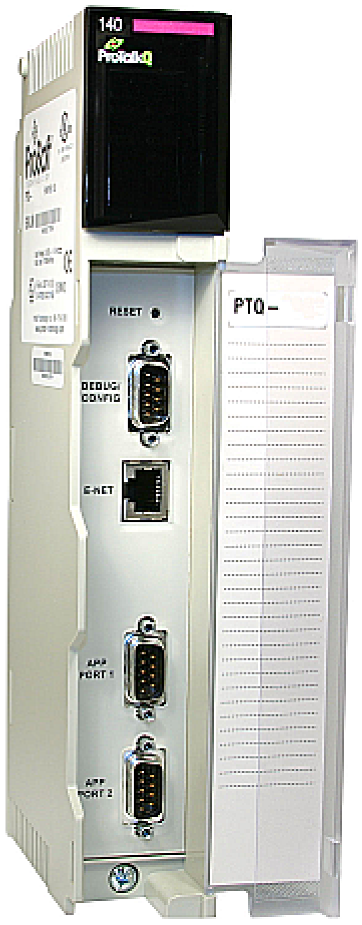
ProSoft® TECHNOLOGY

Where Automation Connects.



ProTalk®

PTQ-DNPSNET-Q

Quantum / Unity Platform

Distributed Network Protocol Interface Module

July 07, 2009

USER MANUAL

Information for ProTalk® Product Users

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.

WARNING - EXPLOSION HAZARD - DO NOT DISCONNECT EQUIPMENT UNLESS POWER HAS BEEN SWITCHED OFF OR THE AREA IS KNOWN TO BE NON-HAZARDOUS.

AVERTISSEMENT - RISQUE D'EXPLOSION - AVANT DE DÉCONNECTER L'EQUIPMENT, COUPER LE COURANT OU S'ASSURER QUE L'EMPLACEMENT EST DÉSIGNÉ NON DANGEREUX.

CL I Div 2 GPs A, B, C, D

Temp Code T5

II 3 G

Ex nA IIC T5 X

0° C <= Ta <= 60° C

II - Equipment intended for above ground use (not for use in mines).

3 - Category 3 equipment, investigated for normal operation only.

G - Equipment protected against explosive gasses.

Warnings

North America Warnings

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 or rewiring modules.
Warning - Explosion Hazard - Do not disconnect equipment unless power has been switched off or the area is known to be nonhazardous.
C Suitable for use in Class I, division 2 Groups A, B, C and D Hazardous Locations or Non-Hazardous Locations.

ATEX Warnings and Conditions of Safe Usage:

Power, Input, and Output (I/O) wiring must be in accordance with the authority having jurisdiction

A Warning - Explosion Hazard - When in hazardous locations, turn off power before replacing or wiring modules.
B Warning - Explosion Hazard - Do not disconnect equipment unless power has been switched off or the area is known to be non-hazardous.
C These products are intended to be mounted in an IP54 enclosure. The devices shall provide external means to prevent the rated voltage being exceeded by transient disturbances of more than 40%. This device must be used only with ATEX certified backplanes.
D DO NOT OPEN WHEN ENERGIZED.

Electrical Ratings

- Backplane Current Load: 800 mA @ 5 V DC; 3mA @ 24V DC
- Operating Temperature: 0 to 60°C (32 to 140°F)
- Storage Temperature: -40 to 85°C (-40 to 185°F)
- Shock: 30g Operational; 50g non-operational; Vibration: 5 g from 10 to 150 Hz
- Relative Humidity 5% to 95% (non-condensing)
- All phase conductor sizes must be at least 1.3 mm(squared) and all earth ground conductors must be at least 4mm(squared).

Markings:

ANSI / ISA	ISA 12.12.01 Class I Division 2, GPs A, B, C, D
CSA/cUL	C22.2 No. 213-1987
CSA CB Certified	IEC61010
ATEX	EN60079-0 Category 3, Zone 2 EN60079-15

243333

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 µA. Maximum battery charge voltage = 3.4 VDC. Maximum battery charge current = 500 µA. Maximum battery discharge current = 30 µA.

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
5201 Truxtun Ave., 3rd Floor
Bakersfield, CA 93309
+1 (661) 716-5100
+1 (661) 716-5101 (Fax)
www.prosoft-technology.com

PTQ-DNPSNET-Q User Manual
July 07, 2009

ProSoft Technology® Product Documentation

In an effort to conserve paper, ProSoft Technology no longer includes printed manuals with our product shipments. User Manuals, Datasheets, Sample Ladder Files, and Configuration Files are provided on the enclosed CD, and are available at no charge from our web site: www.prosoft-technology.com

Printed documentation is available for purchase. Contact ProSoft Technology for pricing and availability.

Asia Pacific: +603.7724.2080

Europe, Middle East, Africa: +33 (0) 5.3436.87.20

Latin America: +1.281.298.9109

North America: +1.661.716.5100

Contents

Guide to the PTQ-DNPSNET-Q User Manual

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 77) Diagnostics and Troubleshooting (page 65)	This section describes how to verify communications with the network. Diagnostic and Troubleshooting procedures.
Reference Product Specifications Functional Overview	→	Reference (page 79) Functional Overview (page 83) Product Specifications (page 79)	These sections contain general references associated with this product, Specifications, and the Functional Overview.
Support, Service, and Warranty Index	→	Support, Service and Warranty (page 161)	This section contains Support, Service and Warranty information. Index of chapters.

1 Start Here

In This Chapter

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.

1.2 Install ProSoft Configuration Builder Software

You must install the ProSoft Configuration Builder (PCB) software in order to configure the module. You can always get the newest version of ProSoft Configuration Builder from the ProSoft Technology web site.

To install ProSoft Configuration Builder from the ProSoft Web Site

1 Open your web browser and navigate to *http://www.prosoft-technology.com/pcb*
2 Click the **DOWNLOAD HERE** link to download the latest version of ProSoft Configuration Builder.
3 Choose **"SAVE"** or **"SAVE FILE"** when prompted.
4 Save the file to your Desktop, so that you can find it easily when you have finished downloading.
5 When the download is complete, locate and open the file, and then follow the instructions on your screen to install the program.

If you do not have access to the Internet, you can install ProSoft Configuration Builder from the ProSoft Solutions CD-ROM, included in the package with your module.

To install ProSoft Configuration Builder from the Product CD

1. Insert the ProSoft Solutions Product CD into the CD drive of your PC. Wait for the startup screen to appear.
2. On the startup screen, click **PRODUCT DOCUMENTATION**. This action opens an explorer window.
3. Click to open the **UTILITIES** folder. This folder contains all of the applications and files you will need to set up and configure your module.
4. Double-click the **SETUPCONFIGURATIONTOOL** folder, double-click the **"PCB_*.EXE"** file and follow the instructions on your screen to install the software on your PC. The information represented by the "*****" character in the file name is the PCB version number and, therefore, subject to change as new versions of PCB are released.

Note: Many of the configuration and maintenance procedures use files and other utilities on the CD-ROM. You may wish to copy the files from the Utilities folder on the CD-ROM to a convenient location on your hard drive.

2 Configuring the Processor with Concept

In This Chapter

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 consider this when monitoring the status of the PTQ module.

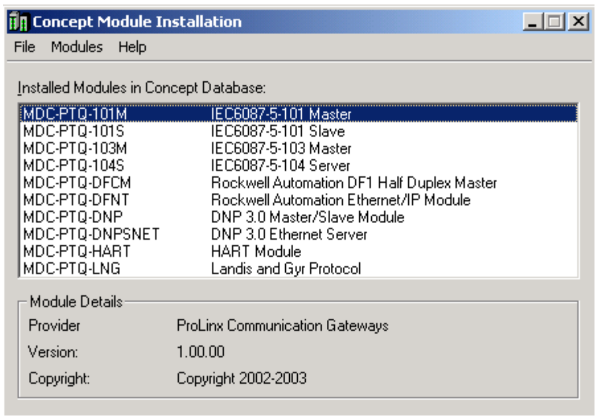
2.1 Information for Concept Version 2.6 Users

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.



Installed Modules in Concept Database:

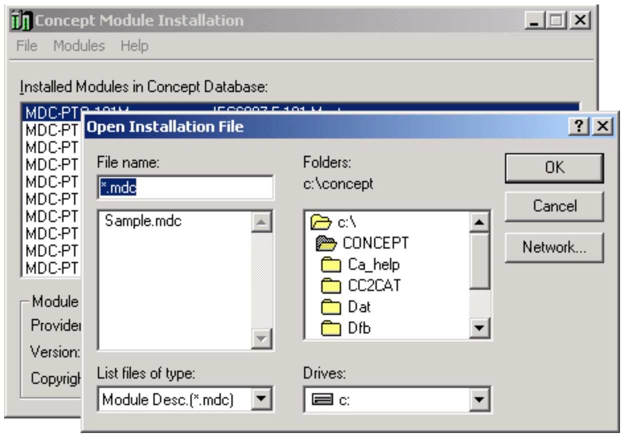
MDC-PTQ-101M	IEC6087-5-101 Master
MDC-PTQ-101S	IEC6087-5-101 Slave
MDC-PTQ-103M	IEC6087-5-103 Master
MDC-PTQ-104S	IEC6087-5-104 Server
MDC-PTQ-DFCM	Rockwell Automation DF1 Half Duplex Master
MDC-PTQ-DFNT	Rockwell Automation Ethernet/IP Module
MDC-PTQ-DNP	DNP 3.0 Master/Slave Module
MDC-PTQ-DNPSNET	DNP 3.0 Ethernet Server
MDC-PTQ-HART	HART Module
MDC-PTQ-LNG	Landis and Gyr Protocol

Module Details

Provider	ProLinx Communication Gateways
Version:	1.00.00
Copyright:	Copyright 2002-2003

2 Choose **FILE / OPEN INSTALLATION FILE.**

This action opens the Open Installation File dialog box:



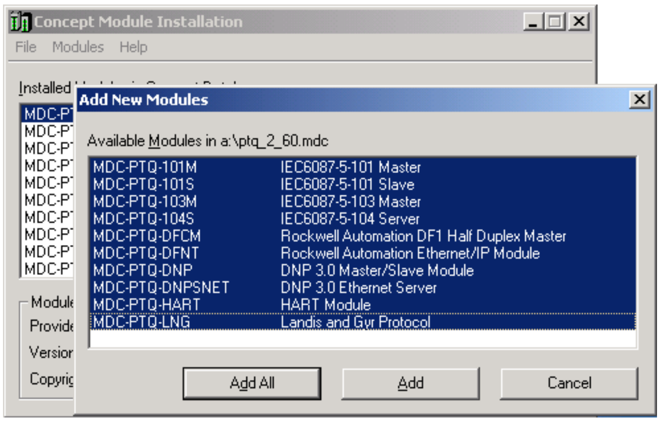
File name: *.mdc
Sample.mdc
Folders: c:\concept
List files of type: Module Desc.(*.mdc)
Drives: c:

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:

- Concept 2.6 users: select PTQ_2_60.mdc and PTQMDC.hlp
- 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.



MDC-PTQ-101M	IEC6087-5-101 Master
MDC-PTQ-101S	IEC6087-5-101 Slave
MDC-PTQ-103M	IEC6087-5-103 Master
MDC-PTQ-104S	IEC6087-5-104 Server
MDC-PTQ-DFCM	Rockwell Automation DF1 Half Duplex Master
MDC-PTQ-DFNT	Rockwell Automation Ethernet/IP Module
MDC-PTQ-DNP	DNP 3.0 Master/Slave Module
MDC-PTQ-DNPSNET	DNP 3.0 Ethernet Server
MDC-PTQ-HART	HART Module
MDC-PTQ-LNG	Landis and Gyr Protocol

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.

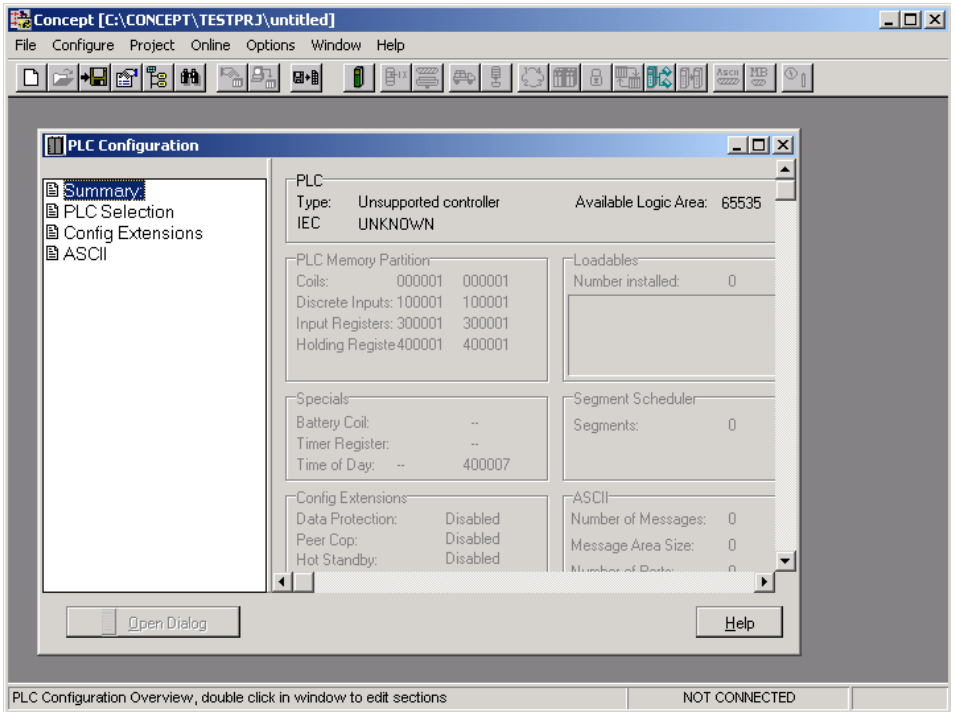
6 When the process is complete, open the File menu and choose Exit to save your changes.

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.


Concept [C:\CONCEPT\TESTPRJ\untitled]
File Configure Project Online Options Window Help
PLC Configuration
Summary
PLC Selection
Config Extensions
ASCII
Open Dialog
Help

PLC
Type: Unsupported controller   Available Logic Area: 65535
IEC UNKNOWN

PLC Memory Partition
Coils: 000001 000001
Discrete Inputs: 100001 100001
Input Registers: 300001 300001
Holding Registe 400001 400001

Loadables
Number installed: 0

Specials
Battery Coil: --
Timer Register: --
Time of Day: -- 400007

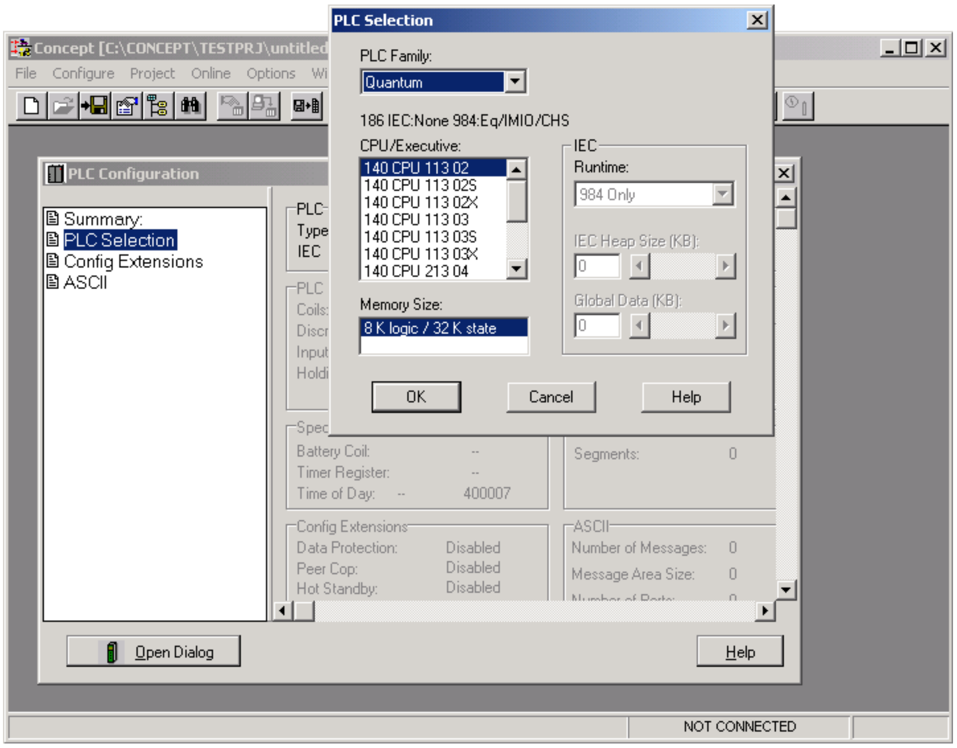
Segment Scheduler
Segments: 0

Config Extensions
Data Protection: Disabled
Peer Cop: Disabled
Hot Standby: Disabled

ASCII
Number of Messages: 0
Message Area Size: 0

PLC Configuration Overview, double click in window to edit sections   NOT CONNECTED

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.



PLC Selection

PLC Family: Quantum

186 IEC:None 984:Eq/IMIO/CHS

CPU/Executive:
140 CPU 113 02
140 CPU 113 02S
140 CPU 113 02X
140 CPU 113 03
140 CPU 113 03S
140 CPU 113 03X
140 CPU 213 04

Memory Size: 8 K logic / 32 K state

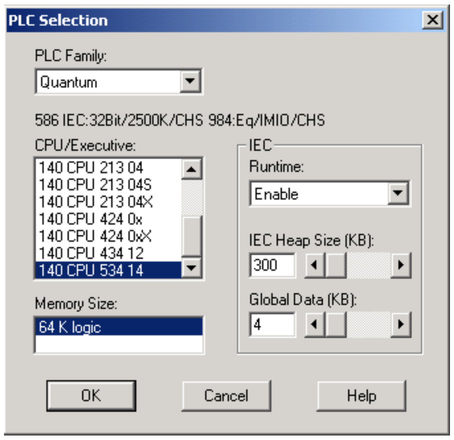
IEC
Runtime: 984 Only
IEC Heap Size (KB): 0
Global Data (KB): 0

Battery Coil: --
Timer Register: --
Time of Day: -- 400007
Segments: 0

Config Extensions
Data Protection: Disabled
Peer Cop: Disabled
Hot Standby: Disabled

ASCII
Number of Messages: 0
Message Area Size: 0

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



PLC Selection

PLC Family: Quantum

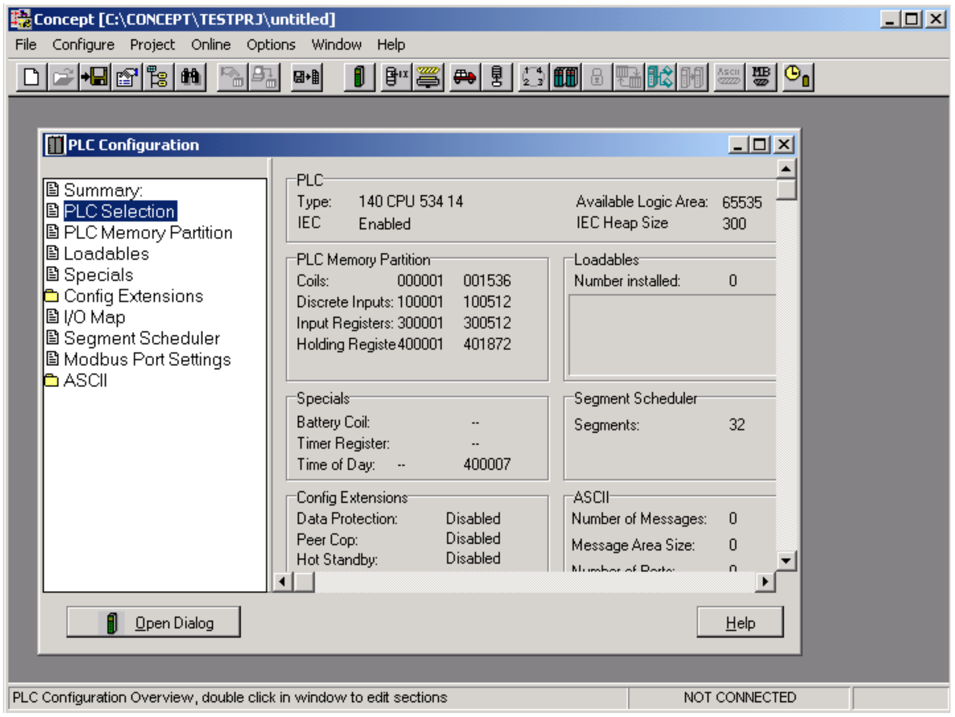
586 IEC:32Bit/2500K/CHS 984:Eq/IMIO/CHS

CPU/Executive:
140 CPU 213 04
140 CPU 213 04S
140 CPU 213 04X
140 CPU 424 0x
140 CPU 424 0xX
140 CPU 434 12
140 CPU 534 14

Memory Size: 64 K logic

IEC
Runtime: Enable
IEC Heap Size (KB): 300
Global Data (KB): 4

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



PLC
Type: 140 CPU 534 14 Available Logic Area: 65535
IEC Enabled IEC Heap Size 300

PLC Memory Partition

Coils:	000001	001536
Discrete Inputs:	100001	100512
Input Registers:	300001	300512
Holding Registe	400001	401872

Loadables
Number installed: 0

Specials

Battery Coil:		--
Timer Register:		--
Time of Day:	--	400007

Segment Scheduler
Segments: 32

Config Extensions

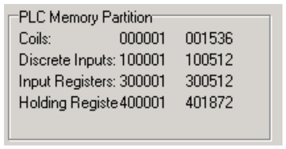
Data Protection:	Disabled
Peer Cop:	Disabled
Hot Standby:	Disabled

ASCII

Number of Messages:	0
Message Area Size:	0
Number of Ports:	0

PLC Configuration Overview, double click in window to edit sections NOT CONNECTED

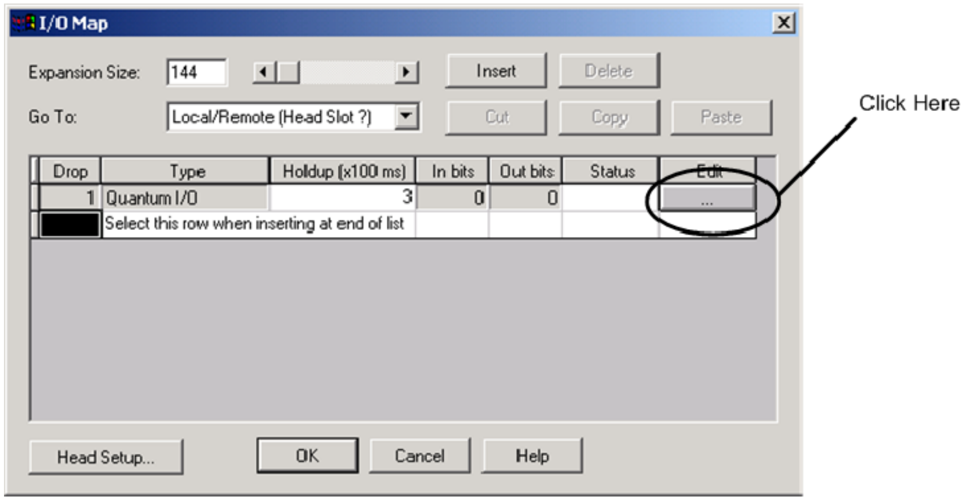
6 Make a note of the holding registers for the module. You will need this information when you modify your application. The Holding Registers are displayed in the PLC Memory Partition pane of the **PLC CONFIGURATION** dialog box.


PLC Memory Partition

Coils:	000001	001536
Discrete Inputs:	100001	100512
Input Registers:	300001	300512
Holding Registe	400001	401872

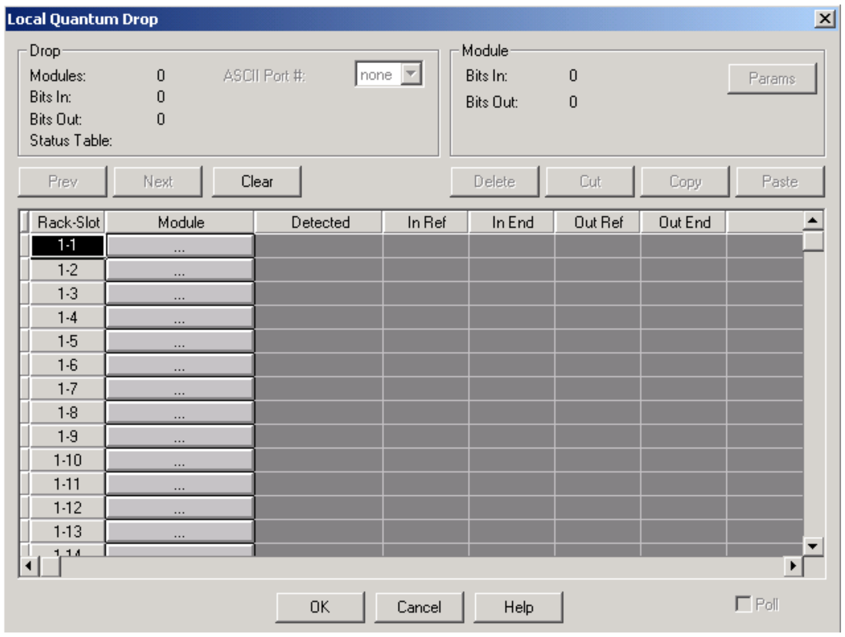
2.3 Add the PTQ Module to the Project

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.



Drop	Type	Holdup (x100 ms)	In bits	Out bits	Status	Edit
1	Quantum I/O	3	0	0		...
	Select this row when inserting at end of list					

2 Click the **EDIT** button to open the **LOCAL QUANTUM DROP** dialog box. This dialog box is where you identify rack and slot locations.

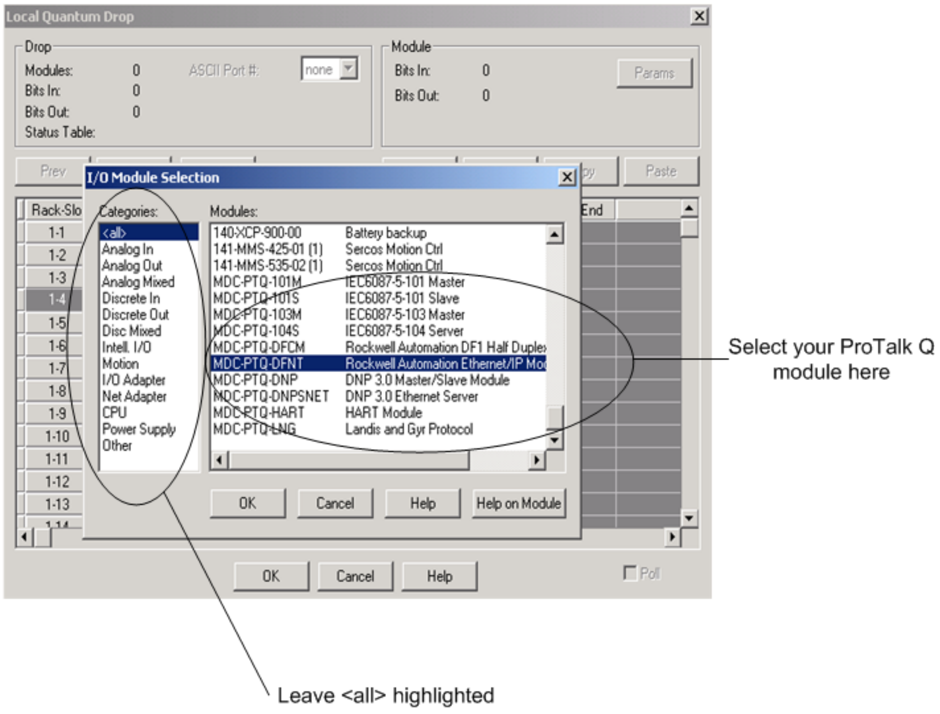


Drop: Modules: 0, Bits In: 0, Bits Out: 0, Status Table:

Module: Bits In: 0, Bits Out: 0

Rack-Slot	Module	Detected	In Ref	In End	Out Ref	Out End
1-1	...					
1-2	...					
1-3	...					
1-4	...					
1-5	...					
1-6	...					
1-7	...					
1-8	...					
1-9	...					
1-10	...					
1-11	...					
1-12	...					
1-13	...					

3 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.

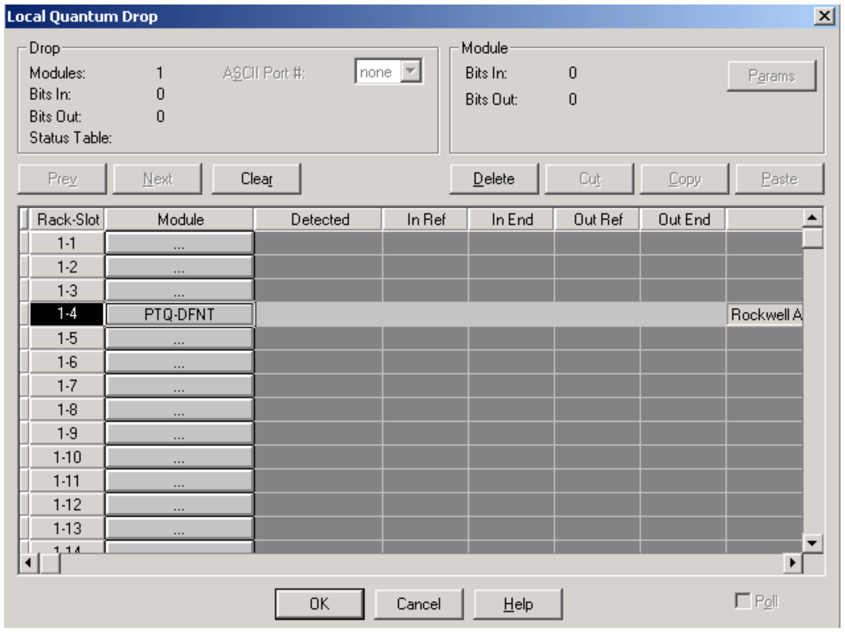


Drop: Modules: 0; Bits In: 0; Bits Out: 0; Status Table:; ASCII Port #: none
Module: Bits In: 0; Bits Out: 0

Categories: <all>, Analog In, Analog Out, Analog Mixed, Discrete In, Discrete Out, Disc Mixed, Intell. I/O, Motion, I/O Adapter, Net Adapter, CPU, Power Supply, Other

Modules:	
140-XCP-900-00	Battery backup
141-MMS-425-01 (1)	Sercos Motion Ctrl
141-MMS-535-02 (1)	Sercos Motion Ctrl
MDC-PTQ-101M	IEC6087-5-101 Master
MDC-PTQ-101S	IEC6087-5-101 Slave
MDC-PTQ-103M	IEC6087-5-103 Master
MDC-PTQ-104S	IEC6087-5-104 Server
MDC-PTQ-DFCM	Rockwell Automation DF1 Half Duplex
MDC-PTQ-DFNT	Rockwell Automation Ethernet/IP Mod
MDC-PTQ-DNP	DNP 3.0 Master/Slave Module
MDC-PTQ-DNPSNET	DNP 3.0 Ethernet Server
MDC-PTQ-HART	HART Module
MDC-PTQ-LNG	Landis and Gyr Protocol

4 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.



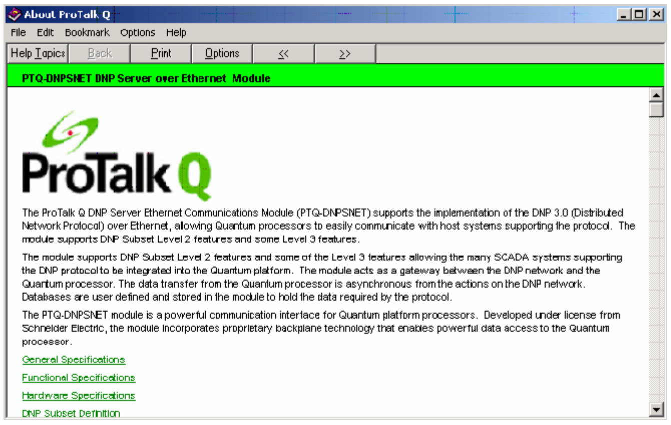
Drop
Modules: 1
Bits In: 0
Bits Out: 0
Status Table:

Module
Bits In: 0
Bits Out: 0

Rack-Slot	Module	Detected	In Ref	In End	Out Ref	Out End	
1-1	...						
1-2	...						
1-3	...						
1-4	PTQ-DFNT						Rockwell A
1-5	...						
1-6	...						
1-7	...						
1-8	...						
1-9	...						
1-10	...						
1-11	...						
1-12	...						
1-13	...						

5 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.



PTQ-DNPSNET DNP Server over Ethernet Module

The ProTalk Q DNP Server Ethernet Communications Module (PTQ-DNPSNET) supports the implementation of the DNP 3.0 (Distributed Network Protocol) over Ethernet, allowing Quantum processors to easily communicate with host systems supporting the protocol. The module supports DNP Subset Level 2 features and some Level 3 features.

The module supports DNP Subset Level 2 features and some of the Level 3 features allowing the many SCADA systems supporting the DNP protocol to be integrated into the Quantum platform. The module acts as a gateway between the DNP network and the Quantum processor. The data transfer from the Quantum 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.

The PTQ-DNPSNET module is a powerful communication interface for Quantum platform processors. Developed under license from Schneider Electric, the module incorporates proprietary backplane technology that enables powerful data access to the Quantum processor.

General Specifications

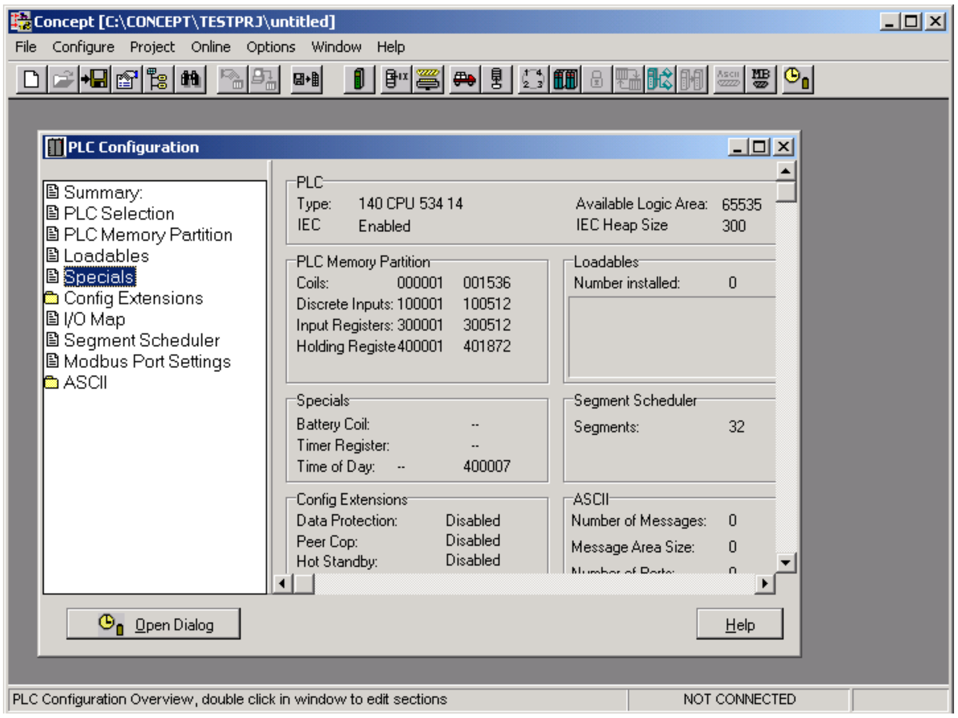
Functional Specifications

Hardware Specifications

DNP Subset Definition

2.4 Set up Data Memory in Project

1 In the list of options on the left side of the **PLC CONFIGURATION** dialog box, double-click **SPECIALS.**



PLC
Type: 140 CPU 534 14 Available Logic Area: 65535
IEC Enabled IEC Heap Size 300

PLC Memory Partition
Coils: 000001 001536
Discrete Inputs: 100001 100512
Input Registers: 300001 300512
Holding Registe 400001 401872

Loadables
Number installed: 0

Specials
Battery Coil: --
Timer Register: --
Time of Day: -- 400007

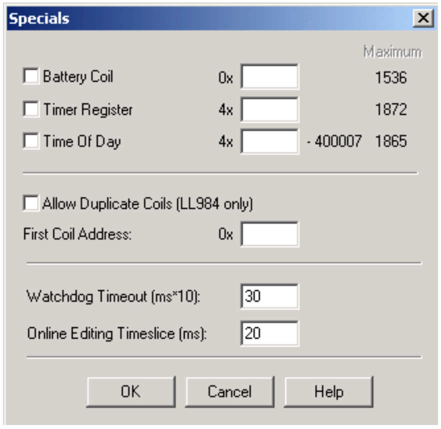
Segment Scheduler
Segments: 32

Config Extensions
Data Protection: Disabled
Peer Cop: Disabled
Hot Standby: Disabled

ASCII
Number of Messages: 0
Message Area Size: 0

PLC Configuration Overview, double click in window to edit sections NOT CONNECTED

2 This action opens the **SPECIALS** dialog box.



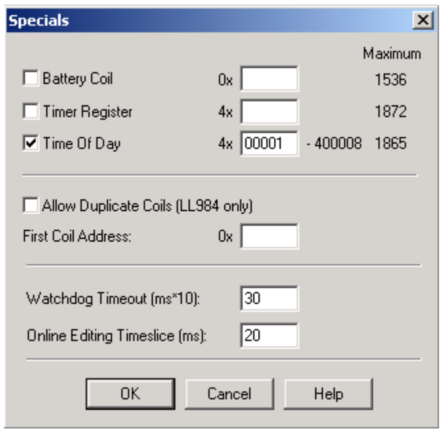
			Maximum
☐ Battery Coil	0x		1536
☐ Timer Register	4x		1872
☐ Time Of Day	4x	- 400007	1865

☐ Allow Duplicate Coils (LL984 only)
First Coil Address: 0x

Watchdog Timeout (ms*10): 30
Online Editing Timeslice (ms): 20

Selecting the Time of Day

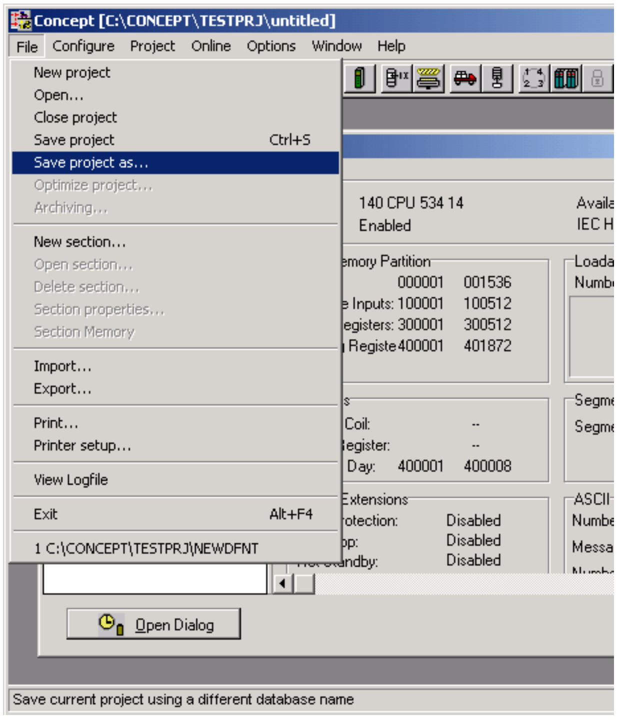
1 Select (check) the **TIME OF DAY** box, and then enter the value 00001 as shown in the following illustration. This value sets the first time of day register to 400001.


Specials
Maximum
Battery Coil 0x 1536
Timer Register 4x 1872
☑ Time Of Day 4x 00001 - 400008 1865
Allow Duplicate Coils (LL984 only)
First Coil Address: 0x
Watchdog Timeout (ms*10): 30
Online Editing Timeslice (ms): 20
OK
Cancel
Help

2 Click **OK** to save your settings and close the **SPECIALS** dialog box.

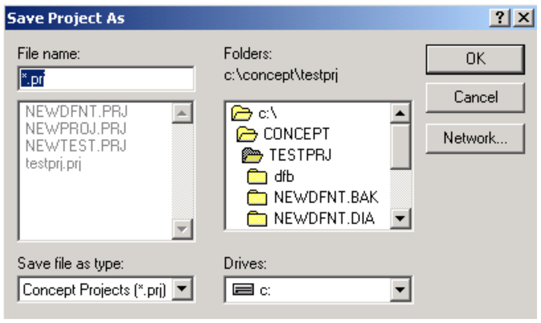
Saving your project

1 In the **PLC CONFIGURATION** dialog box, choose **FILE / SAVE PROJECT AS.**



Save current project using a different database name

2 This action opens the **SAVE PROJECT AS** dialog box.

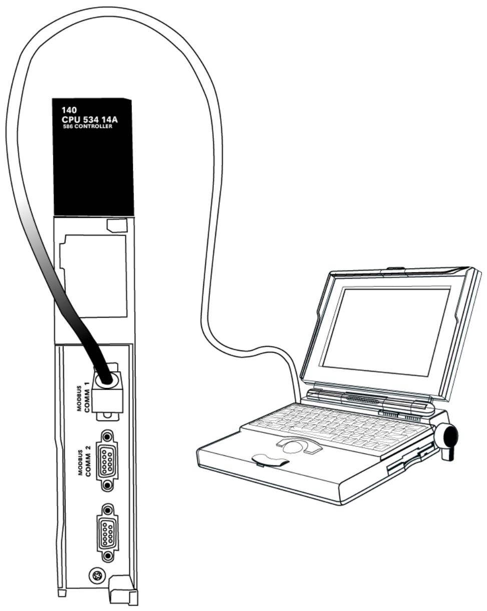


3 Name the project, and then click **OK** to save the project to a file.

2.5 Download the Project to the Processor

Next, download (copy) the project file to the Quantum Processor.

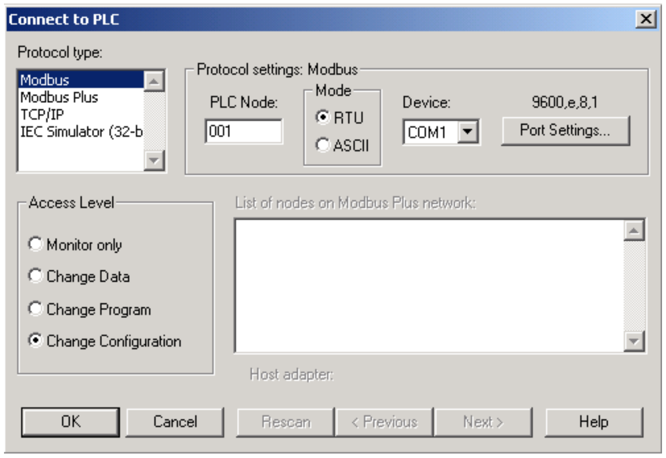
1 Use the null modem cable to connect your PC's serial port to the Quantum processor, as shown in the following illustration.



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.



Protocol type: [✓ Modbus] Modbus Plus TCP/IP IEC Simulator (32-b

Protocol settings: Modbus

PLC Node: 001

Mode: [✓ RTU] ASCII

Device: COM1

9600,e,8,1

Access Level: Monitor only / Change Data / Change Program / [✓ Change Configuration]

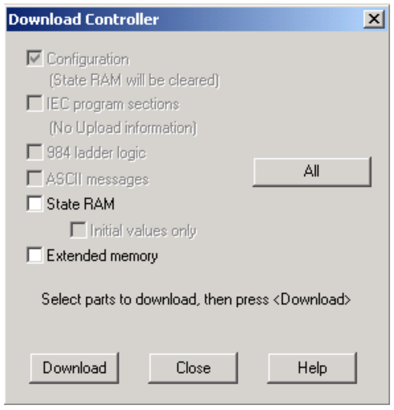
List of nodes on Modbus Plus network:

Host adapter:

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.



[✓] Configuration
(State RAM will be cleared)

☐ IEC program sections
(No Upload information)

☐ 984 ladder logic

☐ ASCII messages

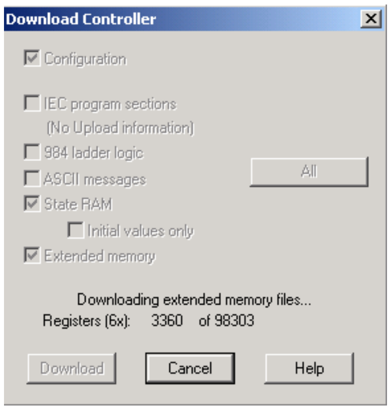
☐ State RAM

☐ Initial values only

☐ Extended memory

Select parts to download, then press <Download>

6 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.



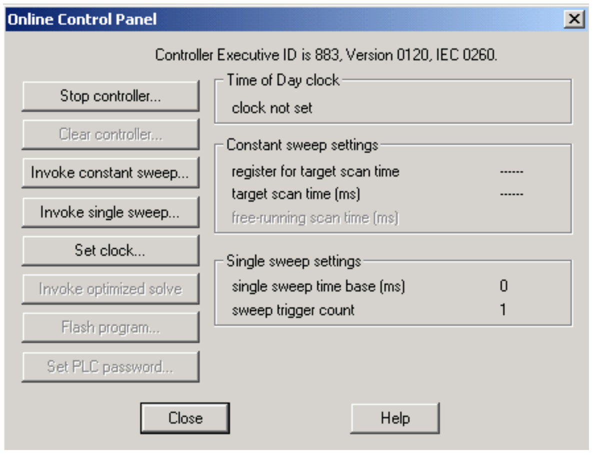
Downloading extended memory files...
Registers (6x): 3360 of 98303

7 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.



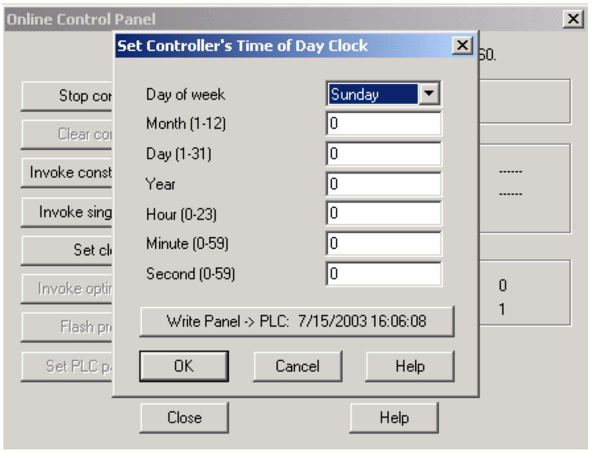
Controller Executive ID is 883, Version 0120, IEC 0260.

Time of Day clock
clock not set

Constant sweep settings
register for target scan time ------
target scan time (ms) ------
free-running scan time (ms)

Single sweep settings
single sweep time base (ms) 0
sweep trigger count 1

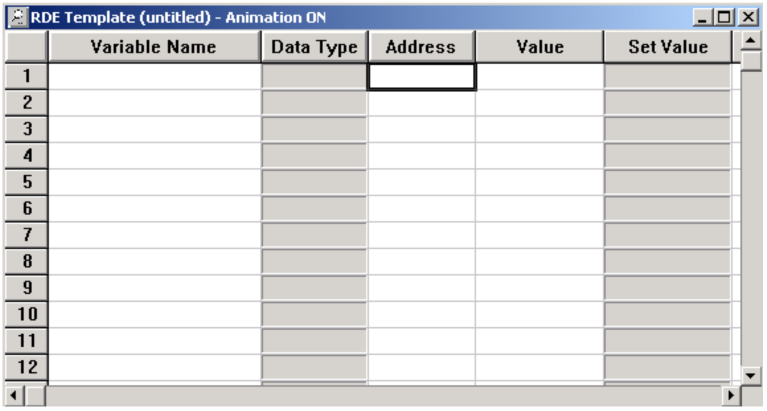
2 Click the **SET CLOCK** button to open the **SET CONTROLLER'S TIME OF DAY CLOCK** dialog box.



Day of week	Sunday
Month (1-12)	0
Day (1-31)	0
Year	0
Hour (0-23)	0
Minute (0-59)	0
Second (0-59)	0

Write Panel -> PLC: 7/15/2003 16:06:08

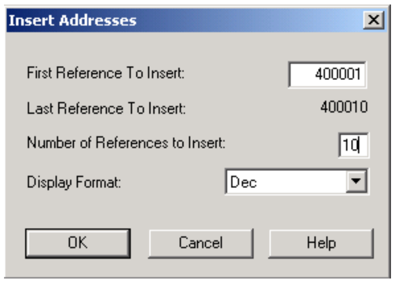
3 Click the **WRITE PANEL** button. This action updates the date and time fields in this dialog box. Click **OK** to close this dialog box and return to the previous window.
4 Click **CLOSE** to close the **ONLINE CONTROL PANEL** dialog box.
5 In the **PLC CONFIGURATION** window, open the **ONLINE** 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					

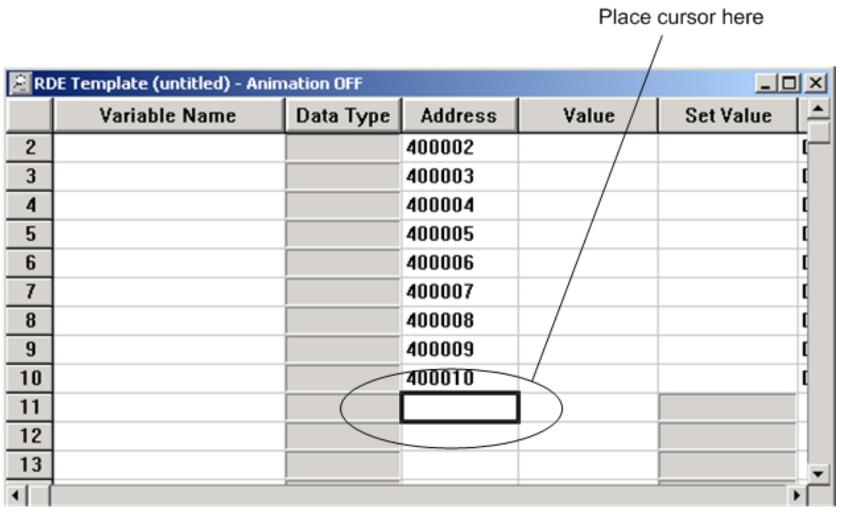
7 In the **PLC CONFIGURATION** window, open the **TEMPLATES** menu, and then choose **INSERT ADDRESSES.** This action opens the Insert addresses dialog box.

8 On the **INSERT ADDRESSES** dialog box, enter the values shown in the following illustration, and then click **OK.**



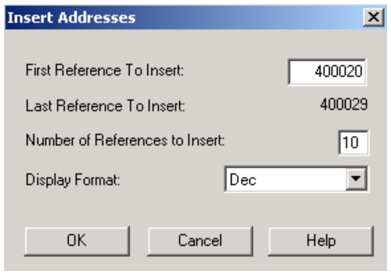
First Reference To Insert: 400001
Last Reference To Insert: 400010
Number of References to Insert: 10
Display Format: Dec

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.



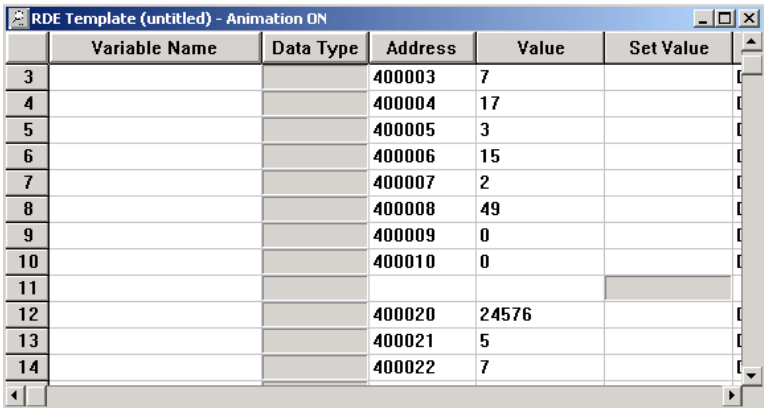
	Variable Name	Data Type	Address	Value	Set Value
2			400002		
3			400003		
4			400004		
5			400005		
6			400006		
7			400007		
8			400008		
9			400009		
10			400010		
11					
12					
13					

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



First Reference To Insert: 400020
Last Reference To Insert: 400029
Number of References to Insert: 10
Display Format: Dec

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 and 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.

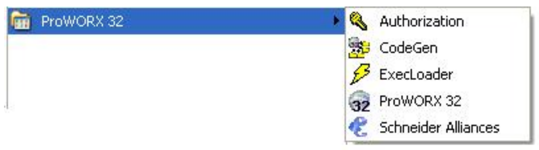
You are now ready to complete the installation and setup of the ProTalk module.

3 Configuring the Processor with ProWORX

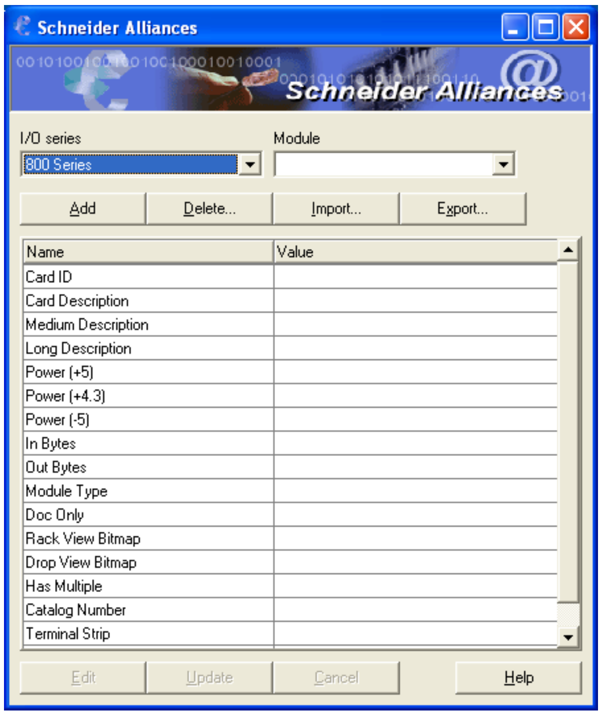
When you use ProWORX 32 software to configure the processor, use the example SAF file provided on the ProTalk Solutions CD-ROM.

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 consider this when monitoring the status of the PTQ module.

1 Run the **SCHNEIDER_ALLIANCES.EXE** application that is installed with the ProWORX 32 software:

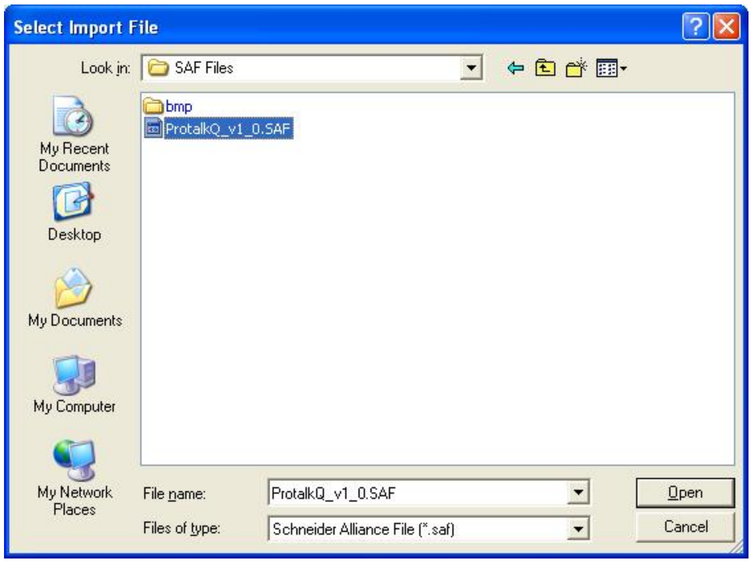


2 Click on **IMPORT…**

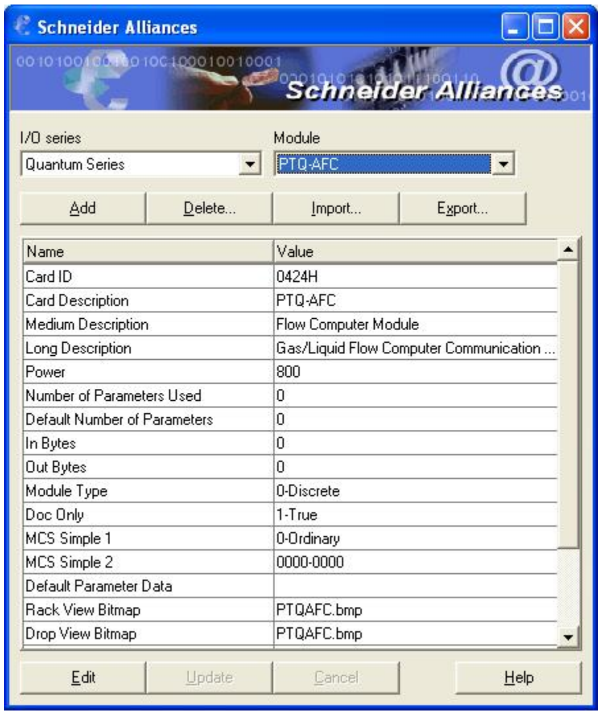


Name	Value
Card ID	
Card Description	
Medium Description	
Long Description	
Power (+5)	
Power (+4.3)	
Power (-5)	
In Bytes	
Out Bytes	
Module Type	
Doc Only	
Rack View Bitmap	
Drop View Bitmap	
Has Multiple	
Catalog Number	
Terminal Strip	

3 Select the .SAF File that is located on the CD-ROM shipped with the PTQ module.



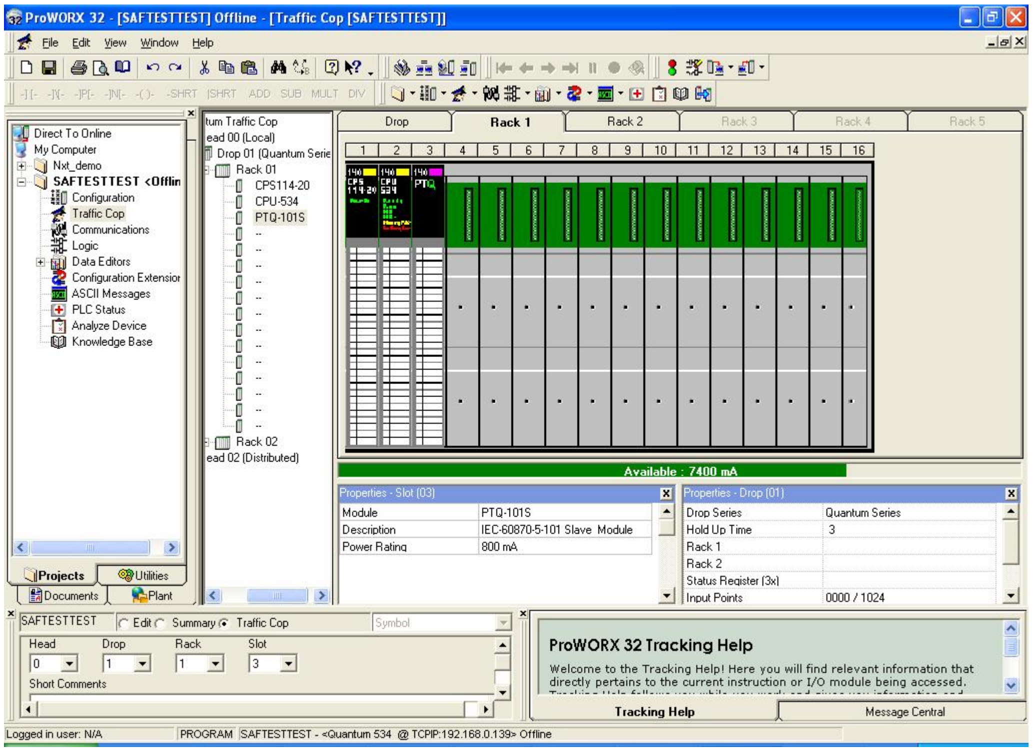
4 After you click on **OPEN** you should see the PTQ modules imported (select **I/O SERIES** as **QUANTUM**):



I/O series: Quantum Series
Module: PTQ-AFC

Name	Value
Card ID	0424H
Card Description	PTQ-AFC
Medium Description	Flow Computer Module
Long Description	Gas/Liquid Flow Computer Communication ...
Power	800
Number of Parameters Used	0
Default Number of Parameters	0
In Bytes	0
Out Bytes	0
Module Type	0-Discrete
Doc Only	1-True
MCS Simple 1	0-Ordinary
MCS Simple 2	0000-0000
Default Parameter Data	
Rack View Bitmap	PTQAFC.bmp
Drop View Bitmap	PTQAFC.bmp

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:



Properties - Slot (03)

Module	PTQ-101S
Description	IEC-60870-5-101 Slave Module
Power Rating	800 mA

Properties - Drop (01)

Drop Series	Quantum Series
Hold Up Time	3
Rack 1	
Rack 2	
Status Register (3x)	
Input Points	0000 / 1024

SAFTESTTEST (Edit (Summary (• Traffic Cop

Head: 0 Drop: 1 Rack: 1 Slot: 3

Short Comments

ProWORX 32 Tracking Help

Welcome to the Tracking Help! Here you will find relevant information that directly pertains to the current instruction or I/O module being accessed.

Tracking Help | Message Central

Logged in user: N/A | PROGRAM | SAFTESTTEST - <Quantum 534 @ TCPIP:192.168.0.139> Offline

4 Configuring the Processor with UnityPro XL

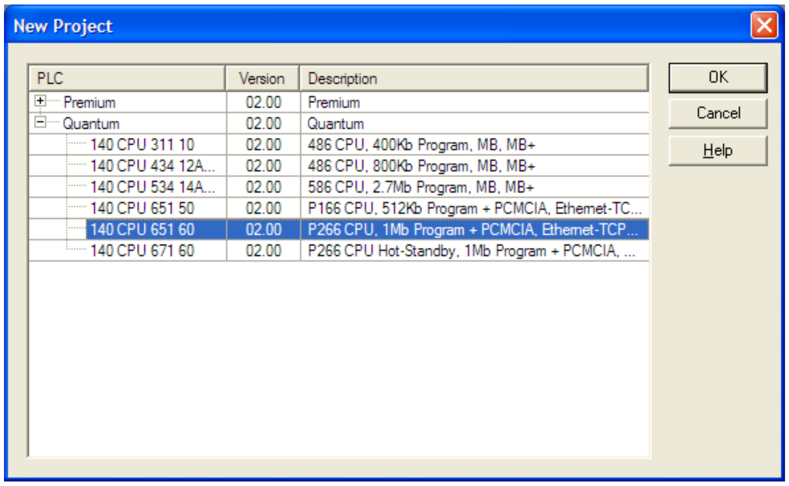
In This Chapter

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

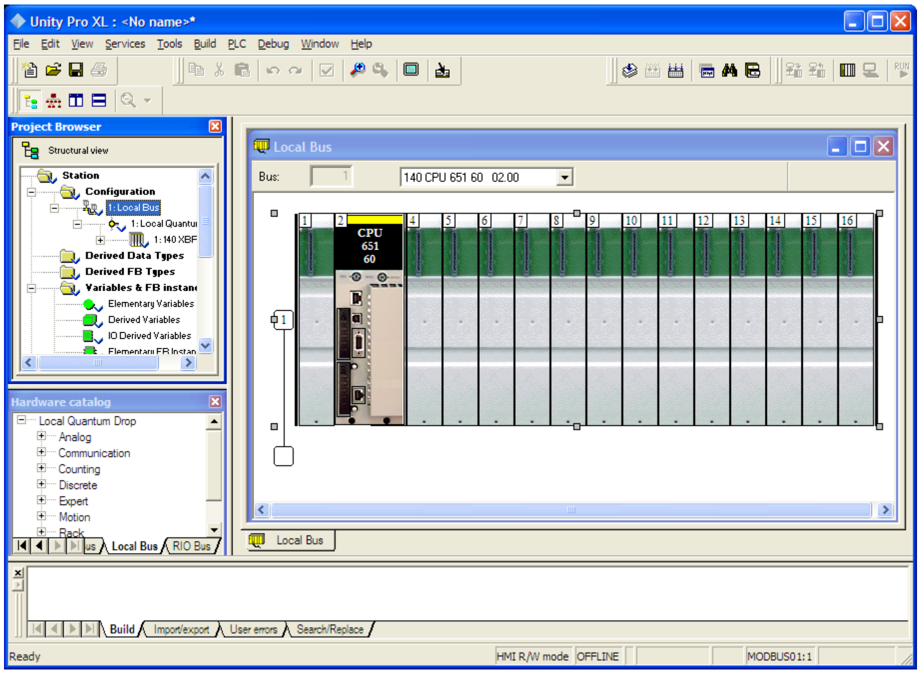
The first step is to open UnityPro XL and create a new project.

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.

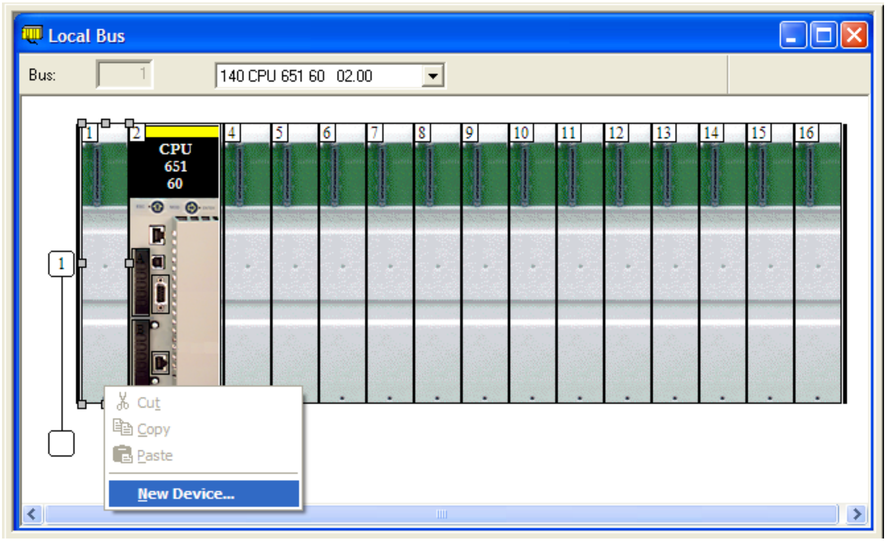


PLC	Version	Description
Premium	02.00	Premium
Quantum	02.00	Quantum
140 CPU 311 10	02.00	486 CPU, 400Kb Program, MB, MB+
140 CPU 434 12A...	02.00	486 CPU, 800Kb Program, MB, MB+
140 CPU 534 14A...	02.00	586 CPU, 2.7Mb Program, MB, MB+
140 CPU 651 50	02.00	P166 CPU, 512Kb Program + PCMCIA, Ethernet-TC...
140 CPU 651 60	02.00	P266 CPU, 1Mb Program + PCMCIA, Ethernet-TCP...
140 CPU 671 60	02.00	P266 CPU Hot-Standby, 1Mb Program + PCMCIA, ...

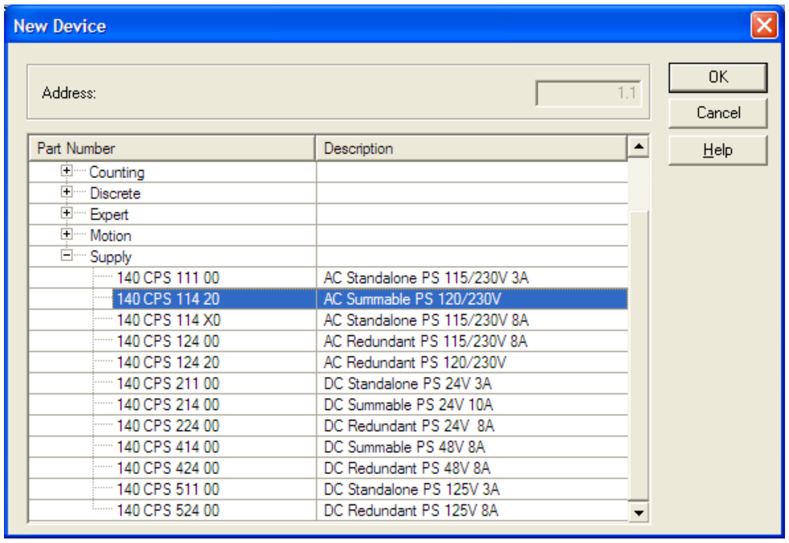
2 Next, add a power supply to the project. In the **PROJECT BROWSER**, expand the **CONFIGURATION** folder, and then double-click the **1:LOCALBUS** icon. This action opens a graphical window showing the arrangement of devices in your Quantum rack.



3 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**.



4 Expand the **SUPPLY** folder, and then select your power supply from the list. Click **OK** to continue.



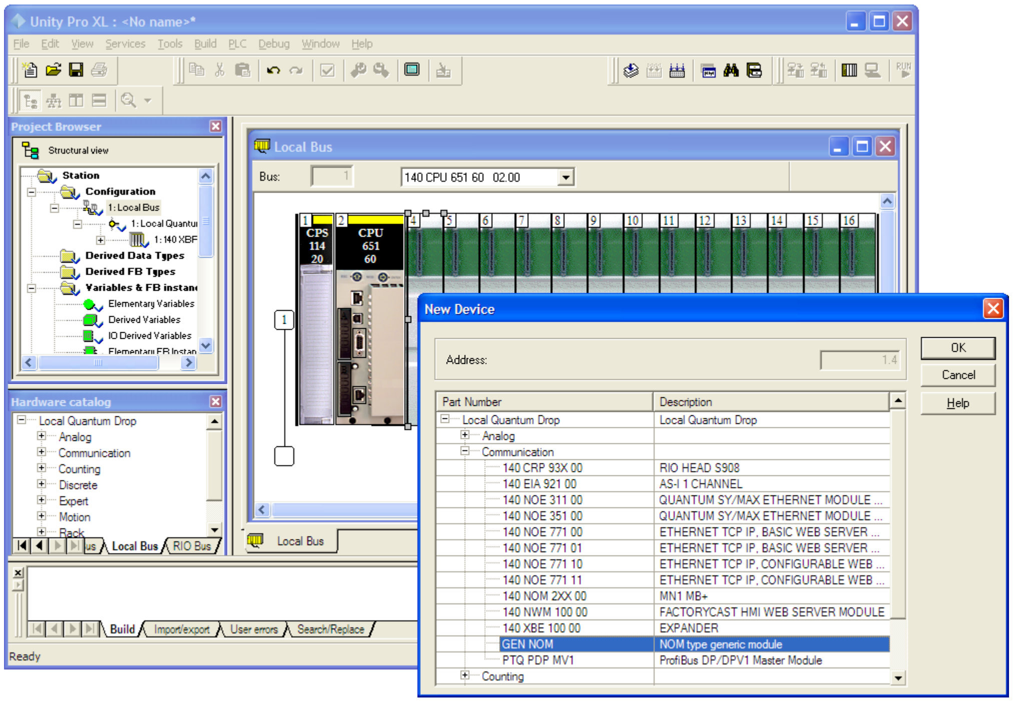
Address: 1.1

Part Number	Description
Counting	
Discrete	
Expert	
Motion	
Supply	
140 CPS 111 00	AC Standalone PS 115/230V 3A
140 CPS 114 20	AC Summable PS 120/230V
140 CPS 114 X0	AC Standalone PS 115/230V 8A
140 CPS 124 00	AC Redundant PS 115/230V 8A
140 CPS 124 20	AC Redundant PS 120/230V
140 CPS 211 00	DC Standalone PS 24V 3A
140 CPS 214 00	DC Summable PS 24V 10A
140 CPS 224 00	DC Redundant PS 24V 8A
140 CPS 414 00	DC Summable PS 48V 8A
140 CPS 424 00	DC Redundant PS 48V 8A
140 CPS 511 00	DC Standalone PS 125V 3A
140 CPS 524 00	DC Redundant PS 125V 8A

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

4.2 Add the PTQ Module to the Project

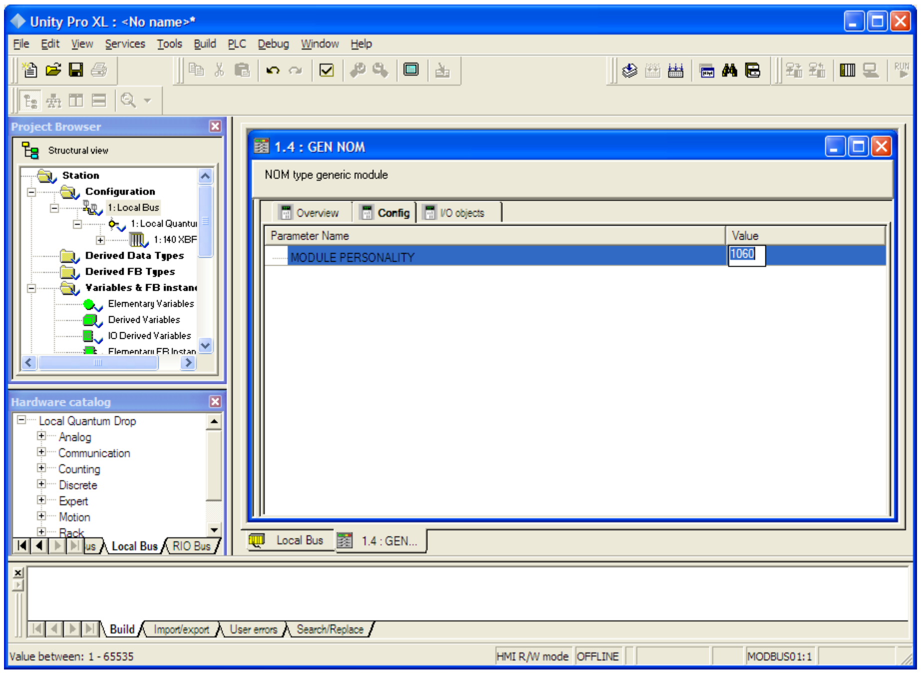
1 Expand the **COMMUNICATION** tree, and select **GEN NOM**. This module type provides extended communication capabilities for the Quantum system, and allows communication between the PLC and the PTQ module without requiring additional programming.



Address: 1.4

Part Number	Description
Local Quantum Drop	Local Quantum Drop
Analog	
Communication	
140 CRP 93X 00	RIO HEAD S908
140 EIA 921 00	AS-I 1 CHANNEL
140 NOE 311 00	QUANTUM SY/MAX ETHERNET MODULE ...
140 NOE 351 00	QUANTUM SY/MAX ETHERNET MODULE ...
140 NOE 771 00	ETHERNET TCP IP, BASIC WEB SERVER ...
140 NOE 771 01	ETHERNET TCP IP, BASIC WEB SERVER ...
140 NOE 771 10	ETHERNET TCP IP, CONFIGURABLE WEB ...
140 NOE 771 11	ETHERNET TCP IP, CONFIGURABLE WEB ...
140 NOM 2XX 00	MN1 MB+
140 NWM 100 00	FACTORYCAST HMI WEB SERVER MODULE
140 XBE 100 00	EXPANDER
GEN NOM	NOM type generic module
PTQ PDP MV1	ProfiBus DP/DPV1 Master Module
Counting	

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



Parameter Name	Value
MODULE PERSONALITY	1060

Value between: 1 - 65535

3 Before you can save the project in UnityPro XL, you must validate the modifications. Open the **EDIT** menu, and then choose **VALIDATE.** If no errors are reported, you can save the project.

4 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.

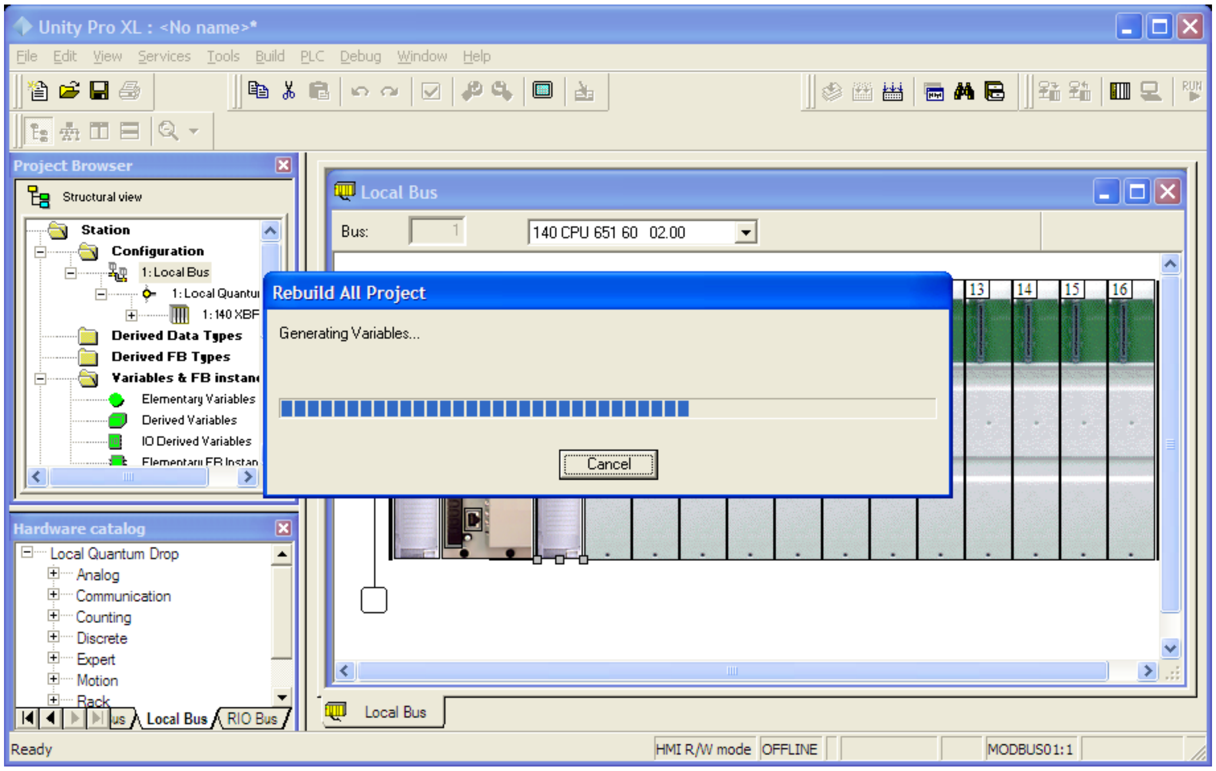
Note: The following steps show you how to build the project in Unity Pro XL. This is not intended to provide detailed information on using Unity Pro XL, or debugging your programs. Refer to the documentation for your processor and for Unity Pro XL for specialized information.

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.



Generating Variables...

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

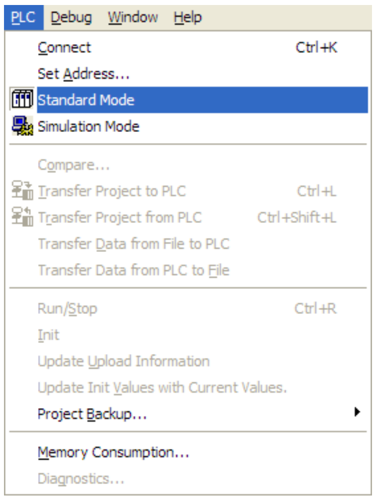
4.4 Connect Your PC to the Processor

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.

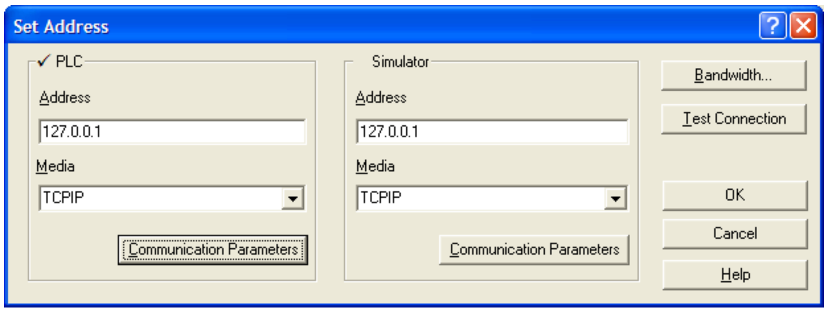
Note: If you have never connected from the PC to your processor before, you must verify that the necessary port drivers are installed and available to UnityPro XL.

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.


PLC
Debug
Window
Help
Connect Ctrl+K
Set Address...
Standard Mode
Simulation Mode
Compare...
Transfer Project to PLC Ctrl+L
Transfer Project from PLC Ctrl+Shift+L
Transfer Data from File to PLC
Transfer Data from PLC to File
Run/Stop Ctrl+R
Init
Update Upload Information
Update Init Values with Current Values.
Project Backup...
Memory Consumption...
Diagnostics...

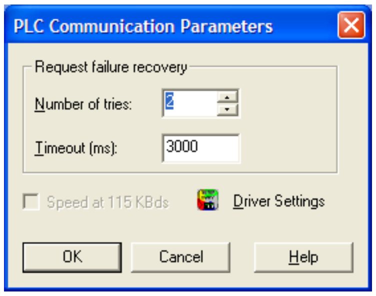
2 Open the **PLC** menu, and choose **SET ADDRESS...** This action opens the **SET ADDRESS** dialog box. Open the **MEDIA** dropdown list and choose the connection type to use (TCPIP or USB).



✓ PLC
Address: 127.0.0.1
Media: TCPIP

Simulator
Address: 127.0.0.1
Media: TCPIP

3 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.

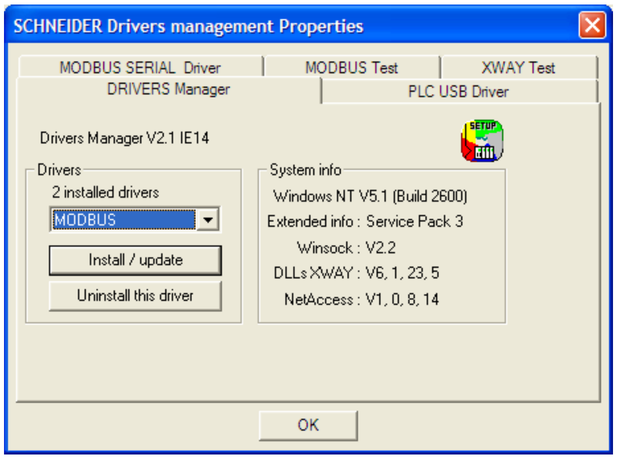

PLC Communication Parameters

Request failure recovery
Number of tries: 2
Timeout (ms): 3000

Speed at 115 KBds
Driver Settings

OK Cancel Help

4 Click the **DRIVER SETTINGS** button to open the **SCHNEIDER DRIVERS MANAGEMENT PROPERTIES** dialog box.


SCHNEIDER Drivers management Properties

MODBUS SERIAL Driver | MODBUS Test | XWAY Test
DRIVERS Manager | PLC USB Driver

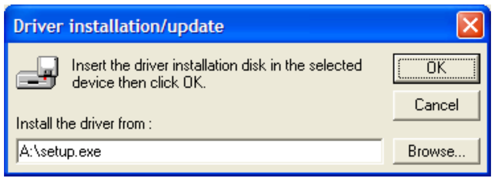
Drivers Manager V2.1 IE14

Drivers
2 installed drivers
MODBUS
Install / update
Uninstall this driver

System info
Windows NT V5.1 (Build 2600)
Extended info : Service Pack 3
Winsock : V2.2
DLLs XWAY : V6, 1, 23, 5
NetAccess : V1, 0, 8, 14

OK

5 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.


Driver installation/update

Insert the driver installation disk in the selected device then click OK.

Install the driver from :
A:\setup.exe

OK
Cancel
Browse...

6 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.

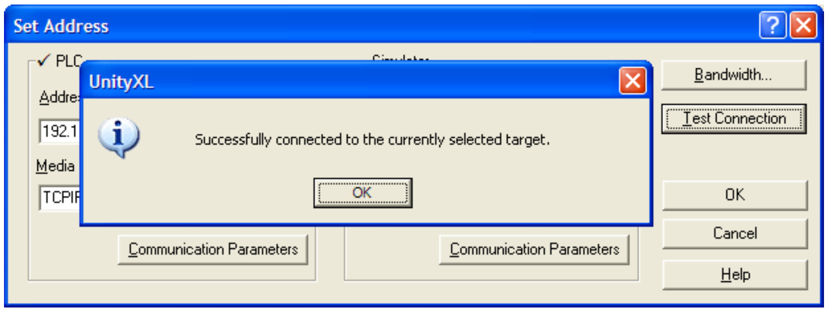
4.4.1 Connecting to the Processor with TCPIP

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.

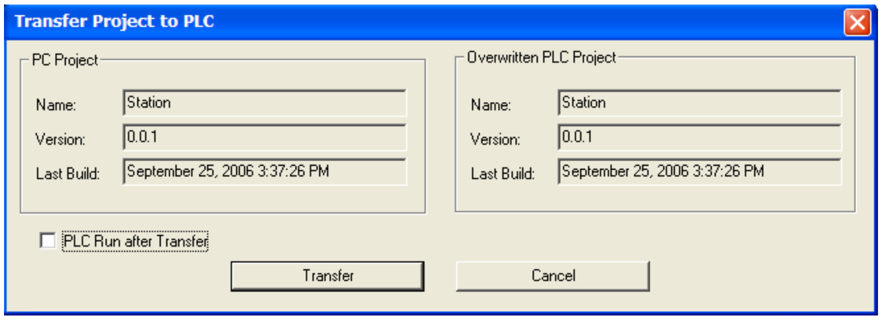
3 Enter the IP address in the address field. In the **MEDIA** dropdown list, choose TCPIP.
4 Click the **TEST CONNECTION** button to verify that your settings are correct.



Successfully connected to the currently selected target.

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** check box.



PC Project
Name: Station
Version: 0.0.1
Last Build: September 25, 2006 3:37:26 PM

Overwritten PLC Project
Name: Station
Version: 0.0.1
Last Build: September 25, 2006 3:37:26 PM

☐ PLC Run after Transfer

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

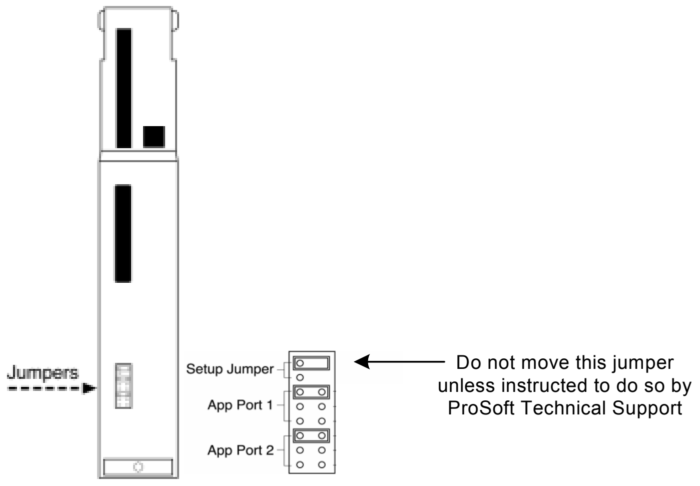
After you complete the following procedures, the ProTalk module will actively be transferring data bi-directionally with the processor.

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.

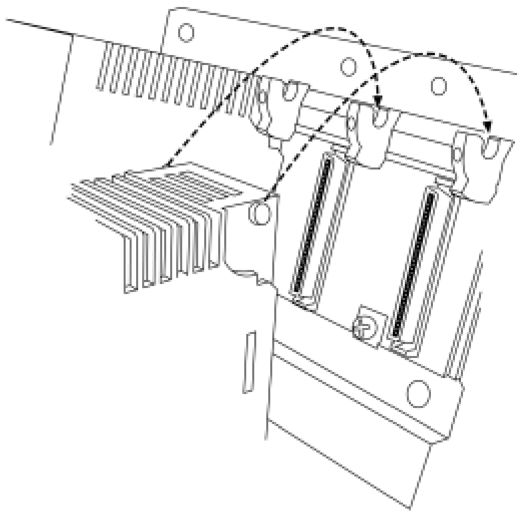
The jumpers are located on the back of the module as shown in the following illustration:


Jumpers
Setup Jumper
App Port 1
App Port 2
Do not move this jumper unless instructed to do so by ProSoft Technical Support

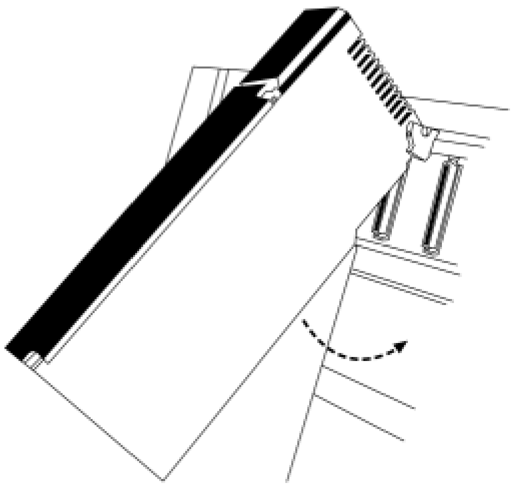
5.1.2 Install the ProTalk Module in the Quantum Rack

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.

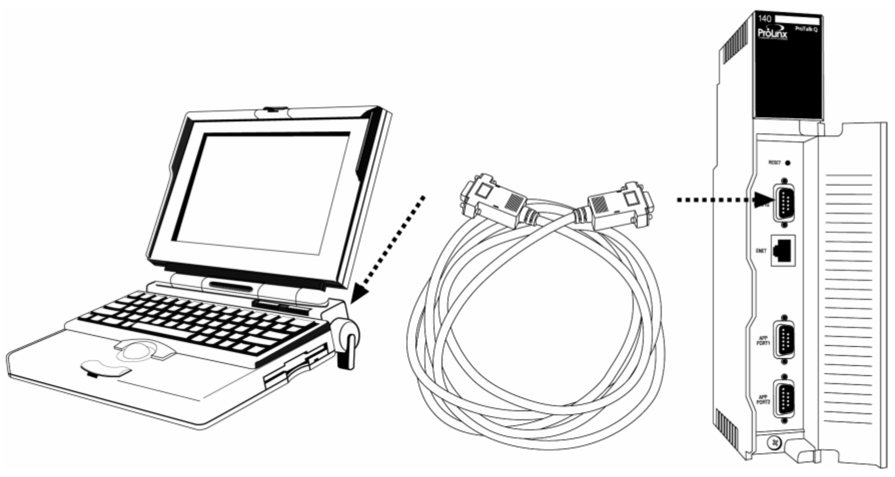


Caution: The PTQ module is hot-swappable, meaning that you can install and remove it while the rack is powered up. You should not assume that this is the case for all types of modules unless the user manual for the product explicitly states that the module is hot-swappable. Failure to observe this precaution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

5.2 Connect the PC to the ProTalk Configuration/Debug Port

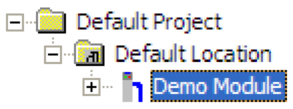
Make sure you have exited the Quantum programming software before performing these steps. This action will avoid serial port conflict.

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

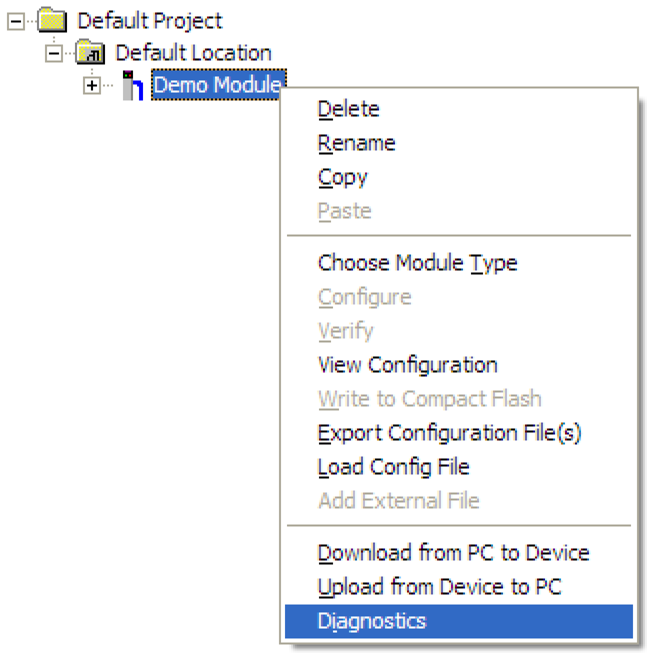


To connect to the module's Configuration/Debug serial port,

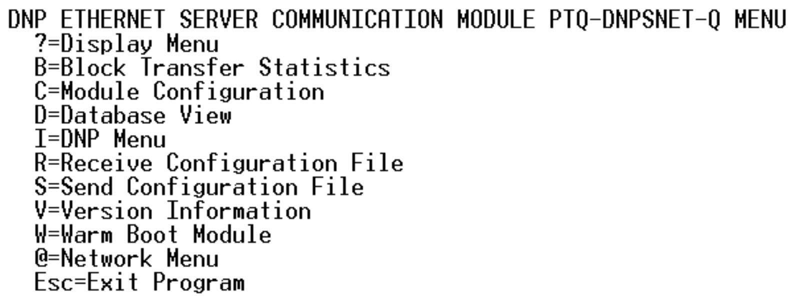
1 Start PCB, and then select the module to test. Click the right mouse button to open a shortcut menu.



2 On the shortcut menu, choose **DIAGNOSTICS.**



3 This action opens the **DIAGNOSTICS** dialog box. Press **[?]** to open the Main Menu.



```
DNP ETHERNET SERVER COMMUNICATION MODULE PTQ-DNPSNET-Q MENU
  ?=Display Menu
  B=Block Transfer Statistics
  C=Module Configuration
  D=Database View
  I=DNP Menu
  R=Receive Configuration File
  S=Send Configuration File
  V=Version Information
  W=Warm Boot Module
  @=Network Menu
  Esc=Exit Program
```

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.

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 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.
3 If you are still not able to establish a connection, contact ProSoft Technology for assistance.

6 Modifying the Configuration File

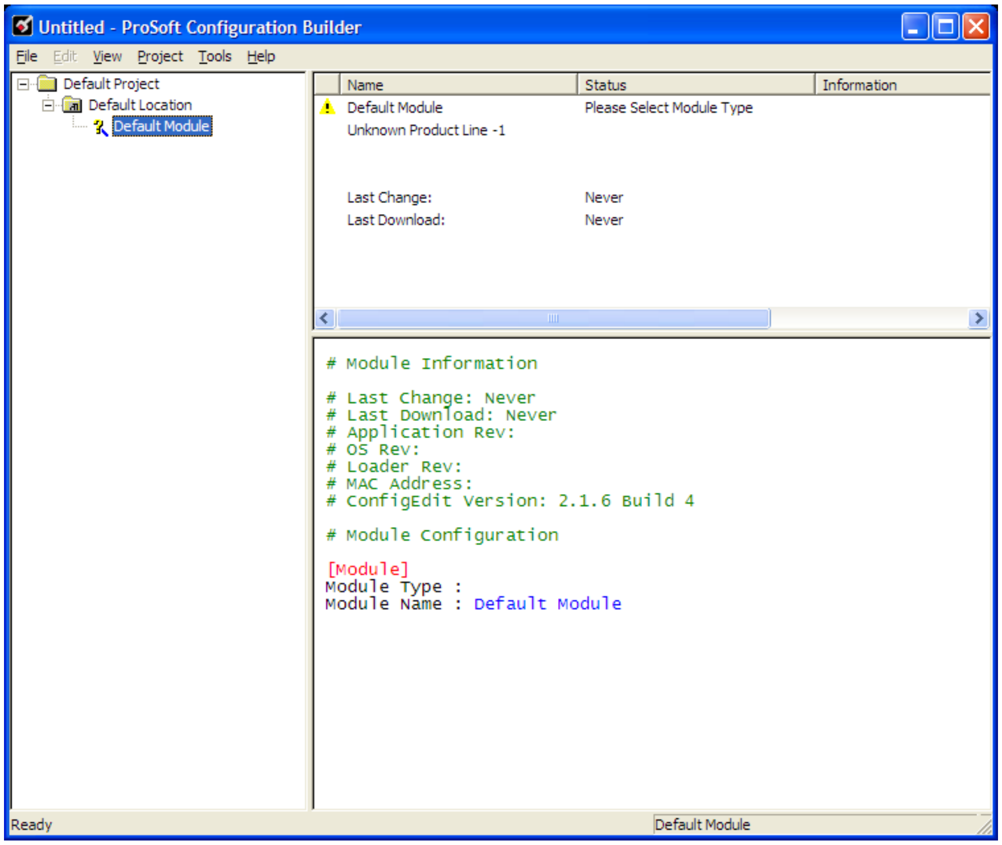
In This Chapter

6.1 ProSoft Configuration Builder

ProSoft Configuration Builder (PCB) provides a quick and easy way to manage module configuration files customized to meet your application needs. *PCB* is not only a powerful solution for new configuration files, but also allows you to import information from previously installed (known working) configurations to new projects.

6.1.1 Set Up the Project

To begin, start ProSoft Configuration Builder. If you have used other Windows configuration tools before, you will find the screen layout familiar. ProSoft Configuration Builder's window consists of a tree view on the left, an information pane and a configuration pane on the right side of the window. When you first start ProSoft Configuration Builder, the tree view consists of folders for Default Project and Default Location, with a Default Module in the Default Location folder. The following illustration shows the ProSoft Configuration Builder window with a new project.


Untitled - ProSoft Configuration Builder
File Edit View Project Tools Help
Default Project
Default Location
Default Module
Ready
Default Module

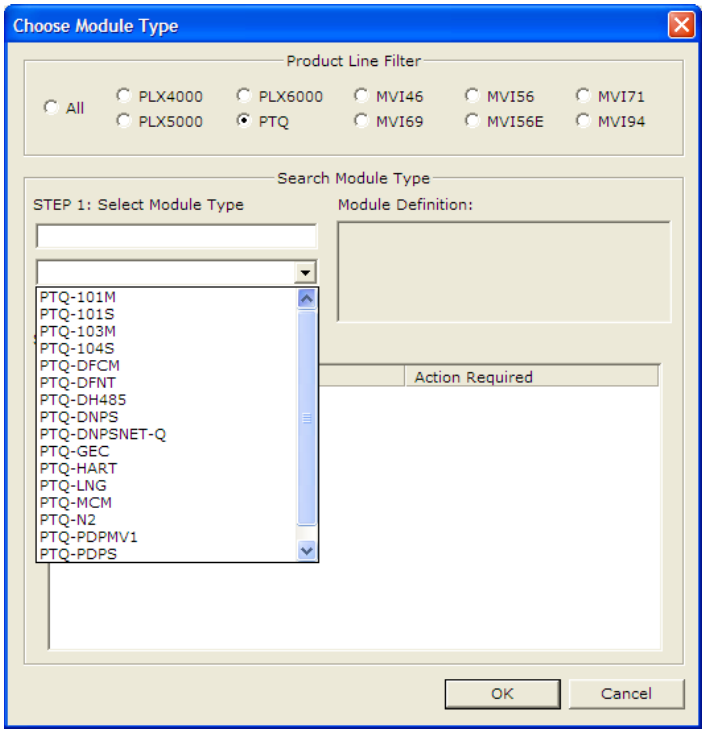
Name	Status	Information
Default Module	Please Select Module Type	
Unknown Product Line -1		
Last Change:	Never	
Last Download:	Never	

```
# Module Information

# Last Change: Never
# Last Download: Never
# Application Rev:
# OS Rev:
# Loader Rev:
# MAC Address:
# ConfigEdit Version: 2.1.6 Build 4

# Module Configuration

[Module]
Module Type :
Module Name : Default Module
```

Your first task is to add the PTQ-DNPSNET-Q module to the project.

1 Use the mouse to select **DEFAULT MODULE** in the tree view, and then click the right mouse button to open a shortcut menu.

2 On the shortcut menu, choose **CHOOSE MODULE TYPE**. This action opens the **CHOOSE MODULE TYPE** dialog box.



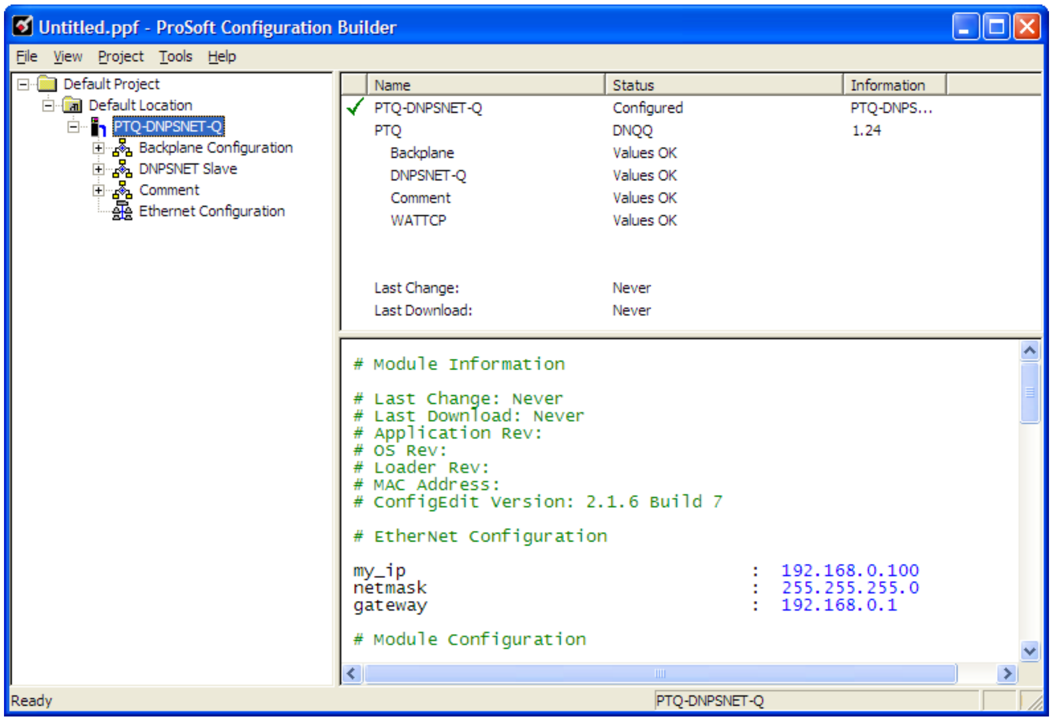
PTQ-101M
PTQ-101S
PTQ-103M
PTQ-104S
PTQ-DFCM
PTQ-DFNT
PTQ-DH485
PTQ-DNPS
PTQ-DNPSNET-Q
PTQ-GEC
PTQ-HART
PTQ-LNG
PTQ-MCM
PTQ-N2
PTQ-PDPMV1
PTQ-PDPS

3 In the **PRODUCT LINE FILTER** area of the dialog box, select **PTQ.** In the **SELECT MODULE TYPE** dropdown list, select PTQ-DNPSNET-Q, and then click **OK** to save your settings and return to the **PROSOFT CONFIGURATION BUILDER** window.

The next task is to set the module parameters.

6.1.2 Set Module Parameters

Notice that the contents of the information pane and the configuration pane changed when you added the PTQ-DNPSNET-Q module to the project.



Name	Status	Information
PTQ-DNPSNET-Q	Configured	PTQ-DNPS...
PTQ	DNQQ	1.24
Backplane	Values OK	
DNPSNET-Q	Values OK	
Comment	Values OK	
WATTCP	Values OK	
Last Change:	Never	
Last Download:	Never	

```
# Module Information

# Last Change: Never
# Last Download: Never
# Application Rev:
# OS Rev:
# Loader Rev:
# MAC Address:
# ConfigEdit Version: 2.1.6 Build 7

# EtherNet Configuration

my_ip                                      :  192.168.0.100
netmask                                    :  255.255.255.0
gateway                                    :  192.168.0.1

# Module Configuration
```

At this time, you may wish to rename the "Default Project" and "Default Location" folders in the tree view.

To rename an object:

1 Select the object, and then click the right mouse button to open a shortcut menu. From the shortcut menu, choose **RENAME.**
2 Type the name to assign to the object.
3 Click away from the object to save the new name.

Module Entries

To configure module parameters

1 Click on the plus sign next to the icon to expand module information.
2 Double-click the icon to open the **EDIT** dialog box.
3 To edit a parameter, select the parameter in the left pane and make your changes in the right pane.
4 Click **OK** to save your changes.

Printing a Configuration File

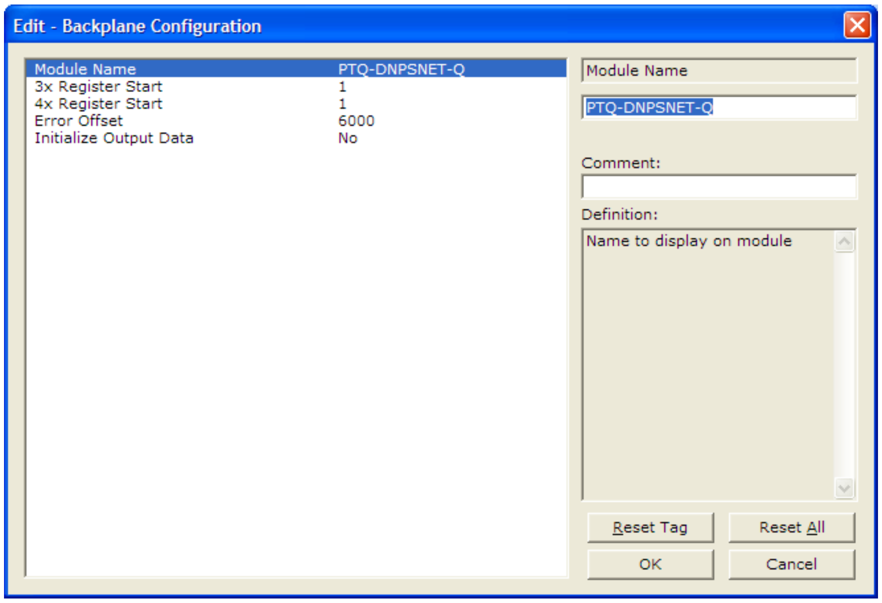
To print a configuration file:

1 Select the **MODULE** icon, and then click the right mouse button to open a shortcut menu.
2 On the shortcut menu, choose **VIEW CONFIGURATION.** This action opens the **VIEW CONFIGURATION** window.
3 On the **VIEW CONFIGURATION** window, open the **FILE** menu, and choose **PRINT.** This action opens the **PRINT** dialog box.
4 On the **PRINT** dialog box, choose the printer to use from the dropdown list, select printing options, and then click **OK.**

6.2 [Backplane Configuration]

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 illustration shows a sample [Backplane Configuration] section:


Edit - Backplane Configuration

Module Name	PTQ-DNPSNET-Q
3x Register Start	1
4x Register Start	1
Error Offset	6000
Initialize Output Data	No

Module Name
PTQ-DNPSNET-Q
Comment:
Definition:
Name to display on module

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

6.2.1 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 3x Register Start

1 to n

The 3x Register Start parameter defines the starting address in the processor's 3x (Quantum) or %iw (Unity) memory area to use for data being moved from the module. Take care to use a starting address that will accommodate the entire block from the module, but that will not overwrite data that is used for other purposes.

6.2.3 4x Register Start

1 to n

The 4x Register Start parameter defines the starting address in the processor's 4x (Quantum) or %iw (Unity) memory area to use for data being moved from the processor to the module. Take care to use a starting address that does not contain data in the processor's registers that is used for other purposes.

6.2.4 Error Offset

0 to 8966, -1 to disable

The Error Offset parameter specifies the register location in the module’s database where module status data will be stored. If a value less than 0 is entered, the data will not be stored in the database. If the value specified is in the range of 0 to 8966, the data will be placed in the modules database. A value of -1 = disable.

6.2.5 Initialize Output Data

Yes or No

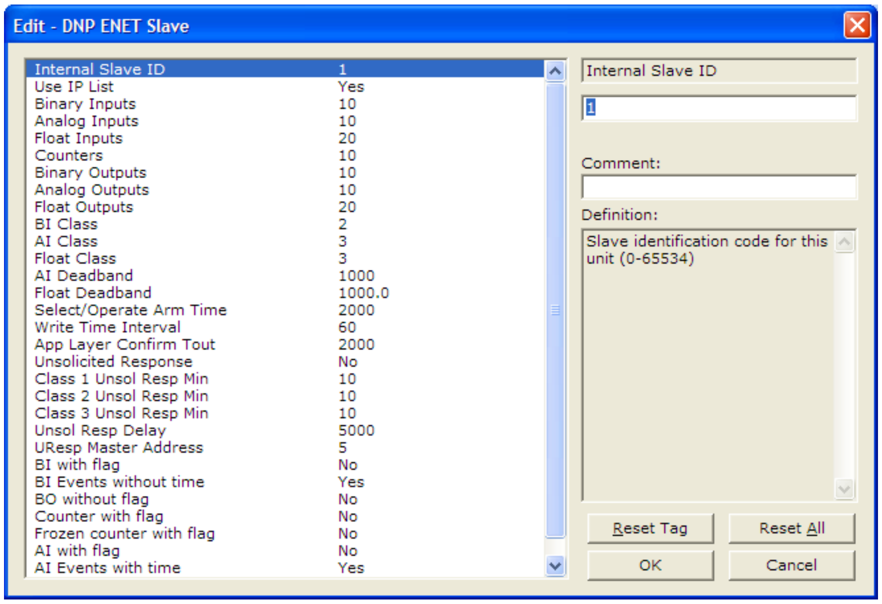
This parameter determines if the output data for the module should be initialized with values from the processor. If the value is set to No (0), the output data will be initialized to 0. If the value is set to Yes (1), the data will be initialized with data from the processor. Use of this option requires associated ladder logic to pass the data from the processor to the module.

6.3 [DNP ENET 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.

Note: A limitation of the DNP slave driver is that all points defined in the module slave database must fit within one Class 0 poll. The maximum packet size for a Class 0 poll is 2048 bytes. A DNP Message Size Calculator is available on the ProSoft Technology web site. This calculator will help you ensure that the packet size fits within this requirement.

The following illustration shows a sample [DNP ENET Slave] section:



Internal Slave ID	1
Use IP List	Yes
Binary Inputs	10
Analog Inputs	10
Float Inputs	20
Counters	10
Binary Outputs	10
Analog Outputs	10
Float Outputs	20
BI Class	2
AI Class	3
Float Class	3
AI Deadband	1000
Float Deadband	1000.0
Select/Operate Arm Time	2000
Write Time Interval	60
App Layer Confirm Tout	2000
Unsolicited Response	No
Class 1 Unsol Resp Min	10
Class 2 Unsol Resp Min	10
Class 3 Unsol Resp Min	10
Unsol Resp Delay	5000
UResp Master Address	5
BI with flag	No
BI Events without time	Yes
BO without flag	No
Counter with flag	No
Frozen counter with flag	No
AI with flag	No
AI Events with time	Yes

1

Slave identification code for this unit (0-65534)

Modify each parameter based on the needs of your application:

6.3.1 *Internal Slave ID*

0 to 65534

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

6.3.2 *Use IP List*

Y or N

This parameter specifies if the IP address of the host connected to the system will be validated. If the parameter is set to N, any host may connect to the unit. If the parameter is set to Y, only hosts in the IP list will be permitted to connect to the module. All other IP addresses will be ignored by the module and the module will issue a RST to the TCP/IP connection.

DNP Database Definition Note: The databases are in the memory of the module in this sequence and are placed directly adjacent to each other. In other words when you change the size of a database you must adjust the transfer commands to accommodate the new location.

6.3.3 Binary Inputs

0 to 500 points

This parameter specifies the number of digital input points to configure in the DNP slave device based on a word count. The valid range is 0 to 500 words.

6.3.4 Analog Inputs

0 to 500 points

This parameter sets the number of analog input points to configure in the DNP slave device. Each point will occupy a one-word area in the module memory. Valid values are 0 to 512 points.

6.3.5 Float Inputs

0 to 128 points

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.

6.3.6 Counters

0 to 128 points

This parameter sets the 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. Valid values are 0 to 128 points. This example show the parameter set to 10 points of counter data.

6.3.7 Binary Outputs

0 to 512 points

This parameter sets the number of digital output words to configure in the DNP slave device based on a word count. Each word stores 16 points. Therefore, if the parameter is set to 2, 32 binary outputs will be defined for the application. Valid values are 0 to 512 words to hold Binary Output data. This example shows the parameter set to 100 words.

6.3.8 Analog Outputs

0 to 512 points

This parameter sets the number of analog output points to configure in the DNP slave device. Each point will occupy a one-word area in the module memory. Valid values are 0 to 512 points of analog output data.

6.3.9 Float Outputs

0 to 128 points

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.

6.3.10 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.

6.3.11 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.

6.3.12 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.

6.3.13 AI Deadband

0 to 32767 data units

This value sets the global deadband for all analog input points. When the current value for an analog input point is not within the deadband limit set based on the last event for the point, an event will be generated.

6.3.14 Float Deadband

0 to 32767 data units

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.

6.3.15 Select/Operate Arm Time

1 to 65535 milliseconds

This parameter sets the 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. Valid arm timeout values are 1 to 65535 milliseconds. This example shows the value set to 2000 milliseconds.

6.3.16 Write Time Interval

0 to 1440 minutes

This parameter sets the time interval to set the need time IIN bit (0=never), which will cause the master to write the time. Stored in milliseconds in the module memory.

6.3.17 App Layer Confirm Tout

1 to 65535 milliseconds

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.

6.3.18 Unsolicited Response

Yes or No

This parameter is set if the slave unit will send unsolicited response messages. If set to N, the slave will not send unsolicited responses. If set to Y, the slave will send unsolicited responses.

6.3.19 Class 1 Unsol Resp Min

1 to 255 events

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

6.3.20 Class 2 Unsol Resp Min

1 to 255 events

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

6.3.21 Class 3 Unsol Resp Min

1 to 255 events

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

6.3.22 Unsol Resp Delay

0 to 65535 milliseconds

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.

6.3.23 Uresp Master Address

0 to 65534

DNP destination address where unsolicited response messages are sent.

6.3.24 BI with Flag

Yes or No

This parameter determines which variation will be returned for object 1 when the master requests variation 0. If the parameter is set to N, variation 1 will be returned. If the parameter is set to Y, variation 2 will be returned.

Note: Flag will always be set for Online and cannot be changed through by the PLC or user program. Only the default variation returned by the module will be affected by changing this parameter.

6.3.25 BI Events Without Time

Yes or No

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

6.3.26 BO Without Flag

Yes or No

This parameter determines which variation will be returned for object 10 when the master requests variation 0. If the parameter is set to N, variation 2 will be returned. If the parameter is set to Y, variation 1 will be returned.

6.3.27 Counter with Flag

Yes or No

This parameter determines which variation will be returned for object 20 when the master requests variation 0. If the parameter is set to N, variation 5 will be returned. If the parameter is set to Y, variation 1 will be returned.

Note: Flag will always be set for Online and cannot be changed through by the PLC or user program. Only the default variation returned by the module will be affected by changing this parameter.

6.3.28 Frozen Counter with Flag

Yes or No

This parameter determines which variation will be returned for object 21 when the master requests variation 0. If the parameter is set to N, variation 9 will be returned. If the parameter is set to Y, variation 1 will be returned.

Note: Flag will always be set for Online and cannot be changed through by the PLC or user program. Only the default variation returned by the module will be affected by changing this parameter.

6.3.29 AI with Flag

Yes or No

This parameter determines which variation will be returned for object 30 when the master requests variation 0. If the parameter is set to N, variation 4 will be returned. If the parameter is set to Y, variation 2 will be returned.

Note: Flag will always be set for Online and cannot be changed through by the PLC or user program. Only the default variation returned by the module will be affected by changing this parameter.

6.3.30 AI Events with Time

Yes or No

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

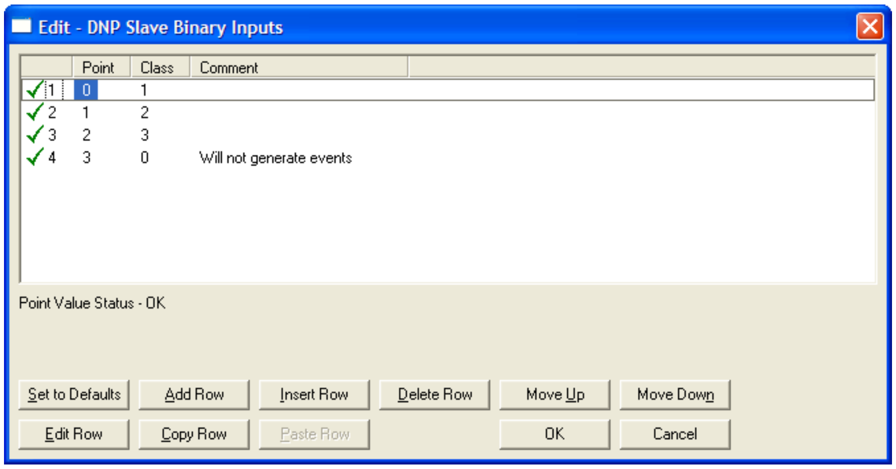
6.3.31 Time Sync Before Events

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 N, events will be generated irrespective of the module’s time sync status. If the parameter is set to Y, events will be generated only if the module’s time is synchronized.

6.4 [DNP Slave Binary Inputs]

This section of the configuration file overrides the Class 2 binary database points. Enter the list of points in this table.

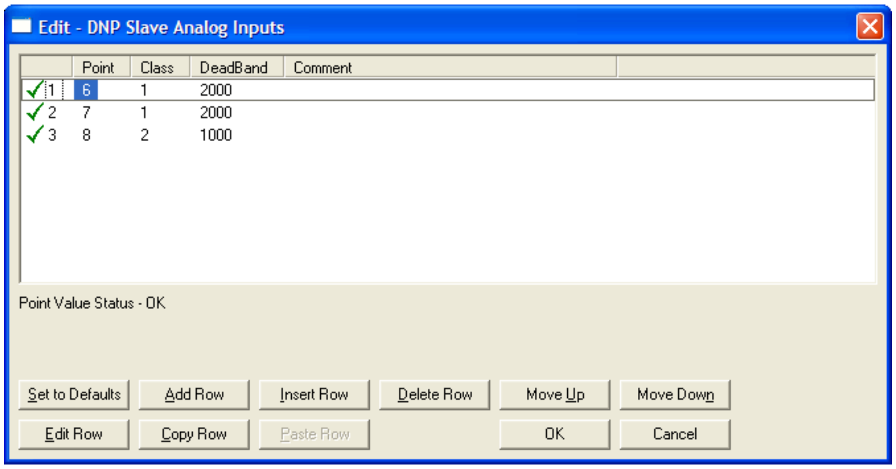


	Point	Class	Comment
✓ 1	0	1	
✓ 2	1	2	
✓ 3	2	3	
✓ 4	3	0	Will not generate events

Point Value Status - OK

6.5 [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.

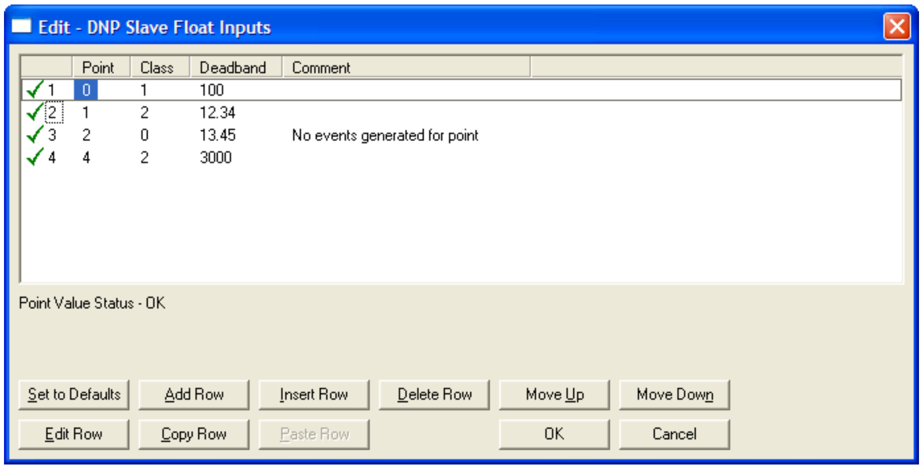


	Point	Class	DeadBand	Comment
✓ 1	6	1	2000	
✓ 2	7	1	2000	
✓ 3	8	2	1000	

Point Value Status - OK

6.6 [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.

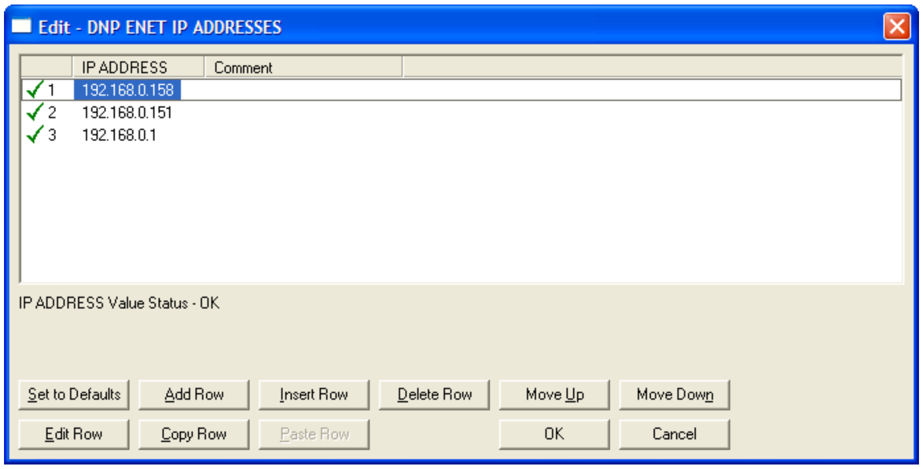


	Point	Class	Deadband	Comment
✓ 1	0	1	100	
✓ 2	1	2	12.34	
✓ 3	2	0	13.45	No events generated for point
✓ 4	4	2	3000	

Point Value Status - OK

6.7 [DNP ENET IP ADDRESSES]

This section of the configuration file only applies if the directive labeled **Use IP List** is set to Yes or Y. If **Use IP List** is enabled, the module will refuse to answer a request unless the IP address of the client is listed in this section. This section may contain no more then 10 addresses.



	IP ADDRESS	Comment
✓ 1	192.168.0.158	
✓ 2	192.168.0.151	
✓ 3	192.168.0.1	

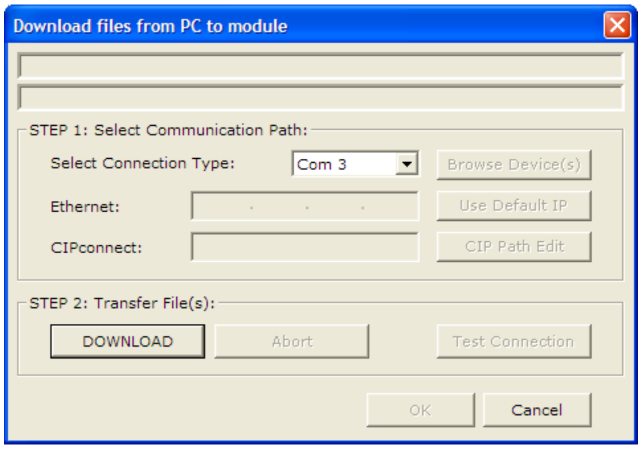
IP ADDRESS Value Status - OK

6.8 Download the Project to the Module

In order for the module to use the settings you configured, you must download (copy) the updated Project file from your PC to the module.

To Download the Project File

1 In the tree view in ProSoft Configuration Builder, click once to select the PTQ-DNPSNET-Q module.
2 Open the **PROJECT** menu, and then choose **MODULE / DOWNLOAD.** The program will scan your PC for a valid com port (this may take a few seconds). When PCB has found a valid com port, the **DOWNLOAD** dialog box will open.



Download files from PC to module

STEP 1: Select Communication Path:

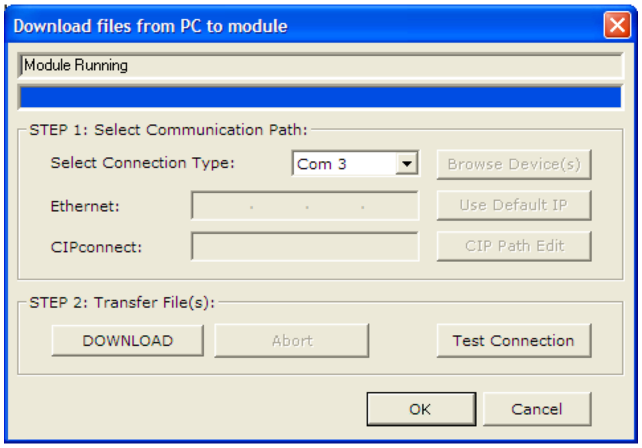
Select Connection Type: Com 3

Ethernet:

CIPconnect:

3 Choose the com port to use from the dropdown list, and then click the **DOWNLOAD** button.

The module will perform a platform check to read and load its new settings. When the platform check is complete, the status bar in the **DOWNLOAD** dialog box with the message *"Module Running*".



Download files from PC to module

Module Running

STEP 1: Select Communication Path:

Select Connection Type: Com 3

Ethernet:

CIPconnect:

7 Diagnostics and Troubleshooting

In This Chapter

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 Reading Status Data from the Module

The PTQ-DNPSNET-Q module returns a Status Data block that can be used to determine the module's operating status. This data is located in the module's database status database and error status list. This data is transferred to the Quantum / Unity processor read blocks with an identification code of 100. For a complete listing of the status data object, refer to the Installing and Configuring the Module section.

7.1.1 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.

7.1.2 The Configuration/Debug Menu

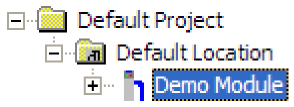
The Configuration and Debug menu for this module is arranged as a tree structure, with the Main Menu at the top of the tree, and one or more sub-menus for each menu command. The first menu you see when you connect to the module is the Main menu.

Because this is a text-based menu system, you enter commands by typing the command letter from your computer keyboard in the diagnostic window in ProSoft Configuration Builder (PCB). 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.

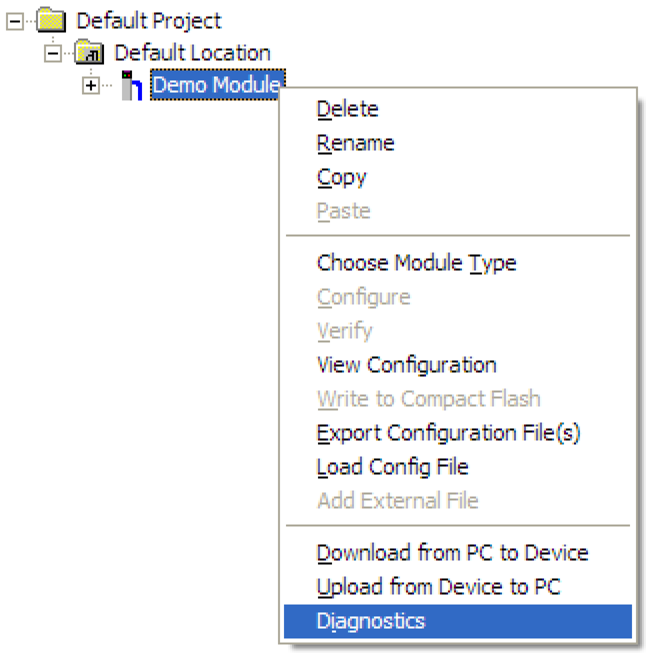
Using the Diagnostic Window in ProSoft Configuration Builder

To connect to the module's Configuration/Debug serial port,

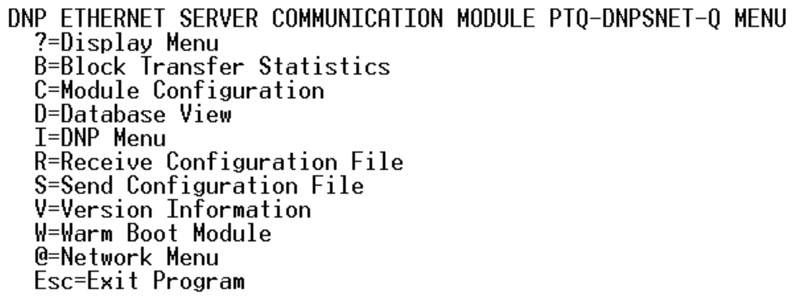
1 Start PCB, and then select the module to test. Click the right mouse button to open a shortcut menu.



2 On the shortcut menu, choose **DIAGNOSTICS.**



3 This action opens the **DIAGNOSTICS** dialog box. Press **[?]** to open the Main Menu.



```
DNP ETHERNET SERVER COMMUNICATION MODULE PTQ-DNPSNET-Q MENU
  ?=Display Menu
  B=Block Transfer Statistics
  C=Module Configuration
  D=Database View
  I=DNP Menu
  R=Receive Configuration File
  S=Send Configuration File
  V=Version Information
  W=Warm Boot Module
  @=Network Menu
  Esc=Exit Program
```

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.

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.

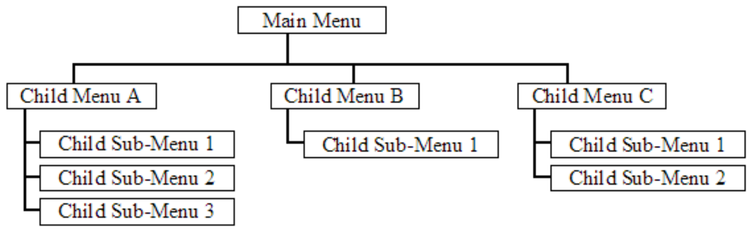
2 On computers with more than one serial port, verify that your communication program is connected to the same port that is connected to the module.

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

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.

Keystrokes

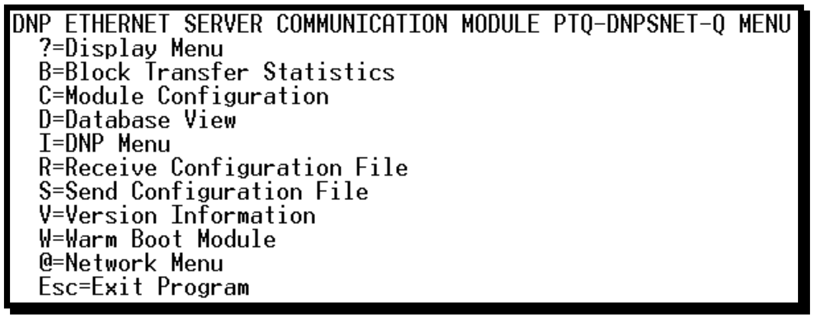
The keyboard commands on these menus are almost always non-case sensitive. You can enter most commands in lower case or capital letters.

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 **[L]** (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.

7.1.3 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:



```
DNP ETHERNET SERVER COMMUNICATION MODULE PTQ-DNPSNET-Q MENU
  ?=Display Menu
  B=Block Transfer Statistics
  C=Module Configuration
  D=Database View
  I=DNP Menu
  R=Receive Configuration File
  S=Send Configuration File
  V=Version Information
  W=Warm Boot Module
  @=Network Menu
  Esc=Exit 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.

Viewing Block Transfer Statistics

Press **[N]** 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: Repeat this command at one-second intervals to determine the number of blocks transferred each second.

Viewing Module Configuration

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

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

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.

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 153).

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 153).

Viewing Version Information

Press **[G]** 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 **[R]** 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.

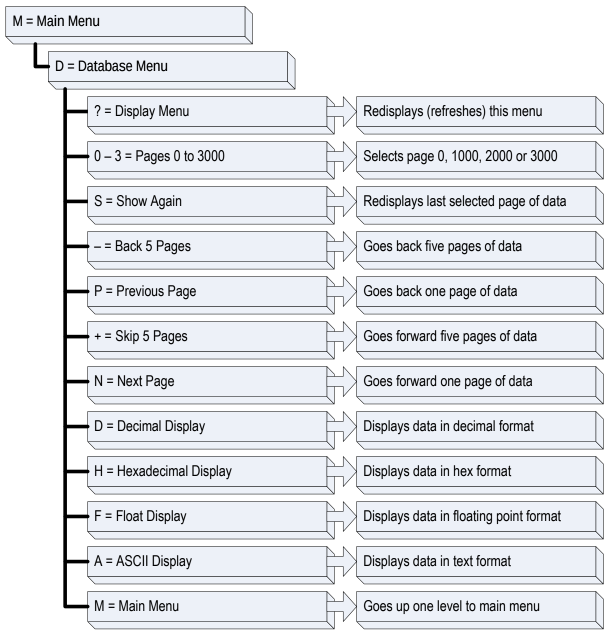
Exiting the Program

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 memory to configure the module.

7.1.4 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.


M = Main Menu
D = Database Menu
? = Display Menu
Redisplays (refreshes) this menu
0 – 3 = Pages 0 to 3000
Selects page 0, 1000, 2000 or 3000
S = Show Again
Redisplays last selected page of data
– = Back 5 Pages
Goes back five pages of data
P = Previous Page
Goes back one page of data
+ = Skip 5 Pages
Goes forward five pages of data
N = Next Page
Goes forward one page of data
D = Decimal Display
Displays data in decimal format
H = Hexadecimal Display
Displays data in hex format
F = Float Display
Displays data in floating point format
A = ASCII Display
Displays data in text format
M = Main Menu
Goes up one level to main menu

Viewing Register Pages

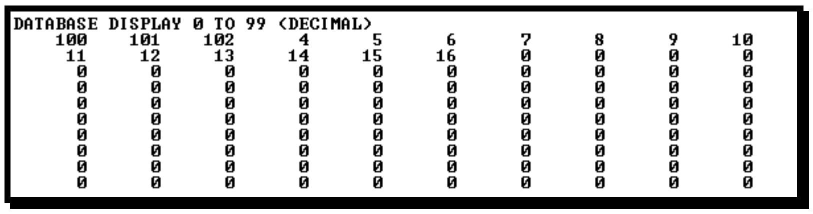
To view sets of register pages, use the keys described below:

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

Press **[S]** from the Database View menu to show the current page of registers again.



```
DATABASE DISPLAY 0 TO 99 (DECIMAL)
   100     101     102       4       5       6       7       8       9      10
    11      12      13      14      15      16       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       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 five pages back in the database to see the previous 100 registers of data.

Moving Forward Through 5 Pages of Registers

Press **[+]** from the Database View menu to skip five pages ahead in the database to see the next 100 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.

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

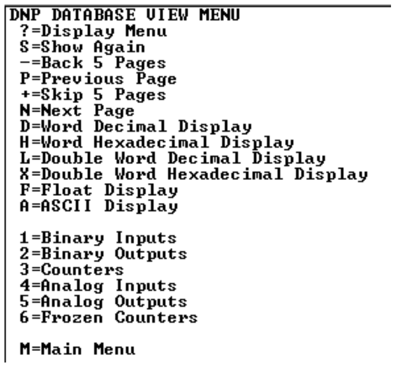
Press **[A]** to display the data on the current page in ASCII format. This is useful for regions of the database that contain ASCII data.

Returning to the Main Menu

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

7.1.5 DNP Database View Menu

Use this menu command to view the current contents of the selected database. Press **[?]** to view a list of commands available on this menu.

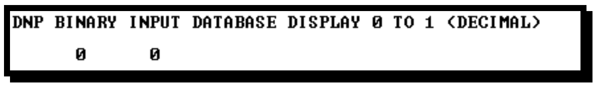


```
DNP DATABASE VIEW MENU
 ?=Display Menu
 S=Show Again
 -=Back 5 Pages
 P=Previous Page
 +=Skip 5 Pages
 N=Next Page
 D=Word Decimal Display
 H=Word Hexadecimal Display
 L=Double Word Decimal Display
 X=Double Word Hexadecimal Display
 F=Float Display
 A=ASCII Display

 1=Binary Inputs
 2=Binary Outputs
 3=Counters
 4=Analog Inputs
 5=Analog Outputs
 6=Frozen Counters

 M=Main Menu
```

Viewing Data Type Databases

Press **[D]** from the DNP menu, then hold down the **[Shift]** key and press the **[/]** key.

Use the number keys 1 to 6 to select the display of the data type you wish to view. For example, if the [1] key is pressed, the following is displayed:



```
DNP BINARY INPUT DATABASE DISPLAY 0 TO 1 (DECIMAL)
      0      0
```

Viewing Register Pages

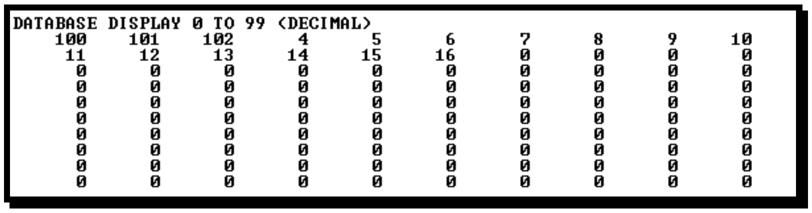
To view sets of register pages, use the keys described below:

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

Press **[S]** from the Database View menu to show the current page of registers again.



```
DATABASE DISPLAY 0 TO 99 (DECIMAL)
     100     101     102       4       5       6       7       8       9      10
      11      12      13      14      15      16       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       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 five pages back in the database to see the previous 100 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.

Moving Forward Through 5 Pages of Registers

Press **[+]** from the Database View menu to skip five pages ahead in the database to see the next 100 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.

Viewing Data in Double Word Decimal Format

Press **[L]** to display the data on the current page in Double Word Decimal format. This is useful for regions of the database that contain Double Word Decimal data.

Viewing Data in Double Word Decimal Format

Press **[X]** to display the data on the current page in Double Word Hexadecimal format. This is useful for regions of the database that contain Double Word Hexadecimal data.

Viewing DNP Binary Inputs

Press **[1]** to view a list of DNP Binary Inputs.

Viewing DNP Binary Outputs

Press **[2]** to view a list of DNP Binary Outputs.

Viewing DNP Counters

Press **[3]** to view a list of DNP Counters.

Viewing DNP Analog Inputs

Press **[4]** to view a list of DNP Analog Inputs.

Viewing DNP Analog Outputs

Press **[5]** to view a list of DNP Analog Outputs.

Viewing DNP Frozen Counters

Press **[6]** to view a list of DNP Frozen Counters.

Viewing DNP Float Inputs

Press **[7]** to view a list of DNP Float Inputs.

Viewing DNP Float Outputs

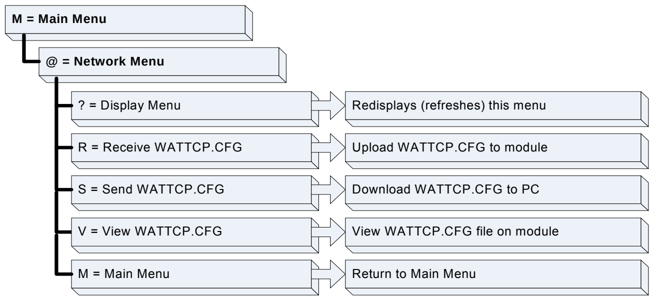
Press **[9]** to view a list of DNP Float Outputs.

Returning to the Main Menu

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

7.1.6 Network Menu

The network menu allows you to send, receive, and view the WATTCP.CFG file that contains the IP and gateway addresses, and other network information.



<u>Transferring WATTCP.CFG to the module</u>

Press **[R]** to transfer a new WATTCP.CFG file from the PC to the module. Use this command to change the network configuration for the module (for example, the module's IP address).

Press **[Y]** to confirm the file transfer, and then follow the instructions on the terminal screen to complete the file transfer process.

<u>Transferring WATTCP.CFG to the PC</u>

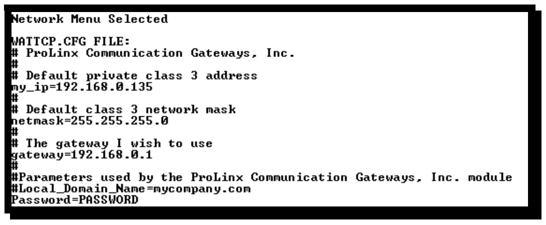
Press **[S]** to transfer the WATTCP.CFG file from the module to your PC.

Press **[Y]** to confirm the file transfer, and then follow the instructions on the terminal screen to complete the file transfer process.

After the file has been successfully transferred, you can open and edit the file to change the module's network configuration.

<u>Viewing the WATTCP.CFG file on the module</u>

Press **[V]** to view the module's WATTCP.CFG file. Use this command to confirm the module's current network settings.



```
Network Menu Selected

WATTCP.CFG FILE:
# ProLinx Communication Gateways, Inc.
#
# Default private class 3 address
my_ip=192.168.0.135
#
# Default class 3 network mask
netmask=255.255.255.0
#
# The gateway I wish to use
gateway=192.168.0.1
#
#Parameters used by the ProLinx Communication Gateways, Inc. module
#Local_Domain_Name=mycompany.com
Password=PASSWORD
```

<u>Returning to the Main Menu</u>

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

7.2 LED Status Indicators

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

Module	Color	Status	Indication
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.
E-Link	Green	On	The Ethernet port is connected to the TCP/IP network
		Off	No Connection
E-Data	Green	On	There is data being transferred through the Ethernet port.
		Off	No data transfer
BAT Low	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.
DEBUG	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.
CFG ERR		On	
		Off	
PRT1	Green	On	Port not used in application
		Off	Port not used in application
PRT2	Green	On	Port not used in application
		Off	Port not used in application
ERR1	Red	Off	The PTQ-DNPSNET-Q is working normally.
		On	The PTQ-DNPSNET-Q module program has recognized an application error. This LED will also be turned on if any command presents an error.
ERR2	N/A		Not used in application

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

7.2.1 Ethernet LED Indicators

LED	State	Description
Data	Off	No activity on the Ethernet port.
	Green Flash	The Ethernet port is actively transmitting or receiving data.
Link	Off	No physical network connection is detected. No Ethernet communication is possible. Check wiring and cables.
	Green Solid	Physical network connection detected. This LED must be on solid for Ethernet communication to be possible.

7.2.2 Clearing a Fault Condition

Typically, if the OK LED on the front of the module turns red for more than ten seconds, a hardware problem has been detected in the module, or the program has exited.

To clear the condition, follow these steps:

1 Turn off power to the rack
2 Remove the card from the rack
3 Verify that all jumpers are set correctly
4 If the module requires a Compact Flash card, verify that the card is installed correctly
5 Re-insert the card in the rack and turn the power back on
6 Verify the configuration data being transferred to the module from the Quantum / Unity processor.

If the module's OK LED does not turn green, verify that the module is inserted completely into the rack. If this does not cure the problem, contact ProSoft Technology Support.

7.2.3 Troubleshooting

Use the following troubleshooting steps if you encounter problems when the module is powered up. If these steps do not resolve your problem, please contact ProSoft Technology Technical Support.

Processor Errors

Problem Description	**Steps to take**
Processor Fault	Verify that the module is plugged into the slot that has been configured for the module. Verify that the slot location in the rack has been configured correctly in the ladder logic.
Processor I/O LED flashes	This indicates a problem with backplane communications. Verify that all modules in the rack are configured in the ladder logic.

Module Errors

Problem Description	**Steps to take**
BP ACT LED remains off or blinks slowly	This indicates that backplane transfer operations are failing. Connect to the module's Configuration/Debug port to check this.
MVI56E modules with scrolling LED display: <*Backplane Status*> condition reads ERR	To establish backplane communications, verify the following items: ▪ The processor is in Run mode. ▪ The backplane driver is loaded in the module. ▪ The module is configured for read and write block data transfer. ▪ The ladder logic handles all read and write block situations. ▪ The module is configured in the processor.
OK LED remains red	The program has halted or a critical error has occurred. Connect to the Configuration/Debug port to see if the module is running. If the program has halted, turn off power to the rack, remove the card from the rack and re-insert the card in the rack, and then restore power to the rack.

8 Reference

In This Chapter

8.1 Product Specifications

8.1.1 PTQ-DNPSNET-Q

The PTQ DNP 3.0 Server over Ethernet Communications Module supports the implementation of the DNP 3.0 (Distributed Network Protocol) over Ethernet, allowing Quantum / Unity processors to easily communicate with host systems supporting the protocol. The module supports DNP Subset Level 2 features and some Level 3 features.

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 programming tools: UnityPro XL, Concept, ProWORX
- Quantum data types supported: 3x, 4x
- High speed data transfer across backplane provides quick data update times
- Sample ladder file available

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 (PRT1)	DB-9M PC Compatible RS-232 only No hardware handshaking
Application Ethernet port	RJ45 Connector Link and Activity LED indicators Electrical Isolation 1500 V rms at 50 Hz to 60 Hz for 60 s, applied as specified in section 5.3.2 of IEC 60950: 1991 Ethernet Broadcast Storm Resiliency = less than or equal to 5000 [ARP] frames-per-second and less than or equal to 5 minutes duration

Functional Specifications

The PTQ-DNPSNET-Q module accepts data read/write commands from a master/client on the network. The module accepts DNP commands to control and monitor the data stored in the DNP databases. In addition, the module can be configured to generate unsolicited messages.

The module has 4000 data words of internal register space that are accessible to the protocol driver and to the Quantum processors memory. Memory usage is user configurable within these 4000 words. Each of the supported database types can be individually sized and each database point is mapped within the module. The supported database point types are:

- Binary Input 512 Points (512 words)
- Binary Output 512 Points (512 words)
- Counter 128 Points (128 words)
- Analog Input 512 Words
- Analog Output 512 Words

Total point counts must be configured such that Class 0 responses do not exceed 2048 bytes in size

DNP 3.0 Server over Ethernet Specifications

Operating in the Server mode, the module accepts commands from a Client(s) to read/write data stored in the module's internal registers. This data is easily and continuously transferred between the ProTalk module and the Quantum processor's data registers.

The Server functionality supported by the module includes:

- The DNP on Ethernet communication driver is built in accordance with the DNP organizations WAN/LAN Recommended Practices
- Ethernet port supporting TCP (events, monitor and control) and UDP (monitor and control)
- Supports DNP 3.0 in a Level 2 implementation
- The module functions as a Server on the network supporting data read/write commands from masters/clients on the network
- Report-by-Exception data is logged to the module's database and reported on TCP socket
- Supports unsolicited messaging
- Analog deadband configurable at module level
- Supports clock synchronization from client or from Quantum (configurable synchronization frequency)
- Supports timestamp events (BI/AI). Configurable enable/disable for AI
- User defined list of acceptable host IP addresses
- Event queue supports 200 events per data type(BI/AI)
- DNP Object Definition documents are available

8.1.2 PTQ-DNPSNET

The PTQ DNP 3.0 Server over Ethernet Communications Module supports the implementation of the DNP 3.0 (Distributed Network Protocol) over Ethernet, allowing Quantum / Unity processors to easily communicate with host systems supporting the protocol. The module supports DNP Subset Level 2 features and some Level 3 features.

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
- Does not currently support Hot-Standby processors or applications

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 (PRT1)	DB-9M PC Compatible RS-232 only No hardware handshaking
Application Ethernet port	RJ45 Connector Link and Activity LED indicators Electrical Isolation 1500 V rms at 50 Hz to 60 Hz for 60 s, applied as specified in section 5.3.2 of IEC 60950: 1991 Ethernet Broadcast Storm Resiliency = less than or equal to 5000 [ARP] frames-per-second and less than or equal to 5 minutes duration

Functional Specifications

The PTQ-DNPSNET module accepts data read/write commands from a master/client on the network. The module accepts DNP commands to control and monitor the data stored in the DNP databases. In addition, the module can be configured to generate unsolicited messages.

The module has 4000 data words of internal register space that are accessible to the protocol driver and to the Quantum processors memory. Memory usage is user configurable within these 4000 words. Each of the supported database types can be individually sized and each database point is mapped within the module. The supported database point types are:

- Binary Input: 500 Points (500 words)
- Binary Output: 500 Points (500 words)
- Counter: 250 Points (250 words)
- Analog Input: 500 Words
- Analog Output: 500 Words

Total point counts must be configured such that Class 0 responses do not exceed 2048 bytes in size

DNP 3.0 Server over Ethernet Specifications

Operating in the Server mode, the module accepts commands from a Client(s) to read/write data stored in the module's internal registers. This data is easily and continuously transferred between the ProTalk module and the Quantum processor's data registers.

The Server functionality supported by the module includes:

- The DNP on Ethernet communication driver is built in accordance with the DNP organizations WAN/LAN Recommended Practices
- Ethernet port supporting TCP (events, monitor and control) and UDP (monitor and control)
- Supports DNP 3.0 in a Level 2 implementation
- The module functions as a Server on the network supporting data read/write commands from masters/clients on the network
- Report-by-Exception data is logged to the module's database and reported on TCP socket
- Supports unsolicited messaging
- Analog deadband configurable at module level
- Supports clock synchronization from client or from Quantum (configurable synchronization frequency)
- Supports timestamp events (BI/AI). Configurable enable/disable for AI
- User defined list of acceptable host IP addresses
- Event queue supports 200 events per data type(BI/AI)
- DNP Object Definition documents are available

8.2 Functional Overview

This section describes how the PTQ-DNPSNET-Q module transfers data between itself and the processor, and how it implements the DNPSNET-Q protocol.

The DNPSNET protocol driver exists as a single service port (DNPSNET port 20000) implementation that supports a single TCP port connection and multiple UDP ports on a TCP/IP Ethernet network. The DNPSNET port operates as a server, supporting the DNP 3.0 protocol in a Level 2 implementation using the DNP User Group recommended extension for use on LAN/WAN. This is published in "Transporting DNP V3.00 over Local and Wide Area Networks", December 15, 1998 by the DNP Users Group and is available on the Internet at http://www.dnp.org.

8.2.1 General Concepts

The following discussion explains several concepts that are important for understanding the operation of the PTQ-DNPSNET-Q module.

Module Power Up

On power up the module begins performing the following logical functions:

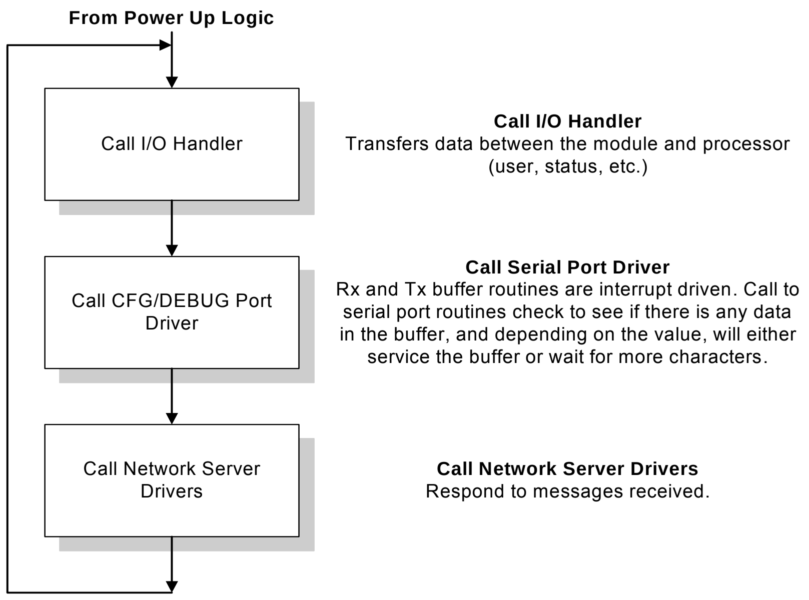
1 Initialize hardware components
- o Initialize Quantum / Unity backplane driver
- o Test and clear all RAM
- o Initialize the serial communication ports

2 Read configuration file from Random Access Memory
3 Enable Slave Driver

After the module has received the configuration, the module will begin communicating with other nodes on the network, depending on the configuration.

Main Logic Loop

Upon completing the power up configuration process, the module enters an infinite loop that performs the functions shown in the following diagram.


From Power Up Logic
Call I/O Handler
Call CFG/DEBUG Port Driver
Call Network Server Drivers
Call I/O Handler
Transfers data between the module and processor (user, status, etc.)
Call Serial Port Driver
Rx and Tx buffer routines are interrupt driven. Call to serial port routines check to see if there is any data in the buffer, and depending on the value, will either service the buffer or wait for more characters.
Call Network Server Drivers
Respond to messages received.

Backplane Data Transfer

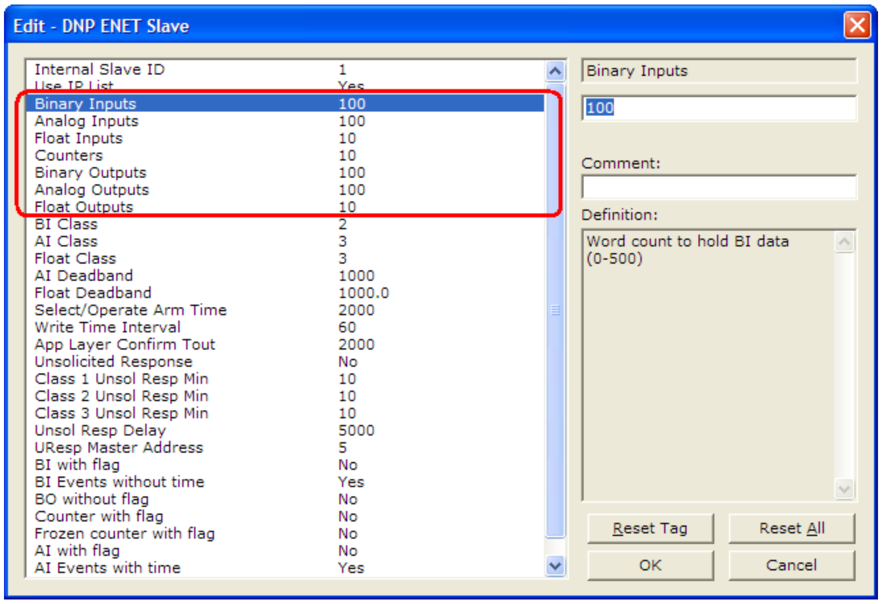
The current version of the PTQ-DNPSNET-Q backplane driver (version 2.0) uses a Large I/O model, which differs from previous versions of the backplane driver in that it transfers all of the data in the input and output databases between the module and the processor on every scan.

The [Backplane Configuration] section of the configuration file defines the starting register for input and output. The typical configuration is 5000 words read (3x/ % IW) and 5000 words write (4x / % MW).

The following parameters control how much database content will be transferred between the Processor and the Module:

- Binary Inputs
- Analog Inputs
- Float Inputs
- Counters
- Binary Outputs
- Analog Outputs
- Float Outputs

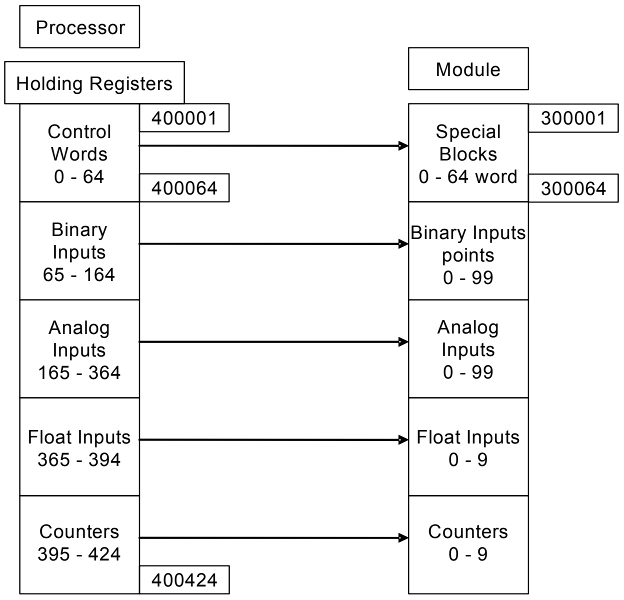
Example 1 (Sample Default Values)

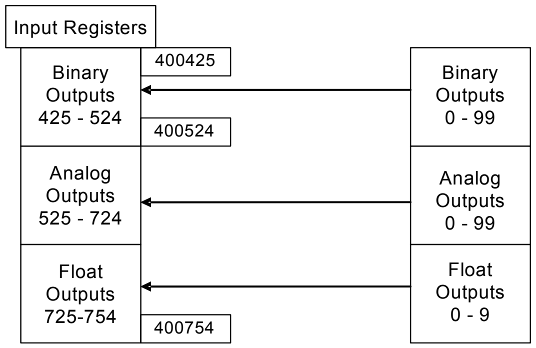


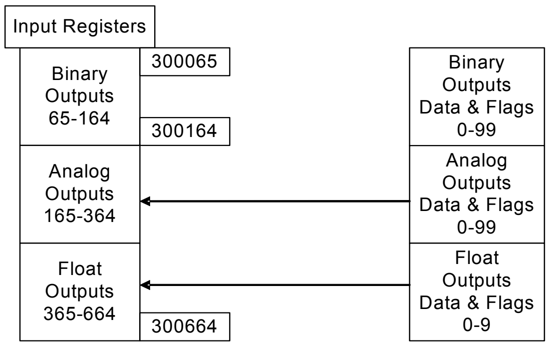
Internal Slave ID	1
Use IP List	Yes
Binary Inputs	100
Analog Inputs	100
Float Inputs	10
Counters	10
Binary Outputs	100
Analog Outputs	100
Float Outputs	10
BI Class	2
AI Class	3
Float Class	3
AI Deadband	1000
Float Deadband	1000.0
Select/Operate Arm Time	2000
Write Time Interval	60
App Layer Confirm Tout	2000
Unsolicited Response	No
Class 1 Unsol Resp Min	10
Class 2 Unsol Resp Min	10
Class 3 Unsol Resp Min	10
Unsol Resp Delay	5000
UResp Master Address	5
BI with flag	No
BI Events without time	Yes
BO without flag	No
Counter with flag	No
Frozen counter with flag	No
AI with flag	No
AI Events with time	Yes

Binary Inputs
100
Comment:
Definition:
Word count to hold BI data (0-500)

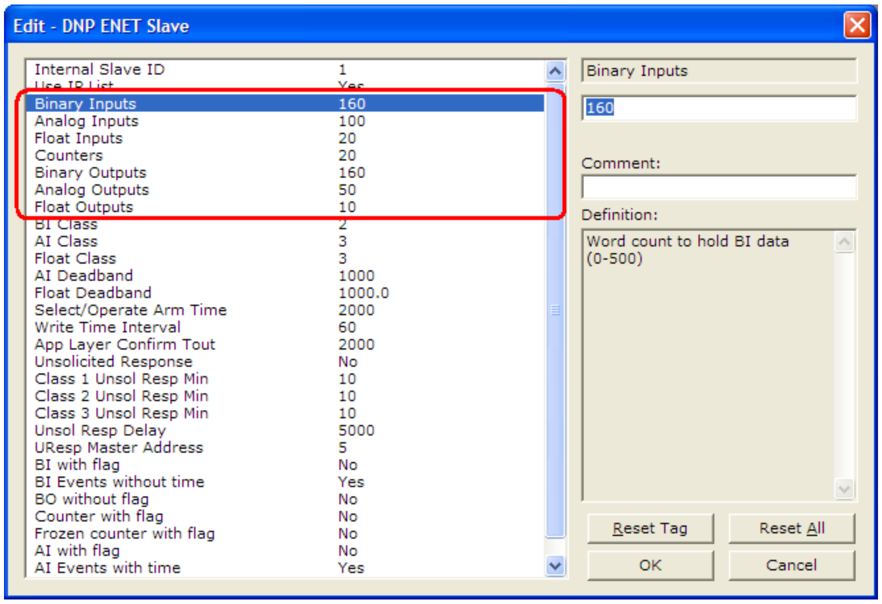
The following illustration shows the data transfer between processor and Module using the configuration values in Example 1:







Example 2

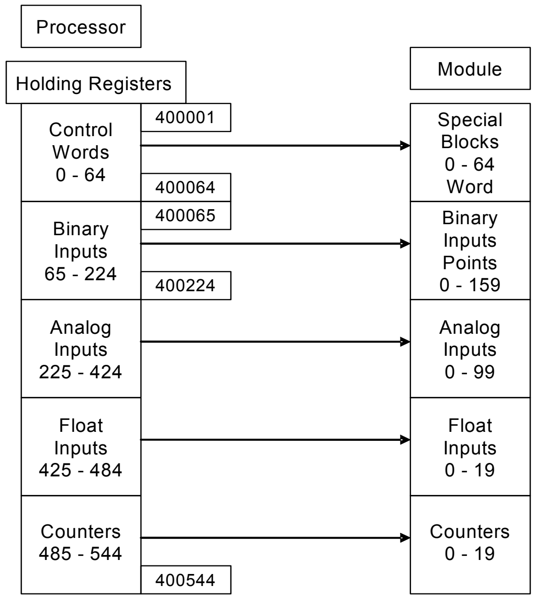


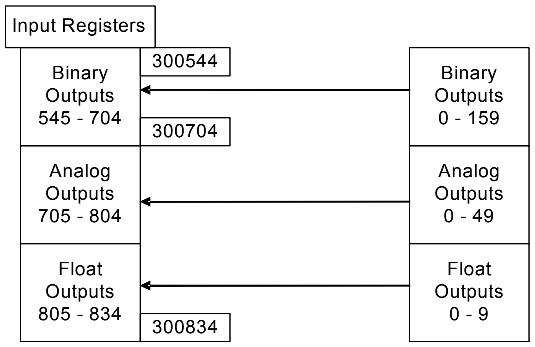
Internal Slave ID	1
Use IP List	Yes
Binary Inputs	160
Analog Inputs	100
Float Inputs	20
Counters	20
Binary Outputs	160
Analog Outputs	50
Float Outputs	10
BI Class	2
AI Class	3
Float Class	3
AI Deadband	1000
Float Deadband	1000.0
Select/Operate Arm Time	2000
Write Time Interval	60
App Layer Confirm Tout	2000
Unsolicited Response	No
Class 1 Unsol Resp Min	10
Class 2 Unsol Resp Min	10
Class 3 Unsol Resp Min	10
Unsol Resp Delay	5000
UResp Master Address	5
BI with flag	No
BI Events without time	Yes
BO without flag	No
Counter with flag	No
Frozen counter with flag	No
AI with flag	No
AI Events with time	Yes

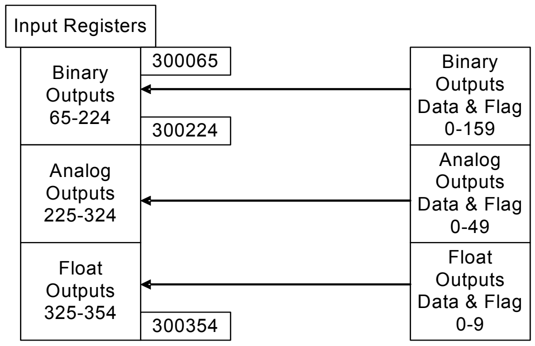
Binary Inputs: 160

Definition: Word count to hold BI data (0-500)

The following illustration shows the data transfer between processor and Module using the configuration values in Example 2:







Input and Output Data Blocks

Status Block 9250

Block 9250 identification code requests the module's status data. The module supports a buffer queue of 99 events per data type. The application can verify the status of the queue (free space in the queue) through the module's status data (Block 9250). When the queue is full, the module will delete the older event in the queue if a new event is received.

There are two ways to request status data:

- Read the data in the database, starting at word 4000. Refer to Error Status Table for detailed information.
- Request the entire status block with a block 9250 block request.

The module responds to a valid block 9250 request with a block containing the requested status data. The status data area for the module starts at address 4000 in the database.

Block Format for Read

Word Offset	Variable Name	Description
0	Sequence number	This word contains a new sequence number when the status block is filled.
1	Status request block	This word will contain the value of 9250 when the operation is complete.
2	Program Scan Count	This value is incremented each scan of the program. Use this command to check the program scan frequency and to make sure the module is still operational.
3 to 4	Product Name (ASCII)	These two words contain the product name of the module in ASCII format (DNQQ).
5 to 6	Revision (ASCII)	These two words contain the product revision level of the firmware in ASCII format.
7 to 8	Operating System Revision (ASCII)	These two words contain the module's internal operating system revision level in ASCII format.
9 to 10	Production Run Number (ASCII)	These two words contain the production "batch" number for the particular chip in the module in ASCII format.
11	Output Transfer Count	This value is incremented each time output data is transferred from the PLC to the module.
12	Input Transfer Count	This value is incremented each time input data is transferred from the module to the PLC.
13	I/O Error Transfer Count	This value is incremented each time an input or output transfer error occurs.
14	DNP Slave Port total number of message frames received by slave	This value represents the TCP/IP 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.
15	DNP Slave Port total number of response message frames sent from slave	This value represents the number of good (non-error) TCP/IP 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.
16	DNP Slave Port total number of message frames seen by slave	This value represents the total number of TCP/IP message frames received by the slave, regardless of the slave address.
17	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.
18	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.
19	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.
20	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.

Word Offset	Variable Name	Description
21	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.
22	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.
23	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.
24	DNP Slave Binary Input Event count	This value contains the total number of binary input events which have occurred.
25	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.
26	DNP Slave Analog Input Event count	This value contains the total number of analog input events which have occurred.
27	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.
28	DNP Float Event Count	Total number of events generated for analog floating-point input data points.
29	DNP Double Event Count	Total number of events generated for analog double, floating-point input data points.
30	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.
31	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.
32	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.
33	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.
34	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.
35	UDP Receive	This counter is incremented each time a UDP packet is received.
36	UDP Transmit	This counter is incremented each time a UDP packet is transmitted.
37	Unsolicited Transmit Error Count	This counter is incremented each time an unsolicited transmit error occurs.

Word Offset	**Variable Name**	**Description**
38	P1 State	This value contains the value of the P1 Tx State (see below).
39	TCP Socket State	This value contains the current state value of the TCP/IP socket. If the value is < 1, the socket is not connected to a remote client. If the value is >= 100, the socket is disconnecting. All other values represent a connected socket condition.
40	UDP Socket State	This value contains the current state value of the UDP/IP socket.
41	Busy Flag	This flag determines if either the UDP or TCP socket is busy processing a message and building a response: 0=not busy, 1=TCP message processing, 2=UDP message processing and 3=TCP processing unsolicited message.
42	Application Layer Fragment Flag	This flag indicates if a multiple application fragment is being handled for transmission (0=no, 1=yes).
43	Tx Frame Flag	This flag indicates if a frame is ready to be transmitted at the data link level (0=no, 1=yes)
44	TCP Length	This value contains the current length of receive data for the TCP/IP socket.
45	UDP Length	This value contains the current length of receive data for the UDP/IP socket.
46	Free Memory LSB	Free memory in module
47	Free Memory MSB	
48	P1 Tx State	This flag indicates if either socket is busy sending data (0=no, 1=yes).
49	Processor State	This status register will contain a value of 1 if the processor is in run mode and 0 if it is not.
50	I/O parameters set	This status register contains a value of 0 if the I/O sizes have been read from the processor and 1 if not.
51	Hot-Standby status word	This status register contains the hot-standby status word. This will be utilized when the hot-standby support is added to the product.

Block 9958 - PLC Binary Input Event data

Block 9958 identification code is used by the PLC to send a set of binary input events to the module.

Block Format from Processor

Word Offset in Block	**Data Field(s)**	**Description**
0	Sequence Counter	This field contains a new value each time the user wishes to send events
1	Block ID	This field contains the value of 9958 identifying the event block to the module.
2	Event Count	This field contains the number of events contained in the block. Valid values for this field are 1 to 10.

Word Offset in Block	Data Field(s)	Description
3	Sequence Counter	This field holds the sequence counter for each 9958 block transfer. This synchronizes and confirms receipt of the block by the module.
4	DNP Binary Input Data point	This is the data point in the DNP binary input database represented by the event.
5	Event Value	This word contains the new value for the point and the event. Only the LSB byte portion of the word is valid for the DNP protocol.
6	Month/Day	Formatted: bits 0 to 4 = Day, bits 8 to 11 = Month. All other bits are ignored.
7	Hour/Minute	Formatted: bits 0 to 5 = Minutes, bits 8 to 12 = Hour. All other bits are ignored.
8	Sec/Millisecond	Formatted: bits 0 to 9 = Milliseconds, bits 10 to 15 = Seconds.
9	Year	This is the four digit year for the event.
10 to 15		Five words of data for Event #2.
16 to 21		Five words of data for Event #3.
22 to 27		Five words of data for Event #4.
28 to 33		Five words of data for Event #5.
34 to 39		Five words of data for Event #6.
40 to 45		Five words of data for Event #7.
46 to 51		Five words of data for Event #8.
52 to 57		Five words of data for Event #9.
58 to 63		Five words of data for Event #10.

Block Format from Module

To insure the receipt of this block of information, the module returns a block 9958 with the sequence counter set to the value of the last successful block 9958 received.

Word Offset in Block	Data Field(s)	Description
0	Sequence Counter	This field contains a new value each time the block is handled.
1	Block ID	This field contains the value of 9958 identifying the event block to the module.
2	Event Count	This field contains the number of events processed by the module.
3	Sequence Counter	This field contains the sequence counter of the last successful block 9958 received.

Block 9959 - PLC Analog Input Event Data

Block 9959 identification code is used by the PLC to send a set of analog input events to the module.

Block Format from Processor

Word Offset in Block	Data Field(s)	Description
0	Sequence Counter	This field contains a new value each time the user wishes to send events
1	Block ID	This field contains the value of 9959 identifying the event block to the module.
2	Event Count	This field contains the number of events contained in the block. Valid values for this field are 1 to 10.
3	Sequence Counter	This field holds the sequence counter for each 9959 block transfer. This synchronizes and confirms receipt of the block by the module.
4	DNP Analog Input Data point	This is the data point in the DNP analog input database represented by the event.
5	Analog Input Value	This is the new analog input value represented in the event.
6	Month/Day	Formatted: bits 0 to 4 = Day, bits 8 to 11 = Month. All other bits are ignored.
7	Hour/Minute	Formatted: bits 0 to 5 = Minutes, bits 8 to 12 = Hour. All other bits are ignored.
8	Sec/Millisecond	Formatted: bits 0 to 9 = Milliseconds, bits 10 to 15 = Seconds.
9	Year	Four digit year value for event.
10 to 15		Six words of data for Event #2.
16 to 21		Six words of data for Event #3.
22 to 27		Six words of data for Event #4.
28 to 33		Six words of data for Event #5.
34 to 39		Six words of data for Event #6.
40 to 45		Six words of data for Event #7.
46 to 51		Six words of data for Event #8.
52 to 57		Six words of data for Event #9.
58 to 63		Six words of data for Event #10.

Block Format from Module

To insure the receipt of this block of information, the module returns a BTR block 9959 with the sequence counter set to the value of the last successful block 9959 received.

Word Offset in Block	Data Field(s)	Description
0	Sequence Counter	This field contains a new value each time the block is handled.
1	Block ID	Block identification code for request from PLC by the module.

Word Offset in Block	Data Field(s)	Description
2	Event Count	This field contains the number of events processed by the module.
3	Sequence Counter	This field contains the sequence counter of the last successful block 9959 received.

Block 9970 - Set PLC time using module's DNP time

Block 9970 identification code requests the module's DNP date and time. Use this data to set the PLC clock.

Block Format from Processor

Word Offset in Block	Data Field(s)	Description
0	Sequence Counter	This field contains a new value each time the user wishes to send events
1	Block ID	This field contains the value of 9970 identifying the block type to the module.

Block Format from Module

Response to a block 9970 request: The module will respond to a valid block 9970 request with a block containing the requested date and time. The format for the block is shown below:

Word Offset in Block	Data Field(s)	Description
0	Sequence Counter	This field contains a new value each time the block is handled.
1	Block Write ID	This is the next block requested by the module.
2	Year	This field contains the four-digit year to be used with the new time value.
3	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	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	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	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	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	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	Remote Time Synchronization	This field informs the PLC if the date and time passed has been synchronized with a remote DNP master device on the module's slave port.

Block 9971 - Set Module's Time using the Quantum / Unity Processor Time

Block identification code 9971 passes the clock time in the PLC to the module. The date and time provided will be used to set the module's DNP clock.

Word Offset in Block	Data Field(s)	Description
0	Block ID	This field contains the block identification code of 9971 for the block.
1	Year	This field contains the four-digit year to be used with the new time value.
2	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	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	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	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	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	Milliseconds	This field contains the millisecond value for the new time. Valid entry for this field is in the range of 0 to 999.
8 to 247	Not Used	Not Used

Block Format from Processor

Word Offset in Block	Data Field(s)	Description
0	Sequence Counter	This field contains a new value each time the user wishes to send events
1	Block ID	This field contains the block identification code of 9971 for the block.
2	Year	This field contains the four-digit year to be used with the new time value.
3	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	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	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	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	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	Milliseconds	This field contains the millisecond value for the new time. Valid entry for this field is in the range of 0 to 999.

Block Format from Module

Word Offset in Block	Data Field(s)	Description
0	Sequence Counter	This field contains a new value each time the block is handled.
1	Block ID	This field contains the block identification code of 9971 for the block.

Block 9998 - Warm Boot Module

If the Quantum / Unity sends a block number 9998, the module will perform a warm-boot operation. The module will reconfigure the communication ports and reset the error and status counters.

Block 9999 - Cold Boot Module

If the Quantum / Unity processor sends a block number 9999, the application will perform the cold-boot operation. The module's program will request a block 9000 from the Quantum / Unity processor for the configuration information. After the module receives the block, the program will configure the module using the data received and reset all DNPSNET-Q memory, error and status data.

8.2.2 PTQ-DNPSNET-Q Application Design

This documentation describes the PTQ-DNPSNET-Q module configuration and setup as it applies to application design. The design of the entire system must be complete before you attempt to implement this module with a DNP network. 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. 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.

Design

In order to implement a solution using the module, the Quantum 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 Quantum processor contains the data to be used by the module and the configuration information is stored in the text file, DNPSNET_Q.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 Quantum processor with the DNP master. The module provides the following data types:

- Digital Input
- Digital Output
- Counter
- Floating Point
- Analog Input
- 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.

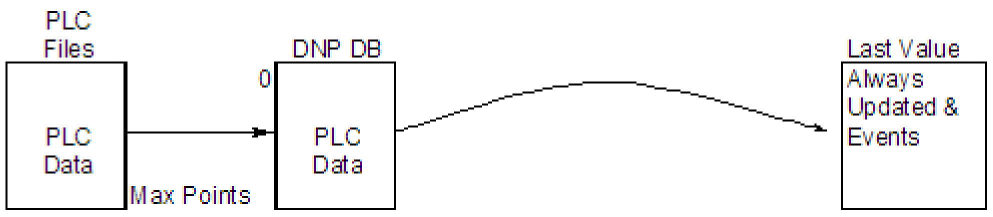
The following illustration shows the databases maintained by the module for the DNP data.

DATA AREA	
DNP DATA	BINARY INPUTS
	ANALOG INPUTS
	FLOAT INPUTS
	COUNTER DATA
	BINARY OUTPUTS
	ANALOG OUTPUTS

The module is responsible for maintaining 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.

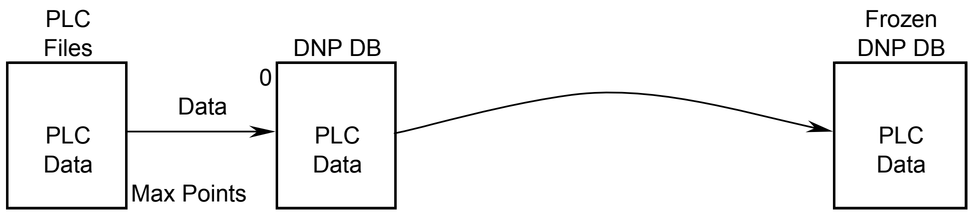
Binary and Analog Input Databases



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.

Counter Databases

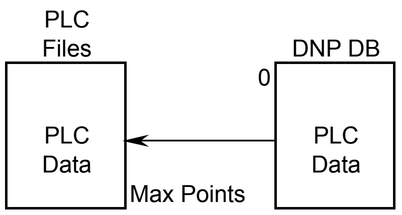


This data is constantly sourced from the processor and placed in the module's internal database. This information is available to the remote master for monitoring. When the module receives a freeze command from the master unit, it will copy the current counter values into the frozen counter database 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.

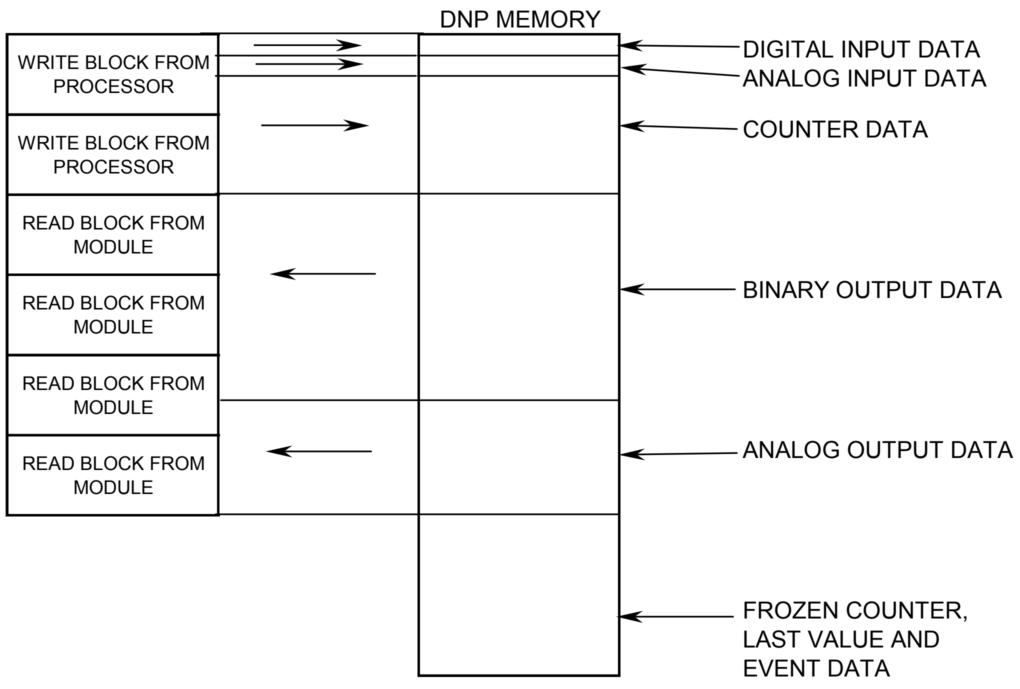
Binary and Analog Output Databases



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.



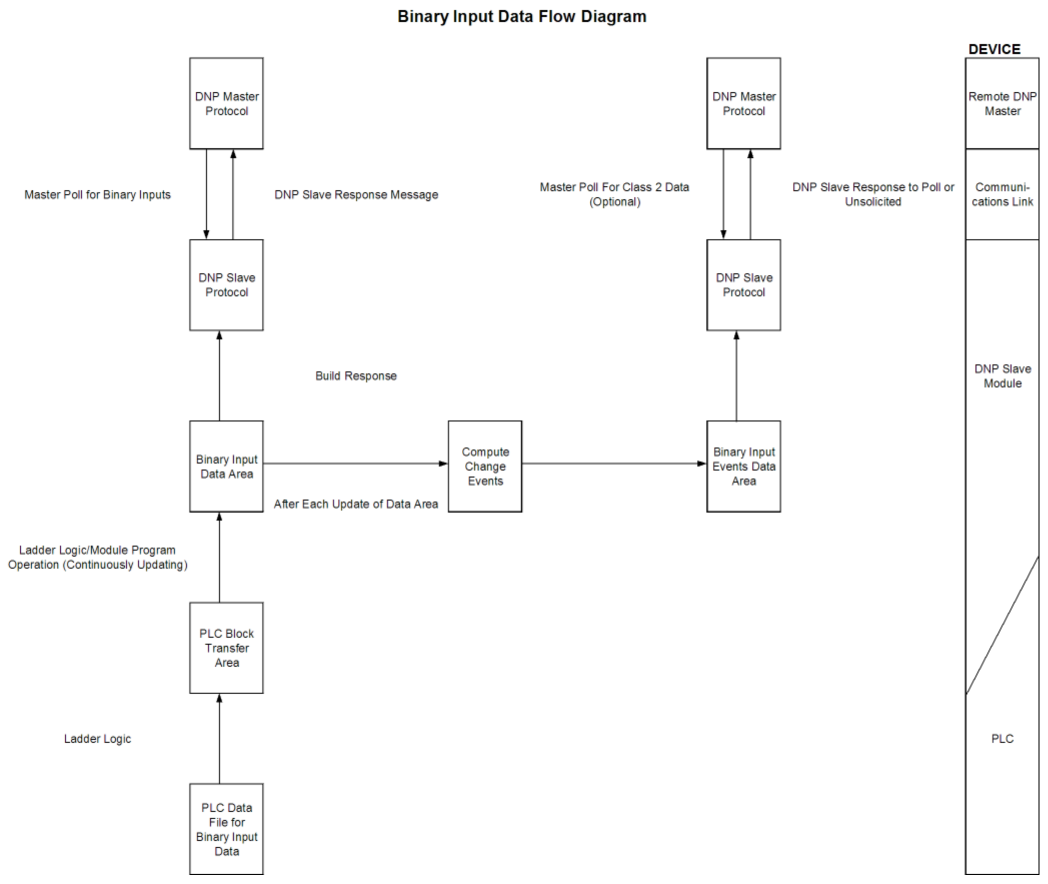
It is important to understand the relationship of the block identifications and the data in the module.

The Reference chapter contains forms to aid in designing your system. They can be used to document the relationship between the point assignments, block identification numbers and the PLC file and offset values and to define the program configuration. Use these forms during your design phase.

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 1 binary input point). 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.

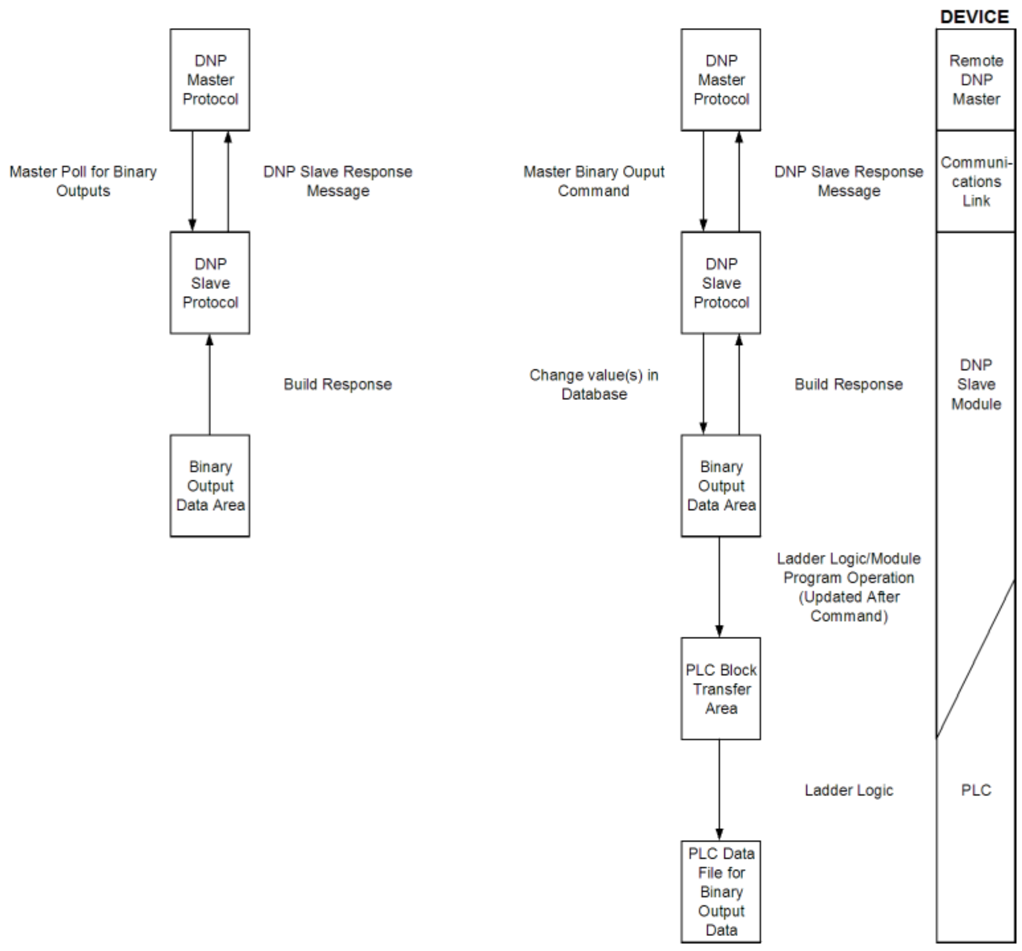
The remote DNP master unit can read the current status data and the event data from the module. Event messages generated by the module can be retrieved using a poll for Class 2 data, as all digital input events are considered a Class 2 data type. If unsolicited message generation is enabled in the application, the events will automatically be sent by the module to the DNP master unit when the maximum event count for Class 2 data is reached or when the timeout for unsolicited messages is exceeded. A data flow diagram for the digital input data is shown in the following figure.


Binary Input Data Flow Diagram
DEVICE
DNP Master Protocol
DNP Master Protocol
Remote DNP Master
Master Poll for Binary Inputs
DNP Slave Response Message
Master Poll For Class 2 Data (Optional)
DNP Slave Response to Poll or Unsolicited
Communi-cations Link
DNP Slave Protocol
DNP Slave Protocol
Build Response
DNP Slave Module
Binary Input Data Area
Compute Change Events
Binary Input Events Data Area
After Each Update of Data Area
Ladder Logic/Module Program Operation (Continuously Updating)
PLC Block Transfer Area
Ladder Logic
PLC
PLC Data File for Binary Input Data

<u>DNP Digital Output Data</u>

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 1 binary output point). 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.

Binary Output Data Flow Diagram



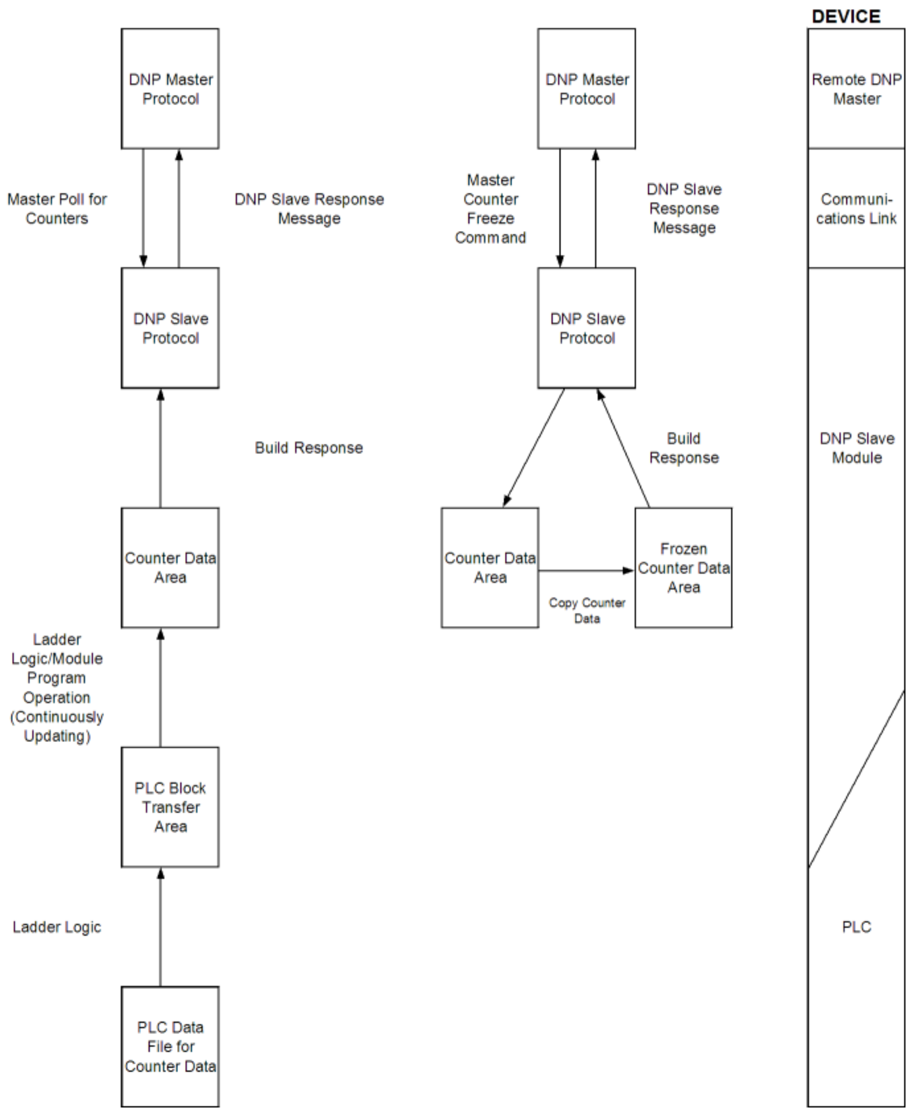
DNP Counter Data

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.

Counter Data Flow Diagram

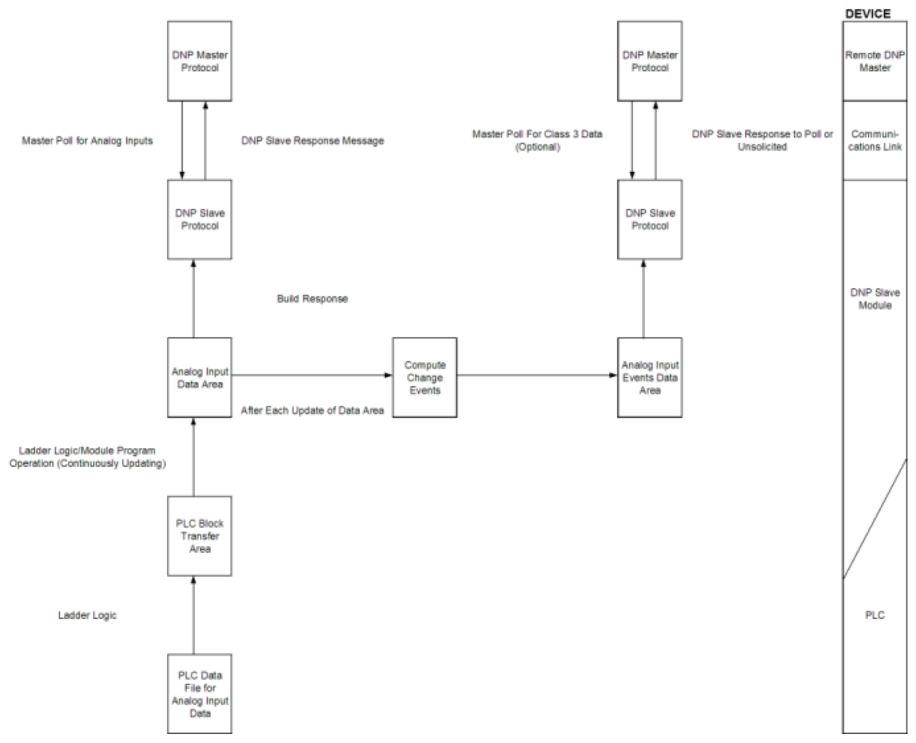


DNP Analog Input Data

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 PLC 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 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. 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 maximum 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.

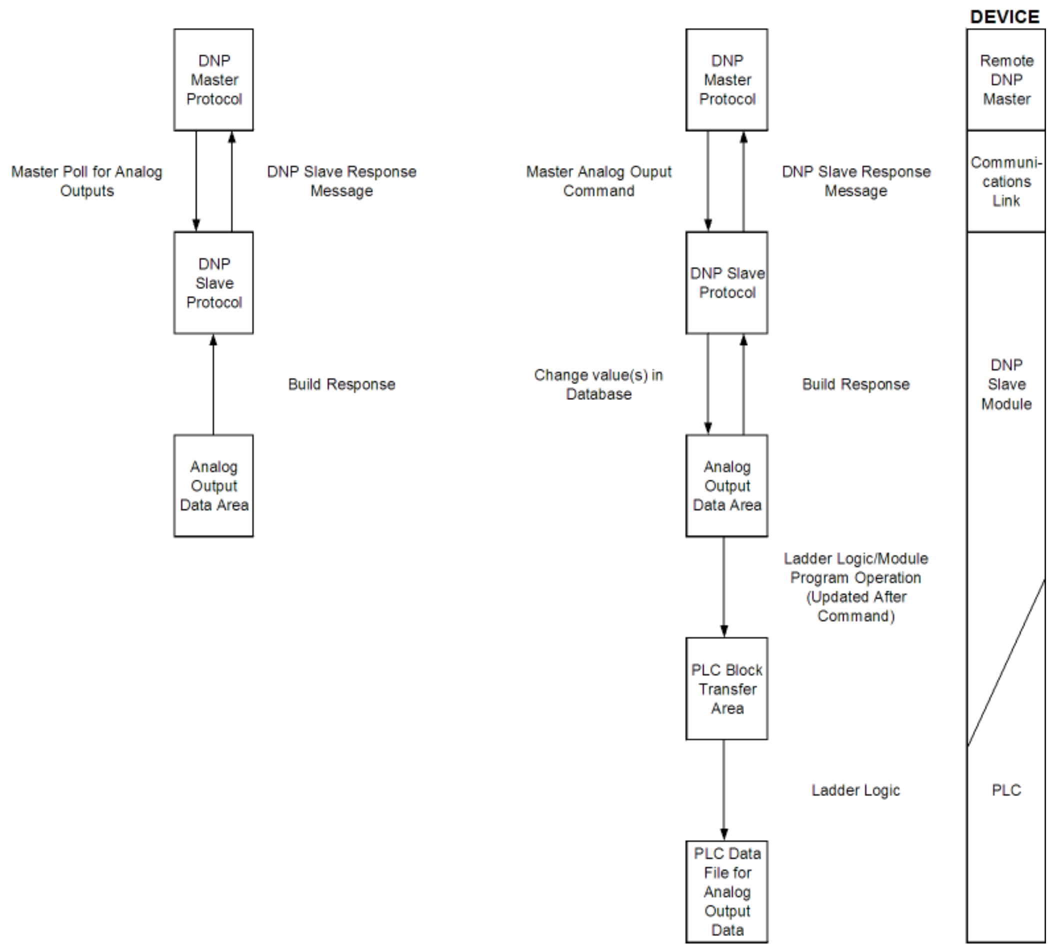
Analog Input Data Flow Diagram


DEVICE
DNP Master Protocol
DNP Master Protocol
Remote DNP Master
Master Poll for Analog Inputs
DNP Slave Response Message
Master Poll For Class 3 Data (Optional)
DNP Slave Response to Poll or Unsolicited
Communi-cations Link
DNP Slave Protocol
DNP Slave Protocol
Build Response
DNP Slave Module
Analog Input Data Area
Compute Change Events
Analog Input Events Data Area
After Each Update of Data Area
Ladder Logic/Module Program Operation (Continuously Updating)
PLC Block Transfer Area
Ladder Logic
PLC
PLC Data File for Analog Input Data

DNP Analog Output Data

This data type stores analog values sent from the DNP master unit to the module and PLC 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 PLC using the write operation. Therefore, these data are read-only for the PLC, as the PLC 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.

Analog Output Data Flow Diagram



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 Module Setup (page 49).

Data Transfer at Startup

The module can be configured to have the internal databases initialized with data contained in the processor. This feature requires ladder logic. 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.

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 PLC 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 once 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 is also responsible for enabling 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.

Additionally, the program will listen on Port 1 for requests. This is the debug port for the module and transfers module information to an attached terminal. Refer to Diagnostics and Troubleshooting (page 65) for a complete discussion on the use of this important feature.

8.3 Cable Connections

The PTQ-DNPSNET-Q module has the following communication connections on the module:

- One Ethernet port (RJ45 connector)
- One RS-232 Configuration/Debug port (DB9 connector)

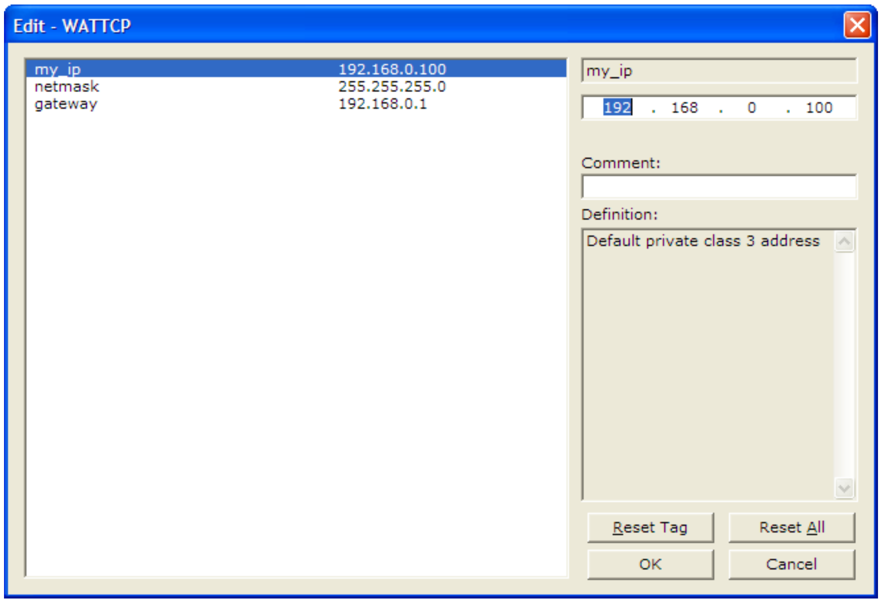
8.3.1 Ethernet Connection

The PTQ-DNPSNET-Q module has an RJ45 port located on the front of the module labeled "Ethernet", for use with the TCP/IP network. The module is connected to the Ethernet network using an Ethernet cable between the module’s Ethernet port and an Ethernet switch or hub.

Note: Depending on hardware configuration, you may see more than one RJ45 port on the module. The Ethernet port is labeled "Ethernet".
Warning: The PTQ-DNPSNET-Q module is NOT compatible with Power Over Ethernet (IEEE802.3af / IEEE802.3at) networks. Do NOT connect the module to Ethernet devices, hubs, switches or networks that supply AC or DC power over the Ethernet cable. Failure to observe this precaution may result in damage to hardware, or injury to personnel.
Important: The module requires a static (fixed) IP address that is not shared with any other device on the Ethernet network. Obtain a list of suitable IP addresses from your network administrator BEFORE configuring the Ethernet port on this module.

Ethernet Port Configuration - wattcp.cfg

The wattcp.cfg file must be set up properly in order to use a TCP/IP network connection. You can view the current network configuration using an ASCII terminal by selecting **[@]** (Network Menu) and **[V]** (View) options when connected to the Debug port.



Edit - WATTCP

```
my_ip       192.168.0.100
netmask     255.255.255.0
gateway     192.168.0.1
```

my_ip

192 . 168 . 0 . 100

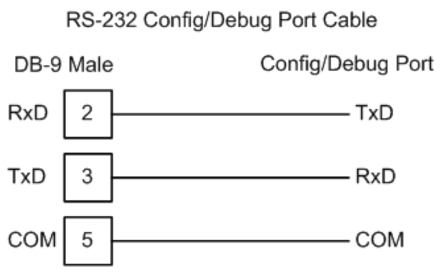
Comment:

Definition:

Default private class 3 address

8.3.2 RS-232 Configuration/Debug Port

This port is physically an RJ45 connection. An RJ45 to DB-9 adapter cable is included with the module. 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:



8.4 Configuration Error Word

Bit	Code	Description
0	0x0001	Invalid baud rate selected
1	0x0002	Invalid address assigned (0 to 65534)
2	0x0004	Database defined will not fit into memory
3	0x0008	Invalid binary input point count
4	0x0010	Invalid binary output point count
5	0x0020	Invalid counter point count
6	0x0040	Invalid analog or float input point count
7	0x0080	Invalid analog or float output point count
8	0x0100	
9	0x0200	
10	0x0400	
11	0x0800	
12	0x1000	
13	0x2000	
14	0x4000	
15	0x8000	

8.4.1 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.
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.

Error Code	Name	Description
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.

8.4.2 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 512.
101	Too many binary output points	Too many binary output points are configured for the module. Maximum value is 512.
102	Too many counter points	Too many counter points are configured for the module. Maximum value is 128.
103	Too many analog input points	Too many analog input points are configured for the module. Maximum value is 512.
104	Too many analog output points	Too many analog output points are configured for the module. Maximum value is 512.
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.
123	Too many float input or output points	Too many float input or output points are configured for the module. Maximum value is 128 for each type.
334	Baud rate of secondary port	The baud rate configured for the secondary slave port is invalid.

8.4.3 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.

Error Code	Name	Description
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.

8.4.4 Error Status Table

The program maintains an error/status table. This table of data is available to the Quantum / Unity processor automatically through block 100. Ladder logic should be programmed to accept this block of data and place it in the module's controller tag. You can use the error/status data to determine the "health" of the module.

The data in the block is structured as shown in the following table.

Word	Block Offset	Variable Name	Description
0	2	Current PTQ-DNPSNET-QSlave 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	3	PTQ-DNPSNET-QSlave Port last transmitted error code	This value represents the last error code transmitted to the master by this slave port.
2	4	PTQ-DNPSNET-QSlave 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.
3	5	PTQ-DNPSNET-QSlave 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	6	PTQ-DNPSNET-QSlave 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.

Word	Block Offset	Variable Name	Description
5	7	PTQ-DNPSNET-QSlave 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	8	PTQ-DNPSNET-QSlave 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	9	PTQ-DNPSNET-QSlave 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	10	PTQ-DNPSNET-QSlave bad CRC error (Data Link Layer Error)	This value counts the number of times a bad CRC value is received in a message.
9	11	PTQ-DNPSNET-QSlave 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	12	PTQ-DNPSNET-QSlave 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	13	PTQ-DNPSNET-QSlave 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	14	PTQ-DNPSNET-QSlave Binary Input Event count	This value contains the total number of binary input events which have occurred.
13	15	PTQ-DNPSNET-QSlave 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	16	PTQ-DNPSNET-QSlave Analog Input Event count	This value contains the total number of analog input events which have occurred.
15	17	PTQ-DNPSNET-QSlave 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	18	PTQ-DNPSNET-QSlave 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	19	PTQ-DNPSNET-QSlave 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	20	PTQ-DNPSNET-QSlave 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	21	PTQ-DNPSNET-QSlave 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	Block Offset	Variable Name	Description
20	22	PTQ-DNPSNET-QSlave multi-frame message from PTQ-DNPSNET-QMaster 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	23	Total blocks transferred	Total BTR/BTW or side-connect interface transfers attempted by the module.
22	24	Successful blocks transferred	This value represents the total number of transfer operations between the Quantum / Unity processor and module that are successful.
23	25	Total errors in block transfer	Total number of transfers that resulted in an error condition.
24	26	Total BTR or write errors	Total number of BTR or write transfers that resulted in an error.
25	27	Total BTW or read errors	Total number of BTW or read transfers that resulted in an error.
26	28	Block number error	Number of BTW requests that resulted in an incorrect BTW identification code.
27	29	Continuous block error counter	Count of sequential data transfer errors. When this value exceeds that specified for the data transfer operation, the error flag below will be set.
28	30	Reserved	Not used
29	31	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	32 to 33	Product Name (ASCII)	These two words contain the product name of the module in ASCII format.
32 to 33	34 to 35	Revision (ASCII)	These two words contain the product revision level of the firmware in ASCII format.
34 to 35	36 to 37	Operating System Revision (ASCII)	These two words contain the module's internal operating system revision level in ASCII format.
36 to 37	38 to 39	Production Run Number (ASCII)	These two words contain the production "batch" number for the particular chip in the module in ASCII format.
38	40	PTQ-DNPSNET-QMaster Port Slave Count	This is the total number of slaves configured for the PTQ-DNPSNET-QMaster port. This may not represent the number of active slaves as it includes slaves that are not enabled.
39	41	PTQ-DNPSNET-QMaster Port Command Count	This is the total number of commands configured for the PTQ-DNPSNET-QMaster port. This may not represent the number of active commands as it includes commands that are disabled.

Word	Block Offset	Variable Name	Description
40	42	PTQ-DNPSNET-QMaster 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	43	PTQ-DNPSNET-QMaster Port Frame Block Count	This value represents the number of physical layer frame memory allocation blocks used by the program.
42	44	PTQ-DNPSNET-QMaster Port Data Link Receive Block Count	This value represents the number of receive data link layer memory blocks allocated.
43	45	PTQ-DNPSNET-QMaster Port Data Link Transmit Block Count	This value represents the number of transmit data link layer memory blocks allocated.
44	46	PTQ-DNPSNET-QMaster Port Application Layer Receive Block Count	This value represents the number of application layer receive memory blocks allocated.
45	47	PTQ-DNPSNET-QMaster Port Application Layer Receive Block Count	This value represents the number of application layer transmit memory blocks allocated.
46	48	PTQ-DNPSNET-QMaster Port Device Memory Allocation Error Count	This value represents the number of memory allocation errors for device blocks.
47	49	PTQ-DNPSNET-QMaster Port Physical Layer Memory Allocation Error Count	This value represents the number of memory allocation errors for physical layer frame blocks.
48	50	PTQ-DNPSNET-QMaster 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	51	PTQ-DNPSNET-QMaster 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	52	PTQ-DNPSNET-QMaster Port Application Layer Receive Memory Allocation Error Count	This value represents the number of memory allocation errors for application layer receive blocks.
51	53	PTQ-DNPSNET-QMaster Port Application Layer Transmit Memory Allocation Error Count	This value represents the number of memory allocation errors for application layer transmit blocks.
52	54	PTQ-DNPSNET-QMaster 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	55	PTQ-DNPSNET-QMaster 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.

Word	Block Offset	Variable Name	Description
54	56	PTQ-DNPSNET-QMaster Bad CRC Error Count (Physical Layer Error)	This value counts the number of times a bad CRC value is received in a message.
55	57	Scan Counter LSB	Program scan counter
56	58	Scan Counter MSB	
57	59	Free Memory LSB	Free memory in module
58	60	Free Memory MSB	
59	61	PTQ-DNPSNET-QSlave Port Transmit State	Value of the PTQ-DNPSNET-QSlave state machine for transmit.
60	62	PTQ-DNPSNET-QFloat Event Count	Total number of events generated for analog floating-point input data points.
61	63	PTQ-DNPSNET-QDouble Event Count	Total number of events generated for analog double, floating-point input data points.
62	64	Event Message Queue Count	Number of event messages waiting to send to processor.
63	65	Event Message Queue Overflow	Flag to indicate if the event message queue has overflowed. If more than 200 event messages are received on the master port and they are not sent to the processor, this flag will be set (1). The flag will clear after the messages are sent to the processor.
64 to 77	66 to 79	Reserved	Future Use
78	80	Error_List[0]	First value in error list
79	81	Error_List[1]	Second value in error list
-	-	-	-
137	139	Error_List[59]	Last value in error list

8.4.5 Module Error Codes

If the module's program encounters an error during execution, it will log the error to the error list. This list is transferred to the Quantum / Unity processor using block identification code 100 (see section above) in at offsets 62 to 119. This data is also available for viewing on the debug monitor port. The following tables list the error codes generated by the program with their associated description. Use the errors to help define where problems exist in the system.

Slave Port Communication Errors

Error Code	Name	Description
0	OK	The module is operating correctly and there are no errors.
10	PTQ-DNPSNET-Qsynchronization error (Physical Layer Error)	Extra bytes are received before the start bytes (0x05 and 0x64).
11	PTQ-DNPSNET-Qoverrun error (Physical Layer Error)	Mainline Data Link Layer routine could not read data received on PTQ-DNPSNET-Qport before it was overwritten.

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

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 output 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.

Error Code	Name	Description
109	Invalid block transfer delay for blocks 251 and 252 (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.

DNPSNET-Q Port Configuration Errors

Error Code	Name	Description
212	Invalid PTQ-DNPSNET-Qaddress	The PTQ-DNPSNET-Qaddress specified in the configuration is not valid (0 to 65534).
213	Invalid PTQ-DNPSNET-Qport baud rate	The baud rate code specified in the configuration is not valid.
219	Invalid PTQ-DNPSNET-Qdata link layer confirm mode	The data link confirmation mode code is not valid in the configuration.
220	Invalid PTQ-DNPSNET-Qdata 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 PTQ-DNPSNET-Qselect/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 PTQ-DNPSNET-Qapplication 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 PTQ-DNPSNET-Qwrite 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 PTQ-DNPSNET-Qunsolicited response mode	The unsolicited response mode code is not valid in the configuration.
226	Invalid PTQ-DNPSNET-Qunsolicited 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 PTQ-DNPSNET-Qunsolicited 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 PTQ-DNPSNET-Qunsolicited 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 PTQ-DNPSNET-Qunsolicited response destination address	The unsolicited response destination address is not valid in the configuration. Value must be in the range of 1 to 65534.

8.4.6 Command Error Codes

General Command Errors

Error Code	Name	Description
-1 (65535)	Current command being issued on the port	Command has been issued out the port, and the module is waiting for the slave to respond.
0	OK	The command was issued and responded to correctly.
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 PTQ-DNPSNET-Qsubset 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 PTQ-DNPSNET-Qsubset for the Master Port.
10	Invalid binary input poll command	This binary input object command is not valid.
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 function command 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 function command	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.
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.

8.5 DNP Subset Definition

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.

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				Slave will return variation 1 data (user can override and have variation 2 returned)
	1	Binary Input	1	06	129, 130	00, 01	1	Slave will return this variation
	2	Binary Input with Status	1	06	129, 130	00, 01	8	Slave will return this variation
2	0	Binary Input Change - All Variations	1	06, 07, 08				Slave will return variation 2 data (user can override and have variation 1 returned)
	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	00, 06				Slave will return variation 2 data (user can override and have variation 1 returned)
	1	Binary Output	1	00, 06			1	Slave will return this variation
	2	Binary Output Status	1	00, 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				Slave will return variation 5 data (user can override and have variation 1 returned)
	1	32-Bit Binary Counter	1, 7, 8, 9, 10	06	129, 130	00, 01	40	Slave will return this variation
	2	16-Bit Binary Counter	1, 7, 8, 9, 10	06	129, 130	00, 01	24	Slave will return this variation (counter upper 16-bits removed)
	3	32-Bit Delta Counter			129, 130	00, 01	40	Slave will return Unknown Object to this request

OBJECT			REQUEST		RESPONSE			
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	NOTES
	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				Slave will return variation 9 data (user can override and have variation 1 returned)
	1	32-Bit Frozen Counter	1	06	129, 130	00, 01	40	Slave will return this variation
	2	16-Bit Frozen Counter	1	06	129, 130	00, 01	24	Slave will return this variation (counter upper 16-bits removed)
	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

OBJECT			REQUEST		RESPONSE			
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	NOTES
	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
	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

OBJECT			REQUEST		RESPONSE			
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	NOTES
	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				Slave will respond with variation 4 data (user can override and have variation 2 returned)
	1	32-Bit Analog Input	1	06	129, 130	00, 01	40	Slave will return this variation (Note: Data will only be 16-bit)
	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

OBJECT			REQUEST		RESPONSE			
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	NOTES
	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				Slave will return variation 2 data (user can override and have variation 4 returned)
	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
	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

OBJECT			REQUEST		RESPONSE			
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	NOTES
	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
	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

OBJECT			REQUEST		RESPONSE			
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	NOTES
	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
	1	Application Identifier						
100	0							
	1	Short Floating Point					48	

OBJECT			REQUEST		RESPONSE			
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	NOTES
	2	Long Floating Point					80	
	3	Extended Floating Point					88	
101	0							
	1	Small Packed Binary-Coded Decimal					16	
	2	Medium Packed Binary-Coded Decimal					32	
	3	Large Packed Binary-Coded Decimal					64	
110	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

8.6 Device Profile

DNP V3.00 **DEVICE PROFILE DOCUMENT**	
Vendor Name: ProSoft Technology, Inc.	
Device Name: PTQ-DNPSNET (VERSION 1.00)	
Highest DNP Level Supported : For Request: L2 For Responses: L2	Device Function: Slave (TCP/IP Server (Data Provider))
Notable objects, functions, and/or qualifiers supported in addition to the highest DNP level stated above (see attached table for complete list): Definition of selected IIN bits: Device Trouble - PLC data transfer operation is not taking place and Supports both TCP and UDP protocols as specified in the recommendation document. Supports new function 24 and object 50 variation 3 for time synchronization. Supports list of valid IP addresses for clients to connect (may be disabled by user). Setting of IP list secure. Supports receipt of multiple messages in a single network packet. The following features are configurable on the module: Time sync before events are generated and default analog input events, Obj32V4 or O32V2, select option. Counter Freeze with reset will not zero values in the processor. Therefore, this function should not be utilized. Module will not generate events until Restart IIN bit is cleared by DNP master.	
Maximum Data Link Frame Size (octets): Transmitted : 292 Received : 292	Maximum Application Fragment Size (octets): Transmitted : 2048 Received : 2048
Maximum Data Link Re-tries: Configurable	Maximum Application Layer Re-tries: None
Requires Data Link Layer Confirmation: Always set to Never as defined in recommendation	
Requires Application Layer Confirmation: When reporting Event Data as a slave unit	

Time-outs while waiting for: Data Link Confirm : NA Complete Application Fragment : Configurable at module start-up Application Confirm : Configurable at module start-up (1 to 65535 mSec) Complete Application Response : None	
Sends/Executes Control Operations: WRITE Binary Outputs : Never SELECT/OPERATE : Always DIRECT OPERATE : Always DIRECT OPERATE-NO ACK : Always Count > 1 : Always (1 to 65535) Pulse On : Always Pulse Off : Always Latch On : Always Latch Off : Always Queue : Never Clear Queue : Never	
Reports Binary Input Change Events when no specific variation requested: Only time-tagged	Reports time-tagged Binary Input Change Events when no specific variation requested: Binary Input Change with Time
Sends Unsolicited Responses: This is configurable at module start-up. If the number of events for the Binary or Analog Input Events is greater than 0, unsolicited responses are supported. Use the Enable/Disable Unsolicited function code from the DNP master for control.	Sends Static Data in Unsolicited Responses: Never
Default Counter Object/Variation: Object : 20 Variation : 5	Counters Roll Over at: 32 Bits
Sends Multi-Fragment Responses: Yes	

8.7 Internal Indication 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	Not Used
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.
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.

8.8 PTQ-DNPSNET Note

The PTQ-DNPSNET module differs from the PTQ-DNPSNET-Q Module in the following ways:

Specification	PTQ-DNPSNET	PTQ-DNPSNET-Q
Point Counts	0-500 word count to hold BI data	0-512 point count to hold BI data
	0-500 points of analog input data	0-512 points of analog input data
	0-250 points of counter data	0-128 points of floating-point format data
	0-500 word count to hold BO data	0-128 points of counter data
	0-500 points of analog output data	0-512 point count to hold BO data
		0-512 points of analog output data
		0-128 points of floating-point format data
Backplane Data Transfer	Small I/O	Large I/O
Class settings	N/A	Default class for binary input events (1-3)
		Default class for analog input events (1-3)
		Default class for float input events (1-3)
Flags	N/A	Return BI data with flag data
		Return BI events without time/date
		Return BO data without flag data (packed)
		Return counters with flag byte
		Return frozen counters with flag byte
		Return AI with flag byte

8.8.1 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 because it is intended to solve very specialized operations its size must be restricted.

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

The only time you should need it is if you are using 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.

8.8.3 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.

8.8.4 PTQ-DNPSNET Configuration

Note: This section describes how to configure the PTQ-DNPSNET module. Please refer to PTQ-DNPSNET-Q Configuration if you are configuring a PTQ-DNPSNET module with Quality Bits.
Important: You should plan your configuration before modifying the configuration files. The remainder of this step provides the information to 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 DNPSNET.CFG file consists of the following sections:

- [Backplane Configuration]
- [DNP ENET Slave]
- [DNP ENET IP Addresses]
- [Backplane Data Exchange]

Important notes to consider when editing the sample configuration file:

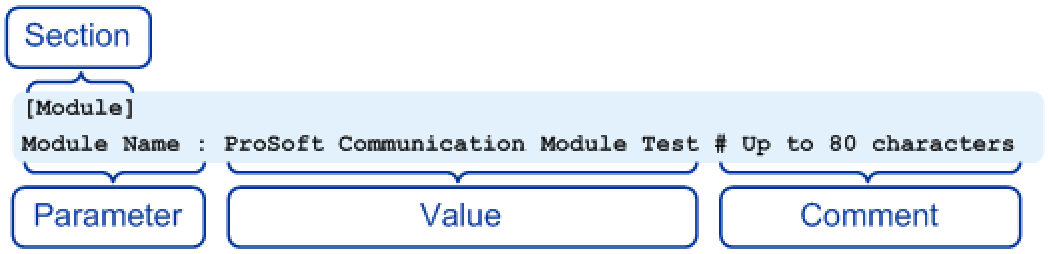
- Comments within the file are preceded by the pound (#) sign. Any text on a line that occurs after the # character will be ignored.
- Do not use tabs or other non-printing characters instead of spaces to separate parameters (spacebar).
- Parameter names must begin in the first column of a line, and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

Configuration File

The PTQ-DNPSNET-Q module stores its configuration in a text file called DNPSNET_Q.CFG, located in the module's flash memory. When the module starts up, it reads the configuration file and uses the information to control how the DNPSNET-Q protocol interacts with the module's application port(s).

The configuration file is arranged in *Sections,* with a heading in **[ ]** characters at the beginning of each section. Each *Section* contains a list of *Parameters* and *Values,* followed by an optional *Comment* that explains the parameter.

The following illustration shows an example of a *Section,* a *Parameter,* a *Value,* and a *Comment.*



```
[Module]
Module Name : ProSoft Communication Module Test # Up to 80 characters
```

The *Parameter* must be followed by a **[:]** (colon) character. The text following the **[:]** is a *Value.*

The module ignores *"comment"* text following the **[#]** character. Use comments to document your configuration settings.

You can get a sample configuration file for the module in the following places:

- Copy (page 153) the DNPSNET_Q.CFG from the module's flash memory to your PC
- Copy the DNPSNET_Q.CFG from the ProSoft Solutions CD-ROM supplied with the module
- Download the DNPSNET_Q.CFG from the ProSoft Technology web site at www.prosoft-technology.com

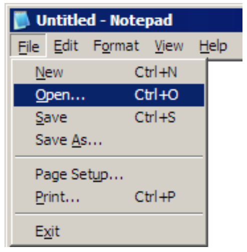
Editing the Configuration File

The DNPSNET_Q.CFG file is a plain ASCII text file. Use a text editor such as Notepad.exe (included with Microsoft Windows) to open and edit the file.

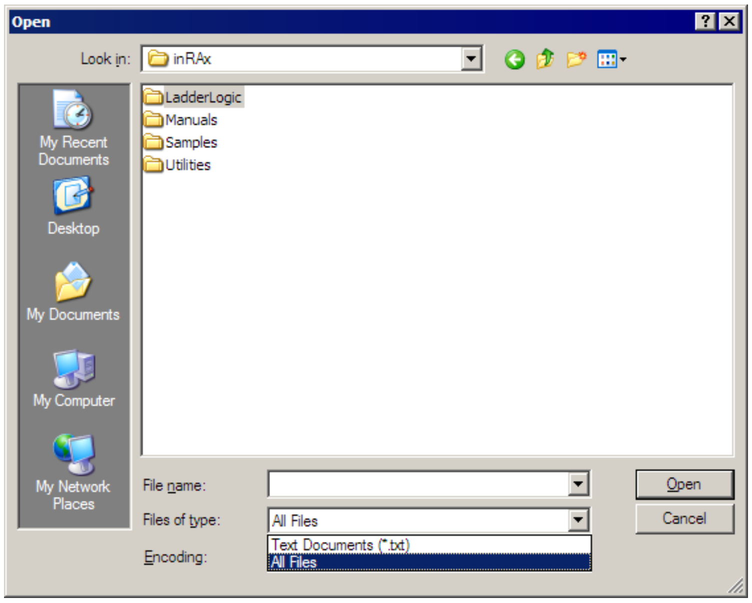
To open the configuration file in Notepad

1. Click the Start button, and then choose Programs
2. Expand the Programs menu, and then choose Accessories.
3. On the Accessories menu, choose **Notepad.**

4. In Notepad, open the File menu, and then choose Open



5 In the Open dialog box, select "All Files" in the Files of Type: dropdown list.



Tip: Sample configuration files are stored under the LadderLogic folder on the ProSoft Solutions CD-ROM.

6 Navigate to the folder containing the configuration file, and then select the file to edit.
7 Click Open to open the file.
8 When you have finished editing, save the file and close Notepad.

Important: Changes to the configuration file will not take effect until you download the file to the module, and then reboot the module.

[Backplane Configuration]

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 [Backplane Configuration] section:

```
[Backplane Configuration]
Module Name: ProTalk-DNPSNET COMMUNICATION MODULE
Failure Flag Count: 0 #Determines if BP failure will cause protocol to be
 #disabled (0=Ignore, >0 = failure count to disable)
Error Offset: 8000 #Location of where to write status data (-1=disable)
Initialize Output Data: N #N=No, Y=Yes read output values from controller
```

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.

Failure Flag Count

0 through 65535

This parameter specifies the number of successive transfer errors that must occur before the communication ports are shut down. If the parameter is set to 0, the communication ports will continue to operate under all conditions. If the value is set larger than 0 (1 to 65535), communications will cease if the specified number of failures occur.

Error Offset

```
Error Offset : 8000 #Location of where to write status data (-1=disable)
```

The Error Offset parameter specifies the register location in the module's database where module status data will be stored. If a value less than 0 is entered, the data will not be stored in the database. If the value specified is in the range of 0 to 8966, the data will be placed in the modules database. A value of -1 = disable.

Initialize Output Data

Yes or No

This parameter determines if the output data for the module should be initialized with values from the processor. If the value is set to No (0), the output data will be initialized to 0. If the value is set to Yes (1), the data will be initialized with data from the processor. Use of this option requires associated ladder logic to pass the data from the processor to the module.

[DNP ENET 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.

Note: A limitation of the DNP slave driver is that all points defined in the module slave database must fit within one Class 0 poll. The maximum packet size for a Class 0 poll is 2048 bytes. A DNP Message Size Calculator is available on the ProSoft Technology web site. This calculator will help you ensure that the packet size fits within this requirement.

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

```
# This section defines the configuration for the Module.
# port. This port will receive requests from a remote DNP master unit.
#
[DNP ENET Slave]
Internal Slave ID: 2 #0-65534 slave identification code for this unit
Use IP List: N #Use IP list to validate connection (N=No, Y=Yes)
# DNP database definition
# Please Note. The databases are in the memory of the module in this sequence
# and are placed directly adjacent to each other. In other words when you
# change the size of a # database you must adjust the transfer commands to
# accommodate the new location.
Binary Inputs: 100 #0-500 word count to hold BI data
Analog Inputs: 100 #0-500 points of analog input data
Counters: 50 #0-250 points of counter data
Binary Outputs: 100 #0-125 word count to hold BO data
Analog Outputs: 100 #0-500 points of analog output data
# DNP specific parameters
AI Deadband: 1000 #0-32767 analog deadband value for events
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
App Layer Confirm Tout: 2000 #1-65535 milliseconds App Layer confirm timeout
Unsolicited Response: N #Generate Unsolicited responses (N=No,Y=Yes)
Class 1 Unsol Resp Min: 10 #1-255 min number of events before send
Class 2 Unsol Resp Min: 10 #1-255 min number of events before send
Class 3 Unsol Resp Min: 10 #1-255 min number of events before send
Unsol Resp Delay: 5000 #0-65535 milliseconds before events sent
UResp Master Address: 5 #DNP address of master to send UResp data
AI Events with time: Y #timestamp AI Event data default (N=No,Y=Yes)
Time Sync Before Events: Y #timesync module before events gen (N=No,Y=Yes)
```

Modify each parameter based on the needs of your application:

Internal Slave ID

0 to 65534

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

Use IP List

```
Use IP List : N #Use IP list to validate connection (N=No, Y=Yes)
```

This parameter specifies if the IP address of the host connected to the system will be validated. If the parameter is set to N, any host may connect to the unit. If the parameter is set to Y, only hosts in the IP list will be permitted to connect to the module. All other IP addresses will be ignored by the module and the module will issue a RST to the TCP/IP connection. This example shows the parameter set to No.

DNP Database Definition Note: The databases are in the memory of the module in this sequence and are placed directly adjacent to each other. In other words when you change the size of a database you must adjust the transfer commands to accommodate the new location.

Binary Inputs

```
Binary Inputs : 100 #0-500 word count to hold BI data
```

This parameter specifies the number of digital input points to configure in the DNP slave device based on a word count. Each word stores 16 points. Therefore, if the parameter is set to 2, 32 binary inputs will be defined for the application. The valid range is 0 to 500 words. This example shows the parameter set to 100 words.

Analog Inputs

```
Analog Inputs : 100 #0-500 points of analog input data
```

This parameter sets the number of analog input points to configure in the DNP slave device. Each point will occupy a one-word area in the module memory. Valid values are 0 to 500 points. This example shows the parameter set to 100 points of analog input data.

Counters

```
Counters : 50 #0-250 points of counter data
```

This parameter sets the 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. Valid values are 0 to 250 points. This example show the parameter set to 50 points of counter data.

Binary Outputs

```
Binary Outputs : 100 #0-125 word count to hold BO data
```

This parameter sets the number of digital output words to configure in the DNP slave device based on a word count. Each word stores 16 points. Therefore, if the parameter is set to 2, 32 binary outputs will be defined for the application. Valid values are 0 to 500 words to hold Binary Output data. This example shows the parameter set to 100 words.

Analog Outputs

```
Analog Outputs : 100 #0-500 points of analog output data
```

This parameter sets the number of analog output points to configure in the DNP slave device. Each point will occupy a one-word area in the module memory. Valid values are 0 to 500 points of analog output data. This example shows the parameter set to 100.

AI Deadband

0 to 32767 data units

This value sets the global deadband for all analog input points. When the current value for an analog input point is not within the deadband limit set based on the last event for the point, an event will be generated.

Select/Operate Arm Time

1 to 65535 milliseconds

This parameter sets the 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. Valid arm timeout values are 1 to 65535 milliseconds. This example shows the value set to 2000 milliseconds.

Write Time Interval

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

This parameter sets the time interval to set the need time IIN bit (0=never), which will cause the master to write the time. Stored in milliseconds in the module memory. Valid values are 0 to 1440. This example shows the value set to 60 milliseconds.

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. Valid values are 1 to 65535 milliseconds. This example shows the value set to 2000 milliseconds.

Unsolicited Response

```
Unsolicited Response: N #Generate Unsolicited responses (N=No,Y=Yes)
```

This parameter is set if the slave unit will send unsolicited response messages. If set to N, the slave will not send unsolicited responses. If set to Y, the slave will send unsolicited responses. This example shows the parameter set to No.

Class 1 Unsol Resp Min

1 to 255 events

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

Class 2 Unsol Resp Min

1 to 255 events

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

Class 3 Unsol Resp Min

1 to 255 events

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

Unsol Resp Delay

```
Unsol Resp Delay: 5000 #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. Valid values are 0 to 65535 milliseconds. This example show the parameter set to 5000 milliseconds.

Uresp Master Address

0 to 65534

DNP destination address where unsolicited response messages are sent.

AI Events with Time

```
AI Events with time: Y #timestamp AI Event data default (N=No,Y=Yes)
```

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 N, the default is set to no time data. If the parameter is set to Y, the default object will include the time of the event. This example shows the parameter set to Yes.

Time Sync Before Events

```
Time Sync Before Events: Y #timesync module before events gen (N=No,Y=Yes)
```

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 N, no events will be generated until the module's time has been synchronized. If the parameter is set to Y, events will always be generated. This example shows the parameter set to Yes.

[DNP ENET IP Addresses]

This section of the configuration file only applies if the directive labeled **Use IP List** is set to Yes or Y. If **Use IP List** is enabled, the module will refuse to answer a request unless the IP address of the client is listed in this section. This section may contain no more then 10 addresses.

```
[DNP ENET IP ADDRESSES]
START
192.168.0.203
192.168.0.55
192.168.0.26
END
```

[Backplane Data Exchange]

Before modifying the [Backplane Data Exchange] section of the configuration file, you must understand some important concepts. The following topics describe these concepts.

If you have used the parameters defined in the [Module] section, you have created the following memory map. We will use this map to explain how data transfer works between the processor and the ProTalk module.

PTQ Memory Address		Application Memory Address
0	=	0
10	=	10
20	=	20
30	=	30
40	=	40
50	=	50
…		…
206	=	206
207	=	207
…	=	…
312	=	312

A thorough understanding of the information contained in this section is required for successful implementation of the module in a user application.

Data Transfer

The module uses a concept referred to as "Command Functions". The command functions reside in the [Backplane Data Exchange] section of the configuration file. This method of data transfer is probably different from other methods you might have used, but does offer some unique advantages:

- The amount of ladder logic required will be substantially reduced; in fact in many applications no ladder logic will be required.
- The module may be placed in any position in the chassis containing the PLC and will operate with no modifications.

Defining Data to be Sent to the PTQ Database

You might be asked to provide access to 207 words of information for other devices on the network. This information resides in the PLC at addresses 400001 to 400207 and you must make this the first 207 words of the database inside the module. This would require the use of "**Command Function 1**".

Because the total amount of data exceeds the maximum length of any single command function, you will need two entries in the [Backplane Data Exchange] section of your configuration file.

This might look like the following:

```
[Backplane Data Exchange]
# Cmd PTQ Point QUANTUM Word
# Type Address Type Address Count
START
 1 0 4 1 100 # move data from Quantum to the PTQ
 1 100 4 101 107 # move data from Quantum to the PTQ
END
```

The first command states:

Field	Value	Meaning
CMD TYPE	1	The type of operation to perform 1 = Read data from the Quantum into the PTQ
PTQ Address	0	The destination address within the PTQ
Point Type	4	The range of registers to read from the Quantum 4 = 40:x style register
Quantum Address	1	The starting address of the data within the Quantum This would be Point Type + offset Example: 40000 + 1 = 40001
Word Count	100	The number of registers to transfer

The second command states:

Field	Value	Meaning
CMD TYPE	1	The type of operation to perform 1 = Read data from the Quantum into the PTQ
PTQ Address	100	The destination address within the PTQ
Point Type	4	The range of registers to read from the Quantum 4 = 40:x style register
Quantum Address	101	The starting address of the data within the Quantum This would be Point Type + Quantum Address Example: 40000 + 101 = 40101
Word Count	107	The number of registers to transfer

The following diagram shows the result of this example.

Quantum Memory Address		PTQ Memory Address
40001		0
40010	First Command	9
40020		19
40030		29
40040		39
…		
40060	→	59
40070		69
40080		79
40090		89

Quantum Memory Address		PTQ Memory Address
40100		99
40101		100
40111	Second Command	110
…	→	
40200		199
40207		206
…		…

Defining Data to be Retrieved from the PTQ Database.

Your application may need to retrieve 105 words of data from other devices on the network These devices have either sent you the data if you are a slave, or you have obtained it for your use if you happen to be a master in your application. Assuming that the data resides in registers 207 to 312 within the PTQ modules memory and you wish to place this information in addresses 400208 to 400313 within the Quantum you could use **Command Function 2** to transfer the information.

Because the total amount of data does not exceed 130 words in length, a single command can be used to transfer the data. This additional entry will be added to our [Backplane Data Exchange] section and it would look like the third command below:

```
[Backplane Data Exchange]
# Cmd PTQ Point QUANTUM Word
# Type Address Type Address Count
START
 1 0 4 1 100 # move data from Quantum to the PTQ
 1 100 4 101 107 # move data from Quantum to the PTQ
 2 207 4 208 105 # move data from PTQ to the Quantum
END
```

The third command states:

Field	Value	Meaning
CMD TYPE	2	The type of operation to perform 2 = Write data from the PTQ to the Quantum
PTQ Address	207	The destination address within the PTQ
Point Type	4	The range of registers to read from the Quantum 4 = 40:x style register
Quantum Address	208	The starting address of the data within the Quantum This would be Point Type + Quantum Address Example: 40000 + 207 = 40207
Word Count	105	The number of registers to transfer

The following diagram shows the result of this example:

Quantum Memory Address		**PTQ Memory Address**
40001		0
40010	First Command	9
40020		19
40030		29
40040		39
…		
40060	→	59
40070		69
40080		79
40090		89
40100		99
40101		100
40111	Second Command	110
…	→	
40200		199
40207		206
…		…
40208		207
40210	Third Command	209
…	←	…
40310		309
40313		312

Defining Special 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 might be used.

Assuming that you have chosen registers 400500 to 400563 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]
# Cmd PTQ Point QUANTUM Word
# Type Address Type Address Count
START
 1 0 4 1 100 # move data from Quantum to the PTQ
 1 100 4 101 107 # move data from Quantum to the PTQ
 2 207 4 208 105 # move data from PTQ to the Quantum
 3 0 4 500 64 # Special Function 3
END
```

The fourth command states:

Field	Value	Meaning
CMD TYPE	3	The type of operation to perform 3 = Read/Write special function to the Quantum.
PTQ Address	0	This is ALWAYS 0 and will not overwrite your database.
Point Type	4	The range of registers to read from the Quantum 4 = 40:x style register
Quantum Address	500	The starting address of the data within the Quantum This would be Point Type + Quantum Address Example: 40000 + 500 = 40500
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 "Quantum Address" registers, process the information, clear the registers and post the status if applicable.

The following diagram shows this example:

Quantum Memory Address		PTQ Memory Address
40001		0
40010	First Command	9
40020		19
40030		29
40040		39
…		
40060	→	59
40070		69
40080		79
40090		89
40100		99
40101		100
40111	Second Command	110
…	→	
40200		199
40207		206
…		…
40208		207
40210	Third Command	209
…	←	…
40310		309
40313		312
40500		N/A
40510	Fourth Command	N/A

Quantum Memory Address		PTQ Memory Address
40520	1st Scan	N/A
40530		N/A
40540	2nd Scan	N/A
40550	←	N/A
40560		N/A
40563		N/A

Implementing Ladder to Support Special Functions.

The previous discussions about Command Function 1 and Command Function 2 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
MyFunction3 is an alias for register 400500
MyData1-MyData63 are aliases for 400501-400563
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 and so on... values for time and date represent aliases for
your time source.
MyTrigger is an alias for register 401000.
*)
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 and so on... values for time and date are representative of
your aliases for your time source.
MyTrigger is an alias for register 400010.
*)
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 400501 to 400563 as required.
2 As your last step, copy the BLOCK ID number of the special function into register 400500.
3 Clear your permissive condition.

The module will read the data in and either clear the registers in the array, or return the requested data and clear the **Block ID** register (400500 in this example).

Modify the [Backplane Data Exchange] Section

The previous sections provided an overview of what is required to modify the [Backplane Data Exchange] section. With this information, you are now ready to make the necessary modifications to the configuration file to work with your application.

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

```
# This section is used by the PTQ module to define the data transferred
# between the module and processor.
#
# Cmd Type --> 0=Disable
# 1=Quantum to PRQ (Read from Quantum)
# 2=PTQ to Quantum (Write to Quantum)
# 3=Control data block for module
# DB Address --> address of starting word in database
# Point Type --> 0=0:x
# 1=1:x
# 3=3:x
# 4=4:x
# Point Address --> point address (1 based)
# (0x and 1x must be at start of word (that is, 1, 17, 33, ...))
# Word Count --> number of words to transfer (1 to 130)
# CMD TYPE is ALWAYS 64 words in length
[Backplane Data Exchange]
# Cmd PTQ Point QUANTUM Word
# Type Address Type Address Count
START
 1 0 4 1 100 # move data from Quantum to the PTQ
 1 100 4 101 107 # move data from Quantum to the PTQ
 2 207 4 208 105 # move data from PTQ to the Quantum
END
```

This example shows an application that reads 207 words from the Quantum to the module and writes 105 words from the module to the Quantum.

The [Backplane Data Exchange] section is a series of messages that instruct the module how to transfer data to/from the Quantum. What is missing from the message is the ability to schedule its execution. This ability is normally left to the programmer in the PLC environment, however in the PTQ module this is not included so that the commands may run as fast as possible to maintain the synchronization of the two databases. One command from the list will execute during each I/O service interval at the end of the PLC ladder logic evaluation. So as an example if your configuration contains 10 "Backplane Data Exchange" commands it will require 10 PLC scans to process the entire list.

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

Set Up Command Function 1 (Read data from the Quantum)

This section provides information on how to read data from the Quantum into the module.

Command Function 1 (one) is designed to transfer data from the Quantum to the module on a continuous basis. The command(s) required to transfer your application data should be entered in the [BACKPLANE DATA EXCHANGE] section of your configuration file as required.

This command takes the following parameters:

- **Command type:** 1 (Read data from the Quantum)
- **PTQ Database Address:** The destination for the data retrieved from the Quantum.
- **Point Type:** The type of register within the Quantum (0:x = 0, 10:x = 1 30:x = 3 or 40:x = 4)
- **Quantum Address:** The source of the data within the Quantum. 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:** 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 30:x or 40:x Register Transfer

The following **example** shows a typical command used to read 40:x data from the Quantum. In this **example,** registers 400001 to 400099 from the Quantum will be transferred to registers 0 to 99 within the module.

```
# Word Count Number of words to transfer (1 to 130)
#
# Cmd PTQ Point Quantum Word
# Type Address Type Address Count
[Backplane Data Exchange]
START
 1 0 4 1 100
END
```

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 Quantum to word 21 in the module you would have to transfer the **word** within the Quantum containing bits 000001 to 000016 to a word within the modules memory.

Take care with the transfer of bits while planning the application so as to optimize the usage of the available bits and to preserve the integrity of your information.

The following **example** shows how this could be accomplished.

```
# Cmd PTQ DB Point Quantum Word
# Type Address Type Address Count
START
 1 21 0 1 1
END
```

Set Up Command Function 2 (Write data to the Quantum)

This section provides information on how to write data from the module to the Quantum.

Command Function 2 (two) is designed to transfer data from the module to the Quantum on a continuous basis. The command(s) required to transfer your application data should be entered in the [BACKPLANE DATA EXCHANGE] section of your configuration file as required.

This command takes the following parameters:

- **Command type:** 2 (Write data to the Quantum)
- **PTQ Database Address:** The source of the data within the PTQ to be sent to the Quantum.
- **Point Type:** The type of register within the Quantum (0:x = 0, 10:x = 1 30:x = 3 or 40:x = 4)
- **Quantum Address:** The destination register within the Quantum. The address is expressed without the use of the register range, for example 400001 would be entered as 1 (400001 to 400000 = 1 or 40001 to 40000 = 1)
- **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 30:x or 40:x Register Transfer

The following **example** shows a typical command used to write to the 40:x registers within the Quantum. In this **example,** registers 207 to 312 from the PTQ will be transferred to registers 400208 to 400313 within the Quantum.

```
# Word Count Number of words to transfer (1 to 130)
#
# Cmd PTQ DB Point Quantum Word
# Type Address Type Address Count
[Backplane Data Exchange]
START
   2 207 4 208 105
END
```

Example 0:x or 10:x Register Transfer

The transfer of Coils and Inputs require some forethought as the command transfers **words** and not bits. This means that if you wanted to transfer the **word** containing the bits 805 to 806 from the module to the Quantum you would transfer the entire 50th word of the modules memory into the destination register in the Quantum. The following command transfer bits 800 to 815 (Word x Bits = Bit Address or 50 * 16 = 800) from the modules memory to word 1 of the coils (000001 to 000016) within the Quantum.

Take care with the transfer of bits while planning the application so as to optimize the usage of the available bits and to preserve the integrity of your information.

The following example shows how this could be accomplished.

```
# Cmd PTQ DB Point Quantum Word
# Type Address Type Address Count
START
 1 49 0 1 1
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 with all modules to implement the following functionality:

- Force a reboot of the PTQ module (Special Function 9998 or 9999 available on all products)
- Set / Retrieve Time and Date (DNP and IEC only!)
- Register events with the protocol (DNP and IEC only!)

Other modules may implement additional functionality, which will be described in the Special Functions section of this manual.

This command takes the following parameters:

- **Command type:** 3 (Write data to the Quantum)

PTQ Database 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 Quantum (0:x = 0, 10:x = 1 30:x = 3 or 40:x = 4)
- **Quantum Address:** The source register within the Quantum. 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. Care should be taken to assure that 64 words of memory are available within the Quantum.

Example 30:x or 40:x Register Transfer

The following **example** shows a typical command used to retrieve a special function command from the Quantum. In this **example,** registers 400500 to 400563 from the Quantum will be used to provide the information required by the module.

```
# Word Count Number of words to transfer (1 to 130)
#
# Cmd PTQ DB Point Quantum Word
# Type Address Type Address Count
[Backplane Data Exchange]
START
   3 0 4 500 64
END
```

The following section shows the functions that may be performed by using the Command Function 3.

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, the contents of the block would look as follows:

Word Offset In Block	Example Address	Data	Description
0	400500	Block ID	This field contains the value of 9958 identifying the event block to the module
1	400501	Event Count	This filed contains the number of events in the block. Valid values for this field are 1 to 12
2	400502	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	DNP Binary Input Data Point	This is the data point in the DNP binary input database represented by the event.

Word Offset In Block	Example Address	Data	Description
4	400504	Month/Day/State	Formatted: bits 0 to 4 = Minutes, bits 8 to 11 = Month, bit 15 = digital state for point. All other bits are ignored.
5	400505	Hour/Minute	Formatted: bits 0 to 5 = minutes, bits 8 to 12 = hour, All other bits are ignored.
6	400506	Sec/Millisecond	Formatted: bits 0 to 9 = milliseconds, bits 10 to 15 = seconds
7	400507	Year	This is the four digit year for the event
8 to 12	400508 to 400512		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		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	Example Address	Data	Description
0	400500	0	If it completed successfully
1	400501	Block Id	9958
2	400502	Event Count	This field contains the number of events processed by the module.
3	400503	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, the contents of the block would look as follows:

Word Offset in Block	Example Address	Data	Description
0	400500	Block ID	This field contains the value of 9959 identifying the event block to the module
1	400501	Event Count	This filed contains the number of events in the block. Valid values for this field are 1 to 12
2	400502	Sequence Counter	This field holds the sequence counter for each 9959 block transfer. This synchronizes and confirms receipt of the block by the module.

Word Offset in Block	Example Address	Data	Description
3	400503	DNP Analog Input Data Point	This is the data point in the DNP Analog Input database represented by the event.
4	400504	Month/Day/State	Formatted: bits 0 to 4 = Minutes, bits 8 to 11 = Month, bit 15 = digital state for point. All other bits are ignored.
5	400505	Hour/Minute	Formatted: bits 0 to 5 = minutes, bits 8 to 12 = hour, All other bits are ignored.
6	400506	Sec/Millisecond	Formatted: bits 0 to 9 = milliseconds, bits 10 to 15 = seconds
7	400507	Year	This is the four digit year for the event
8 to 12	400508 to 400512		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		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	Example Address	Data	Description
0	400500	0	If completed successfully
1	400501	Block Id	9959
2	400502	Event Count	This field contains the number of events processed by the module.
3	400503	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 PLC's Time using the module

If the module retrieves a BLOCK ID of 9970 from the PLC 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 PLC. Using the example data buffer of 400500 to 563, the contents of the block would look as follows:

Word Offset in Block	Example Address	Data	Description
0	400500	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	Example Address	Data Field(s)	Description
0	400500	0	If completed successfully
1	400501	Block ID	9970.
2	400502	Year	This field contains the four-digit value to be used with the new time value.
3	400503	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	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	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	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	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	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	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 PLC's Time

If the module retrieves a BLOCK ID of 9971 from the PLC 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, the contents of the block would look as follows:

Word Offset in Block	Example Address	Data Field(s)	Description
0	400500	Block ID	9971.
1	400501	Year	This field contains the four-digit value to be used with the new time value.
2	400502	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	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	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	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	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	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	**Example Address**	**Data Field(s)**	**Description**
0	400500	0	If completed successfully
1	400501	Block ID	9971

Block ID 9998 or 9999 - Reboot Module

If the Quantum 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	**Example Address**	**Data Field(s)**	**Description**
0	400500	9998 or 9999	Block ID to reboot module

Uploading and Downloading the Configuration File

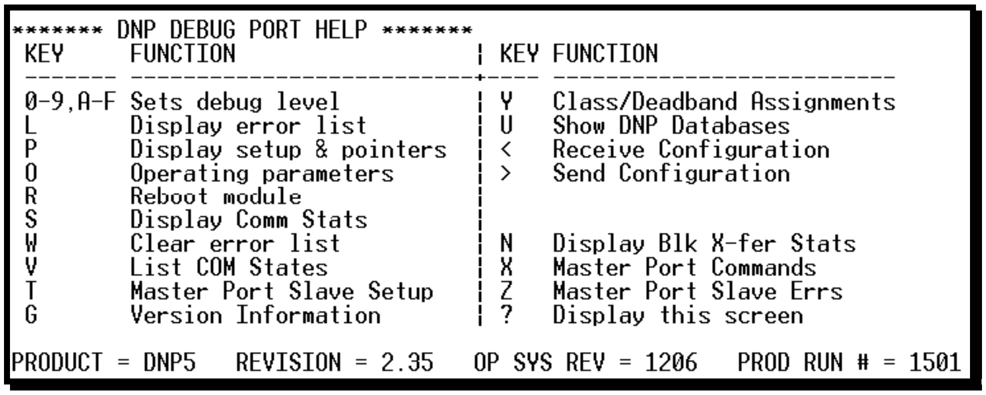
ProSoft modules are shipped with a pre-loaded configuration file. In order to edit this file, you must transfer the file from the module to your PC. After editing, you must transfer the file back to the module.

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.

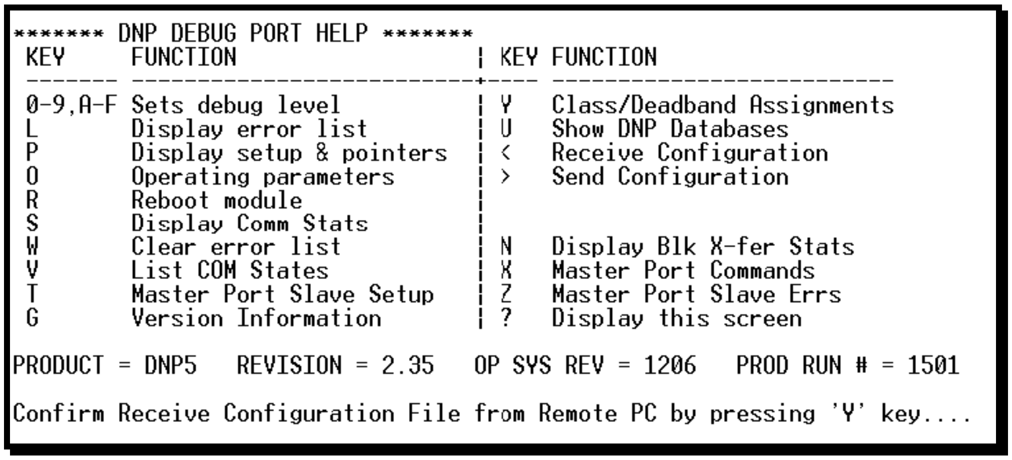
Transferring the Configuration File to Your PC

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.



```
******* 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         |
 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 = DNP5   REVISION = 2.35   OP SYS REV = 1206   PROD RUN # = 1501
```

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

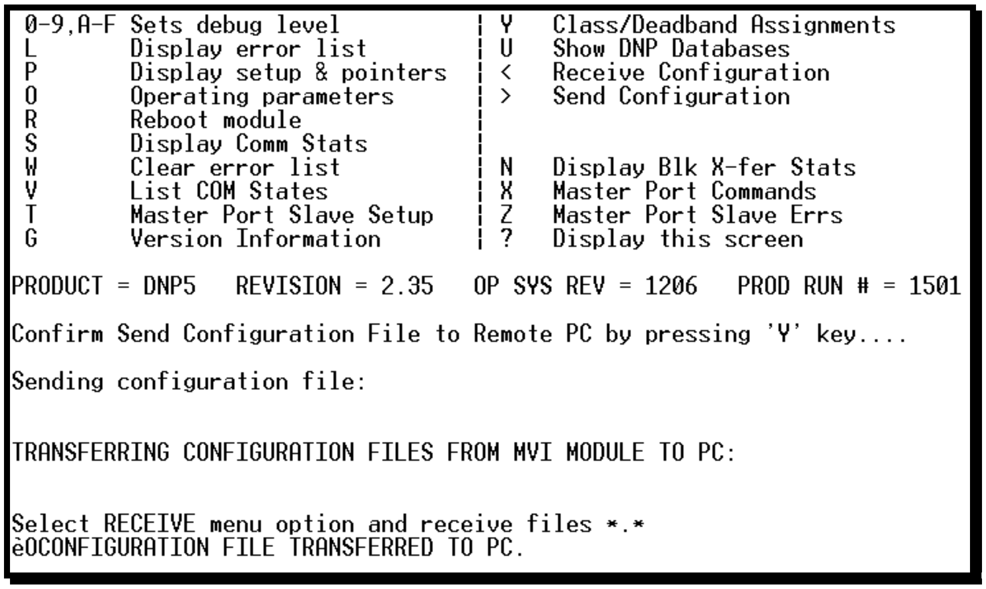


```
******* DNP DEBUG PORT HELP *******
 KEY     FUNCTION                   | KEY FUNCTION
 ------- ---------------------------+---- -------------------------
 0-9,A-F Sets debug level           | Y   Class/Deadband Assignments
 L       Display error list         | U   Show DNP Databases
 P       Display setup & pointers   | <   Receive Configuration
 O       Operating parameters       | >   Send Configuration
 R       Reboot module              |
 S       Display Comm Stats         |
 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 = DNP5   REVISION = 2.35   OP SYS REV = 1206   PROD RUN # = 1501

Confirm Receive Configuration File from Remote PC by pressing 'Y' key....
```

3 Press **[Y].** The module will automatically start a Zmodem file transfer. The configuration file will be stored in the default file transfer folder on your PC.

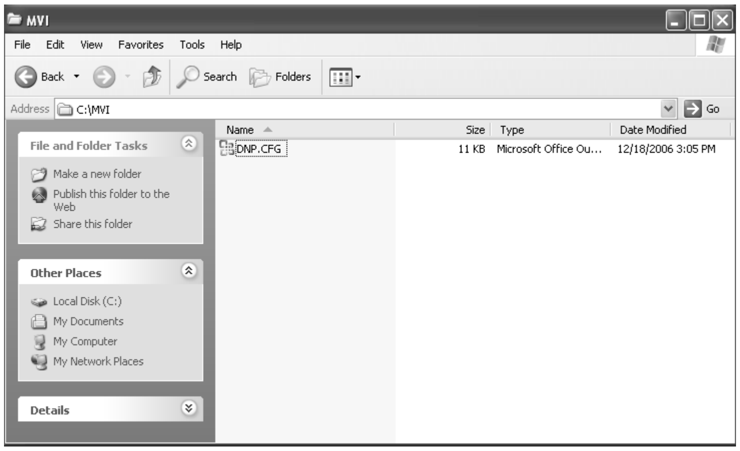
▪ Note: ProSoft Technology suggests that you download the configuration file pre-loaded on your module. However, configuration files are also available on the ProSoft CD as well as the ProSoft Technology web site at www.prosoft-technology.com.

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



```
 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         |
 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 = DNP5   REVISION = 2.35   OP SYS REV = 1206   PROD RUN # = 1501

Confirm Send Configuration File to Remote PC by pressing 'Y' key....

Sending configuration file:


TRANSFERRING CONFIGURATION FILES FROM MVI MODULE TO PC:


Select RECEIVE menu option and receive files *.*
èOCONFIGURATION FILE TRANSFERRED TO PC.
```

The configuration file is now in a folder on your PC. To view the location of this folder, open the Transfer menu in Hyperterminal and choose Receive File.



Name	Size	Type	Date Modified
DNP.CFG	11 KB	Microsoft Office Ou...	12/18/2006 3:05 PM

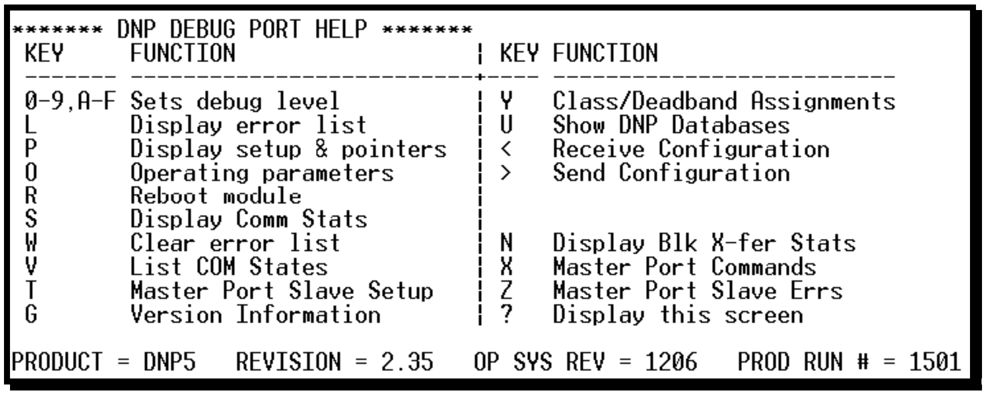
4 You can now open and edit the file in a text editor such as Notepad. When you have finished editing the file, save it and close Notepad.

Important: You must name this file DNP.CFG before you transfer it to the module. The module will not recognize configuration files with any other name or extension.

Transferring the Configuration File to the Module

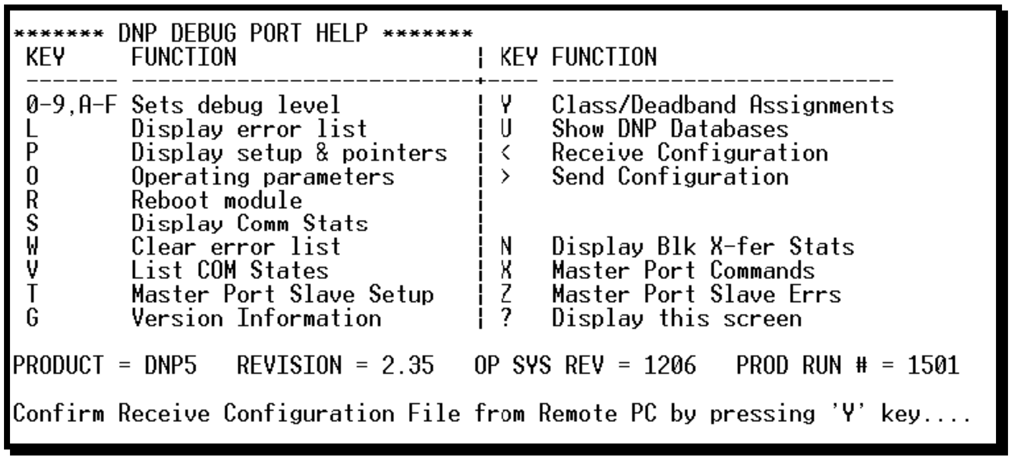
Perform the following steps to transfer a configuration file from your PC to the module.

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.



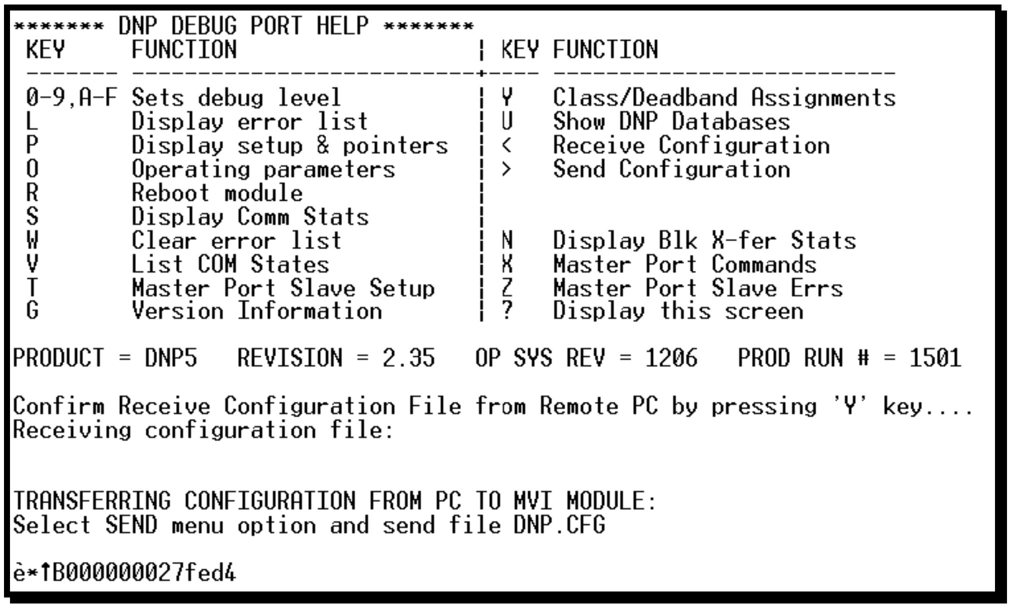
```
******* DNP DEBUG PORT HELP *******
 KEY     FUNCTION                   | KEY FUNCTION
 ------- ---------------------------+---- -------------------------
 0-9,A-F Sets debug level           | Y   Class/Deadband Assignments
 L       Display error list         | U   Show DNP Databases
 P       Display setup & pointers   | <   Receive Configuration
 O       Operating parameters       | >   Send Configuration
 R       Reboot module              |
 S       Display Comm Stats         |
 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 = DNP5   REVISION = 2.35   OP SYS REV = 1206   PROD RUN # = 1501
```

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



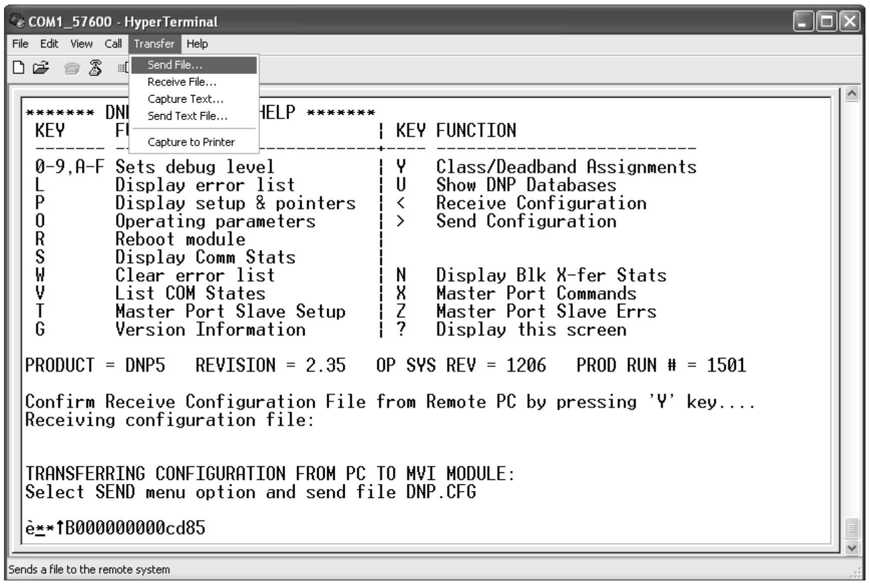
```
******* DNP DEBUG PORT HELP *******
 KEY     FUNCTION                  | KEY FUNCTION
 ------- --------------------------+---- -------------------------
 0-9,A-F Sets debug level          | Y   Class/Deadband Assignments
 L       Display error list        | U   Show DNP Databases
 P       Display setup & pointers  | <   Receive Configuration
 O       Operating parameters      | >   Send Configuration
 R       Reboot module             |
 S       Display Comm Stats        |
 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 = DNP5   REVISION = 2.35   OP SYS REV = 1206   PROD RUN # = 1501

Confirm Receive Configuration File from Remote PC by pressing 'Y' key....
```

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



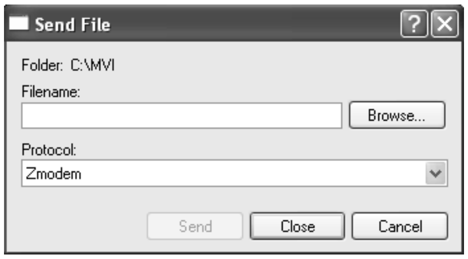
```
******* DNP DEBUG PORT HELP *******
 KEY     FUNCTION                  | KEY FUNCTION
 ------- --------------------------+---- -------------------------
 0-9,A-F Sets debug level          | Y   Class/Deadband Assignments
 L       Display error list        | U   Show DNP Databases
 P       Display setup & pointers  | <   Receive Configuration
 O       Operating parameters      | >   Send Configuration
 R       Reboot module             |
 S       Display Comm Stats        |
 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 = DNP5   REVISION = 2.35   OP SYS REV = 1206   PROD RUN # = 1501

Confirm Receive Configuration File from Remote PC by pressing 'Y' key....
Receiving configuration file:


TRANSFERRING CONFIGURATION FROM PC TO MVI MODULE:
Select SEND menu option and send file DNP.CFG

è*↑B000000027fed4
```

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


COM1_57600 - HyperTerminal
File Edit View Call Transfer Help
Send File...
Receive File...
Capture Text...
Send Text File...
Capture to Printer
Sends a file to the remote system

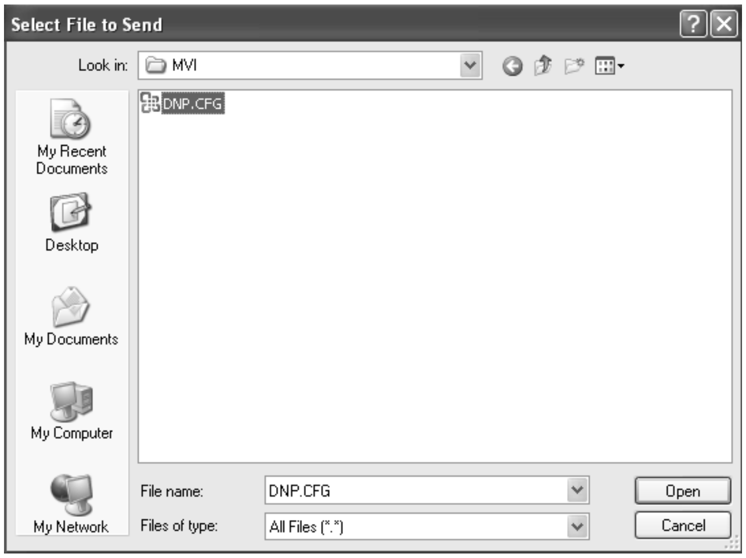
```
******* DN                 HELP *******
 KEY     F                             | KEY FUNCTION
 -------  -                ------------+---- --------------------------
 0-9,A-F Sets debug level              | Y   Class/Deadband Assignments
 L       Display error list            | U   Show DNP Databases
 P       Display setup & pointers      | <   Receive Configuration
 O       Operating parameters          | >   Send Configuration
 R       Reboot module                 |
 S       Display Comm Stats            |
 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 = DNP5   REVISION = 2.35   OP SYS REV = 1206   PROD RUN # = 1501

Confirm Receive Configuration File from Remote PC by pressing 'Y' key....
Receiving configuration file:


TRANSFERRING CONFIGURATION FROM PC TO MVI MODULE:
Select SEND menu option and send file DNP.CFG

è**↑B000000000cd85
```

The Send File dialog appears.



Folder: C:\MVI
Filename:
Protocol: Zmodem

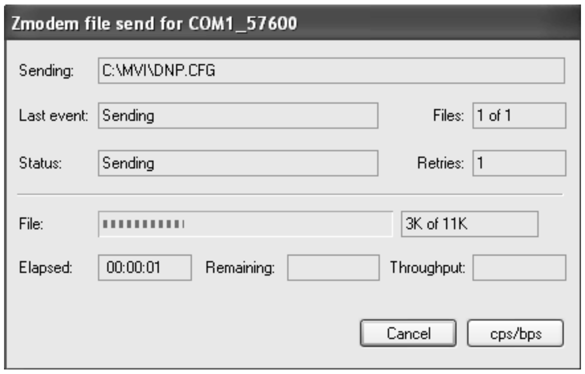
5 Use the Browse button to locate the configuration file your computer.



File name: DNP.CFG
Files of type: All Files (*.*)

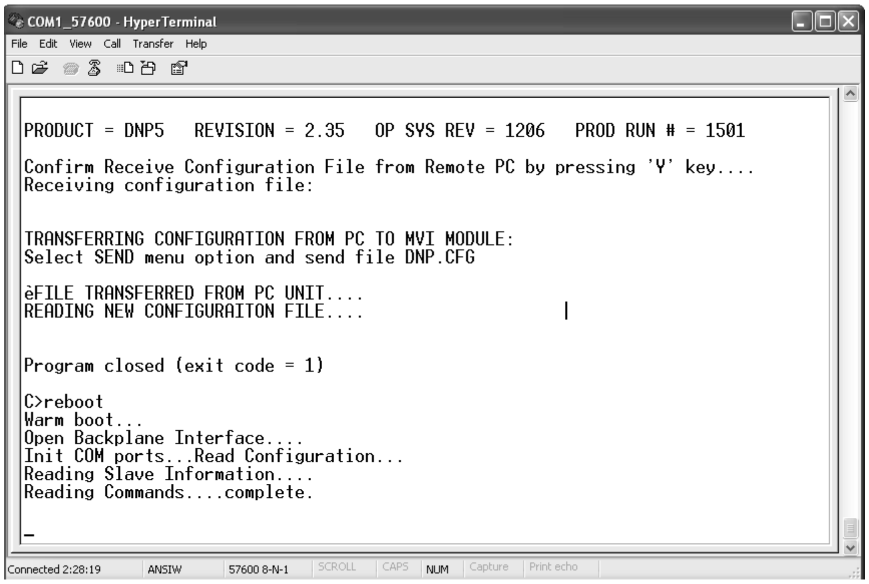
Note: This procedure assumes that you are uploading a newly edited configuration file from your PC to the module. However, configuration files are also available on the ProSoft CD as well as the ProSoft Technology web site.

6 Select Zmodem as the protocol.
7 Click the Send button. This action opens the Zmodem File Send dialog box.



Sending: C:\MVI\DNP.CFG
Last event: Sending
Files: 1 of 1
Status: Sending
Retries: 1
File: 3K of 11K
Elapsed: 00:00:01
Remaining:
Throughput:

When the upload is complete, the screen indicates that the module has reloaded program values and displays information about the module.


COM1_57600 - HyperTerminal
File Edit View Call Transfer Help
Connected 2:28:19 ANSIW 57600 8-N-1 SCROLL CAPS NUM Capture Print echo

```
PRODUCT = DNP5   REVISION = 2.35   OP SYS REV = 1206   PROD RUN # = 1501

Confirm Receive Configuration File from Remote PC by pressing 'Y' key....
Receiving configuration file:


TRANSFERRING CONFIGURATION FROM PC TO MVI MODULE:
Select SEND menu option and send file DNP.CFG

èFILE TRANSFERRED FROM PC UNIT....
READING NEW CONFIGURAITON FILE....


Program closed (exit code = 1)

C>reboot
Warm boot...
Open Backplane Interface....
Init COM ports...Read Configuration...
Reading Slave Information....
Reading Commands....complete.

_
```

8 Your module now contains the new configuration. Press **[?]** to see the module's main menu.

8.9 DNPSNET-Q 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 DNPSNET-Q configuration file you should create sufficient data size for your application plus any anticipated growth. For example, if 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 one or more commands 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

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
 o Module Operation
 o Configuration/Debug status information
 o 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 How to Contact Us: Technical Support

Internet	Web Site: www.prosoft-technology.com/support
	E-mail address: support@prosoft-technology.com

Asia Pacific

+603.7724.2080, 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
Languages spoken include: French, English

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

+1.661.716.5100, 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
Languages spoken include: Portuguese, English

9.2 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.

9.2.1 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 161). 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, and so on.

9.2.2 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.

9.2.3 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

9.3 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.

9.3.1 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 warranteed 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.

9.3.2 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.

9.3.3 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 of 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.

9.3.4 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.

9.3.5 Disclaimer of all Other Warranties

The Warranty set forth in What Is Covered By This Warranty (page 163) 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.

9.3.6 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.

9.3.7 Time Limit for Bringing Suit

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

9.3.8 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.

9.3.9 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.

9.3.10 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.

Index