:
DNP INTERNAL INDICATION BITS:
| RESTART || TIME |
CONFIGURATION ERROR WORD : 0X0000
MEMORY ALLOCATION COUNTERS: CONFIG  FRAME  LINK REC/TRANS  APPL. REC/TRANS
                           2         1          0 0          0 1
MEMORY ALLOCATION ERROR CNT: CONFIG  FRAME  LINK REC/TRANS  APPL. REC/TRANS
                           0         0          0 0          0 0
PORT  PORT  ERROR  FRAME  BAD
INDEX HANDLE SYNC  LENGTH  CRC
  0    COM3:  0      0      0
```


After the module receives the clear restart and time sync commands, the same sub-menu shows that the IIN Bits are now cleared:

```

PTQDNP - HyperTerminal
File Edit View Call Transfer Help
*** DNP STATISTICS *** 05/27/1980 06:42:55.451
COMMUNICATION STATISTICS: SLAVE = 32 CURRENT SLAVE PORT = COM2:
DNP PHYSICAL LAYER : SYNC = 0 OVERRUN = 0 LENGTH = 0
DNP DATA LINK LAYER: BAD CRC = 0
DNP TRANSPORT LAYER: DATA OVERFLOW = 0 SEQ = 0 ADDRESS = 0
DNP APPL LAYER: BAD FUNC = 0 OBJECT UNKN = 0 OUT OF RANGE = 0
OVERFLOW = 0 MASTER MULTI-FRAME = 0
DNP MESSAGE STATS: TOT FRAMES REC = 0
FRAMES REC FOR THIS UNIT = 0 FRAMES TRANS = 0
EVENT STATISTICS: (UNSOLED ENABLED: CLASS 1: NO CLASS 2: NO CLASS 3: NO)
BINARY INPUT: TOTAL COUNT = 0 EVENT COUNT = 0 MAXIMUM = 400
ANALOG INPUT: TOTAL COUNT = 0 EVENT COUNT = 0 MAXIMUM = 400
FLOAT EVENTS: 0 DOUBLE EVENTS: 0:
DNP INTERNAL INDICATION BITS:
CONFIGURATION ERROR WORD : 0x0000
MEMORY ALLOCATION COUNTERS: CONFIG FRAME LINK REC/TRANS APPL. REC/TRANS
2 1 0 0 0 1
MEMORY ALLOCATION ERROR CNT: CONFIG FRAME LINK REC/TRANS APPL. REC/TRANS
0 0 0 0 0 0
PORT PORT ERROR FRAME BAD
INDEX HANDLE SYNC LENGTH CRC
0 COM3: 0 0 0

```

Additionally, Port 1 is used for debug and troubleshooting. This is the debug port for the module and transfers module information to an attached terminal. Refer to the Diagnostics and Troubleshooting section for a complete discussion on the use of this important feature.

8.3 PTQ-DNP Error Status Table

The program maintains an error/status table that is transferred to the processor in each read block. You can use the error/status data to determine the "health" of the module.

Word Offset	Variable Name	Description
0	Current DNP Slave Port status	This value represents the current value of the error code for the port. This value will only be valid if the port is configured as a slave. The possible values are described in the application documentation.
1	DNP Slave Port last transmitted error code	This value represents the last error code transmitted to the master by this slave port.
2	DNP Slave Port total number of message frames received by slave	This value represents the total number of message frames that have matched this slaves address on this port. This count includes message frames which the slave may or may not be able to parse and respond.

Word Offset	Variable Name	Description
3	DNP Slave Port total number of response message frames sent from slave	This value represents the number of good (non-error) responses that the slave has sent to the master on this port. The presumption is that if the slave is responding, the message was good. Note: This is a frame count.
4	DNP Slave Port total number of message frames seen by slave	This value represents the total number of message frames received by the slave, regardless of the slave address.
5	DNP Slave synchronization error count (Physical Layer Error)	This value counts the number of times a sync error occurs. The error occurs when extra bytes are received before the start bytes (0x05 and 0x64) are received.
6	DNP Slave overrun error count (Physical Layer Error)	This value counts the number of times the overrun error occurs. This error occurs when the mainline Data Link Layer routine cannot read the data received on the communication port before it is overwritten.
7	DNP Slave length error count (Physical Layer Error)	This value counts the number of times an invalid length byte is received. If the length of the message does not match the length value in the message, this error occurs.
8	DNP Slave bad CRC error (Data Link Layer Error)	This value counts the number of times a bad CRC value is received in a message.
9	DNP Slave user data overflow error (Transport Layer Error)	This value counts the number of times the application layer receives a message fragment buffer which is too small.
10	DNP Slave sequence error (Transport Layer Error)	This value counts the number of times the sequence numbers of multi-frame request fragments do not increment correctly.
11	DNP Slave address error (Transport Layer Error)	This value counts the number of times the source addresses contained in a multi-frame request fragments do not match.
12	DNP Slave Binary Input Event count	This value contains the total number of binary input events which have occurred.
13	DNP Slave Binary Input Event count in buffer	This value represents the number of binary input events which are waiting to be sent to the master.
14	DNP Slave Analog Input Event count	This value contains the total number of analog input events which have occurred.
15	DNP Slave Analog Input Event count in buffer	This value represents the number of analog input events which are waiting to be sent to the master.
16	DNP Slave bad function code error (Application Layer Error)	This value counts the number of times a bad function code for a selected object/variation is received by the slave device.
17	DNP Slave object unknown error (Application Layer Error)	This value counts the number of times a request for an unsupported object is received by the slave device.
18	DNP Slave out of range error (Application Layer Error)	This value counts the number of times a parameter in the qualifier, range or data field is not valid or out of range.
19	DNP Slave message overflow error (Application Layer Error)	This value counts the number of times an application response message from the slave is too long to transmit.

Word Offset	Variable Name	Description
20	DNP Slave multi-frame message from DNP Master error (Application Layer Error)	This value counts the number of times the slave receives a multi-frame message from the master. The application does not support multi-frame master messages.
21	Total blocks transferred	Total BTR/BTW or side-connect interface transfers attempted by the module.
22	Successful blocks transferred	This value represents the total number of transfer operations between the PLC and module that are successful.
23	Total errors in block transfer	Total number of transfers that resulted in an error condition.
24	Reserved	Reserved
25	Reserved	Reserved
26	Reserved	Reserved
27	Reserved	Reserved
28	Block transfer error flag	This flag indicates that data is not being successfully transferred between the PLC and the module. This flag corresponds to the Device Trouble IIN bit.
29	Configuration Type	This is a coded field that defines the configuration of the module. The codes are as follows: 0=Single Slave Configuration, 1=Dual Slave Configuration, 2=Slave/Master Configuration
30 to 31	Product Name (ASCII)	These two words contain the product name of the module in ASCII format.
32 to 33	Revision (ASCII)	These two words contain the product revision level of the firmware in ASCII format.
34 to 35	Operating System Revision (ASCII)	These two words contain the module's internal operating system revision level in ASCII format.
36 to 37	Production Run Number (ASCII)	These two words contain the production 'batch' number for the particular chip in the module in ASCII format.
38	DNP Master Port Slave Count	This is the total number of slaves configured for the DNP Master port. This may not represent the number of active slaves as it includes slaves that are not enabled.
39	DNP Master Port Command Count	This is the total number of commands configured for the DNP Master port. This may not represent the number of active commands as it includes commands that are disabled.
40	DNP Master Port Device Memory Block Count	This value represents the number of memory allocation blocks for slave devices. This number should be one greater than the number of slave devices. The extra device is held for the broadcast device.
41	DNP Master Port Frame Block Count	This value represents the number of physical layer frame memory allocation blocks used by the program.
42	DNP Master Port Data Link Receive Block Count	This value represents the number of receive data link layer memory blocks allocated.
43	DNP Master Port Data Link Transmit Block Count	This value represents the number of transmit data link layer memory blocks allocated.
44	DNP Master Port Application Layer Receive Block Count	This value represents the number of application layer receive memory blocks allocated.

Word Offset	Variable Name	Description
45	DNP Master Port Application Layer Receive Block Count	This value represents the number of application layer transmit memory blocks allocated.
46	DNP Master Port Device Memory Allocation Error Count	This value represents the number of memory allocation errors for device blocks.
47	DNP Master Port Physical Layer Memory Allocation Error Count	This value represents the number of memory allocation errors for physical layer frame blocks.
48	DNP Master Port Data Link Layer Receive Memory Allocation Error Count	This value represents the number of memory allocation errors for data link layer receive blocks.
49	DNP Master Port Data Link Layer Transmit Memory Allocation Error Count	This value represents the number of memory allocation errors for data link layer transmit blocks.
50	DNP Master Port Application Layer Receive Memory Allocation Error Count	This value represents the number of memory allocation errors for application layer receive blocks.
51	DNP Master Port Application Layer Transmit Memory Allocation Error Count	This value represents the number of memory allocation errors for application layer transmit blocks.
52	DNP Master Synchronization Error Count (Physical Layer Error)	This value counts the number of times a sync error occurs. The error occurs when extra bytes are received before the start bytes (0x05 and 0x64) are received.
53	DNP Master Length Error Count (Physical Layer Error)	This value counts the number of times an invalid length byte is received. If the length of the message does not match the length value in the message, this error occurs.
54	DNP Master Bad CRC Error Count (Physical Layer Error)	This value counts the number of times a bad CRC value is received in a message.
55	Scan Counter LSB	Program scan counter
56	Scan Counter MSB	
57	Free Memory LSB	Free memory in module
58	Free Memory MSB	
59	DNP Slave Port Transmit State	Value of the DNP Slave state machine for transmit.
60	DNP Float Event Count	Total number of events generated for analog floating-point input data points.
61	Reserved	Reserved

8.3.1 Slave Port Communication Errors

Error Code	Name	Description
0	OK	The module is operating correctly and there are no errors.
10	DNP synchronization error (Physical Layer Error)	Extra bytes are received before the start bytes (0x05 and 0x64).
11	DNP overrun error (Physical Layer Error)	Mainline Data Link Layer routine could not read data received on DNP port before it was overwritten.
12	DNP length error (Physical Layer Error)	Length of message does not match length value in message.

Error Code	Name	Description
13	DNP bad CRC error (Data Link Layer Error)	Computed CRC value for message does not match that received in message.
14	DNP user data overflow error (Transport Layer Error)	Application layer received a message fragment buffer which is too small.
15	DNP sequence error (Transport Layer Error)	Sequence numbers of multi-frame request fragments do not increment correctly.
16	DNP address error (Transport Layer Error)	Source addresses contained in multi-frame request fragments do not match.
17	DNP bad function code error (Application Layer Error)	Function code received from DNP master is not supported for selected object/variation.
18	DNP object unknown error (Application Layer Error)	Slave does not have the specified objects or there are no objects assigned to the requested class.
19	DNP out of range error (Application Layer Error)	Qualifier, range or data fields are not valid or out of range for the selected object/variation.
20	DNP message overflow error (Application Layer Error)	Application response buffer overflow condition. The response message from the slave is too long to transmit.
21	DNP master multi-frame message error (Application Layer Error)	Received a multi-frame message from the DNP master. This application does not support multi-frame messages from the master.

8.3.2 System Configuration Errors

Error Code	Name	Description
100	Too many binary input points	Too many binary input points are configured for the module. Maximum value is 15360.
101	Too many binary output points	Too many binary output points are configured for the module. Maximum value is 15360.
102	Too many counter points	Too many counter points are configured for the module. Maximum value is 480.
103	Too many analog input points	Too many analog input points are configured for the module. Maximum value is 960.
104	Too many analog input points	Too many analog output points are configured for the module. Maximum value is 960.
105	Too many binary input events	Too many binary input events are configured for the module. Maximum value is 400.
106	Too many analog input events	Too many analog input events are configured for the module. Maximum value is 400.
107	Invalid analog input deadband	Deadband value for analog input events is out of range. Value must be in the range of 0 to 32767.
108	Not enough memory	There is not enough memory in the module to configure the module as specified.
109	Invalid block transfer delay for blocks error/status blocks	Block transfer delay value specified is too low.
110	File count invalid	The file count must be in the range of 0 to 6.
111	Invalid file record size	The file record size must be in the range of 1 to 120.
112	Invalid block identification code for file	The file block transfer code must be in the range of 100 to 120.

8.3.3 DNP Port Configuration Errors

Error Code	Name	Description
212	Invalid DNP address	The DNP address specified in the configuration is not valid (0 to 65534).
213	Invalid DNP port baud rate	The baud rate code specified in the configuration is not valid.
219	Invalid DNP data link layer confirm mode	The data link confirmation mode code is not valid in the configuration.
220	Invalid DNP data link confirm time-out	The data link time-out period specified in the configuration is 0. It must be an integer in the range of 1 to 65535.
222	Invalid DNP select/operate arm time duration	The select/operate arm timer is set to 0. It must be an integer in the range of 1 to 65535.
223	Invalid DNP application layer confirm time-out	The application layer confirm time-out value is set to 0. It must be an integer in the range of 1 to 65535.
224	Invalid DNP write time interval	The write time interval is not in the data range in the configuration. The value must be in the range of 0 to 1440.
225	Invalid DNP unsolicited response mode	The unsolicited response mode code is not valid in the configuration.
226	Invalid DNP unsolicited response minimum quantity for Class 1	The unsolicited response minimum quantity for Class 1 is not valid in the configuration. Value must be an integer in the range of 1 to 255.
227	Invalid DNP unsolicited response minimum quantity for Class 2	The unsolicited response minimum quantity for Class 2 is not valid in the configuration. Value must be an integer in the range of 1 to 255.
228	Invalid DNP unsolicited response minimum quantity for Class 3	The unsolicited response minimum quantity for Class 3 is not valid in the configuration. Value must be an integer in the range of 1 to 255.
230	Invalid DNP unsolicited response destination address	The unsolicited response destination address is not valid in the configuration. Value must be in the range of 1 to 65534.

8.3.4 Command Error Codes

General Command Errors

Error Code	Name	Description
1	Device not defined	The IED slave address referenced in the command is not defined in the module. Check to make sure there is an entry in the slave table for each slave device referenced in the command list.
2	Invalid command	This command is not valid. Check to make sure the slave address parameter is greater than or equal to zero and that the point count is not set to zero.
3	Object not supported	The data object in the command is not supported by the module. Refer to the DNP subset for the Master Port.
4	Command function not supported	The function specified in the command is not supported for the object type selected. Refer to the DNP subset for the Master Port.
10	Invalid binary input poll command	This binary input object command is not valid.

Error Code	Name	Description
11	Invalid binary input event poll command	This binary input event object poll command is not valid.
20	Invalid binary output command function	This binary output command function is not valid.
30	Invalid counter poll command function	The counter object poll command contains an invalid function code.
31	Invalid counter poll command	This counter object poll command is not valid.
40	Invalid frozen counter poll command	This frozen counter object poll command is not valid.
50	Invalid analog input poll command	This analog input poll command is not valid.
51	Invalid analog input event poll command	This analog input event poll command is not valid.
60	Invalid analog output poll command function	This analog output poll command contains an invalid function code.
61	Invalid analog output poll command	This analog output poll command is not valid.
70	Invalid time/date poll command	This time/date object poll command is not valid.
80	Invalid event poll command	This event poll command is not valid.

Application Layer Errors

Error Code	Name	Description
1000	Device index invalid	The device index in the request or response message is not found in the slave list.
1001	Duplicate request in application layer queue	The newly submitted message to the application layer already exists in the queue. The message is ignored.
1002	COM port device removed from system	The communication port for the message has been uninstalled on the system. This error should never occur as the communication ports are only uninstalled when the module's program is terminated.
1003	Sequence number error	The application sequence number in the response message does not match that based on the last request message. This indicates application layer messages are received out of order.
1004	Response to select before operate does not match	The select response message received from the slave module is not that expected from the last select request. This indicates a synchronization problem between the master and slave devices.
1005	Response does not contain date/time object	The response message from the slave device does not contain a date/time object. The master expects this object for the response message.
1006	Time-out condition on response	The slave device did not respond to the last request message from the master within the time-out set for the IED device. The application layer time-out value is specified for each IED unit in the slave configuration table in the module. This table is established each time the module performs the restart operation.

Error CodeName		Description
1007	Function code in application layer message not supported	The function code returned in the response message is not valid for the application layer or not supported by the module.
1008	Read operation not supported for object/variation	The application layer response message contains an object that does not support the read function.
1009	Operate function not supported for the object/variation	The application layer response message contains an object that does not support the operate function.
1010	Write operation not supported for the object/variation	The application layer response message contains an object that does not support the write function.

8.4 PTQ-DNP Module Internal Indication Bits (IIN Bits) for DNP Server

The internal indication bits are stored in a word that follows the function code in all response messages. These bits report status and error information to the master DNP device. The following is a description of the word:

First Byte:

Bit	Description
0	All stations message received. Set when a request is received with the destination address set to 0xffff. Cleared after next response. Used to let master station know broadcast received.
1	Class 1 data available. Set when class 1 data is ready to be sent from the slave to the master. Master should request class 1 data when this bit is set.
2	Class 2 data available. Set when class 2 data is ready to be sent from the slave to the master. Master should request class 2 data when this bit is set.
3	Class 3 data available. Set when class 3 data is ready to be sent from the slave to the master. Master should request class 3 data when this bit is set.
4	Time synchronization required from master. The master should write the date and time when this bit is set. After receiving the write command the bit will be cleared.
5	Slave digital outputs are in local control. This bit is not used in this application.
6	Device trouble. When this bit is set, the data reported by the module may not be that currently present in the PLC because the block transfer operation is not successful.
7	Device restart. This bit is set when the slave either warm or cold boots. It is cleared after a master writes a 0 to the bit.

Second Byte:

Bit	Description
0	Bad function code. The function code contained in the master request is not supported for the specified object/variation.
1	Requested object(s) unknown. Object requested by master is not supported by the application.
2	Parameters in the qualifier, range or data fields are not valid or out of range for the slave.
3	Event buffer(s) or other application buffers have overflowed. This bit is also set if the slave receives a multi-frame message from the master.

Second Byte:

Bit	Description
4	Request understood but requested operation is already executing. The slave will never set this bit.
5	Bad configuration. The slave configuration is invalid and should be re-configured. If the configuration is invalid, the slave will set the invalid parameters to default values and continue to run. Check error log using debug port.
6	Reserved, always 0.
7	Reserved, always 0.

8.5 DNP Subset Definition**Module Slave Port**

Object		Request		Response				
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	Notes
1	0	Binary Input - All Variations	1	06			1	Slave will return variation 1 data
	1	Binary Input	1	06	129, 130	00, 01	1	Slave will return this variation
	2	Binary Input with Status			129, 130	00, 01	8	Slave will return Unknown Object to this request
2	0	Binary Input Change - All Variations	1	06, 07, 08			56	Slave will return variation 2 data
	1	Binary Input Change Without Time	1	06, 07, 08	129, 130	17, 28	8	Slave will return this variation
	2	Binary Input Change With Time	1	06, 07, 08	129, 130	17, 28	56	Slave will return this variation
	3	Binary Input Change With Relative Time	1	06, 07, 08	129, 130	17, 28	24	Slave will parse this message and return no data
10	0	Binary Output - All Variations	1	06			8	Slave will return variation 2 data
	1	Binary Output					1	Slave will return Unknown Object to this request
	2	Binary Output Status	1	06	129, 130	00, 01	8	Slave will return this variation
12	0	Control Block - All Variations					88	Slave will use variation 1 control
	1	Control Relay Output Block	3, 4, 5, 6	17, 28	129	Echo of request	88	Slave will respond correctly to this variation
	2	Pattern Control Block					88	Slave will return Unknown Object to this request
	3	Pattern Mask					16	Slave will return Unknown Object to this request
20	0	Binary Counter - All Variations	1, 7, 8, 9, 10	06			32	Slave will return variation 5 data

Object		Request		Response			Notes	
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)		Data Size (bits)
1		32-Bit Binary Counter			129, 130	00, 01	40	Slave will return Unknown Object to this request
2		16-Bit Binary Counter			129, 130	00, 01	24	Slave will return Unknown Object to this request
3		32-Bit Delta Counter			129, 130	00, 01	40	Slave will return Unknown Object to this request
4		16-Bit Delta Counter			129, 130	00, 01	24	Slave will return Unknown Object to this request
5		32-Bit Binary Counter Without Flag	1, 7, 8, 9, 10	06	129, 130	00, 01	32	Slave will return this variation
6		16-Bit Binary Counter Without Flag	1, 7, 8, 9, 10	06	129, 130	00, 01	16	Slave will return this variation (counter upper 16-bits removed)
7		32-Bit Delta Counter Without Flag			129, 130	00, 01	32	Slave will return Unknown Object to this request
8		16-Bit Delta Counter Without Flag			129, 130	00, 01	16	Slave will return Unknown Object to this request
21	0	Frozen Counter - All Variations	1	06			32	Slave will return variation 9 data
	1	32-Bit Frozen Counter			129, 130	00, 01	40	Slave will return Unknown Object to this request
	2	16-Bit Frozen Counter			129, 130	00, 01	24	Slave will return Unknown Object to this request
	3	32-Bit Frozen Delta Counter					40	Slave will return Unknown Object to this request
	4	16-Bit Frozen Delta Counter					24	Slave will return Unknown Object to this request
	5	32-Bit Frozen Counter With Time Of Freeze					88	Slave will return Unknown Object to this request
	6	16-Bit Frozen Counter With Time Of Freeze					72	Slave will return Unknown Object to this request
	7	32-Bit Frozen Delta Counter With Time Of Freeze					88	Slave will return Unknown Object to this request
	8	16-Bit Frozen Delta Counter With Time Of Freeze					72	Slave will return Unknown Object to this request
	9	32-Bit Frozen Counter Without Flag	1	06	129, 130	00, 01	32	Slave will return this variation
	10	16-Bit Frozen Counter Without Flag	1	06	129, 130	00, 01	16	Slave will return this variation (counter upper 16-bits removed)
	11	32-Bit Frozen Delta Counter Without Flag					32	Slave will return Unknown Object to this request
	12	16-Bit Frozen Delta Counter Without Flag					16	Slave will return Unknown Object to this request
22	0	Counter Change Event - All Variations	1	06, 07, 08				Slave will parse this request and return no data

Object		Request		Response				
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	Notes
	1	32-Bit Counter Change Event Without Time			129, 130	17, 28	40	Slave will return Unknown Object to this request
	2	16-Bit Counter Change Event Without Time			129, 130	17, 28	24	Slave will return Unknown Object to this request
	3	32-Bit Delta Counter Change Event Without Time					40	Slave will return Unknown Object to this request
	4	16-Bit Delta Counter Change Event Without Time					24	Slave will return Unknown Object to this request
	5	32-Bit Counter Change Event With Time					88	Slave will return Unknown Object to this request
	6	16-Bit Counter Change Event With Time					72	Slave will return Unknown Object to this request
	7	32-Bit Delta Counter Change Event With Time					88	Slave will return Unknown Object to this request
	8	16-Bit Delta Counter Change Event With Time					72	Slave will return Unknown Object to this request
23	0	Frozen Counter Event - All Variations						Slave will return Unknown Object to this request
	1	32-Bit Frozen Counter Event Without Time					40	Slave will return Unknown Object to this request
	2	16-Bit Frozen Counter Event Without Time					24	Slave will return Unknown Object to this request
	3	32-Bit Frozen Delta Counter Event Without Time					40	Slave will return Unknown Object to this request
	4	16-Bit Frozen Delta Counter Event Without Time					24	Slave will return Unknown Object to this request
	5	32-Bit Frozen Counter Event With Time					88	Slave will return Unknown Object to this request
	6	16-Bit Frozen Counter Event With Time					72	Slave will return Unknown Object to this request
	7	32-Bit Frozen Delta Counter Event With Time					88	Slave will return Unknown Object to this request
	8	16-Bit Frozen Delta Counter Event With Time					72	Slave will return Unknown Object to this request
30	0	Analog Input - All Variations	1	06			16	Slave will respond with variation 4 data
	1	32-Bit Analog Input	1	06	129, 130	00, 01	40	Slave will return this variation (Note: Data will only be 16-bit)

Object		Request		Response				
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	Notes
	2	16-Bit Analog Input	1	06	129, 130	00, 01	24	Slave will return this variation
	3	32-Bit Analog Input Without Flag	1	06	129, 130	00, 01	32	Slave will return this variation (Note: Data will only be 16-bit)
	4	16-Bit Analog Input Without Flag	1	06	129, 130	00, 01	16	Slave will return this variation
	5	Short Floating Point Analog Input	1	06	129, 130	00, 01	40	Slave will return this variation
	6	Long Floating Point Analog Input	1	06	129, 130	00, 01	72	Slave will return this variation
31	0	Frozen Analog Input - All Variations						Slave will return Unknown Object to this request
	1	32-Bit Frozen Analog Input					40	Slave will return Unknown Object to this request
	2	16-Bit Frozen Analog Input					24	Slave will return Unknown Object to this request
	3	32-Bit Frozen Analog Input With Time To Freeze					88	Slave will return Unknown Object to this request
	4	16-Bit Frozen Analog Input With Time To Freeze					72	Slave will return Unknown Object to this request
	5	32-Bit Frozen Analog Input Without Flag					32	Slave will return Unknown Object to this request
	6	16-Bit Frozen Analog Input Without Flag					16	Slave will return Unknown Object to this request
	7	Short Floating Point Frozen Analog Input					40	Slave will return Unknown Object to this request
	8	Long Floating Point Frozen Analog Input					72	Slave will return Unknown Object to this request
32	0	Analog Change Event - All Variations	1	06, 07, 08			24	Slave will return variation 2 data
	1	32-Bit Analog Change Event Without Time	1	06, 07, 08	129, 130	17, 28	40	Slave will return this variation (Note: Data only 16-bit)
	2	16-Bit Analog Change Event Without Time	1	06, 07, 08	129, 130	17, 28	24	Slave will return this variation
	3	32-Bit Analog Change Event With Time	1	06, 07, 08	129, 130	17, 28	88	Slave will return this variation (Note: Data only 16-bit)
	4	16-Bit Analog Change Event With Time	1	06, 07, 08	129, 130	17, 28	72	Slave will return this variation
	5	Short Floating Point Analog Change Event	1	06, 07, 08	129, 130	17, 28	40	Slave will return this variation
	6	Long Floating Point Analog Change Event	1	06, 07, 08	129, 130	17, 28	72	Slave will return this variation
	7	Short Floating Point Analog Change Event With Time	1	06, 07, 08	129, 130	17, 28	88	Slave will return this variation

Object		Request		Response			Notes	
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)		Data Size (bits)
	8	Long Floating Point Analog Change Event With Time	1	06, 07, 08	129, 130	17, 28	120	Slave will return this variation
33	0	Frozen Analog Event - All Variations						Slave will return Unknown Object to this request
	1	32-Bit Frozen Analog Event Without Time					40	Slave will return Unknown Object to this request
	2	16-Bit Frozen Analog Event Without Time					24	Slave will return Unknown Object to this request
	3	32-Bit Frozen Analog Event With Time					88	Slave will return Unknown Object to this request
	4	16-Bit Frozen Analog Event With Time					72	Slave will return Unknown Object to this request
	5	Short Floating Point Frozen Analog Event					40	Slave will return Unknown Object to this request
	6	Long Floating Point Frozen Analog Event					72	Slave will return Unknown Object to this request
	7	Short Floating Point Frozen Analog Event With Time					88	Slave will return Unknown Object to this request
	8	Long Floating Point Frozen Analog Event With Time					120	Slave will return Unknown Object to this request
40	0	Analog Output Status - All Variations	1	06			24	Slave will return variation 2 data
	1	32-Bit Analog Output Status	1	06	129,130	00,01	40	Slave will return this variation but data only 16-bit accuracy
	2	16-Bit Analog Output Status	1	06	129, 130	00, 01	24	Slave will return this variation
	3	Short Floating Point Analog Output Status	1	06	129, 130	00, 01	40	Slave will return this variation
	4	Long Floating Point Analog Output Status	1	06	129, 130	00, 01	72	Slave will return this variation
41	0	Analog Output Block - All Variations					24	Slave will respond to this request using variation 2 data
	1	32-Bit Analog Output Block	3, 4, 5, 6	17, 28	129,130	00,01	40	Slave will respond to this request but data only 16-bit
	2	16-Bit Analog Output Block	3, 4, 5, 6	17, 28	129	Echo of Request	24	Slave will respond to this request
	3	Short Floating Point Analog Output Block	3, 4, 5, 6	17, 28	129	Echo of Request	40	Slave will respond to this request
	4	Long Floating Point Analog Output Block	3, 4, 5, 6	17, 28	129	Echo of Request	72	Slave will respond to this request
50	0	Time and Date - All Variations	2	07, With Quant=1			48	Slave will use variation 1

Object		Request		Response			Notes	
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)		Data Size (bits)
	1	Time and Date	2	07, With Quant=1			48	Slave will respond to this variation
	2	Time and Date With Interval					80	Slave will return Unknown Object to this request
51	0	Time and Date CTO - All Variations						Slave will return Unknown Object to this request
	1	Time and Date CTO			129, 130	07, With Quant=1	48	Slave will return Unknown Object to this request
	2	Unsynchronized Time and Date CTO			129, 130	07, With Quant=1	48	Slave will return Unknown Object to this request
52	0	Time Delay - All Variations						
	1	Time Delay Coarse			129	07, With Quant=1	16	Slave will never return this variation
	2	Time Delay Fine			129	07, With Quant=1	16	Slave will return this variation to functions 0D, 0E, and 17
60	0	Not Defined						Not Defined in DNP
	1	Class 0 Data	1	06				Slave will respond to this variation with all static data
	2	Class 1 Data	1	06, 07, 08				Slave will respond to this variation (No class 1 data defined in application)
	3	Class 2 Data	1	06, 07, 08				Slave will respond to this variation with all class 2 data (binary input events)
	4	Class 3 Data	1	06, 07, 08				Slave will respond to this variation with all class 3 data (analog input events)
70	0	Not Defined						Not Defined in DNP
	1	File Identifier						Slave will return Unknown Object to this request
80	0	Not Defined						Not Defined in DNP
	1	Internal Indications	2	00, Index=7			24	Slave will respond to this variation
81	0	Not Defined						Not Defined in DNP
	1	Storage Object						
82	0	Not Defined						Not Defined in DNP
	1	Device Profile						
83	0	Not Defined						Not Defined in DNP
	1	Private Registration Object						
	2	Private Registration Objection Descriptor						
90	0	Not Defined						Not Defined in DNP

Object		Request		Response				
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	Notes
	1	Application Identifier						
10	0							
	1	Short Floating Point					48	
	2	Long Floating Point					80	
	3	Extended Floating Point					88	
10	0							
	1	Small Packed Binary-Coded Decimal					16	
	2	Medium Packed Binary-Coded Decimal					32	
	3	Large Packed Binary-Coded Decimal					64	
11	0	Not Defined						Not Defined as the variation determines the string length
	1 to 100	Octet String	1	00, 01, 06, 07, 08, 17, 28	129, 130	00, 01, 07, 08, 17, 28	8 * Var #	The module will return this variation for the points defined in the module. The variation determines the returned string length.
No Object			13					Slave supports the Cold Restart Function and will return Obj 52, Var 2, Qual 7, Cnt 1
			14					Slave supports the Warm Restart Function and will return Obj 52, Var 2, Qual 7, Cnt 1
			20					Slave supports the Enable Unsolicited Function
			21					Slave supports the Disable Unsolicited Function
			23					Slave supports the Delay Measurement & Time Synchronization Function and will return Obj 52, Var 2, Qual 7, Cnt 1

Note: Objects that we support that are not required within the Level II specification are grayed out. Refer to the associated notes to determine our response to the message.

8.5.1 Module Master Port

Object		Request		Response				Notes
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	
1	0	Binary Input: All Variations	1	06			1	Master will generate this variation
	1	Binary Input	1	06	129, 130	00, 01	1	Master will generate and process this variation
	2	Binary Input with Status	1	06	129, 130	00, 01	8	Master will generate and process this variation
2	0	Binary Input Change: All Variations	1	06, 07, 08			56	Master will generate this variation
	1	Binary Input Change Without Time	1	06, 07, 08	129, 130	17, 28	8	Master will generate and process this variation
	2	Binary Input Change With Time	1	06, 07, 08	129, 130	17, 28	56	Master will generate and process this variation
	3	Binary Input Change With Relative Time	1	06, 07, 08	129, 130	17, 28	24	Master will generate and process this variation
10	0	Binary Output: All Variations	1	06			8	Master does not use this object type and will not generate a message or process this type
	1	Binary Output					1	
	2	Binary Output Status			129, 130	00, 01	8	
12	0	Control Block: All Variations					88	
	1	Control Relay Output Block	3, 4, 5, 6	17, 28	129	Echo of request	88	Master will generate this variation and parse the response
	2	Pattern Control Block					88	
	3	Pattern Mask					16	
20	0	Binary Counter: All Variations	1, 7, 8, 9, 10	06			32	Master will generate this variation
	1	32-Bit Binary Counter			129, 130	00, 01	40	Master will process this variation
	2	16-Bit Binary Counter			129, 130	00, 01	24	Master will process this variation
	3	32-Bit Delta Counter			129, 130	00, 01	40	Master will process this variation
	4	16-Bit Delta Counter			129, 130	00, 01	24	Master will process this variation
	5	32-Bit Binary Counter Without Flag	1, 7, 8, 9, 10	06	129, 130	00, 01	32	Master will generate and process this variation
	6	16-Bit Binary Counter Without Flag	1, 7, 8, 9, 10	06	129, 130	00, 01	16	Master will generate and process this variation
	7	32-Bit Delta Counter Without Flag			129, 130	00, 01	32	Master will process this variation
8	16-Bit Delta Counter Without Flag			129, 130	00, 01	16	Master will process this variation	

Object		Request		Response				Notes
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	
21	0	Frozen Counter: All Variations	1	06			32	Master will generate this variation
	1	32-Bit Frozen Counter			129, 130	00, 01	40	Master will process this variation
	2	16-Bit Frozen Counter			129, 130	00, 01	24	Master will process this variation
	3	32-Bit Frozen Delta Counter					40	
	4	16-Bit Frozen Delta Counter					24	
	5	32-Bit Frozen Counter With Time Of Freeze					88	
	6	16-Bit Frozen Counter With Time Of Freeze					72	
	7	32-Bit Frozen Delta Counter With Time Of Freeze					88	
	8	16-Bit Frozen Delta Counter With Time Of Freeze					72	
	9	32-Bit Frozen Counter Without Flag	1	06	129, 130	00, 01	32	Master will generate and process this variation
	10	16-Bit Frozen Counter Without Flag	1	06	129, 130	00, 01	16	Master will generate and process this variation
	11	32-Bit Frozen Delta Counter Without Flag					32	
	12	16-Bit Frozen Delta Counter Without Flag					16	
22	0	Counter Change Event: All Variations	1	06, 07, 08				Master will not generate a request for this variation
	1	32-Bit Counter Change Event Without Time			129, 130	17, 28	40	Master will process this variation
	2	16-Bit Counter Change Event Without Time			129, 130	17, 28	24	Master will process this variation
	3	32-Bit Delta Counter Change Event Without Time					40	
	4	16-Bit Delta Counter Change Event Without Time					24	
	5	32-Bit Counter Change Event With Time					88	
	6	16-Bit Counter Change Event With Time					72	
	7	32-Bit Delta Counter Change Event With Time					88	

Object		Request		Response				Notes
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	
	8	16-Bit Delta Counter Change Event With Time					72	
23	0	Frozen Counter Event: All Variations						
	1	32-Bit Frozen Counter Event Without Time					40	
	2	16-Bit Frozen Counter Event Without Time					24	
	3	32-Bit Frozen Delta Counter Event Without Time					40	
	4	16-Bit Frozen Delta Counter Event Without Time					24	
	5	32-Bit Frozen Counter Event With Time					88	
	6	16-Bit Frozen Counter Event With Time					72	
	7	32-Bit Frozen Delta Counter Event With Time					88	
	8	16-Bit Frozen Delta Counter Event With Time					72	
30	0	Analog Input: All Variations	1	06			16	Master will generate this variation
	1	32-Bit Analog Input	1	06	129, 130	00, 01	40	Master will generate and process this variation
	2	16-Bit Analog Input	1	06	129, 130	00, 01	24	Master will generate and process this variation
	3	32-Bit Analog Input Without Flag	1	06	129, 130	00, 01	32	Master will generate and process this variation
	4	16-Bit Analog Input Without Flag	1	06	129, 130	00, 01	16	Master will generate and process this variation
31	0	Frozen Analog Input: All Variations						
	1	32-Bit Frozen Analog Input					40	
	2	16-Bit Frozen Analog Input					24	
	3	32-Bit Frozen Analog Input With Time To Freeze					88	
	4	16-Bit Frozen Analog Input With Time To Freeze					72	

Object		Request		Response				Notes
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	
	5	32-Bit Frozen Analog Input Without Flag					32	
	6	16-Bit Frozen Analog Input Without Flag					16	
32	0	Analog Change Event: All Variations	1	06, 07, 08			24	Master will generate this variation
	1	32-Bit Analog Change Event Without Time	1	06, 07, 08	129, 130	17, 28	40	Master will generate and process this variation
	2	16-Bit Analog Change Event Without Time	1	06, 07, 08	129, 130	17, 28	24	Master will generate and process this variation
	3	32-Bit Analog Change Event With Time	1	06, 07, 08	129, 130	17, 28	88	Master will generate and process this variation
	4	16-Bit Analog Change Event With Time	1	06, 07, 08	129, 130	17, 28	72	Master will generate and process this variation
33	0	Frozen Analog Event: All Variations						
	1	32-Bit Frozen Analog Event Without Time					40	
	2	16-Bit Frozen Analog Event Without Time					24	
	3	32-Bit Frozen Analog Event With Time					88	
	4	16-Bit Frozen Analog Event With Time					72	
40	0	Analog Output Status: All Variations	1	06			24	Master does not use this object type and will not generate a message or process this type
	1	32-Bit Analog Output Status					40	
	2	16-Bit Analog Output Status			129, 130	00, 01	24	
41	0	Analog Output Block: All Variations					24	
	1	32-Bit Analog Output Block					40	
	2	16-Bit Analog Output Block	3, 4, 5, 6	17, 28	129	Echo of Request	24	Master will generate this variation and parse the response
50	0	Time and Date: All Variations					48	
	1	Time and Date	2	07, With Quant=1			48	Master will generate this variation
	2	Time and Date With Interval					80	
51	0	Time and Date CTO: All Variations						

Object		Request		Response			Notes	
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)		Data Size (bits)
	1	Time and Date CTO			129, 130	07, With Quant=1	48	Master will process this variation
	2	Unsynchronized Time and Date CTO			129, 130	07, With Quant=1	48	Master will process this variation
52	0	Time Delay: All Variations						
	1	Time Delay Coarse			129	07, With Quant=1	16	Master will not process this variation
	2	Time Delay Fine			129	07, With Quant=1	16	Master will not process this variation
60	0	Not Defined						Not Defined in DNP
	1	Class 0 Data	1	06				Master will generate this variation
	2	Class 1 Data	1	06, 07, 08				Master will generate this variation
	3	Class 2 Data	1	06, 07, 08				Master will generate this variation
	4	Class 3 Data	1	06, 07, 08				Master will generate this variation
70	0	Not Defined						
	1	File Identifier						
80	0	Not Defined						
	1	Internal Indications	2	00, Index=7			24	The Master will generate this variation
81	0	Not Defined						
	1	Storage Object						
82	0	Not Defined						
	1	Device Profile						
83	0	Not Defined						Not Defined in DNP
	1	Private Registration Object						
	2	Private Registration Objection Descriptor						
90	0	Not Defined						Not Defined in DNP
	1	Application Identifier						
10	0							
	1	Short Floating Point					48	
	2	Long Floating Point					80	
	3	Extended Floating Point					88	
10	0							
	1	Small Packed Binary-Coded Decimal					16	

Object		Request		Response				
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	Notes
2		Medium Packed Binary-Coded Decimal					32	
3		Large Packed Binary-Coded Decimal					64	
No Object			13					Master supports the Cold Restart Function
			14					Master supports the Warm Restart Function
			20					Master supports the Enable Unsolicited Function
			21					Master supports the Disable Unsolicited Function

8.6 Event Size Computation

The minimum event buffer size required to avoid overflow can be computed as follows:

$$\frac{((\text{number of static points}) * (\text{rate per second scan of change function}))}{(\text{rate per second of master event data poll})}$$

For example: 51 binary input points are scanned 2 times each second and polled by the master station about every 5 seconds. The minimum number of binary input events is:

$$(51 * 2) / .02 = 510 \text{ events}$$

This computation assumes the unlikely event that all data points will change in consecutive calls to the scan of change function. If an event buffer overflow condition occurs, the internal indication bit, BUFFER OVERFLOW, will be set. If the system you are working with is fairly stable, the following equation can be used to compute the event buffer size:

$$(\text{number of points that change per change function} * \text{rate per second of scan of change function}) * (\text{number of seconds between master event data poll})$$

For example: 1000 binary input points are scanned 2 times each second and polled by the master station about every 5 seconds. Only about 5 points change state every scan of the change function call.

$$(5 * 2) * 5 = 50 \text{ events required}$$

The number of events that can be defined in the system is limited to 200 for bits or analogs. The event buffer will overflow in systems which are very dynamic unless one of the following conditions exist:

- The master frequently polls the slave device for events to keep the buffer empty.
- OR
- The slave is configured to send unsolicited messages to the master station. This method requires full-duplex operation of the network because the slave may be sending a message during a request from the master station.

In order to disable the report by exception feature in the module, set the number of events to 0 for both the binary and analog input events in the configuration. This will cause the DNP slave port driver to never return any data on object 2 and 32 and class 2 and 3 master station requests.

8.7 DNP Collision Avoidance

8.7.1 When Required

Collision avoidance is required under to following network configurations:

- 1 A multi-point network is used (that is, master unit is communicating with several slave units on same physical link). This excludes a dial-up modem network where the master only communicates with one slave at a time in a point-to-point physical link. Will only operate on two-wire, half-duplex communication networks.
- 2 Unsolicited messaging is used where asynchronous, spontaneous messages may be generated by any node on the network.
- 3 Any network where the physical layer does not implement a collision avoidance scheme and permits several nodes to communicate at one time (that is, some radio networks).

8.7.2 Rules

- RTS controls DCD line on all other units.
- No stations transmit while DCD line is high except the one that has the RTS line high.
- After DCD line drops low, slaves wait variable time before attempting to transmit.
- Master has smallest delay (can be set to 0)
- Slaves have higher delays (fixed delay (that is, slave#*20 milliseconds) + random delay (20 to 50 milliseconds))
- Only supported in 2-wire half-duplex mode. Not supported in 4-wire half-duplex mode.
- This option is set in the configuration of the module with the port type.

8.7.3 Cable

This is the cable required for direct connection of the module to a remote unit:

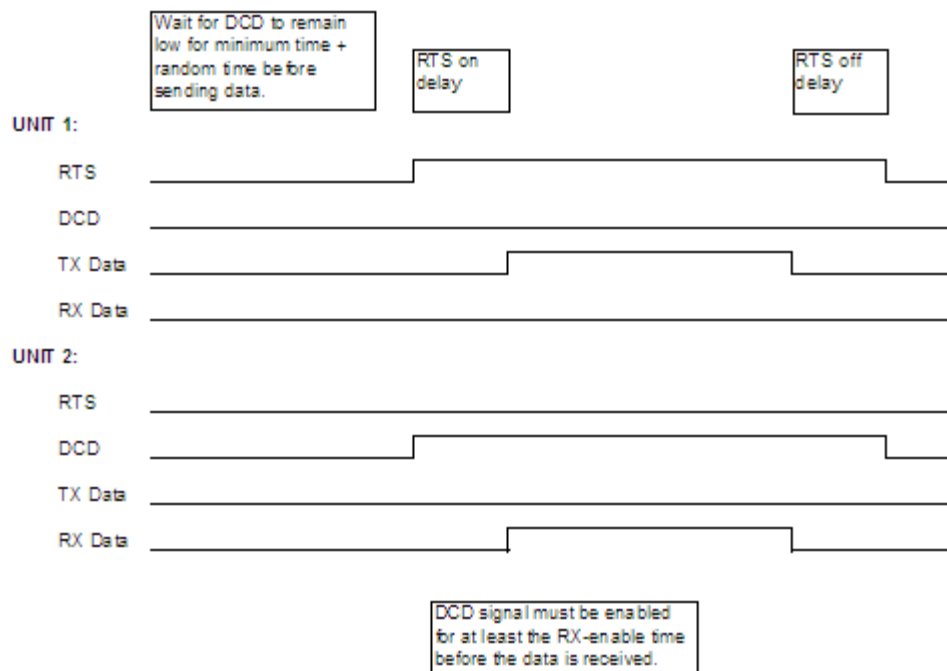
		DB-9	DB-25
DCD	RTS	7	4
RX	TX	3	2
TX	RX	2	3
GND	GND	5	7
RTS	DCD	1	8

8.7.4 Timing Chart

Several timing parameters are required for each unit in order to implement the collision avoidance feature. The parameters are as follows:

Parameter	Description
Fixed DCD Idle Delay Time Before Transmit	This parameter specifies the minimum number of milliseconds to delay before transmitting a message after recognizing that the DCD line is low.
Random DCD Idle Delay Time Before Transmit	This parameter determines the random time to be added to the above fixed delay value above before attempting to transmit a message. The value specified for the parameter determines the range of random values (milliseconds) to be used. For example, if a value of 20 is specified, the random delay time will be from 0 to 20 each time the value is requested.
DCD Time Before Receive	This parameter specifies the number of milliseconds to delay after recognizing that DCD has been asserted before accepting data. The RTS on time of the sending unit must be set greater than the time specified here or else the first part of the data message will be ignored.
RTS On Time	This parameter specifies the number of milliseconds to delay after asserting the RTS modem control line before sending the data.
RTS Off Time	This parameter specifies the number of milliseconds to delay after the data has been transmitted before dropping the RTS modem control line.

The timing parameters defined above must be set correctly for successful use of the collision avoidance feature. A timing diagram displaying the data and modem control lines used with the collision avoidance scheme is shown below. This example displays the state of the signal lines in transmitting a message from Unit 1 to Unit 2.



8.8 Frequently Asked Questions

8.8.1 How fast do the "Backplane Data Exchange" commands run?

The "Backplane Data Exchange" commands will execute one at a time during the I/O service interval of the PLC. What this means is that if you had a list of 10 commands at the end of every PLC scan one command would execute. This would mean that it would take 10 PLC scans to execute the 10 commands contained within the "Backplane Data Exchange" section of the configuration file.

8.8.2 What is the maximum number of words I can transfer with a "Backplane Data Exchange" command?

For command types 1 & 2 you may move up to 130 words with each command. Function 3 is somewhat different in that it provides only 64 words of data movement BUT since it is intended to solve very specialized operations its size must be restricted.

8.8.3 Do I need to use "Backplane Data Exchange" function 3?

The only time you should need it is if you are using the DNP, DNP or one of the IEC protocols. If you are using one of these protocols then you can find sample structured text examples included in the manual for these protocols. In all other instances you should not need to use this function.

8.8.4 How much data can I transfer between the PLC and the Module.

You can enter up to 100 commands in the [BACKPLANE DATA EXCHANGE] section of the configuration file. The limit for any single execution of a Function 1 or 2 is 130 words but you may enter multiple commands to transfer more data.

8.8.5 How do I configure the module?

The ProTalk requires a simple text based configuration file to make it operational. For a really quick tutorial on the modules communications with the PLC you should review the [QUICK START GUIDE] or for more in depth information the chapter on "Backplane Data Exchange" should answer most questions.

8.8.6 What software application is required for my Ladder Logic?

The design of the module should be software independent and for many installations minimal or possibly no ladder will be required. The section on "Backplane Data Exchange" offers to samples to help in the few instances where ladder is required.

8.8.7 Is a .MDC available for configuration of the Module?

Yes. The CDROM that ships with the module should have a version for both Concept 2.5 and 2.6 in the ProTalk directory.

8.8.8 Does the module work in a remote rack?

The module is designed to be located in the chassis with the PLC and will not operate in a remote chassis. If your application requires remote placement of the communication device you should investigate the other members of the ProSoft family such as the 4202-MNET-DFCM. (if you require DF1 connectivity for instance although many others are available) This module for example would allow you to communicate with DF1 devices and allow you to map the contents of its memory using Modbus TCP/IP.

8.8.9 Can I use the module in a hot backup system?

Support for Hot Backup is not currently implemented in the module. We are currently investigating the addition of this functionality but until this development can be finalized, it may be possible to use one of the 4000 series of ProSoft communication products. Please call our technical support technicians when considering this application.

8.8.10 DNP Specific Questions**What does "Initialize Output Data" in the configuration file mean?**

The default of this user parameter is NO. When the module reboots it will reset all of its internal registers to a zero value. In some applications this will cause a problem as the master wishes to see what he/she believes he/she put in that register during the last access. If this is true you should set this parameter to YES, which will cause the module to convert the writes (command function 2) in the [BACKPLANE DATA EXCHANGE] section to reads for one scan and one scan only. This will reload the registers in the module with the information contained within the PLC.

Where do the individual data types actually exist in the modules memory?

The placement of the individual data types is in a pre-defined order, which is the same as they are placed in the configuration file for easy reference. They will be placed in memory sequentially as follows:

- Binary Inputs
- Analog Inputs
- Counter Data
- Binary Outputs
- Analog Outputs

When you describe the database in the DNP configuration file you should create sufficient data size for your application plus any anticipated growth. If for instance you describe 10 Binary Inputs today and later increase the size to 20 you will have effectively changed the location of your Analog Inputs, Counter Data, Binary Outputs and Analog Outputs by 10 locations.

If you choose not to do this then you should enter command(s) for each data transfer. In this instance you could change the data AND change the [BACKPLANE DATA EXCHANGE] commands to maintain your mapping in the PLC.

9 Support, Service & Warranty

In This Chapter

- ❖ Return Material Authorization (RMA) Policies and Conditions..... 163
- ❖ LIMITED WARRANTY..... 165
- ❖ How to Contact Us: Technical Support..... 169

ProSoft Technology, Inc. (ProSoft) is committed to providing the most efficient and effective support possible. Before calling, please gather the following information to assist in expediting this process:

- 1 Product Version Number
- 2 System architecture
- 3 Network details

If the issue is hardware related, we will also need information regarding:

- 1 Module configuration and contents of file
 - Module Operation
 - Configuration/Debug status information
 - LED patterns
- 2 Information about the processor and user data files as viewed through and LED patterns on the processor.
- 3 Details about the serial devices interfaced, if any.

9.1 Return Material Authorization (RMA) Policies and Conditions

The following RMA Policies and Conditions (collectively, "RMA Policies") apply to any returned Product. These RMA Policies are subject to change by ProSoft without notice. For warranty information, see "Limited Warranty". In the event of any inconsistency between the RMA Policies and the Warranty, the Warranty shall govern.

9.1.1 Procedures for Return of Units Under Warranty:

A Technical Support Engineer must approve the return of Product under ProSoft's Warranty:

- a) A replacement module will be shipped and invoiced. A purchase order will be required.
- b) Credit for a product under warranty will be issued upon receipt of authorized product by ProSoft at designated location referenced on the Return Material Authorization.

- If a defect is found and is determined to be customer generated, or if the defect is otherwise not covered by ProSoft's Warranty, there will be no credit given. Customer will be contacted and can request module be returned at their expense.

9.1.2 Procedures for Return of Units Out of Warranty:

- a) Customer sends unit in for evaluation
- b) If no defect is found, Customer will be charged the equivalent of \$100 USD, plus freight charges, duties and taxes as applicable. A new purchase order will be required.
- c) If unit is repaired, charge to Customer will be 30% of current list price (USD) plus freight charges, duties and taxes as applicable. A new purchase order will be required or authorization to use the purchase order submitted for evaluation fee.

The following is a list of non-repairable units:

- 3150 - All
- 3750
- 3600 - All
- 3700
- 3170 - All
- 3250
- 1560 - Can be repaired, only if defect is the power supply
- 1550 - Can be repaired, only if defect is the power supply
- 3350
- 3300
- 1500 - All

9.1.3 All Product Returns:

- a) In order to return a Product for repair, exchange or otherwise, the Customer must obtain a Returned Material Authorization (RMA) number from ProSoft and comply with ProSoft shipping instructions.
- b) In the event that the Customer experiences a problem with the Product for any reason, Customer should contact ProSoft Technical Support at one of the telephone numbers listed above (page 169). A Technical Support Engineer will request that you perform several tests in an attempt to isolate the problem. If after completing these tests, the Product is found to be the source of the problem, we will issue an RMA.
- c) All returned Products must be shipped freight prepaid, in the original shipping container or equivalent, to the location specified by ProSoft, and be accompanied by proof of purchase and receipt date. The RMA number is to be prominently marked on the outside of the shipping box. Customer agrees to insure the Product or assume the risk of loss or damage in transit. Products shipped to ProSoft using a shipment method other than that specified by ProSoft or shipped without an RMA number will be returned to the Customer, freight collect. Contact ProSoft Technical Support for further information.

- d) A 10% restocking fee applies to all warranty credit returns whereby a Customer has an application change, ordered too many, does not need, etc.

9.1.4 Purchasing Warranty Extension:

- a) ProSoft's standard warranty period is three (3) years from the date of shipment as detailed in "Limited Warranty (page 165)". The Warranty Period may be extended at the time of equipment purchase for an additional charge, as follows:
- Additional 1 year = 10% of list price
 - Additional 2 years = 20% of list price
 - Additional 3 years = 30% of list price

9.2 LIMITED WARRANTY

This Limited Warranty ("Warranty") governs all sales of hardware, software and other products (collectively, "Product") manufactured and/or offered for sale by ProSoft, and all related services provided by ProSoft, including maintenance, repair, warranty exchange, and service programs (collectively, "Services"). By purchasing or using the Product or Services, the individual or entity purchasing or using the Product or Services ("Customer") agrees to all of the terms and provisions (collectively, the "Terms") of this Limited Warranty. All sales of software or other intellectual property are, in addition, subject to any license agreement accompanying such software or other intellectual property.

9.2.1 What Is Covered By This Warranty

- a) *Warranty On New Products:* ProSoft warrants, to the original purchaser, that the Product that is the subject of the sale will (1) conform to and perform in accordance with published specifications prepared, approved and issued by ProSoft, and (2) will be free from defects in material or workmanship; provided these warranties only cover Product that is sold as new. This Warranty expires three years from the date of shipment (the "Warranty Period"). If the Customer discovers within the Warranty Period a failure of the Product to conform to specifications, or a defect in material or workmanship of the Product, the Customer must promptly notify ProSoft by fax, email or telephone. In no event may that notification be received by ProSoft later than 39 months. Within a reasonable time after notification, ProSoft will correct any failure of the Product to conform to specifications or any defect in material or workmanship of the Product, with either new or used replacement parts. Such repair, including both parts and labor, will be performed at ProSoft's expense. All warranty service will be performed at service centers designated by ProSoft.

- b) *Warranty On Services:* Materials and labor performed by ProSoft to repair a verified malfunction or defect are warranted in the terms specified above for new Product, provided said warranty will be for the period remaining on the original new equipment warranty or, if the original warranty is no longer in effect, for a period of 90 days from the date of repair.

9.2.2 What Is Not Covered By This Warranty

- a) ProSoft makes no representation or warranty, expressed or implied, that the operation of software purchased from ProSoft will be uninterrupted or error free or that the functions contained in the software will meet or satisfy the purchaser's intended use or requirements; the Customer assumes complete responsibility for decisions made or actions taken based on information obtained using ProSoft software.
- b) This Warranty does not cover the failure of the Product to perform specified functions, or any other non-conformance, defects, losses or damages caused by or attributable to any of the following: (i) shipping; (ii) improper installation or other failure of Customer to adhere to ProSoft's specifications or instructions; (iii) unauthorized repair or maintenance; (iv) attachments, equipment, options, parts, software, or user-created programming (including, but not limited to, programs developed with any IEC 61131-3, "C" or any variant of "C" programming languages) not furnished by ProSoft; (v) use of the Product for purposes other than those for which it was designed; (vi) any other abuse, misapplication, neglect or misuse by the Customer; (vii) accident, improper testing or causes external to the Product such as, but not limited to, exposure to extremes of temperature or humidity, power failure or power surges; or (viii) disasters such as fire, flood, earthquake, wind and lightning.
- c) The information in this Agreement is subject to change without notice. ProSoft shall not be liable for technical or editorial errors or omissions made herein; nor for incidental or consequential damages resulting from the furnishing, performance or use of this material. The user guide included with your original product purchase from ProSoft contains information protected by copyright. No part of the guide may be duplicated or reproduced in any form without prior written consent from ProSoft.

9.2.3 Disclaimer Regarding High Risk Activities

Product manufactured or supplied by ProSoft is not fault tolerant and is not designed, manufactured or intended for use in hazardous environments requiring fail-safe performance including and without limitation: the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines or weapons systems in which the failure of the product could lead directly or indirectly to death, personal injury or severe physical or environmental damage (collectively, "high risk activities"). ProSoft specifically disclaims any express or implied warranty of fitness for high risk activities.

9.2.4 Limitation of Remedies **

In no event will ProSoft or its Dealer be liable for any special, incidental or consequential damages based on breach of warranty, breach of contract, negligence, strict tort or any other legal theory. Damages that ProSoft or its Dealer will not be responsible for included, but are not limited to: Loss of profits; loss of savings or revenue; loss of use of the product or any associated equipment; loss of data; cost of capital; cost of any substitute equipment, facilities, or services; downtime; the claims of third parties including, customers of the Purchaser; and, injury to property.

** Some areas do not allow time limitations on an implied warranty, or allow the exclusion or limitation of incidental or consequential damages. In such areas, the above limitations may not apply. This Warranty gives you specific legal rights, and you may also have other rights which vary from place to place.

9.2.5 Time Limit for Bringing Suit

Any action for breach of warranty must be commenced within 39 months following shipment of the Product.

9.2.6 Intellectual Property Indemnity

Buyer shall indemnify and hold harmless ProSoft and its employees from and against all liabilities, losses, claims, costs and expenses (including attorney's fees and expenses) related to any claim, investigation, litigation or proceeding (whether or not ProSoft is a party) which arises or is alleged to arise from Buyer's acts or omissions under these Terms or in any way with respect to the Products. Without limiting the foregoing, Buyer (at its own expense) shall indemnify and hold harmless ProSoft and defend or settle any action brought against such Companies to the extent based on a claim that any Product made to Buyer specifications infringed intellectual property rights of another party. ProSoft makes no warranty that the product is or will be delivered free of any person's claiming of patent, trademark, or similar infringement. The Buyer assumes all risks (including the risk of suit) that the product or any use of the product will infringe existing or subsequently issued patents, trademarks, or copyrights.

- a) Any documentation included with Product purchased from ProSoft is protected by copyright and may not be duplicated or reproduced in any form without prior written consent from ProSoft.
- b) ProSoft's technical specifications and documentation that are included with the Product are subject to editing and modification without notice.
- c) Transfer of title shall not operate to convey to Customer any right to make, or have made, any Product supplied by ProSoft.
- d) Customer is granted no right or license to use any software or other intellectual property in any manner or for any purpose not expressly permitted by any license agreement accompanying such software or other intellectual property.

- e) Customer agrees that it shall not, and shall not authorize others to, copy software provided by ProSoft (except as expressly permitted in any license agreement accompanying such software); transfer software to a third party separately from the Product; modify, alter, translate, decode, decompile, disassemble, reverse-engineer or otherwise attempt to derive the source code of the software or create derivative works based on the software; export the software or underlying technology in contravention of applicable US and international export laws and regulations; or use the software other than as authorized in connection with use of Product.
- f) **Additional Restrictions Relating To Software And Other Intellectual Property**

In addition to compliance with the Terms of this Warranty, Customers purchasing software or other intellectual property shall comply with any license agreement accompanying such software or other intellectual property. Failure to do so may void this Warranty with respect to such software and/or other intellectual property.

9.2.7 Disclaimer of all Other Warranties

The Warranty set forth in What Is Covered By This Warranty (page 165) are in lieu of all other warranties, express or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose.

9.2.8 No Other Warranties

Unless modified in writing and signed by both parties, this Warranty is understood to be the complete and exclusive agreement between the parties, suspending all oral or written prior agreements and all other communications between the parties relating to the subject matter of this Warranty, including statements made by salesperson. No employee of ProSoft or any other party is authorized to make any warranty in addition to those made in this Warranty. The Customer is warned, therefore, to check this Warranty carefully to see that it correctly reflects those terms that are important to the Customer.

9.2.9 Allocation of Risks

This Warranty allocates the risk of product failure between ProSoft and the Customer. This allocation is recognized by both parties and is reflected in the price of the goods. The Customer acknowledges that it has read this Warranty, understands it, and is bound by its Terms.

9.2.10 Controlling Law and Severability

This Warranty shall be governed by and construed in accordance with the laws of the United States and the domestic laws of the State of California, without reference to its conflicts of law provisions. If for any reason a court of competent jurisdiction finds any provisions of this Warranty, or a portion thereof, to be unenforceable, that provision shall be enforced to the maximum extent permissible and the remainder of this Warranty shall remain in full force and effect. Any cause of action with respect to the Product or Services must be instituted in a court of competent jurisdiction in the State of California.

9.3 How to Contact Us: Technical Support

Internet

Web Site: <http://www.prosoft-technology.com/support>
(<http://www.prosoft-technology.com/support>)

E-mail address: support@prosoft-technology.com
(<mailto:support@prosoft-technology.com>)

Asia Pacific

+603.7724.2080, support.asia@prosoft-technology.com
(<mailto:support.asia@prosoft-technology.com>)

Languages spoken include: Chinese, English

Europe (location in Toulouse, France)

+33 (0) 5.34.36.87.20, support.EMEA@prosoft-technology.com
(<mailto:support.emea@prosoft-technology.com>)

Languages spoken include: French, English

North America/Latin America (excluding Brasil) (location in California)

+1.661.716.5100, support@prosoft-technology.com (<mailto:support@prosoft-technology.com>)

Languages spoken include: English, Spanish

For technical support calls within the United States, an after-hours answering system allows pager access to one of our qualified technical and/or application support engineers at any time to answer your questions.

Brasil (location in Sao Paulo)

+55-11-5084-5178, eduardo@prosoft-technology.com (<mailto:eduardo@prosoft-technology.com>)

Languages spoken include: Portuguese, English

Index

I

[Backplane Data Exchange] • 53
 [DNP Master Commands] • 100
 [DNP Master Slave List] • 99
 [DNP Master] • 98
 [DNP Slave Analog Inputs] • 95
 [DNP Slave Binary Inputs] • 95
 [DNP Slave Database] • 92
 [DNP Slave Float Inputs] • 96
 [DNP Slave] • 86
 [IED Database] • 98
 [Module] • 52
 [Secondary Port] • 96

A

Add the PTQ Module to the Project • 16, 35
 AI Class • 89
 AI Deadband • 89
 AI Events with Time • 92
 All Product Returns: • 164
 Allocation of Risks • 168
 Analog Inputs • 93, 99
 Analog Outputs • 94, 99
 App Layer Confirm Tout • 90
 Application Layer Errors • 143

B

Baud Rate • 87, 97
 Before You Begin • 70
 BI Class • 89
 Binary Input Command Examples • 102
 Binary Inputs • 93, 98
 Binary Output Command Examples • 103
 Binary Outputs • 94, 99
 Block Definition • 61
 Block Format for Read • 68
 Block ID 9901 - CROB Control Block for Digital Outputs • 64
 Block ID 9902 - Command Control Block (Add command to Command List Queue) • 66
 Block ID 9903 - Event Pass-Through Block • 66
 Block ID 9958 - Binary Input Event • 81
 Block ID 9959 - Analog Input Event • 82
 Block ID 9970 - Set Processor's Time using the Module • 84
 Block ID 9971 - Set Module's Time Using Processor's Time • 85
 Block ID 9998 or 9999 - Reboot Module • 85
 Block Transfer Configuration • 62
 Block Transfer Structure • 61
 Build the Project • 37

C

Cable • 158
 Cable Connections • 44
 Can I use the module in a hot backup system? • 161
 CD Idle Time • 89, 97
 CD Random Time • 89, 97
 CD Time Before Receive • 89, 98
 Class 1 Unsol Resp Min • 91
 Class 2 Unsol Resp Min • 91
 Class 3 Unsol Resp Min • 91
 Cmd Function • 101
 Collision Avoidance • 88, 97
 Collision Avoidance (DNP modules only) • 48
 Command Error Codes • 142
 Command Error List • 63
 Configuring the Module • 51, 135
 Configuring the Processor with Concept • 11
 Configuring the Processor with ProWORX • 29
 Configuring the Processor with UnityPro XL • 33
 Connect the PC to the ProTalk Configuration/Debug Port • 43
 Connect Timeout • 88
 Connect Your PC to the Processor • 38
 Connecting to the Processor with TCP/IP • 39
 Controlling Law and Severability • 169
 Convert the EVENTFB Function Block • 71
 Counters • 93, 99
 Create a New Project • 13, 33

D

Data Block Capacities • 55
 Data Link Confirm Mode • 90
 Data Link Confirm Tout • 90
 Data Link Max Retry • 90
 Data Object • 101
 Data Requirements • 126
 Data Transfer at Startup • 135
 Data Transfer Interface • 129
 Data Types Table • 54
 Data Variation • 101
 Database View Menu • 117
 Defining Error/Status Functions - Function 2 • 58
 Defining Special Data Transfer Functions • 56
 Design • 126
 Designing the system • 126
 Device Address • 102
 Diagnostics and Troubleshooting • 7, 110, 111
 Disclaimer of all Other Warranties • 168
 Disclaimer Regarding High Risk Activities • 166
 Displaying the Current Page of Registers Again • 117
 DNP Analog Input Data • 133
 DNP Analog Output Data • 134
 DNP Collision Avoidance • 158
 DNP Counter Data • 132
 DNP DB Address • 102
 DNP Digital Input Data • 130
 DNP Digital Output Data • 132
 DNP Menu • 119
 DNP Port Configuration Errors • 142
 DNP Specific Questions • 161

DNP Subset Definition • 145
Do I need to use • 160
Does the module work in a remote rack? • 161
Double Deadband • 90
Download the Concept Project • 79
Download the Project to the Processor • 21, 40

E

Edit the Configuration File • 51
Error Codes • 142
Event Messages to PLC • 98
Event Pass-Through Block Format • 68
Event Size Computation • 157
EVENTFB Function Block Overview • 69
Example 0
 x or 10
 x Register Transfer • 54
Example 30
 x or 40
 x Register Transfer • 64
Exiting the Program • 116

F

Features and Benefits • 123
First Character Delay • 88
Float Class • 89
Float Deadband • 89
Float Inputs • 93
Float Outputs • 94
Frequently Asked Questions • 160
Functional Overview • 7, 125
Functional Specifications • 124
Functionality • 135

G

General Command Errors • 142
General Specifications • 123
Guide to the PTQ-DNP Application Reference Guide •
 7

H

Hardware and Software Requirements • 9
Hardware Specifications • 124
How do I configure the module? • 160
How fast do the • 160
How much data can I transfer between the PLC and
 the Module. • 160
How to Contact Us
 Technical Support • 164, 169
How to Set up and Use the Sample Function Block for
 Concept • 69

I

Idle Timeout • 88
IED DB Address • 102
Implementing Ladder to Support Special Functions. •
 59
Information for Concept Version 2.6 Users • 12
Initialize DNP Database • 92

Initialize IED Database • 98
Inserting the 1454-9F connector • 42
Install the ProTalk Module in the Quantum Rack • 41,
 42
Installing MDC Configuration Files • 12
Intellectual Property Indemnity • 167
Internal ID • 98
Internal Slave ID • 87
Is a .MDC available for configuration of the Module? •
 161

K

Keystrokes • 112

L

LED Status Indicators • 121
Limitation of Remedies ** • 167
LIMITED WARRANTY • 165

M

Main Menu • 114
Min Response Delay • 97
Minimum Response Delay • 87
Modem • 88
Module Master Port • 152
Module Name • 52
Module Operation • 136
Module Status • 63
Moving Back Through 5 Pages of Registers • 118

N

Navigation • 111
No Other Warranties • 168
Node Address • 101

O

Obtain the Sample Configuration Files • 51
Opening the Database Menu • 115
Opening the DNP Database View Menu • 121
Opening the DNP Menu • 115

P

PC and PC Software • 10
Phone Number • 88
PLC Analog Inputs • 93
PLC Analog Outputs • 94
PLC Binary Inputs • 93
PLC Binary Outputs • 94
PLC Counters • 94
PLC Float Inputs • 93
PLC Float Outputs • 94
Please Read This Notice • 2
Point Count • 102
Poll Interval • 102
Procedures for Return of Units Out of Warranty: • 164
Procedures for Return of Units Under Warranty: • 163
Product Specifications • 7, 123
ProTalk Module Carton Contents • 9
PTQ Installation and Operating Instructions • 2

PTQ-DNP Error Status Table • 137
PTQ-DNP Module Internal Indication Bits (IIN Bits) for
DNP Server • 144
Purchasing Warranty Extension: • 165

Q

Quantum / Unity Hardware • 10

R

Receiving the Configuration File • 115
Redial Delay Time • 88
Redial Random Delay • 88
Reference • 7, 123
Required Hardware • 104, 112
Required Software • 105, 112
Return Material Authorization (RMA) Policies and
Conditions • 163
Returning to the Main Menu • 118
RS-232 • 45
 Modem Connection • 46
 Null Modem Connection (Hardware Handshaking)
 • 46
 Null Modem Connection (No Hardware
 Handshaking) • 47
RS-232 Configuration/Debug Port • 45
RS-422 • 48
RS-485 • 47
RS-485 and RS-422 Tip • 48
RTS Off • 87, 97
RTS On • 87, 97
Rules • 158

S

Select/Operate Arm Time • 90
Sending the Configuration File • 115
Set Up Command Function 2 (Read Status Error /
Status) • 62
Set Up Command Function 3 (Special Functions) • 64
Set up Data Memory in Project • 18
Setting up the Commands • 53
Setting Up the ProTalk Module • 41
Setup the Concept Project • 73
Skipping 500 Registers of Data • 118
Slave Port Communication Errors • 140
Slave Status • 63
Special Control Block Codes • 57
Special Control Block Codes Example • 57
Special Functions • 64
Start Here • 7, 9
Support, Service & Warranty • 7, 163
System Configuration Errors • 141

T

The Configuration/Debug Menu • 104, 111
Time Limit for Bringing Suit • 167
Time Sync Before Events • 92
Timing Chart • 159
Transferring the Configuration File to the Module • 108
Transferring the Configuration File to Your PC • 105
Type • 97

U

Unsol Resp Delay • 91
Unsolicited Response • 91
Uploading and Downloading the Configuration File •
104, 115
UResp Master Address • 91
UResp Retry Count • 91
Using the Configuration/Debug Port • 113
Using the EVENTFB Function Block • 80

V

Verification and Troubleshooting • 7, 110
Verify Jumper Settings • 41
Verify Successful Download • 24
Viewing a List of Valid Hosts • 120
Viewing Block Transfer Statistics • 114
Viewing Data in ASCII (Text) Format • 118
Viewing Data in Decimal Format • 118
Viewing Data in Floating Point Format • 118
Viewing Data in Hexadecimal Format • 118
Viewing DNP Communication Status • 120
Viewing Memory Allocation and Database Setup
Parameters • 119
Viewing Module Configuration • 114
Viewing Register Pages • 117
Viewing Server Configuration Information • 119
Viewing TCP Socket Status • 120
Viewing the Backplane Command List • 115
Viewing the Next 100 Registers of Data • 118
Viewing the Previous 100 Registers of Data • 118
Viewing UDP Socket Status • 121
Viewing Version Information • 115

W

Warm Booting the Module • 116
What does • 161
What Is Covered By This Warranty • 165, 168
What Is Not Covered By This Warranty • 166
What is the maximum number of words I can transfer
with a • 160
What software application is required for my Ladder
Logic? • 160
When Required • 158
Where do the individual data types actually exist in the
modules memory? • 161
Write Time Interval • 90

Y

Your Feedback Please • 3