



Where Automation Connects.



## MVI56-DNPSNET

ControlLogix® Platform

DNP 3.0 Server over Ethernet  
Communication Module

October 14, 2020

**USER MANUAL**

## 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 our products, documentation, or support, please write or call us.

## How to Contact Us

**ProSoft Technology, Inc.**  
+1 (661) 716-5100  
+1 (661) 716-5101 (Fax)  
www.prosoft-technology.com  
support@prosoft-technology.com

**Copyright © 2020 ProSoft Technology, Inc. All rights reserved.**

MVI56-DNPSNET User Manual

October 14, 2020

ProSoft Technology<sup>®</sup>, ProLinx<sup>®</sup>, inRAX<sup>®</sup>, ProTalk<sup>®</sup>, and RadioLinx<sup>®</sup> are Registered Trademarks of ProSoft Technology, Inc. All other brand or product names are or may be trademarks of, and are used to identify products and services of, their respective owners.



### **For professional users in the European Union**

If you wish to discard electrical and electronic equipment (EEE), please contact your dealer or supplier for further information.



**Prop 65 Warning** – Cancer and Reproductive Harm – [www.P65Warnings.ca.gov](http://www.P65Warnings.ca.gov)

## Important Installation Instructions

Power, Input, and Output (I/O) wiring must be in accordance with Class I, Division 2 wiring methods, Article 501-4 (b) of the National Electrical Code, NFPA 70 for installation in the U.S., or as specified in Section 18-1J2 of the Canadian Electrical Code for installations in Canada, and in accordance with the authority having jurisdiction. The following warnings must be heeded:

**WARNING** - EXPLOSION HAZARD - SUBSTITUTION OF COMPONENTS MAY IMPAIR SUITABILITY FOR CLASS I, DIV. 2;

**WARNING** - EXPLOSION HAZARD - WHEN IN HAZARDOUS LOCATIONS, TURN OFF POWER BEFORE REPLACING OR WIRING MODULES

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

### Class 2 Power

## MVI (Multi Vendor Interface) Modules

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'ÉQUIPEMENT, COUPER LE COURANT OU S'ASSURER QUE L'EMPLACEMENT EST DÉSIGNÉ NON DANGEREUX.

## Battery Life Advisory

The MVI46, MVI56, MVI56E, MVI69, and MVI71 modules use a rechargeable Lithium Vanadium Pentoxide battery to backup the real-time clock and CMOS. The battery should last for the life of the module. The module must be powered for approximately twenty hours before the battery becomes fully charged. After it is fully charged, the battery provides backup power for the CMOS setup and the real-time clock for approximately 21 days. When the battery is fully discharged, the module will revert to the default BIOS and clock settings.

**Note:** The battery is not user replaceable.

## Agency Approvals & Certifications

Please visit our website: [www.prosoft-technology.com](http://www.prosoft-technology.com)

# Contents

Your Feedback Please .....	2
How to Contact Us.....	2
Important Installation Instructions.....	3
MVI (Multi Vendor Interface) Modules.....	3
Battery Life Advisory.....	3
<b>Guide to the MVI56-DNPSNET User Manual</b>	<b>7</b>
<b>1 Start Here</b>	<b>8</b>
1.1 System Requirements .....	9
1.2 Package Contents .....	10
1.3 Installing ProSoft Configuration Builder Software.....	11
1.4 Setting Jumpers .....	11
1.5 Installing the Module in the Rack.....	12
1.6 Connecting Your PC to the ControlLogix Processor .....	13
1.7 Opening the Sample Ladder Logic .....	14
1.7.1 Determining the Firmware Version of Your Processor .....	14
1.7.2 Selecting the Slot Number for the Module.....	16
1.7.3 Configuring the RSLinx Driver for the PC COM Port.....	17
1.8 Downloading the Sample Program to the Processor.....	19
1.9 Connecting your PC to the Module.....	20
<b>2 Configuring the MVI56-DNPSNET Module</b>	<b>21</b>
2.1 Using ProSoft Configuration Builder .....	22
2.1.1 Setting Up the Project.....	22
2.1.2 Renaming PCB Objects .....	24
2.2 [Backplane Configuration].....	26
2.2.1 Module Name.....	26
2.2.2 Read Register Start .....	26
2.2.3 Read Register Count .....	26
2.2.4 Write Register Start .....	26
2.2.5 Write Register Count .....	26
2.2.6 Failure Flag Count .....	26
2.2.7 Error Offset .....	27
2.2.8 Initializing Output Data.....	27
2.3 [DNP ENET Slave].....	28
2.3.1 Internal Slave ID .....	28
2.3.2 Use IP List.....	28
2.3.3 Use Trip/Close Single Point.....	29
2.3.4 Binary Inputs .....	30
2.3.5 Analog Inputs .....	30
2.3.6 Float Inputs .....	30
2.3.7 Counters .....	31
2.3.8 Binary Outputs .....	31
2.3.9 Analog Outputs .....	31
2.3.10 Float Outputs .....	31
2.3.11 BI Class.....	31

2.3.12	AI Class .....	31
2.3.13	Float Class .....	31
2.3.14	AI Deadband .....	32
2.3.15	Float Deadband .....	32
2.3.16	Select/Operate Arm Time .....	32
2.3.17	Write Time Interval .....	32
2.3.18	App Layer Confirm Tout .....	32
2.3.19	Unsolicited Response .....	32
2.3.20	Class 1 Unsol Resp Min .....	33
2.3.21	Class 2 Unsol Resp Min .....	33
2.3.22	Class 3 Unsol Resp Min .....	33
2.3.23	Unsol Resp Delay .....	33
2.3.24	Uresp Master Address .....	33
2.3.25	AI Events with Time .....	33
2.3.26	AI with Flag .....	33
2.3.27	BI with Flag .....	34
2.3.28	BI Events Without Time .....	34
2.3.29	BO Without Flag .....	34
2.3.30	Counter with Flag .....	34
2.3.31	Frozen Counter with Flag .....	35
2.3.32	Time Sync Before Events .....	35
2.4	[DNP Slave Binary Inputs] .....	36
2.4.1	Point # .....	36
2.4.2	Class .....	36
2.5	[DNP Slave Analog Inputs] .....	37
2.5.1	Point # .....	37
2.5.2	Class .....	37
2.5.3	Deadband .....	37
2.6	[DNP Slave Float Inputs] .....	38
2.6.1	Point # .....	38
2.6.2	Class .....	38
2.6.3	Deadband .....	38
2.7	[DNP ENET IP ADDRESSES] .....	39
2.8	Ethernet Configuration .....	40
2.9	Downloading the Project to the Module Using a Serial COM Port .....	41
<b>3</b>	<b>Ladder Logic</b> .....	<b>42</b>
3.1	Module Data Objects .....	43
3.1.1	DNPSNETModuleDef Object .....	43
3.1.2	Special Objects .....	46
3.2	Adding the Module to an Existing Project .....	48
<b>4</b>	<b>Diagnostics and Troubleshooting</b> .....	<b>51</b>
4.1	LED Status Indicators .....	52
4.1.1	Ethernet LED Indicators .....	53
4.1.2	Clearing a Fault Condition .....	53
4.1.3	Troubleshooting .....	53
4.1.4	Error Status Table .....	54
4.2	Using ProSoft Configuration Builder (PCB) for Diagnostics .....	56
4.2.1	Using the Diagnostic Window in ProSoft Configuration Builder .....	56
4.2.2	Navigation .....	58

---

4.2.3	Main Menu .....	59
4.2.4	DNP Menu .....	61
4.2.5	Database View Menu.....	63
4.2.6	Network Menu.....	65
4.3	Reading Status Data from the Module .....	67
<b>5</b>	<b>Reference</b> .....	<b>68</b>
5.1	Product Specifications .....	69
5.1.1	General Specifications .....	69
5.1.2	Hardware Specifications .....	70
5.1.3	Functional Specifications .....	71
5.2	Functional Overview .....	72
5.2.1	General Concepts .....	72
5.3	MVI56-DNPSNET Application Design .....	86
5.3.1	Designing the system .....	86
5.3.2	Communication Parameters .....	98
5.3.3	Functionality.....	98
5.3.4	Data Transfer at Startup .....	98
5.3.5	Module Operation .....	99
5.4	Cable Connections .....	100
5.4.1	Ethernet Connection .....	100
5.4.2	RS-232 Configuration/Debug Port .....	101
5.4.3	DB9 to RJ45 Adaptor (Cable 14) .....	104
5.5	Configuration Data .....	105
5.6	MVI56-DNPSNET Status Data .....	109
5.7	Internal Indication Bits (IIN Bits) for DNP Server .....	112
5.7.1	First Byte .....	112
5.7.2	Second Byte.....	112
5.8	DNP Subset Definition .....	113
5.9	Device Profile.....	119
5.10	Event Size Computation .....	121
<b>6</b>	<b>Support, Service &amp; Warranty</b> .....	<b>122</b>
6.1	Contacting Technical Support.....	122
6.2	Warranty Information .....	122
<b>Index</b>		<b>123</b>

---

## Guide to the MVI56-DNPSNET User Manual

Function	Section to Read	Details
Introduction (Must Do)	→ Start Here (page 8)	This section introduces the customer to the module. Included are: package contents, system requirements, hardware installation, and basic configuration.
Diagnostic and Troubleshooting	→ Diagnostics and Troubleshooting (page 51)	This section describes Diagnostic and Troubleshooting procedures.
Reference Product Specifications	→ Reference (page 68) Product Specifications (page 69)	These sections contain general references associated with this product and its Specifications..
Support, Service, and Warranty Index	→ Support, Service and Warranty (page 122) Index	This section contains Support, Service and Warranty information. Index of chapters.

# 1 Start Here

## *In This Chapter*

❖ System Requirements .....	9
❖ Package Contents .....	10
❖ Installing ProSoft Configuration Builder Software .....	11
❖ Setting Jumpers.....	11
❖ Installing the Module in the Rack.....	12
❖ Connecting Your PC to the ControlLogix Processor .....	13
❖ Opening the Sample Ladder Logic .....	14
❖ Downloading the Sample Program to the Processor .....	19
❖ Connecting your PC to the Module.....	20

To get the most benefit from this User Manual, you should have the following skills:

- **Rockwell Automation® RSLogix™ software:** launch the program, configure ladder logic, and transfer the ladder logic to the processor
- **Microsoft Windows:** install and launch programs, execute menu commands, navigate dialog boxes, and enter data
- **Hardware installation and wiring:** install the module, and safely connect Distributed Network Protocol and ControlLogix devices to a power source and to the MVI56-DNPSNET module's application port(s)



## 1.1 System Requirements

The MVI56-DNPSNET module requires the following minimum hardware and software components:

- Rockwell Automation ControlLogix™ processor, with compatible power supply and one free slot in the rack, for the MVI56-DNPSNET module. The module requires 800 mA of available power.
- Rockwell Automation RSLogix 5000 programming software version 2.51 or higher
- Rockwell Automation RSLinx communication software
- Pentium® II 450 MHz minimum. Pentium III 733 MHz (or better) recommended
- Supported operating systems:
  - Microsoft Windows 10
  - Microsoft Windows 7 Professional (32-or 64-bit)
  - Microsoft Windows XP Professional with Service Pack 1 or 2
  - Microsoft Windows Vista
  - Microsoft Windows 2000 Professional with Service Pack 1, 2, or 3
  - Microsoft Windows Server 2003
- 128 Mbytes of RAM minimum, 256 Mbytes of RAM recommended
- 100 Mbytes of free hard disk space (or more based on application requirements)
- 256-color VGA graphics adapter, 800 x 600 minimum resolution (True Color 1024 × 768 recommended)
- ProSoft Configuration Builder, HyperTerminal or other terminal emulator program.

**Note:** You can install the module in a local or remote rack. For remote rack installation, the module requires EtherNet/IP or ControlNet communication with the processor.

## 1.2 Package Contents

The following components are included with your MVI56-DNPSNET module, and are all required for installation and configuration.

**Important:** Before beginning the installation, please verify that all of the following items are present.

Qty.	Part Name	Part Number	Part Description
1	MVI56-DNPSNET Module	MVI56-DNPSNET	DNP 3.0 Server over Ethernet Communication Module
1	Cable	Cable #15 - RS232 Null Modem	For RS232 between a Personal Computer (PC) and the CFG port of the module
1	Cable	Cable #14 - RJ45 to DB9 Male Adapter	For connecting the module's port to Cable #15 for RS-232 connections

If any of these components are missing, please contact ProSoft Technology Support for replacement parts.

### 1.3 Installing ProSoft Configuration Builder Software

You must install the *ProSoft Configuration Builder (PCB)* software to configure the module. You can always get the newest version of *ProSoft Configuration Builder* from the ProSoft Technology website.

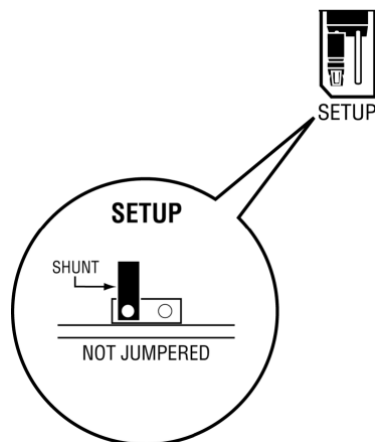
#### To install ProSoft Configuration Builder from the ProSoft Technology website

- 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 *Windows 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.

### 1.4 Setting Jumpers

The Setup Jumper acts as "write protection" for the module's flash memory. In "write protected" mode, the Setup pins are not connected, and the module's firmware cannot be overwritten. Do not jumper the Setup pins together unless you are directed to do so by ProSoft Technical Support.

The following illustration shows the MVI56-DNPSNET jumper configuration.



**Note:** If you are installing the module in a remote rack, you may prefer to leave the Setup pins jumpered. That way, you can update the module's firmware without requiring physical access to the module.

## 1.5 Installing the Module in the Rack

If you have not already installed and configured your ControlLogix processor and power supply, please do so before installing the MVI56-DNPSNET module. Refer to your Rockwell Automation product documentation for installation instructions.

**Warning:** You must follow all safety instructions when installing this or any other electronic devices. Failure to follow safety procedures could result in damage to hardware or data, or even serious injury or death to personnel. Refer to the documentation for each device you plan to connect to verify that suitable safety procedures are in place before installing or servicing the device.

After you have checked the placement of the jumpers, insert MVI56-DNPSNET into the ControlLogix chassis. Use the same technique recommended by Rockwell Automation to remove and install ControlLogix modules.

**Warning:** When you insert or remove the module while backplane power is on, an electrical arc can occur. This could cause an explosion in hazardous location installations. Verify that power is removed or the area is non-hazardous before proceeding. Repeated electrical arcing causes excessive wear to contacts on both the module and its mating connector. Worn contacts may create electrical resistance that can affect module operation.

- 1 Turn power OFF.
- 2 Align the module with the top and bottom guides, and slide it into the rack until the module is firmly against the backplane connector.



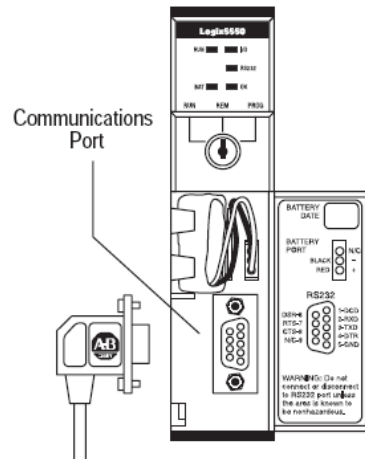
- 3 With a firm but steady push, snap the module into place.
- 4 Check that the holding clips on the top and bottom of the module are securely in the locking holes of the rack.
- 5 Make a note of the slot location. You must identify the slot in which the module is installed in order for the sample program to work correctly. Slot numbers are identified on the green circuit board (backplane) of the ControlLogix rack.
- 6 Turn power ON.

**Note:** If you insert the module improperly, the system may stop working, or may behave unpredictably.

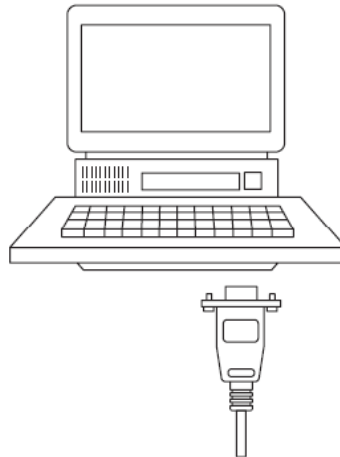
## 1.6 Connecting Your PC to the ControlLogix Processor

There are several ways to establish communication between your PC and the ControlLogix processor. The following steps show how to establish communication through the serial interface. It is not mandatory that you use the processor's serial interface. You may access the processor through whatever network interface is available on your system. Refer to your Rockwell Automation documentation for information on other connection methods.

- 1 Connect the right-angle connector end of the cable to your controller at the communications port.



- 2 Connect the straight connector end of the cable to the serial port on your computer.



## 1.7 Opening the Sample Ladder Logic

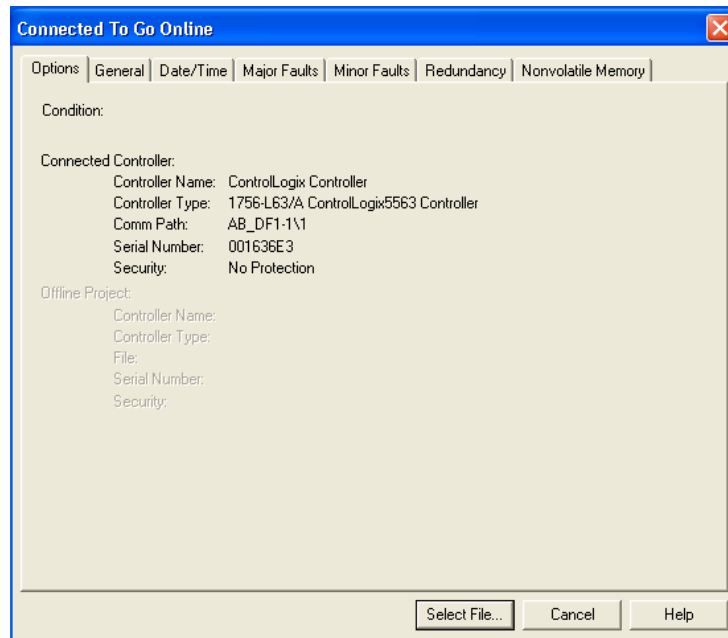
The sample program for your MVI56-DNPSNET module includes custom tags, data types and ladder logic for data I/O and status monitoring. For most applications, you can run the sample ladder program without modification, or, for advanced applications, you can incorporate the sample program into your existing application.

The version number appended to the file name corresponds with the firmware version number of your ControlLogix processor. The firmware version and sample program version must match.

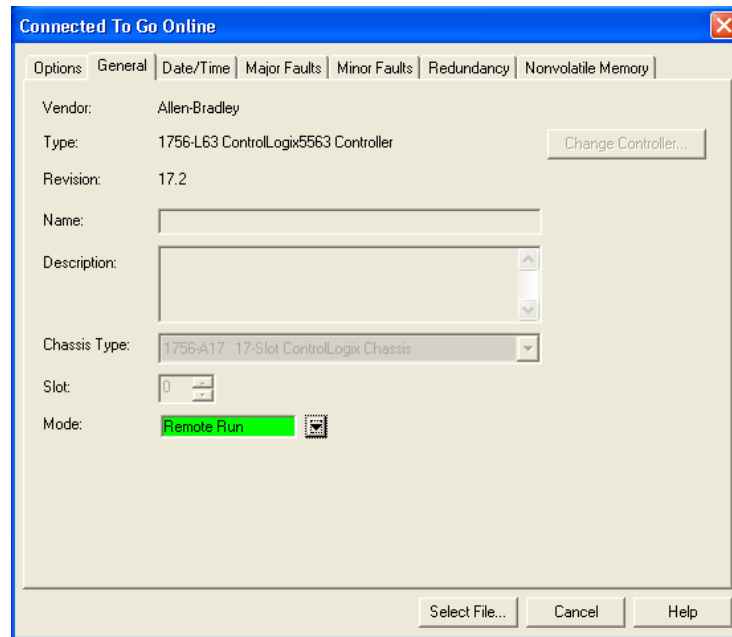
### 1.7.1 Determining the Firmware Version of Your Processor

**Important:** The RSLinx service must be installed and running on your computer in order for RSLogix to communicate with the processor. Refer to your RSLinx and RSLogix documentation for help configuring and troubleshooting these applications.

- 1 Connect an RS-232 serial cable from the COM (serial) port on your PC to the communication port on the front of the processor.
- 2 Start RSLogix 5000 and close any existing project that may be loaded.
- 3 Open the **COMMUNICATIONS** menu and choose **GO ONLINE**. RSLogix will establish communication with the processor. This may take a few moments.
- 4 When RSLogix has established communication with the processor, the *Connected To Go Online* dialog box will open.



- 5 In the *Connected To Go Online* dialog box, click the **GENERAL** tab. This tab shows information about the processor, including the *Revision* (firmware) version. In the following illustration, the firmware version is 17.2.

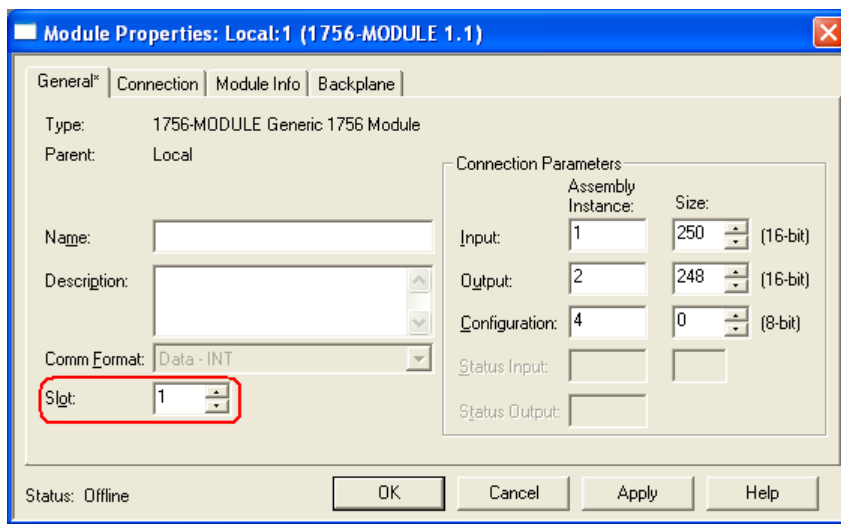


### 1.7.2 Selecting the Slot Number for the Module

This sample application is for a module installed in Slot 1 in a ControlLogix rack. The ladder logic uses the slot number to identify the module. If you are installing the module in a different slot, you must update the ladder logic so that program tags and variables are correct, and do not conflict with other modules in the rack.

#### To change the slot number

- 1 In the *Controller Organization* list, select the module and then click the right mouse button to open a shortcut menu.
- 2 On the shortcut menu, choose **PROPERTIES**. This action opens the *Module Properties* dialog box.



- 3 In the *Slot* field, use the spinners on the right side of the field to select the slot number where the module will reside in the rack, and then click **OK**.

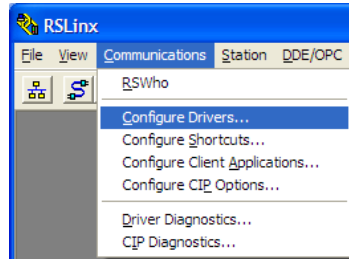
RSLogix will automatically apply the slot number change to all tags, variables and ladder logic rungs that use the MVI56-DNPSNET slot number for computation.



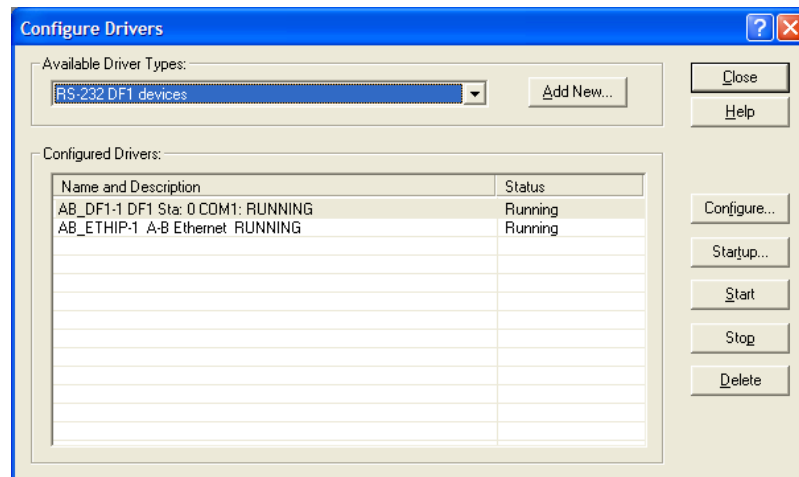
### 1.7.3 Configuring the RSLinx Driver for the PC COM Port

If RSLogix is unable to establish communication with the processor, follow these steps.

- 1 Open *RSLogix*.
- 2 Open the **COMMUNICATIONS** menu, and choose **CONFIGURE DRIVERS**.

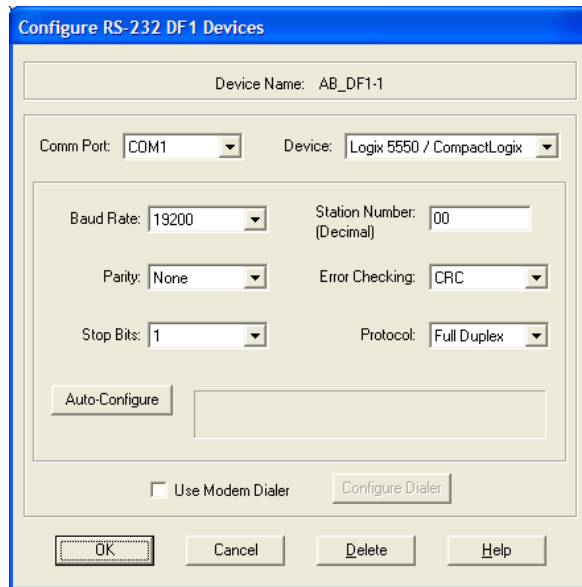


This action opens the *Configure Drivers* dialog box.



**Note:** If the list of configured drivers is blank, you must first choose and configure a driver from the *Available Driver Types* list. The recommended driver type to choose for serial communication with the processor is *RS-232 DF1 Devices*.

- 3 Click to select the driver, and then click **CONFIGURE**. This action opens the *Configure RS-232 DF1 Devices* dialog box.



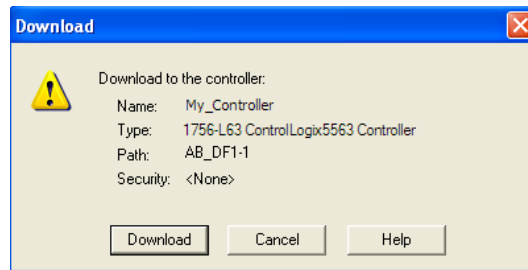
- 4 Click the **AUTO-CONFIGURE** button. RSLinx will attempt to configure your serial port to work with the selected driver.
- 5 When you see the message *Auto Configuration Successful*, click the **OK** button to dismiss the dialog box.

**Note:** If the auto-configuration procedure fails, verify that the cables are connected correctly between the processor and the serial port on your computer, and then try again. If you are still unable to auto-configure the port, refer to your RSLinx documentation for further troubleshooting steps.

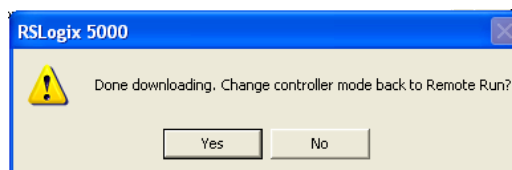
## 1.8 Downloading the Sample Program to the Processor

**Note:** The key switch on the front of the ControlLogix processor must be in the REM or PROG position.

- 1 If you are not already online with the processor, open the *Communications* menu, and then choose **DOWNLOAD**. RSLogix 5000 will establish communication with the processor. You do not have to download through the processor's serial port, as shown here. You may download through any available network connection.
- 2 When communication is established, RSLogix 5000 will open a confirmation dialog box. Click the **DOWNLOAD** button to transfer the sample program to the processor.



- 3 RSLogix 5000 will compile the program and transfer it to the processor. This process may take a few minutes.
- 4 When the download is complete, RSLogix 5000 will open another confirmation dialog box. If the key switch is in the REM position, click **OK** to switch the processor from PROGRAM mode to RUN mode.

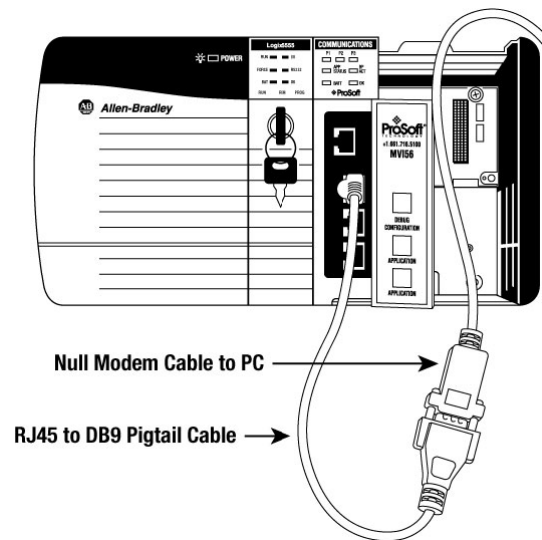


**Note:** If you receive an error message during these steps, refer to your RSLogix documentation to interpret and correct the error.

## 1.9 Connecting your PC to the Module

With the module securely mounted, connect your PC to the *Configuration/Debug* port using an RJ45-DB-9 Serial Adapter Cable and a Null Modem Cable.

- 1 Attach both cables as shown.
- 2 Insert the RJ45 cable connector into the Configuration/Debug port of the module.
- 3 Attach the other end to the serial port on your PC.



## 2 Configuring the MVI56-DNPSNET Module

### *In This Chapter*

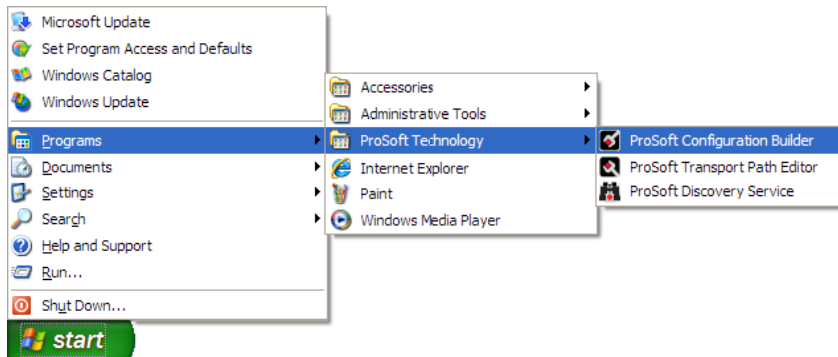
❖ Using ProSoft Configuration Builder.....	22
❖ [Backplane Configuration] .....	26
❖ [DNP ENET Slave] .....	28
❖ [DNP Slave Binary Inputs] .....	36
❖ [DNP Slave Analog Inputs] .....	37
❖ [DNP Slave Float Inputs] .....	38
❖ [DNP ENET IP ADDRESSES].....	39
❖ Ethernet Configuration .....	40
❖ Downloading the Project to the Module Using a Serial COM Port.....	41

## 2.1 Using ProSoft Configuration Builder

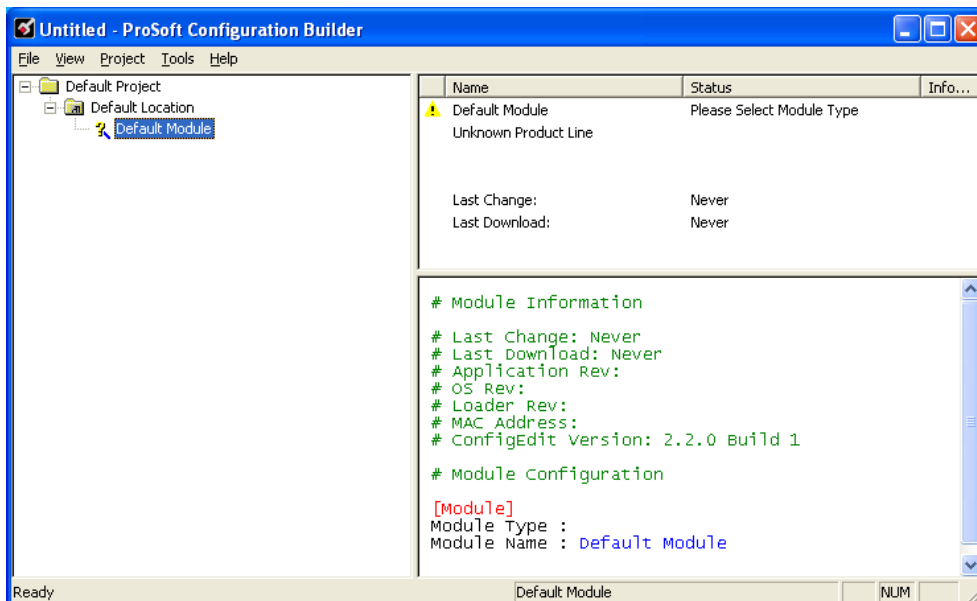
*ProSoft Configuration Builder (PCB)* provides a quick and easy way to manage module configuration files customized to meet your application needs. *PCB* is not only a powerful solution for new configuration files, but also allows you to import information from previously installed (known working) configurations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

### 2.1.1 Setting Up the Project

To begin, start **PROSOFT CONFIGURATION BUILDER (PCB)**.

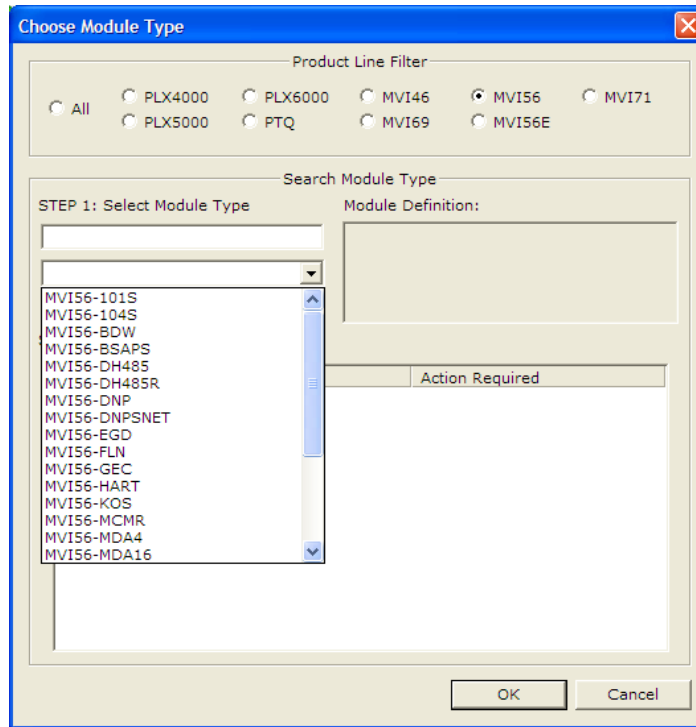


If you have used other Windows configuration tools before, you will find the screen layout familiar. *PCB's* window consists of a tree view on the left, and an information pane and a configuration pane on the right side of the window. When you first start *PCB*, the tree view consists of folders for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



### Adding the MVI56-DNPSNET module to the project

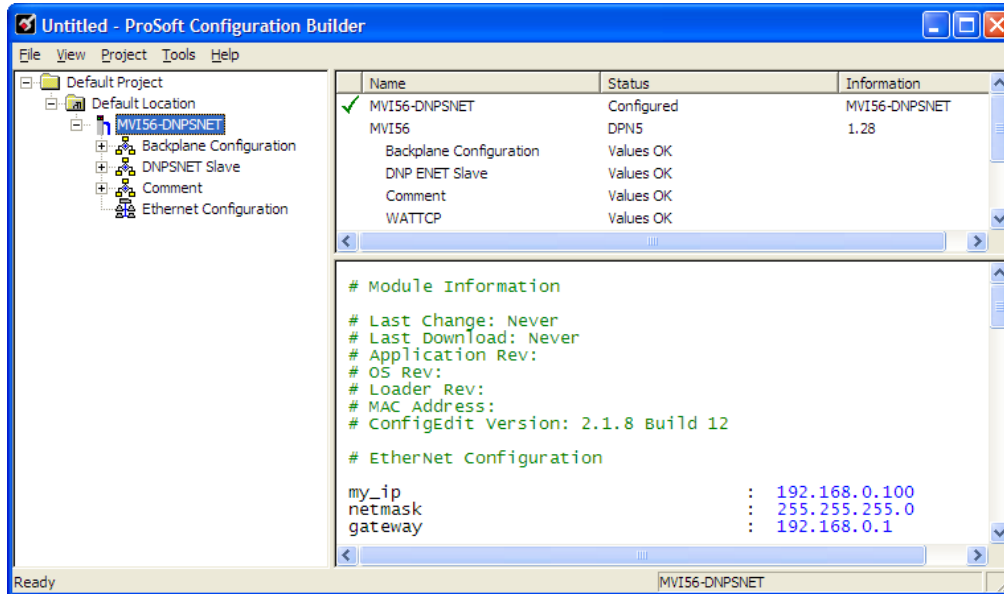
- 1 Use the mouse to select **DEFAULT MODULE** in the tree view, and then click the right mouse button to open a shortcut menu.
- 2 On the shortcut menu, choose **CHOOSE MODULE TYPE**. This action opens the *Choose Module Type* dialog box.



- 3 In the *Product Line Filter* area of the dialog box, select **MVI56**. In the *Select Module Type* dropdown list, select **MVI56-DNPSNET**, and then click **OK** to save your settings and return to the *ProSoft Configuration Builder* window.

### 2.1.2 Renaming PCB Objects



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



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



- 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.

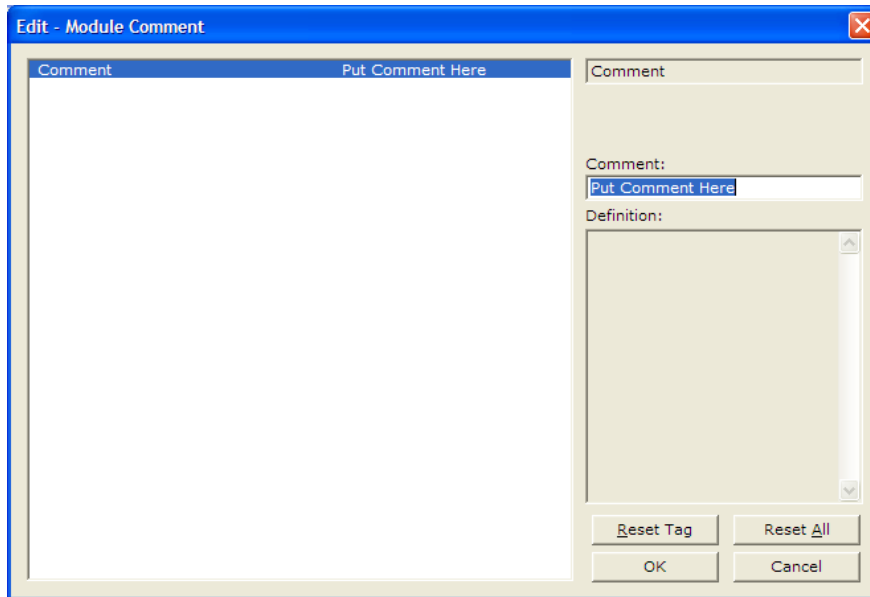
#### Configuring Module Parameters

- 1 Click the **[+]** sign next to the module icon to expand module information.
- 2 Click the **[+]** sign next to any  icon to view module information and configuration options.
- 3 Double-click any  icon to open an *Edit* dialog box.
- 4 To edit a parameter, select the parameter in the left pane and make your changes in the right pane.
- 5 Click **OK** to save your changes.



### Creating Optional Comment Entries

- 1 Click the **[+]** to the left of the  Comment icon to expand the module comments.
- 2 Double-click the  Module Comment icon. The *Edit - Module Comment* dialog box appears.



- 3 Enter your comment and click **OK** to save your changes.

### Printing 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 In the *View Configuration* window, open the **FILE** menu, and choose **PRINT**. This action opens the *Print* dialog box.
- 4 In the *Print* dialog box, choose the printer to use from the drop-down list, select printing options, and then click **OK**.

## 2.2 [Backplane Configuration]

This section of the file describes the database setup and module level parameters.

### 2.2.1 *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.

### 2.2.2 *Read Register Start*

**0** to **8899**

This parameter specifies the starting register in the module where data will be transferred from the module to the processor. Valid range for this parameter is 0 to 8899.

### 2.2.3 *Read Register Count*

**0** to **9000**

This parameter specifies the number of registers to be transferred from the module to the processor. Valid entry for this parameter is 0 to 9000.

### 2.2.4 *Write Register Start*

**0** to **8899**

This parameter specifies the starting register in the module where the data will be transferred from the processor to the module. Valid range for this parameter is 0 to 8899.

### 2.2.5 *Write Register Count*

**0** to **9000**

This parameter specifies the number of registers to transfer from the processor to the module. Valid entry for this parameter is 0 to 9000.

### 2.2.6 *Failure Flag Count*

**0** through **65535**

This parameter specifies the number of successive transfer errors that must occur before halting communication on the application port(s). If the parameter is set to **0**, the application port(s) 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.

### **2.2.7 Error Offset**

0 to 8964

This 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 8964, the data will be placed in the modules database.

### **2.2.8 Initializing 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.

## 2.3 [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.

### 2.3.1 *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.

### 2.3.2 *Use IP List*

**YES or NO**

This parameter specifies if the IP address of the host connected to the system will be validated. If the parameter is set to **No**, any host may connect to the unit. If the parameter is set to **YES**, 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.

### 2.3.3 Use Trip/Close Single Point

Yes or No

If you set this parameter to Yes, Trip/Close events will function like Pulse On operations. Only one bit will be reserved in the DNP BO database.

If you set this parameter to No, the dual-point relay control database (Trip/Close) is overlaid on the DNP Binary Output database of the module. Each DNP point index sent will have an offset of point index times 2 into the database. The first bit of the dual-point relay control database will correspond to the close relay and the second will correspond to the trip relay.

The bit definitions from control byte of CROB are as follows:

- 00 - Null (single bit control or select of Trip/Close)
- 01 - Close relay
- 10 - Trip relay
- 11 - Invalid

If the operate command is used with the Null relay (00), the module will operate on the point as single point control. The following table describes the module's behavior:

Point Index in Command	Point in Database Controlled
0	Bit 0 in BO database
10	Bit 10 in BO database
15	Bit 15 in BO database

If the operate command is used with the close relay selected, the module will operate on the first bit of the two database bits associated with the point. The following table describes the module's behavior when the close relay is selected:

Point Index in Command	Point in Database Controlled
0	Bit 0 in BO database
1	Bit 2 in BO database
10	Bit 20 in BO database
15	Bit 30 in BO database

If the operate command is used with the trip relay selected, the module will operate on the second bit of the two database bits associated with the point. The following table describes the module's behavior when the trip relay is selected:

Point Index in Command	Point in Database Controlled
0	Bit 1 in BO database
1	Bit 3 in BO database
10	Bit 21 in BO database
15	Bit 31 in BO database

It is important to note that the trip and close relays are linked in the module. If a latch-on command is sent to the close relay its bit will be set and the associated trip relay bit will be cleared.

Because the single-point and dual-point control database share the same memory area, caution should be exercised to prevent control of one area by another. This can be accomplished by careful design of the system. The dual-point database could be isolated from the single-point database. For example, DNP point index 0 to 9 could be used for the dual-point database and correspond to bits 0 to 19. The single-point control points would then start at DNP point index 20 which corresponds to bit 20 of the database.

Using this technique, the MVI56-DNPSNET module will not require any configuration for the new dual-point control, and the module will be backward compatible for current customer applications.

### **2.3.4 Binary Inputs**

0 to 500 words

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.

### **2.3.5 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.

### **2.3.6 Float Inputs**

0 to 150

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.

### **2.3.7 Counters**

0 to 250 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 250 points.

### **2.3.8 Binary Outputs**

0 to 500 words

Number of digital output 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 outputs will be defined for the application.

### **2.3.9 Analog Outputs**

0 to 500 points

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

### **2.3.10 Float Outputs**

0 to 150 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.

### **2.3.11 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.

### **2.3.12 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.

### **2.3.13 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.

### **2.3.14 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.

### **2.3.15 Float Deadband**

**0** to **10000** data units

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

### **2.3.16 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.

### **2.3.17 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.

### **2.3.18 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.

### **2.3.19 Unsolicited Response**

**YES** or **No**

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



### **2.3.20 Class 1 Unsol Resp Min**

1 to **255** events

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

### **2.3.21 Class 2 Unsol Resp Min**

1 to **255** events

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

### **2.3.22 Class 3 Unsol Resp Min**

1 to **255** events

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

### **2.3.23 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.

### **2.3.24 Uresp Master Address**

0 to **65535**

DNP destination address where unsolicited response messages are sent.

### **2.3.25 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 **No**, the default is set to no time data. If the parameter is set to **YES**, the default object will include the time of the event.

### **2.3.26 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 **No**, variation 4 will be returned. If the parameter is set to **YES**, 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.

### ***2.3.27 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 **No**, variation 1 will be returned. If the parameter is set to **YES**, 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.

### ***2.3.28 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.

### ***2.3.29 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 **No**, variation 2 will be returned. If the parameter is set to **YES**, variation 1 will be returned.

### ***2.3.30 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 **No**, variation 5 will be returned. If the parameter is set to **YES**, 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.

### **2.3.31 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 **No**, variation 9 will be returned. If the parameter is set to **YES**, 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.

### **2.3.32 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 **No**, events will be generated irrespective of the module's time sync status. If the parameter is set to **YES**, events will be generated only if the module's time is synchronized.

## **2.4 [DNP Slave Binary Inputs]**

This section of the configuration file overrides the Class 2 binary database points.

### **2.4.1 *Point #***

This is the information object address of the point.

### **2.4.2 *Class***

**CLASS 1** - Highest priority

**CLASS 2** - Middle priority

**CLASS 3** - Lowest priority

**0** - Disable.

## **2.5 [DNP Slave Analog Inputs]**

This area is to override the class (3) and deadband for the integer analog input database. The point # is the offset from the start of the analog input database.

### **2.5.1 Point #**

This is the information object address of the point.

### **2.5.2 Class**

**CLASS 1** - Highest priority

**CLASS 2** - Middle priority

**CLASS 3** - Lowest priority

**0** - Disable.

### **2.5.3 Deadband**

The module will generate events only if the data value changes by an amount greater than or equal to the configured deadband value.

## **2.6 [DNP Slave Float Inputs]**

This area is to override the class (3) and deadband for the single float database. The point # is not the address in the analog database, but is the offset from the start of the single floating-point database.

### **2.6.1 Point #**

This is the information object address of the point.

### **2.6.2 Class**

**CLASS 1** - Highest priority

**CLASS 2** - Middle priority

**CLASS 3** - Lowest priority

**0** - Disable.

### **2.6.3 Deadband**

The module will generate events only if the data value changes by an amount greater than or equal to the configured deadband value.

## 2.7 [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 than 10 addresses.

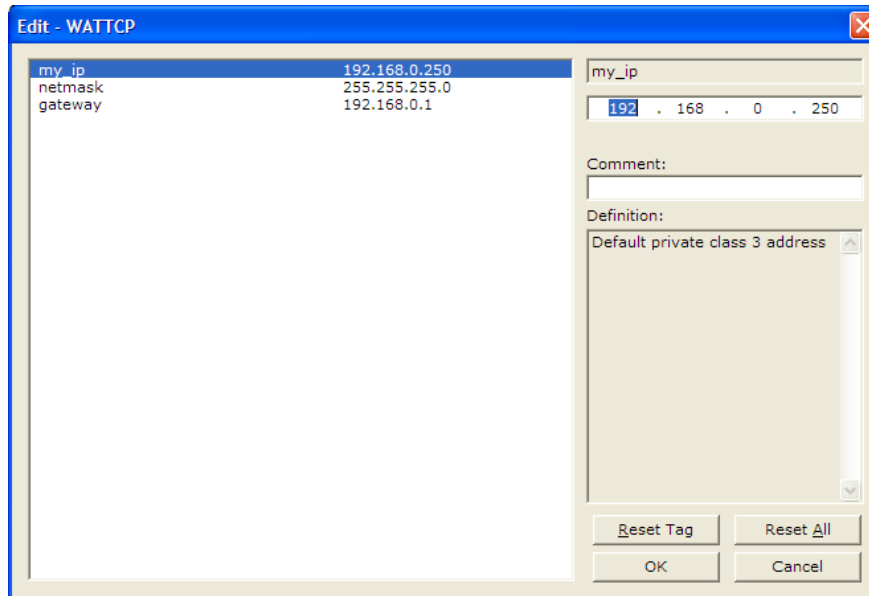
## 2.8 Ethernet Configuration

Use this procedure to configure the Ethernet settings for your module. You must assign an IP address, subnet mask and gateway address. After you complete this step, you can connect to the module with an Ethernet cable.

- 1 Determine the network settings for your module, with the help of your network administrator if necessary. You will need the following information:
  - IP address (fixed IP required) \_\_\_\_\_ . \_\_\_\_\_ . \_\_\_\_\_ . \_\_\_\_\_
  - Subnet mask \_\_\_\_\_ . \_\_\_\_\_ . \_\_\_\_\_ . \_\_\_\_\_
  - Gateway address \_\_\_\_\_ . \_\_\_\_\_ . \_\_\_\_\_ . \_\_\_\_\_

**Note:** The gateway address is optional, and is not required for networks that do not use a default gateway.

- 2 Double-click the **ETHERNET CONFIGURATION** icon. This action opens the *Edit* dialog box.



- 3 Edit the values for *my\_ip*, *netmask* (subnet mask) and *gateway* (default gateway).
- 4 When you are finished editing, click **OK** to save your changes and return to the *ProSoft Configuration Builder* window.

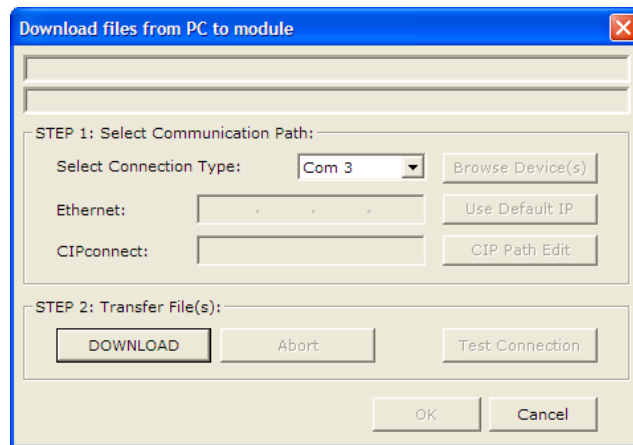


## 2.9 Downloading the Project to the Module Using a Serial COM Port

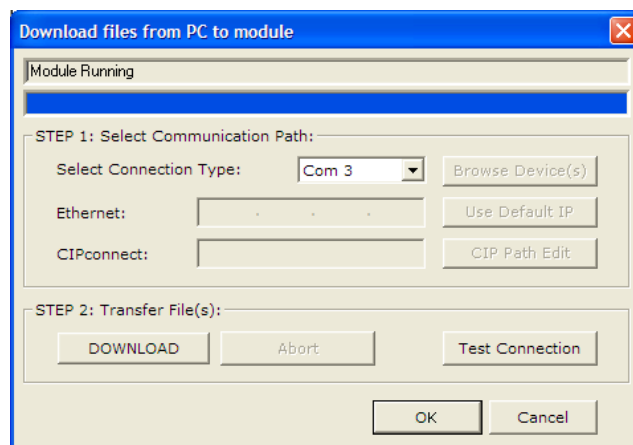
For the module to use the settings you configured, you must download (copy) the updated *Project* file from your PC to the module.

### To download the project file

- 1 In the tree view in *ProSoft Configuration Builder*, click once to select the module.
- 2 Right-click the module icon to open a shortcut menu. From the shortcut menu, choose **DOWNLOAD FROM PC TO DEVICE**. 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.



- 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 will display the message *Module Running*.



## 3 Ladder Logic

### *In This Chapter*

- ❖ Module Data Objects .....43
- ❖ Adding the Module to an Existing Project .....48

Ladder logic is required for application of the MVI56-DNPSNET module. Tasks that must be handled by the ladder logic are module data transfer, special block handling, and status data receipt. Additionally, a power-up handler may be needed to handle the initialization of the module's data and to clear any processor fault conditions.

The sample ladder logic is extensively commented, to provide information on the purpose and function of each rung. For most applications, the sample ladder will work without modification.

### 3.1 Module Data Objects

All data related to the MVI56-DNPSNET is stored in one user defined data type, containing data transfer and status data, and the DNP datasets. Any time an array's size is altered in the RSLogix 5000 software, all the data in the object can be set to zero. Because the array sizes may need to be adjusted for the data types in an application, the user defined data should be adjusted prior to the module being placed in service (if the default configuration does not contain enough data points for the application).

An instance of each data type is required before the module can be used. This is accomplished by declaring variables of the data types in the Controller Tags Edit Tags dialog box. Each object is discussed in the following topics.

#### 3.1.1 DNPSNETModuleDef Object

The DNPSNETModuleDef object contains all the MVI56-DNPSNET module status data and data transfer variables. The following table describes the structure of this object.

Name	Data Type
Status	DNPSNETSlvStat
Data	DNPSNETData
CMDcontrolbits	DNPSNETCMDBits
ReadClock	DNPSNETClock
WriteClock	DNPSNETClock
BI_Events	DNPSNETBIEvntBlk
SOE_BI_Events	DNPSNET_SOE_BIEvntBlk
AI_Events	DNPSNETAIEvntBLK
BP	DNPSNETBackplane

Each of these object types are discussed in the following topics:

### Status

This object holds the module status information transferred with each read data block transferred from the module. The following table describes the structure of this object.

<b>Name</b>	<b>Data Type</b>	<b>Description</b>
Scan_Cnt	INT	Program Scan Counter
Product_Name	SINT[4]	Product Code
Rev_Level	SINT[4]	Revision
Op_Sys	SINT[4]	Operating system revision
Run_Number	SINT[4]	Run number
Blk_Rd_Count	INT	Number of block read transfers
Blk_Wr_Count	INT	Number of block write transfers
Blk_Parse_Cnt	INT	Number of blocks parsed by module
Blk_Err	INT	Number of block errors
Rx_Frames	INT	Number of frames received for this unit
Tx_Frames	INT	Number of frames transmitted for this unit
Rx_Total	INT	Number of frames received
Sync_err	INT	Sync error count
Overrun_err	INT	Overrun error count
len_err	INT	Length error count
CRC_err	INT	CRC error count
Overflow_err	INT	Overflow error count
Seq_err	INT	Sequence error count
Addr_err	INT	Address error count
BI_Events	INT	Number of binary events generated
BI_Queue	INT	Number of binary events in queue
AI_Events	INT	Number of analog input events
AI_Queue	INT	Number of analog input events in queue
FL_Events	INT	Number of float input events
Reserved	INT	Reserved
Bad_func_err	INT	Number of bad function code error count
Ukn_Obj_err	INT	Unknown Object error count
Range_err	INT	Range error count
App_Overflow	INT	Number of application level overflow errors
Multi_Frame_err	INT	Multi-frame error count
UDP_Rx_Count	INT	UDP Recieve Count
UDP_Tx_Count	INT	UDP transmit Count
Unsol_error	INT	Unsolicited Error Count
State_Value	INT	State Value
TCP_ST_value	INT	TCP Socket State Value
UDP_ST_Value	INT	UDP Socket State Value
Busywithmsg	INT	DNP Busy with Message State
App_Fragm	INT	Application fragment
Tx_frame_ST	INT	Transmit frame State
TCP_msg_len	INT	TCP message length
UDP_msg_len	INT	UDP message length
Port_Tx_St	INT	Port Transmit state
Free_Mem	DINT	Free Memory

This information is important as it can be used to view the "health" of the module. If the module is not communicating, examine the object to help find the problem. Additionally, you should use the configuration/debug port on the module to confirm that the desired configuration of the module is implemented.

Data

The DNPSNETData object stores all the data for an MVI56-DNPSNET module. Contained within the object is an array for each data type. The array sizes are set to match the configuration set for the module. If multiple MVI56-DNPSNET modules are used within a rack, a copy of this structure may have to be made to permit each module to have its own database sizes. Ladder logic is required to transfer the data in this structure between the module and the processor. The following table describes the structure of this object.

Name	Data Type	Description
DNPSNET_BI	INT[20]	Number of DNPSNET BI data words
DNPSNET_AI	INT[40]	Number of DNPSNET AI data points
DNPSNET_FLTI	REAL[20]	Number of DNPSNET FLTI data points
DNPSNET_Cntr	DINT[10]	Number of DNPSNET counter double-words
DNPSNET_BO	INT[40]	Number of DNPSNET BO data words
DNPSNET_AO	INT[40]	Number of DNPSNET AO data words
DNPSNET_FLTO	REAL[20]	Number of DNPSNET FLTO data points

CMDcontrolbits

Name	Data Type	Description
Warmboot	BOOL	WarmBoot module
ColdBoot	BOOL	ColdBoot module
GetDateNTime	BOOL	Get Date and Time from module
SetDateNTime	BOOL	Set Date and Time for module
SetBIEvents	BOOL	Send BI Event to Binary Input event buffer
SetAIEvents	BOOL	Send AI Event to Analog Input event buffer
GetFullStatus	BOOL	Request 9250 block for full status information

Clock

Name	Data Type	Description
Year	DINT	Year returned from GSV to processor
Month	DINT	Month returned from GSV to processor
Day	DINT	Day returned from GSV to processor
Hours	DINT	Hours returned from GSV to processor
Minutes	DINT	Minutes returned from GSV to processor
Seconds	DINT	Seconds returned from GSV to processor
MicroSeconds	DINT	MicroSeconds returned from GSV to processor
Synchronized	INT	If 1, time has been set by DNP master. 0 = waiting for time to be set.

**BP (Backplane)**

The DNPSNETBackplane Object stores the variables required for backplane data transfer between the module and the processor. The following table describes the structure of the object.

<b>Name</b>	<b>Data Type</b>	<b>Description</b>
LastRead	INT	Index of last read block
LastWrite	INT	Index of last write block
BlockIndex	INT	Computed block offset for data table
ReadData	INT[600]	Buffer File for data Read from Module
WriteData	INT[600]	Buffer File for data Written to Module

**3.1.2 Special Objects**

These objects utilize some of the advanced features the module provides. If your application does not require the object, then you need not declare an instance of the object. Each of the objects and associated function are discussed in the following topics.

**BI Events**

The DNPSNETAIEvntBLKt object stores the information for a single binary input event to be sent from the processor to the module in a command block 9958. The structure shown in the following example contains all the parameters required for a binary input event.

<b>Name</b>	<b>Data Type</b>	<b>Description</b>
EventCount	INT	Event Count
SeqCounter	INT	Sequence Counter
EventData	DNPSNETAIEvntData[10]	Event Data Points
AIDataPoint	INT	DNPSNET Analog Input Data Point
AIvalue	INT	DNPSNET Analog Input Value
Day	SINT	Day
Month	SINT	Month
Minutes	SINT	Minutes
Hour	SINT	Hour
SecMsec	INT	Bits 0 to 9 = Milliseconds Bits 10 to 15 = Seconds
Year	INT	Year

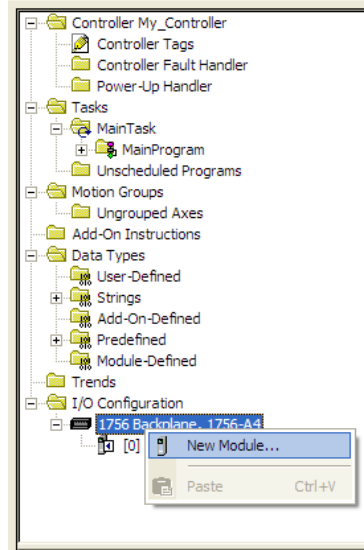
### AI Events

The DNPSNETBIEvntBlk object stores the information for a single analog input event to be sent from the processor to the module in a command block 9959. The structure shown in the following example contains all the parameters required for an analog input event.

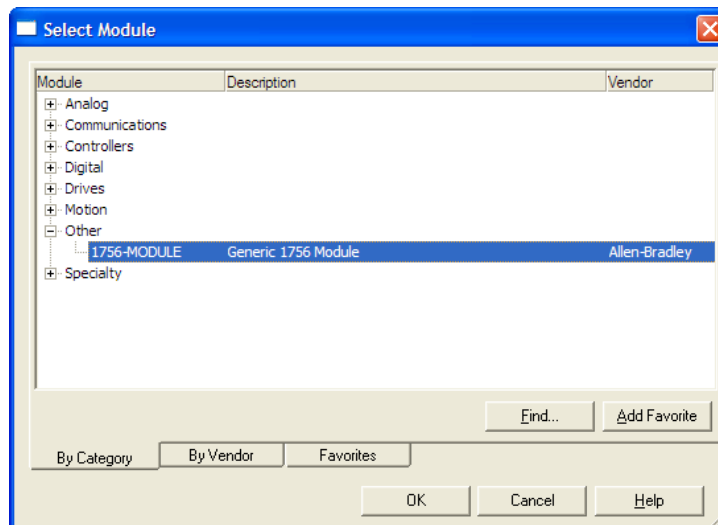
<b>Name</b>	<b>Data Type</b>	<b>Description</b>
EventCount	INT	Event Count
SeqCounter	INT	Sequence Counter
EventData	DNPSNETBIEvntData[12]	Event Data Points
DataPoint	INT	DNPSNET Binary Input Data Point
Day	SINT	Day
MonthState	SINT	Month and State Bit (state is MSB)
Minutes	SINT	Minutes
Hour	SINT	Hours
SecMsecond	INT	Bits 0 to 9 = Milliseconds Bits 10 to 15 = Seconds
Year	INT	Year

### 3.2 Adding the Module to an Existing Project

- 1 Select the *I/O Configuration* folder in the *Controller Organization* window of RSLogix 5000, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose **NEW MODULE**.



This action opens the *Select Module* dialog box:

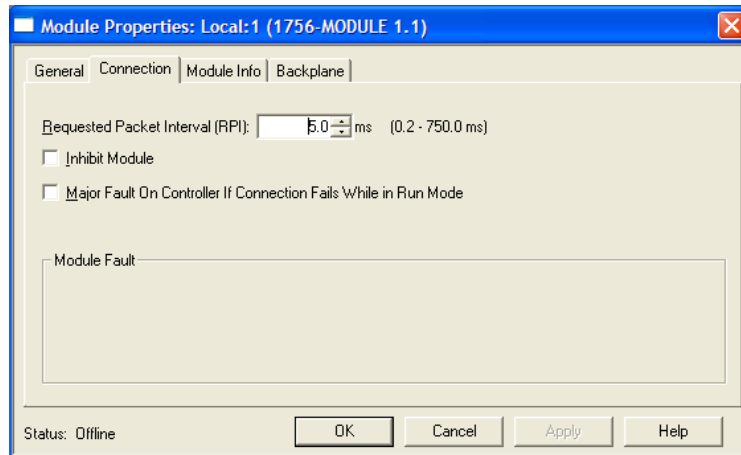


- 2 Select the **1756-MODULE (GENERIC 1756 MODULE)** from the list and click **OK**. This action opens the *New Module* dialog box.
- 3 Enter the *Name*, *Description* and *Slot* options for your application. You must select the *Comm Format* as **DATA - INT** in the dialog box, otherwise the module will not communicate. Click **OK** to continue.

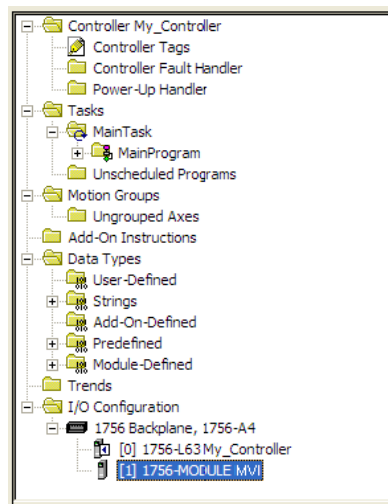


Parameter	Value
Name	Enter a module identification string. Example: <b>DNPSNET_2</b>
Description	Enter a description for the module. Example: <b>DNP 3.0 SERVER OVER ETHERNET COMMUNICATION MODULE</b>
Comm Format	Select <b>DATA-INT</b> .
Slot	Enter the slot number in the rack where the MVI56-DNPSNET module is located.
Input Assembly Instance	<b>1</b>
Input Size	<b>250</b>
Output Assembly Instance	<b>2</b>
Output Size	<b>248</b>
Configuration Assembly Instance	<b>4</b>
Configuration Size	<b>0</b>

- 4 Select the *Requested Packet Interval* value for scanning the I/O on the module. This value represents the minimum frequency that the module will handle scheduled events. This value should not be set to less than **1** millisecond. The default value is **5** milliseconds. Values between **1** and **10** milliseconds should work with most applications.



- 5 Save the module. Click **OK** to dismiss the dialog box. The *Controller Organization* window now displays the module's presence.



- 6 Copy the *User-Defined Data Types* from the sample program into your existing RSLogix 5000 project.
- 7 Copy the *Controller Tags* from the sample program into your project.
- 8 Copy the *Ladder Rungs* from the sample program into your project.

## 4 Diagnostics and Troubleshooting

### In This Chapter

- ❖ LED Status Indicators..... 52
- ❖ Using ProSoft Configuration Builder (PCB) for Diagnostics..... 56
- ❖ Reading Status Data from the Module ..... 67

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

- LED status indicators on the front of the module provide general information on the module's status.
- Status data contained in the module can be viewed through the Configuration/Debug port, using the troubleshooting and diagnostic capabilities of *ProSoft Configuration Builder (PCB)*.
- Status data values can be transferred from the module to processor memory and can be monitored there manually or by customer-created logic.

## 4.1 LED Status Indicators

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

LED	Color	Status	Indication
CFG	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.
P1	Green	On	Port not used in application
		Off	Port not used in application
P2	Green	On	Port not used in application
		Off	Port not used in application
APP	Amber	Off	The MVI56-DNPSNET is working normally.
		On	The MVI56-DNPSNET module program has recognized a communication error on one of its DNP ports.
BP ACT	Amber	On	The LED is on when the module is performing a write operation on the backplane.
		Off	The LED is off when the module is performing a read operation on the backplane. Under normal operation, the LED should blink rapidly on and off.
OK	Red/ Green	Off	The card is not receiving any power and is not securely plugged into the rack.
		Green	The module is operating normally.
		Red	The program has detected an error or is being configured. If the LED remains red for over 10 seconds, the program has probably halted. Remove the card from the rack and re-insert the card to restart the module's program.
BAT	Red	Off	The battery voltage is OK and functioning.
		On	The battery voltage is low or battery is not present. Allow battery to charge by keeping module plugged into rack for 24 hours. If BAT LED still does not go off, contact ProSoft Technology, as this is not a user serviceable item.

If the APP, BP ACT and OK LEDs blink at a rate of every one-second, this indicates a serious problem with the module. Call ProSoft Technology support to arrange for repairs.

### 4.1.1 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.

### 4.1.2 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 correct configuration data is being transferred to the module from the ControlLogix controller.

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 Technical Support.

### 4.1.3 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 in the I/O Configuration of RSLogix. 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. A problem could exist between the processor and any installed I/O module, not just the MVI56-DNPSNET. Verify that all modules in the rack are correctly configured in the ladder logic.

## Module Errors

Problem description	Steps to take
BP ACT LED (not present on MVI56E modules) remains OFF or blinks slowly MVI56E modules with scrolling LED display: <i>&lt;Backplane Status&gt;</i> condition reads ERR	<p>This indicates that backplane transfer operations are failing. Connect to the module's Configuration/Debug port to check this.</p> <p>To establish backplane communications, verify the following items:</p> <ul style="list-style-type: none"> <li>▪ The processor is in RUN or REM RUN mode.</li> <li>▪ The backplane driver is loaded in the module.</li> <li>▪ The module is configured for read and write data block transfer.</li> <li>▪ The ladder logic handles all read and write block situations.</li> <li>▪ The module is properly configured in the processor I/O configuration and ladder logic.</li> </ul>
OK LED remains RED	<p>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 it, and then restore power to the rack.</p>

### 4.1.4 Error Status Table

The program maintains an error/status table that is transferred to the processor in each read block. 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	Variable Name	Description
0	Scan Counter	Program scan counter incremented each time the program loop is executed.
1 to 2	Product Name (ASCII)	These two words contain the product name of the module in ASCII format.
3 to 4	Revision (ASCII)	These two words contain the product revision level of the firmware in ASCII format.
5 to 6	Operating System Revision (ASCII)	These two words contain the module's internal operating system revision level in ASCII format.
7 to 8	Production Run Number (ASCII)	These two words contain the production "batch" number for the particular chip in the module in ASCII format.
9	Read Block Count	Total number of blocks transferred from the module to the processor.
10	Write Block Count	Total number of blocks transferred from the processor to the module.
11	Parse Block Count	Total number of blocks parsed by the module that were received from the processor.
12	Block number error	Number of BTW requests that resulted in an incorrect BTW identification code.
13	DNP Slave Port total number of message frames received by slave	This value represents the total number of message frames that have matched this slaves address on this port. This count includes message frames which the slave may or may not be able to parse and respond.
14	DNP Slave Port total number of response message frames sent from slave	This value represents the number of good (non-error) responses that the slave has sent to the master on this port. The presumption is that if the slave is responding, the message was good. Note: This is a frame count.

Word	Variable Name	Description
15	DNP Slave Port total number of message frames seen by slave	This value represents the total number of message frames received by the slave, regardless of the slave address.
16	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.
17	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.
18	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.
19	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.
20	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.
21	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.
22	DNP Slave address error (Layer Error) (Transport)	This value counts the number of times the source addresses contained in a multi-frame request fragments do not match.
23	DNP Slave Binary Input Event count	This value contains the total number of binary input events which have occurred.
24	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.
25	DNP Slave Analog Input Event count	This value contains the total number of analog input events which have occurred.
26	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.
27	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.
28	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.
29	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.
30	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.
31	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.
32	Free Memory LSB Free Memory MSB	Free memory in module

## 4.2 Using ProSoft Configuration Builder (PCB) for Diagnostics

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 submenus for each menu command. The first menu you see when you connect to the module is the *Main* menu.

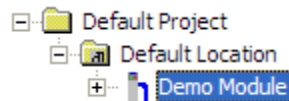
Because this is a text-based menu system, you enter commands by typing the [command letter] from your computer keyboard in the *Diagnostic* window in *ProSoft Configuration Builder (PCB)*. The module does not respond to mouse movements or clicks. The command executes as soon as you press the [COMMAND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

### 4.2.1 Using the Diagnostic Window in ProSoft Configuration Builder

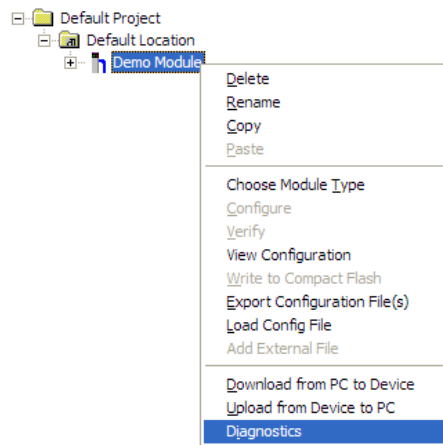
**Tip:** You can have a ProSoft Configuration Builder Diagnostics window open for more than one module at a time.

#### 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.



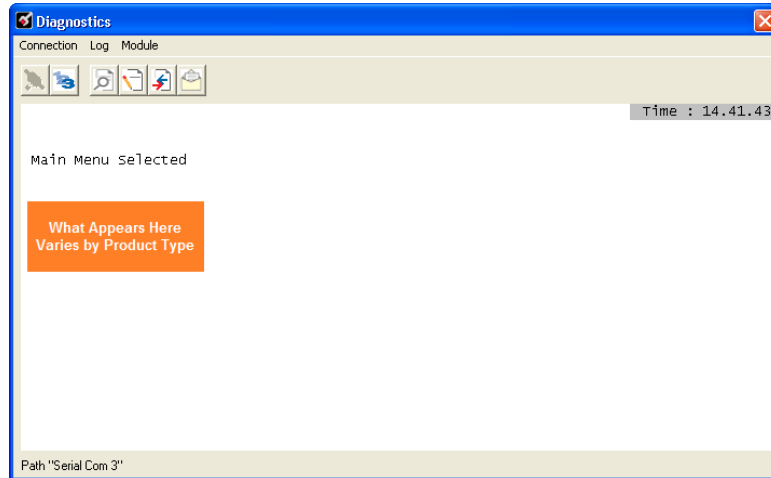
- 2 On the shortcut menu, choose **DIAGNOSTICS**.



This action opens the *Diagnostics* dialog box.

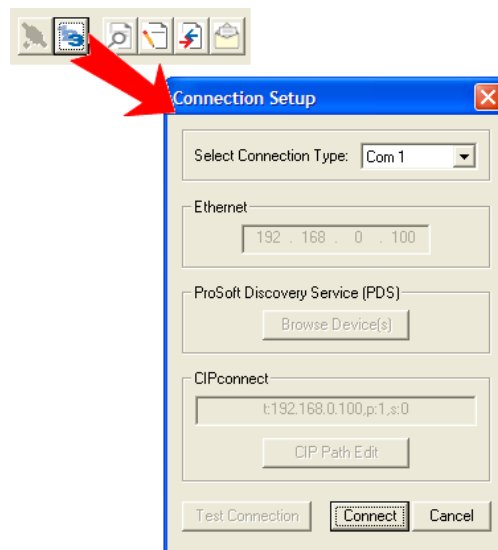


- 3 Press [?] to open the *Main* menu.



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

- 1 Click to configure the connection. On the *Connection Setup* dialog box, select a valid com port or other connection type supported by the module.



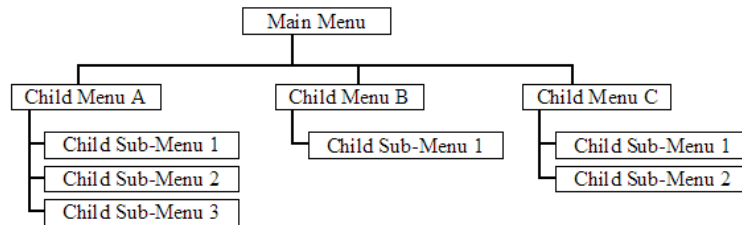
- 2 Verify that the null modem cable is connected properly between your computer's serial port and the module. A regular serial cable will not work.
- 3 On computers with more than one serial port, verify that your communication program is connected to the same port that is connected to the module.

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

### 4.2.2 Navigation

All of the submenus for this module contain commands to redisplay the menu or return to the previous menu. You can always return from a submenu 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 the menus available for this module, and briefly discusses the commands available to you.

#### Keystrokes

The keyboard commands on these menus are usually not case sensitive. You can enter most commands in lowercase or uppercase 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** and **/**.

Also, take care to distinguish the different uses for uppercase letter "eye" (**I**), lowercase letter "el" (**L**), and the number one (**1**). Likewise, uppercase letter "oh" (**O**) and the number zero (**0**) are not interchangeable. Although these characters look alike on the screen, they perform different actions on the module and may not be used interchangeably.

### 4.2.3 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.

```
DNP ETHERNET SERVER COMMUNICATION MODULE
?=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 [**B**] from the *Main* menu to view the *Block Transfer Statistics* screen.

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

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

#### 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 View Menu

Press [**D**] to open the *Database View* menu.

Use this menu command to view the current contents of the module's database. For more information about this submenu, see Database View Menu (page 63).

#### 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 (page 61).

#### Receiving the Configuration File

Press **[R]** to download (receive) the current configuration file from the module.

#### Sending the Configuration File

Press **[S]** to upload (send) a configuration file from the module to your PC.

#### Viewing Version Information

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

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

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

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

#### Warm Booting the Module

Press **[W]** from the *Main* menu to warm boot (restart) the module.

This command will cause the program to exit and reload, refreshing configuration parameters that must be set on program initialization. Only use this command if you must force the module to reboot.

#### Opening the Network Menu

Press **[@]** to open the *Network* menu.

The *Network* menu allows you to send, receive and view the WATTCP.CFG file that contains the IP, gateway and other network specification information. For more information about this submenu, see Network Menu (page 65).

### Exiting the Program

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.

### **4.2.4 DNP Menu**

When you press the **[I]** key, this opens the DNP Ethernet Protocol Menu. After the option is selected, press the **[?]** key to display the menu and the following is displayed:

Each option on the menu is discussed in the following topics.

#### Viewing DNP Set Up & Pointers

Press **[B]** to display the memory allocation and the database setup parameters.

#### Viewing DNP Configuration

Press **[C]** to display the configuration information for the server. Use this command to confirm that the module is configured as desired. If any parameter is not set correctly, adjust the configuration file and download the altered file to the unit.

#### Opening the DNP Database View Menu

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

#### Viewing a List of Valid Hosts

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

#### Returning to the Main Menu

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

#### Viewing DNP Communication Status

Press **[1]** to view DNP Communication Status. Use this command to view the communication status data for the DNP driver.

Viewing TCP Socket Status

Press **[2]** to view the status of the TCP socket in the module. After selecting the option, the following is displayed:

```
TCP SOCKET STATUS
Rx Count      : -20148
Tx Count      : -20146
Tx State      : 0
TCP State     : 1
Busy Flag     : 0
App Frame     : 0
Tx Frame      : 1
Packet Length : 0
```

The parameters displayed have the following definitions:

**Rx Count** - Number of messages received on TCP socket

**Tx Count** - Number of messages transmitted on TCP socket

**Tx State** - 0=not transmitting, 1=transmitting

**TCP State** - Value used for TCP/IP socket state machine

**Busy Flag** - 0=not busy, 1=TCP has control of DNP server, 2=UDP has control of DNP server, 3=Unsolicited message being sent

**App Frame** - 0=no application data frame data, 1=application data available

**Tx Frame** - 0=Data link level frame ready to send, 1=Data link level message not ready to send

**Packet Length** - Length of message left to process

Viewing UDP Socket Status

Press **[3]** to view the status of the UDP socket in the module. After selecting the option, the following is displayed:

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

The parameters displayed have the following definitions:

**Rx Count** - Number of messages received on UDP socket

**Tx Count** - Number of messages transmitted on UDP socket

**Tx State** - 0=not transmitting, 1=transmitting

**TCP State** - Value used for UDP/IP socket state machine

**Busy Flag** - 0=not busy, 1=TCP has control of DNP server, 2=UDP has control of DNP server, 3=Unsolicited message being sent

**App Frame** - 0=no application data frame data, 1=application data available

**Tx Frame** - 0=Data link level frame ready to send, 1=Data link level message not ready to send

**Packet Length** - Length of message left to process

### 4.2.5 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 database. Press **[?]** to view a list of commands available on this menu.

```
DB Menu selected

DATABASE VIEW MENU
?=Display Menu
0-9=Display 0-9000
S=Show Again
-=Back 5 Pages
P=Previous Page
+=Skip 5 Pages
N=Next Page
D=Decimal Display
H=Hexadecimal Display
F=Float Display
A=ASCII Display
M=Main Menu
```

#### Viewing Register Pages

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

Command	Description
<b>[0]</b>	Display registers 0 to 99
<b>[1]</b>	Display registers 1000 to 1099
<b>[2]</b>	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
```

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 100 registers of data starting 500 registers before the currently displayed page.

*Moving Forward (Skipping) Through 5 Pages of Registers*

Press **[+]** from the *Database View* menu to skip five pages ahead in the database to see the 100 registers of data starting 500 registers after the currently displayed page.

*Viewing the Previous Page of Registers*

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

*Viewing the Next Page of Registers*

Press **[N]** from the *Database View* menu to display the next page of data.

*Viewing Data in Decimal Format*

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

*Viewing Data in Hexadecimal Format*

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

*Viewing Data in Floating-Point Format*

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

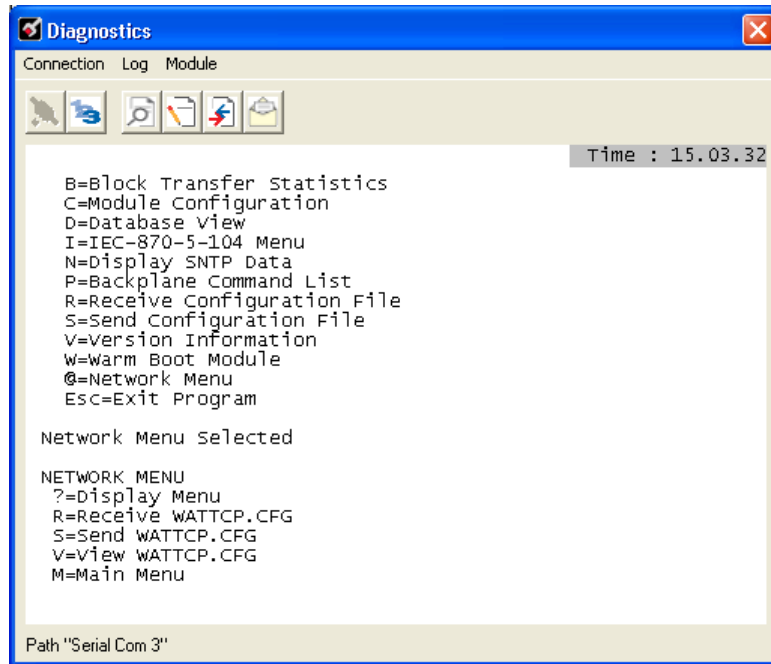
*Returning to the Main Menu*

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



## 4.2.6 Network Menu

From the *IEC-870-5-104 Server* menu press [**@**] to display the *IEC-870-5-104 Network* menu screen. The *Network* menu allows you to send, receive, and view the WATTCP.CFG file that contains the IP and module addresses, and other network information.



### Transferring WATTCP.CFG to the Module

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.

### Transferring WATTCP.CFG to the PC

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.

Viewing the WATTCP.CFG File on the module

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

```
WATTCP.CFG FILE:
# ProLinx Communication Gateways, Inc.
# Default private class 3 address
my_ip=192.168.0.75
# Default class 3 network mask
netmask=255.255.255.0
# name server 1 up to 9 may be included
# nameserver=xxx.xxx.xxx.xxx
# name server 2
# nameserver=xxx.xxx.xxx.xxx
# The gateway I wish to use
gateway=192.168.0.1
# some networks (class 2) require all three parameters
# gateway,network,subnetmask
# gateway 192.168.0.1,192.168.0.0,255.255.255.0
# The name of my network
# domainlist="mynetwork.name"
```

Returning to the Main Menu

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

### 4.3 Reading Status Data from the Module

The MVI56-DNPSNET module provides the status data in each read block. This data can also be located in the module's database. For a complete listing of the status data object, refer to the **Module Set Up** section.

## 5 Reference

### *In This Chapter*

❖ Product Specifications .....	69
❖ Functional Overview .....	72
❖ MVI56-DNPSNET Application Design .....	86
❖ Cable Connections .....	100
❖ Configuration Data .....	105
❖ MVI56-DNPSNET Status Data .....	109
❖ Internal Indication Bits (IIN Bits) for DNP Server .....	112
❖ DNP Subset Definition .....	113
❖ Device Profile .....	119
❖ Event Size Computation .....	121

## 5.1 Product Specifications

The MVI56 DNP 3.0 Server over Ethernet Communications Module supports the implementation of the DNP 3.0 (Distributed Network Protocol) over Ethernet, allowing ControlLogix processors to easily communicate with host systems supporting the protocol. The module supports DNP Subset Level 2 features and some Level 3 features.

### 5.1.1 General Specifications

- Single Slot - 1756 backplane compatible
- The module is recognized as an Input/Output module and has access to processor memory for data transfer between processor and module.
- Ladder Logic is used for data transfer between module and processor. Sample ladder file included.
- Configuration data obtained from configuration text file downloaded to module. Sample configuration file included
- Local or remote rack

### 5.1.2 Hardware Specifications

Specification	Description
Backplane Current Load	800 mA @ 5 Vdc 3 mA @ 24 Vdc
Operating Temperature	32°F to 140°F (0° C to 60°C)
Storage Temperature	-40°F to 185°F (-40° C to 85°C)
Shock	30 g operational 50 g non-operational Vibration: 5 g from 10 Hz to 150 Hz
Relative Humidity	5% to 95% (with no condensation)
LED Indicators	Module Status Backplane Transfer Status Application Status Serial Activity
<b>Application port (Ethernet)</b>	
Ethernet Port (Ethernet modules)	10/100 Base-T 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
Shipped with Unit	RJ45 to DB-9M cables for each port 6-foot RS-232 configuration cable
<b>Debug/Configuration port (CFG)</b>	
CFG Port (CFG)	RJ45 (DB-9M with supplied cable) No hardware handshaking

### 5.1.3 Functional Specifications

The MVI56-DNPSNET module accepts DNP commands to control and monitor the data stored in the DNP databases. This data is passed between the module and the ControlLogix processor over the backplane for use in user applications.

- DNP databases to house data for the slave port supporting the following maximum input counts

Binary input	8000 points (500 words)
Binary output	8000 points (500 words)
Counter	250 (500 words)
Analog input	500
Analog output	500
Float Input	150
Float Output	150

- User-definable module memory usage up to maximum point counts
- Data movement between module using input/output image
- Ethernet port supports both TCP and UDP over Ethernet
- Supports DNP 3.0 in a level 2 implementation
- Supports sending of input event data from the ladder to the module
- Supports time synchronization from/to processor
- Configurable via text file
- Status and error information
- All data in the DNP slave is contained in user-defined files

## 5.2 Functional Overview

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>.

### 5.2.1 General Concepts

The following discussion explains several concepts that are important for understanding module operation.

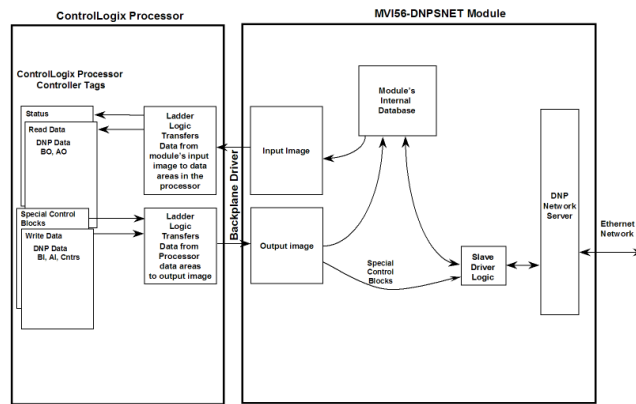
#### Backplane Data Transfer

The MVI56-DNPSNET module communicates directly over the ControlLogix backplane. Data is paged between the module and the ControlLogix processor across the backplane using the module's input and output images. The update frequency of the images is determined by the scheduled scan rate defined by the user for the module and the communication load on the module. Typical updates are in the range of 1 to 10 milliseconds.

This bi-directional transference of data is accomplished by the module filling in data in the module's input image to send to the processor. Data in the input image is placed in the Controller Tags in the processor by the ladder logic. The input image for the module is set to 250 words. This large data area permits fast throughput of data between the module and the processor.

The processor inserts data to the module's output image to transfer to the module. The module's program extracts the data and places it in the module's internal database. The output image for the module is set to 248 words. This large data area permits fast throughput of data from the processor to the module.

The following illustration shows the data transfer method used to move data between the ControlLogix processor, the MVI56-DNPSNET module and the DNP Network.





All data transferred between the module and the processor over the backplane is through the input and output images. Ladder logic must be written in the ControlLogix processor to interface the input and output image data with data defined in the Controller Tags. All data used by the module is stored in its internal databases. These databases are defined as a virtual DNP data tables with addresses from 0 to the maximum number of points for each data type. The following illustration shows the layout of the databases:

<b>Data Area</b>	
DNP DATA	BINARY INPUTS
	ANALOG INPUTS
	COUNTER DATA
	BINARY OUTPUTS
	ANALOG OUTPUTS
FROZEN DATA	FROZEN COUNTER DATA
LAST VALUE DATA	BINARY INPUTS
	ANALOG INPUTS
EVENT DATA	BINARY INPUT EVENTS
	ANALOG INPUT EVENTS

Data contained in this database is paged through the input and output images by coordination of the ControlLogix ladder logic and the MVI56-DNPSNET module's program. Up to 248 words of data can be transferred from the module to the processor at a time. Up to 247 words of data can be transferred from the processor to the module.

Each block transferred from the module to the processor or from the processor to the module contains a block identification code that describes the content of the block.

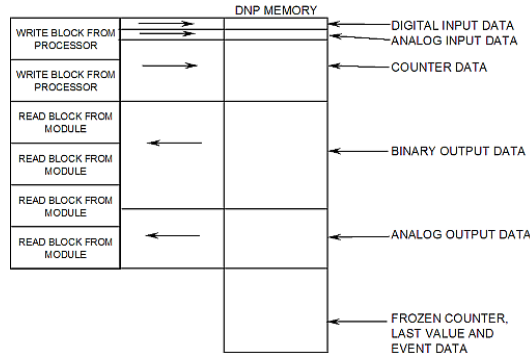
The following table defines the blocks used by this module:

<b>Block Number</b>	<b>Function/Description</b>
0 or -1	Dummy Blocks: Used by module when no data is to be transferred
1 to 45	DNP Data blocks
1000 to 1044	Output initialization blocks
9958	PLC Binary Input Event data
9959	PLC Analog Input Event Data
9970	Set PLC time using module's DNP time
9971	Set module's time using PLC time
9998	Warm Boot Request from PLC (Block contains no data)
9999	Cold Boot Request from PLC (Block contains no data)

Blocks 1 to 45 transfer data between the module and the processor. Blocks 1000 to 1044 are utilized to transfer the initial output databases (binary and analog output data) from the processor to the module at startup. Blocks 9958 to 9999 are used for command control of the module. Each group of blocks are discussed in the following topics.

Normal Data Transfer

Normal data transfer includes the paging of the user data found in the module's internal databases between the module and the controller. These data are transferred through read (input image) and write (output image) blocks. Refer to the **Module Set Up** section for a description of the data objects used with the blocks and the ladder logic required. Each data block transferred between the module and the processor has a specific block identification code that defines the data set contained in the block. The following illustration shows the direction of movement of the DNP data types between the module and the processor:



The structure and function of each block is described in the following topics:

**Read Block**

These blocks of data transfer information from the module to the ControlLogix processor. The following table describes the structure of the input image.

Block Offset	Content
0	Reserved
1	Write block ID
2 to 201	Read data
202 to 248	Status Data
249	Read block ID

The Read Block ID is an index value used to determine the location of where the data will be placed in the ControlLogix processor controller tag array of module read data. Each transfer can move up to 200 words (block offsets 2 to 201) of data.

The Write Block ID associated with the block requests data from the ControlLogix processor. Under normal program operation, the module sequentially sends read blocks and requests write blocks. For example, if two blocks of read data and three blocks of write data are to be moved between the module and the processor, the sequence will be as follows:

R1W1→R2W2→R1W3→R2W1→R1W2→R2W3→R1W1→

This sequence will continue until interrupted by other write block numbers sent by the controller or by a command request from a node on the DNP network or operator control through the module's Configuration/Debug port.

### Write Block

These blocks of data transfer information from the ControlLogix processor to the module. The following table describes the structure of the output image.

Block Offset	Content
0	Write Block ID
1 to 200	Write Data
201 to 247	Spare (Not used)

The Write Block ID is an index value used to determine the location in the module's database where the data will be placed. Each transfer can move up to 200 words (block offsets 1 to 200) of data.

### Trip/Close

The MVI56-DNPSNET module supports Trip/Close functionality for Binary Output points.

This allows Trip/Close commands to be sent to the MVI56-DNPSNET module, for dual point control. Each DNP Trip/Close command will occupy 2 bits within the module memory.

This does overlap the regular pulse on/off and latch on/off Binary Output database, therefore special consideration must be used to make sure that points are not used twice.

The following table describes the address mapping for the module using Latch and Pulse commands, and Trip/Close functionality.

DNP BO Database Point	BO Latch/Pulse Point	BO Trip/Close Point
0	BO 0	Close BO 0
1	BO 1	Trip BO 0
2	BO 2	Close BO 1
3	BO 3	Trip BO 1
4	BO 4	Close BO 2
5	BO 5	Trip BO 2
100	BO 100	Close BO 50
101	BO 101	Trip BO 50
1000	BO 1000	Close BO 500
1001	BO 1001	Trip BO 500
2046	BO 2046	Close BO 1023
2047	BO 2047	Trip BO 1023

As you can see from the above chart, trip/close requires 2 points within the module's DNP database. A trip is represented by the value of 2#10 for those 2 points, and a close is represented by the value of 2#01 for those same 2 points.

The module can only process 4000 trip/close points, as the database for the DNP BO is limited to 8000 bits total.

Special Function Blocks

Special Function blocks are blocks used to control the module or request special data from the module. The current version of the software supports several special function blocks each of which is discussed in the following topics:

**Binary Input Event Block (9958)**

If the PLC sends a block 9958, the module will place the binary input event data in the block into the event buffer and alter the data values for the points in the DNP binary input database. The format for the event message is shown in the following table.

Word Offset in Block	Data Field(s)	Description
0	Block ID	This field contains the value of 9958 identifying the event block to the module.
1	Event Count	This field contains the number of events contained in the block. Valid values for this field are 1 to 12.
2	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	DNP Binary Input Data point	This is the data point in the DNP binary input database represented by the event.
4	Month/Day/State	Formatted: bits 0 to 4 = Day, bits 8 to 11 = Month, bit 15 = digital state for point. All other bits are ignored.
5	Hour/Minute	Formatted: bits 0 to 5 = Minutes, bits 8 to 12 = Hour. All other bits are ignored.
6	Sec/Millisecond	Formatted: bits 0 to 9 = Milliseconds, bits 10 to 15 = Seconds.
7	Year	This is the four digit year for the event.
8 to 12		Five words of data for Event #2.
13 to 17		Five words of data for Event #3.
18 to 22		Five words of data for Event #4.
23 to 27		Five words of data for Event #5.
28 to 32		Five words of data for Event #6.
33 to 37		Five words of data for Event #7.
38 to 42		Five words of data for Event #8.
43 to 47		Five words of data for Event #9.
48 to 52		Five words of data for Event #10.
53 to 57		Five words of data for Event #11.
58 to 62		Five words of data for Event #12.
63 to 247	Spare	Not Used

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 send a response read block 9958 to the PLC. The following table describes the format of the block.

Word Offset in Block	Data Field(s)	Description
0	Reserved	Reserved (0)
1	Block ID	Block identification code for request from PLC by 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.
4 to 248	Spare	Not used
249	Block ID	Identification code for block set to 9958.

The sequence counter field in the returned block is set to the last successfully processed block 9958 from the PLC. Compare this value to that sent by the PLC. If the values match, the events can be removed from the PLC. If the values do not match, or the PLC does not receive a 9958 block, the PLC must re-send the block.

### Analog Input Event Block (9959)

If the PLC sends a block 9959, the module 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. The format for the event message is shown in the following table.

Word Offset in Block	Data Field(s)	Description
0	Block ID	This field contains the value of 9959 identifying the event block to the module.
1	Event Count	This field contains the number of events contained in the block. Valid values for this field are 1 to 10.
2	Sequence Counter	This field holds the sequence counter for each 9959 block transfer. This synchronizes and confirms receipt of the block by the module.
3	DNP Analog Input Data point	This is the data point in the DNP analog input database represented by the event.
4	Analog Input Value	This is the new analog input value represented in the event.
5	Month/Day	Formatted: bits 0 to 4 = Day, bits 8 to 11 = Month. All other bits are ignored.
6	Hour/Minute	Formatted: bits 0 to 5 = Minutes, bits 8 to 12 = Hour. All other bits are ignored.
7	Sec/Millisecond	Formatted: bits 0 to 9 = Milliseconds, bits 10 to 15 = Seconds.
8	Year	Four digit year value for event.
9 to 14		Six words of data for Event #2.
15 to 20		Six words of data for Event #3.
21 to 26		Six words of data for Event #4.
27 to 32		Six words of data for Event #5.
33 to 38		Six words of data for Event #6.
39 to 44		Six words of data for Event #7.
45 to 50		Six words of data for Event #8.
51 to 56		Six words of data for Event #9.
57 to 62		Six words of data for Event #10.
63 to 247	Spare	Not Used

Up to 10 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 send a response read block 9959 to the PLC. The following table describes the format of the block.

Word Offset in Block	Data Field(s)	Description
0	Reserved	Reserved (0)
1	Block ID	Block identification code for request from PLC by 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 9959 received.
4 to 248	Spare	Not used
249	Block ID	Identification code for block set to 9959.

The sequence counter field in the returned block is set to the last successfully processed block 9959 from the PLC. Compare this value to that sent by the PLC. If the values match, the events can be removed from the PLC. If the values do not match, or the PLC does not receive a 9959 block, the PLC must re-send the block.

#### **SOE Input Events Block (9961)**

Block 9961 identification code is used by the PLC to send a set of SOE input events to the module that use 64-bit time. This is based at a starting point of January 1st, 1972. This block is for RS Logix versions prior to version 16.

Block Format for Write (9961)

Word Offset in Block	Data Field(s)	Description
0	Block ID	This field contains the value of 9961 identifying the SOE event block to the module.
1	Event Count	This field contains the number of events contained in the block. Valid values for this field are 1 to 20.
2	Sequence Counter	This field is used to hold the sequence counter for each 9961 block transfer. This is used to synchronize and confirm receipt of the block by the module.
3	DNP Binary Input Data point	This is the data point in the DNP binary input database represented by the event.
4 to 7	64-bit time	This is the 64-bit time value generated by the SOE module.
8	Value	This is the value for the event data. It is either a 0 or 1.
9 to 14		Six words of data for Event #2.
15 to 20		Six words of data for Event #3.
21 to 26		Six words of data for Event #4.
27 to 32		Six words of data for Event #5.
33 to 38		Six words of data for Event #6.
39 to 44		Six words of data for Event #7.
45 to 50		Six words of data for Event #8.
51 to 56		Six words of data for Event #9.
57 to 62		Six words of data for Event #10.
63 to 68		Six words of data for Event #11.
69 to 74		Six words of data for Event #12.
75 to 80		Six words of data for Event #13.
81 to 86		Six words of data for Event #14.
87 to 92		Six words of data for Event #15.
93 to 98		Six words of data for Event #16.
99 to 104		Six words of data for Event #17.
105 to 110		Six words of data for Event #18.
111 to 116		Six words of data for Event #19.
117 to 122		Six words of data for Event #20.
123 to 247	Spare	Not Used

To insure the receipt of this block of information, the module returns a BTR block 9961 with the sequence counter set to the value of the last successful block 9961 received.



Block Format for Read (9961)

<b>Word Offset in Block</b>	<b>Data Field(s)</b>	<b>Description</b>
0	Reserved	Reserved (0)
1	Block ID	Block identification code for request from PLC by 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 9961 received.
4 to 248	Spare	Not used
249	Block ID	Identification code for block set to 9961.

**SOE Input Events Block (9962)**

Block 9962 identification code is used by the PLC to send a set of SOE input events to the module that use 64-bit time. This is based at a starting point of January 1st, 1970. This block is for RS Logix versions 16 and later.

Block Format for Write (9962)

Word Offset in Block	Data Field(s)	Description
0	Block ID	This field contains the value of 9962 identifying the SOE event block to the module.
1	Event Count	This field contains the number of events contained in the block. Valid values for this field are 1 to 20.
2	Sequence Counter	This field is used to hold the sequence counter for each 9961 block transfer. This is used to synchronize and confirm receipt of the block by the module.
3	DNP Binary Input Data point	This is the data point in the DNP binary input database represented by the event.
4 to 7	64-bit time	This is the 64-bit time value generated by the SOE module.
8	Value	This is the value for the event data. It is either a 0 or 1.
9 to 14		Six words of data for Event #2.
15 to 20		Six words of data for Event #3.
21 to 26		Six words of data for Event #4.
27 to 32		Six words of data for Event #5.
33 to 38		Six words of data for Event #6.
39 to 44		Six words of data for Event #7.
45 to 50		Six words of data for Event #8.
51 to 56		Six words of data for Event #9.
57 to 62		Six words of data for Event #10.
63 to 68		Six words of data for Event #11.
69 to 74		Six words of data for Event #12.
75 to 80		Six words of data for Event #13.
81 to 86		Six words of data for Event #14.
87 to 92		Six words of data for Event #15.
93 to 98		Six words of data for Event #16.
99 to 104		Six words of data for Event #17.
105 to 110		Six words of data for Event #18.
111 to 116		Six words of data for Event #19.
117 to 122		Six words of data for Event #20.
123 to 247	Spare	Not Used

To insure the receipt of this block of information, the module returns a BTR block 9962 with the sequence counter set to the value of the last successful block 9962 received.

Block Format for Read (9962)

Word Offset in Block	Data Field(s)	Description
0	Reserved	Reserved (0)
1	Block ID	Block identification code for request from PLC by 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 9962 received.
4 to 248	Spare	Not used
249	Block ID	Identification code for block set to 9962.

**Set Processor Time Block (9970)**

This block transfers the module's time to the ControlLogix processor. Ladder logic must be used to set the processor's clock using the data received. The format of the block sent from the PLC has the following format:

Word Offset in Block	Data Field(s)	Description
0	Block ID	This field contains the value of 9970 identifying the block type to the module.
1 to 247	Not Used	Not Used

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

Word Offset in Block	Data Field(s)	Description
0	Reserved	Reserved (0)
1	Block Write ID	This is the next block requested by the module.
2	Year	This field contains the four-digit year for 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.
10 to 248	Not Used	Not Used
249	Block Read ID	This field contains the block identification code of 9970 for the block.

**Set Module Time Block (9971)**

This block sets the clock in the module to match the clock in the ControlLogix processor. If the PLC sends a block 9971, the module will set its time using the data contained the block. The following table describes the format of the block.

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 for 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

The module will respond to a valid 9971 block with a read block of the following format:

Word Offset in Block	Data Field(s)	Description
0	Reserved	Reserved (0)
1	Block Write ID	This is the next block requested by the module.
2 to 248	Not Used	Not Used
249	Block Read ID	This field contains the block identification code of 9971 for the block.

**Warm Boot Block (9998)**

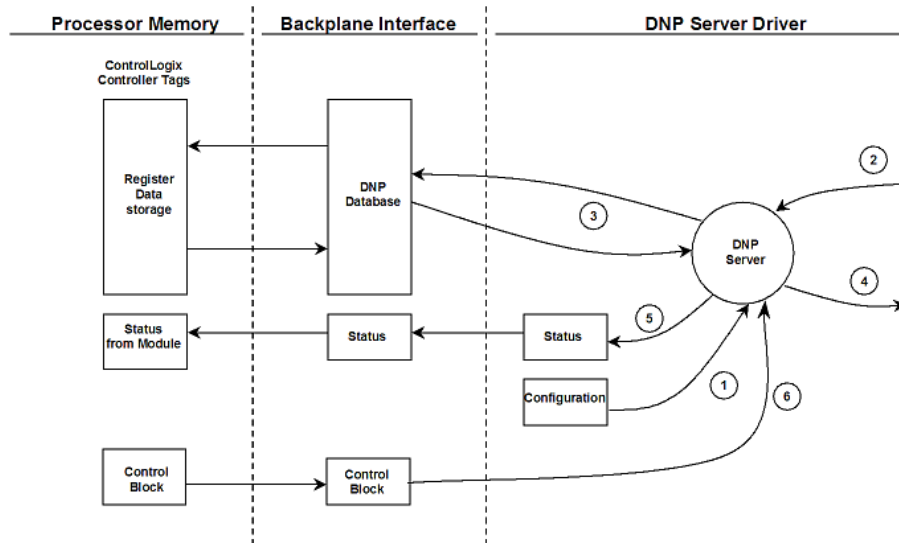
If the ControlLogix processor sends a block number 9998, the module will perform a warm-boot operation.

**Cold Boot Block (9999)**

If the ControlLogix processor sends a block number 9999, the application performs the cold-boot operation. The module exits the program and performs a soft restart on the module.

Data Flow Between MVI56-DNPSNET Module and the ControlLogix Processor

The following topics describe the flow of data between the two pieces of hardware (ControlLogix processor and MVI56-DNPSNET module) and other nodes on the DNP network. The DNP Server Driver allows the MVI56-DNPSNET module to respond to data read and write commands issued by a master on the DNP network. The following flow chart and associated table describe the flow of data into and out of the module.



Step	Description
1	The configuration information for the module is retrieved from the DNPSNET.CFG file on the Compact Flash Disk. This information configures the module and define the Ethernet node characteristics.
2	A Host device (DNP Master unit) issues a read or write command to the module's node address. The driver qualifies the message before accepting it into the module.
3	After the module accepts the command, the data is immediately transferred to or from the appropriate internal database in the module. If the command is a read command, the data is read out of the database and a response message is built. If the command is a write command, the data is written directly into the database and a response message is built.
4	After the data processing has been completed in Step 3, the response is issued to the originating master node.
5	Counters are available in the Status Block that permit the ladder logic program to determine the level of activity of the Slave Driver.
6	The module constantly monitors for command control blocks from the processor. If a valid block is received, the function is executed. Additionally, data is constantly being exchanged between the module and the processor.

Review the **Module Configuration** section for a complete list of the parameters that must be defined for a slave port.

### 5.3 MVI56-DNPSNET Application Design

This documentation describes the MVI56-DNPSNET module configuration and setup as it applies to application design. Before attempting to implement this module with a DNP network, verify that the whole design of the system is complete. This includes definition of all the data types and point counts required for each type, all communication parameters required for the network including media type and the use of advanced features such as unsolicited messaging. These must be defined for all master and slave devices on the network. Additionally, the DNP Device Profiles and DNP Subset Definition documents for each device must be reviewed to make sure all the devices will interact on the network as expected. Failure to fully understand these important documents for all devices on the network will usually lead to many problems when implementing the design.

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

In order to implement a solution using the module, the ControlLogix processor must be set up using predefined user data structures. The data transfer interface requires ladder logic in order to interface data in the module with that in the processor. The program required for data transfer is developed in ladder and is discussed in the **Module Set Up** section. This program will interact with the module by sending and receiving data and issuing special control commands.

Data tags in the ControlLogix processor contain the data to be used by the module and the configuration information is stored in the text file, DNPSNET.CFG, stored on the module's Compact Flash Disk. 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.

#### 5.3.1 *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

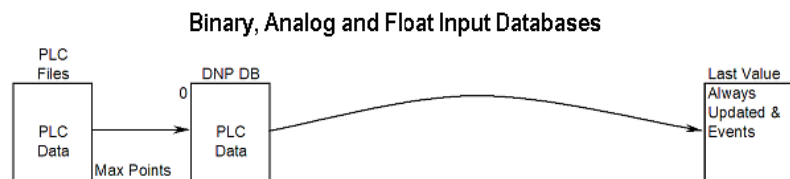
This phase of design defines what data elements are to be interfaced in the ControlLogix processor with the DNP master. The module provides the following data types: digital input, digital output, counter, analog input, analog output, float input and float output. All communications between the DNP master 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
	FLOAT OUTPUTS
FROZEN DATA	FROZEN COUNTER DATA
LAST VALUE DATA	BINARY INPUTS
	ANALOG INPUTS
	FLOAT INPUTS
EVENT DATA	BINARY INPUT EVENTS
	ANALOG INPUT EVENTS
	FLOAT INPUT EVENTS

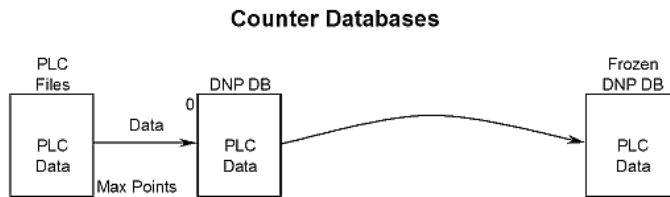
The module is responsible for maintaining the databases using data acquired from the PLC and DNP master attached network port.

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



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

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

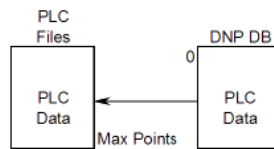


This data is constantly sourced from the processor and placed in the module's internal database. This information is available to the remote master for monitoring. When the module receives a freeze command from the master unit, it will copy the current counter values into the frozen counter database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

**Note:** This data is not sent to the controller, and the zero data can 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, analog and Float output points with the databases.

**Binary, Analog and Float 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

Data is transferred between the ControlLogix processor and the module using module's I/O images. Each block transfer operation transfers up to a maximum of 200 words of data. The other words in the block contain block header identification codes, or not used. The module defines the blocks to be transferred between the PLC and the module when the system is initialized. For the PLC read operations, word 249 of the module's input image identifies the data set contained in the image. Word 1 contains the block index the module is requesting the processor to write. The PLC constructs the write image to send to the module in the module's output image. The first word of the block identifies the data set contained in the block.

The module determines the block numbers required based on the module read and write register counts defined in the configuration file. The user is responsible for defining these parameters and the starting location of these data areas in the module's database correctly. These data must correspond to the DNP database definitions defined. The module stores the data in fixed order for the data types. The size of each data area for each type is determined by the user configuration. An example is given in the following table.

DATA AREA		Cfg	Points	Words	Offset
DNP DATA	BINARY INPUTS	2	32	2	0 to 1
	ANALOG INPUTS	48	48	48	2 to 49
	FLOAT INPUTS	10	10	20	50 to 69
	COUNTER DATA	25	25	50	70 to 119
	BINARY OUTPUTS	4	64	4	120 to 123
	ANALOG OUTPUTS	52	52	52	124 to 175
	FLOAT OUTPUTS	20	20	40	176 to 215

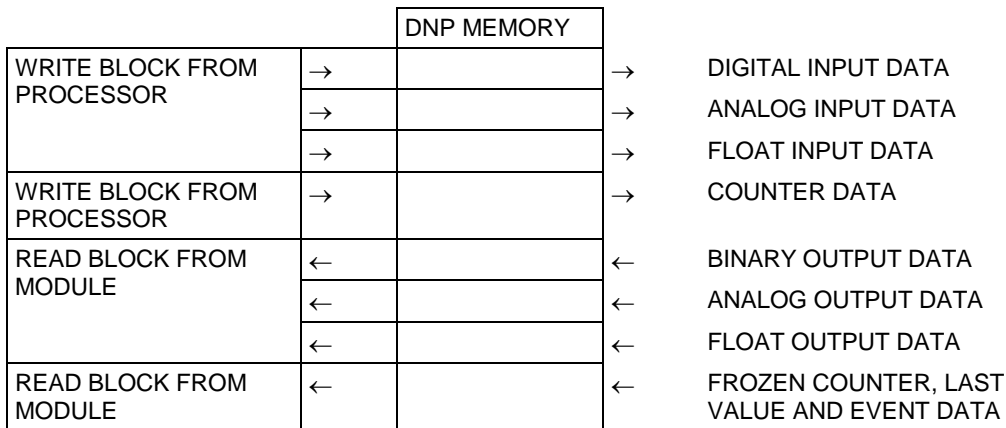
For the example above, 120 registers will be transferred from the processor (all the input data) and 96 registers will be transferred to the processor (all the output data). The data transfer parameters should be defined as follows:

Parameter	Value
Write Register Start:	120
Write Register Count:	96
Read Register Start:	0
Read Register Count:	120

The configuration above will require one block to read and one block to write all the DNP data between the module and the processor.

Note that in one block, one or more data types may be transferred. This is especially important when considering the counter and Float data. They require two registers to store their value. The value of a counter should never be passed in two separate blocks. To avoid this potential problem, always configure the module to have the counter data start on an even word number same rule applies to Float points.

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



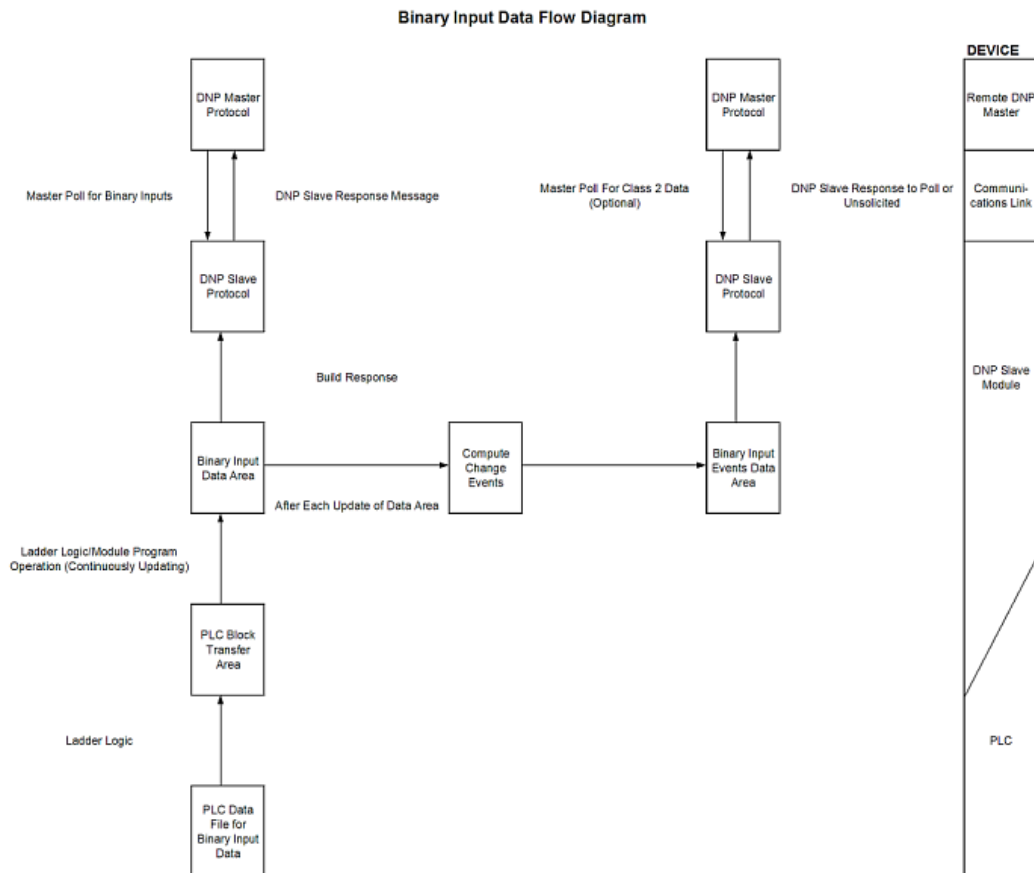
It is important to understand the relationship of the block identifications and the data in the module. Confident data handling in the module is only accomplished if the user defines a consistent set of parameters in the module configuration, handles the read and write operations for the blocks in the module in the PLC ladder logic and understands the requirements of the DNP master unit.

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 16 binary input points). These data are transferred to the module from the PLC using the read operation. Therefore, these data are read-only for the module and the DNP master unit communicating with the module. When the module receives a new block of this data from the PLC, it compares the new values to those currently in the database. If there is a change in any of the data, the module will generate an event message for the points that change.

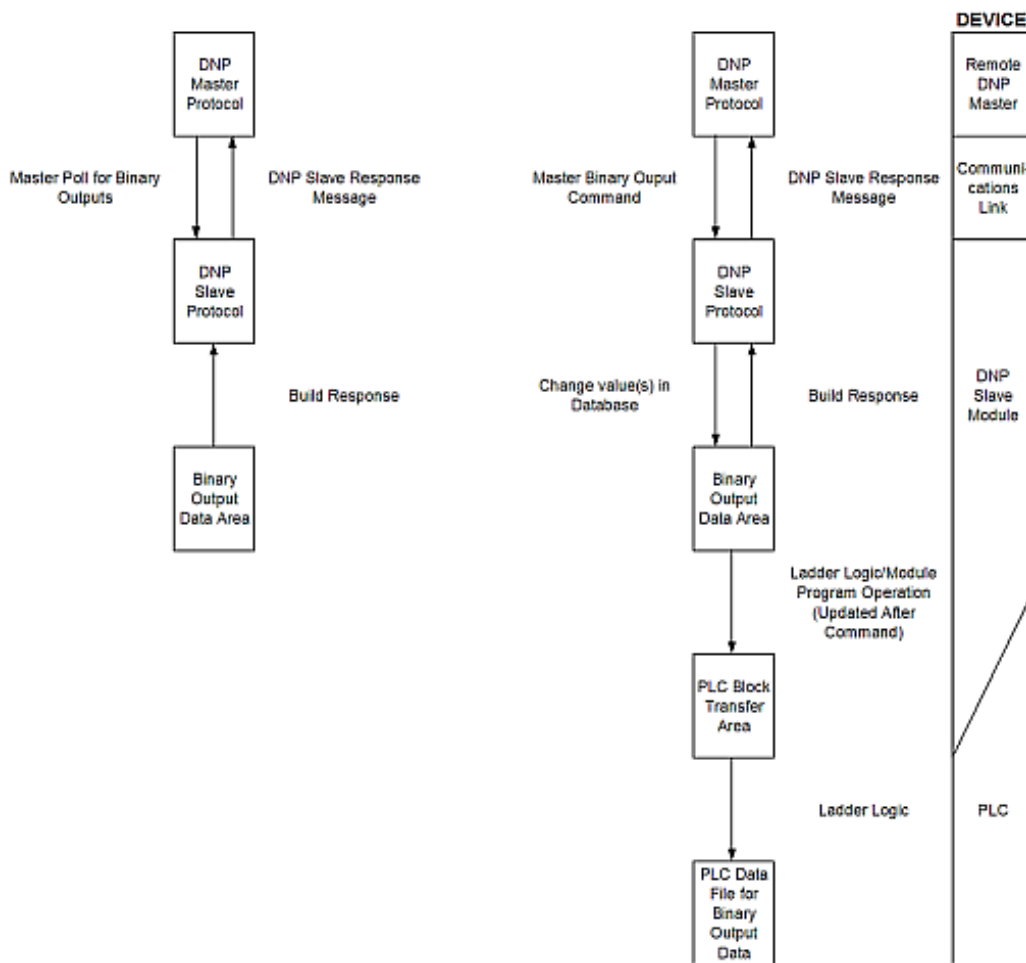
The remote DNP master unit can read the current status data and the event data from the module. Event messages generated by the module can be retrieved using a poll for Class 2 data, as all digital input events are considered a Class 2 data type. If unsolicited message generation is enabled in the application, 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.



DNP Digital Output Data

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

**Binary Output Data Flow Diagram**

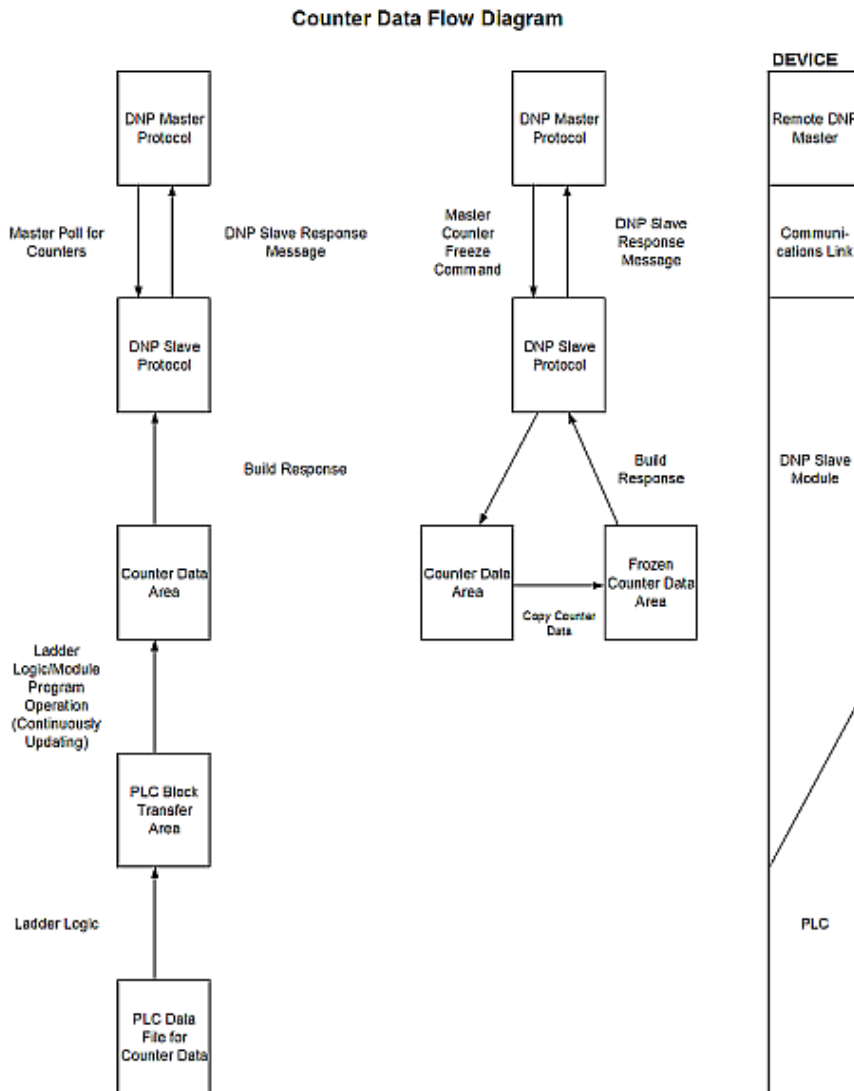


DNP Counter Data

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.

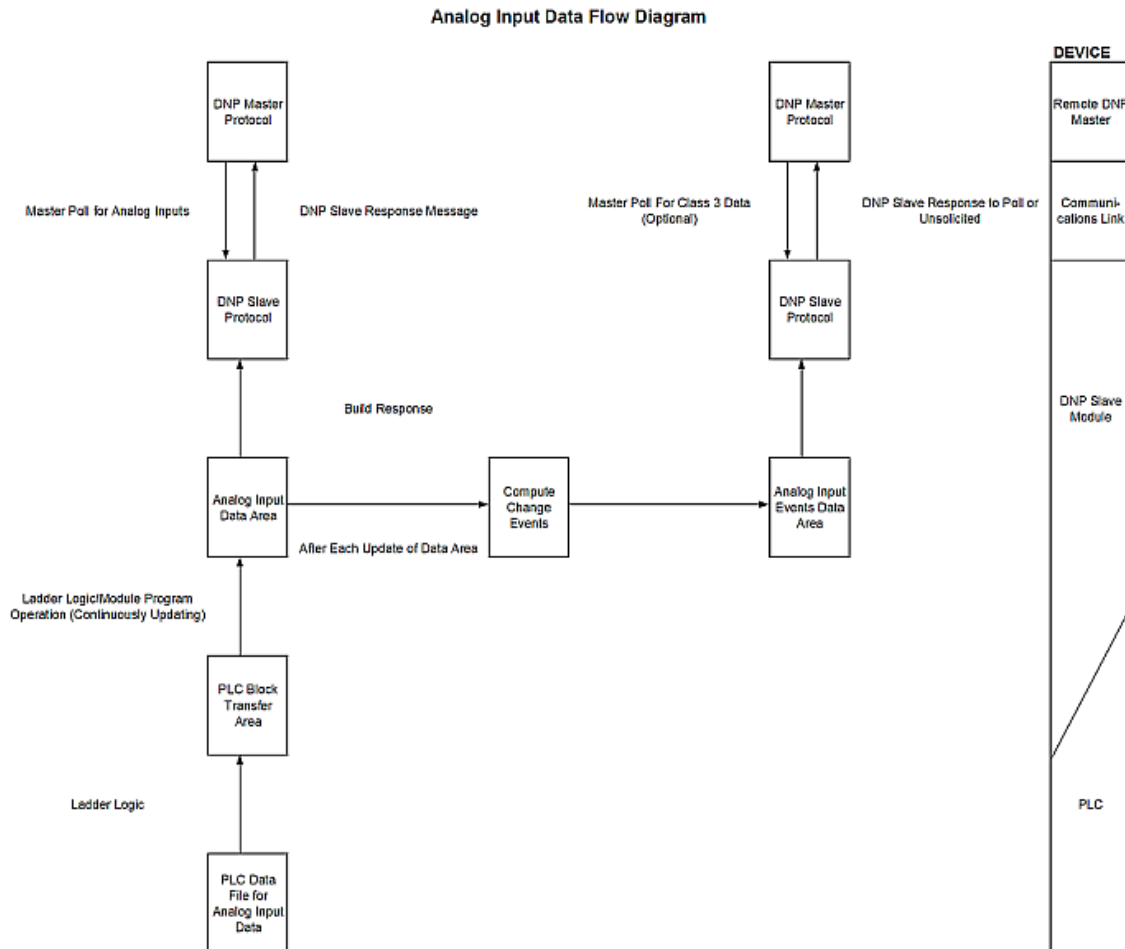
A data flow diagram for the counter data is shown in the following figure.



**DNP Analog Input Data**

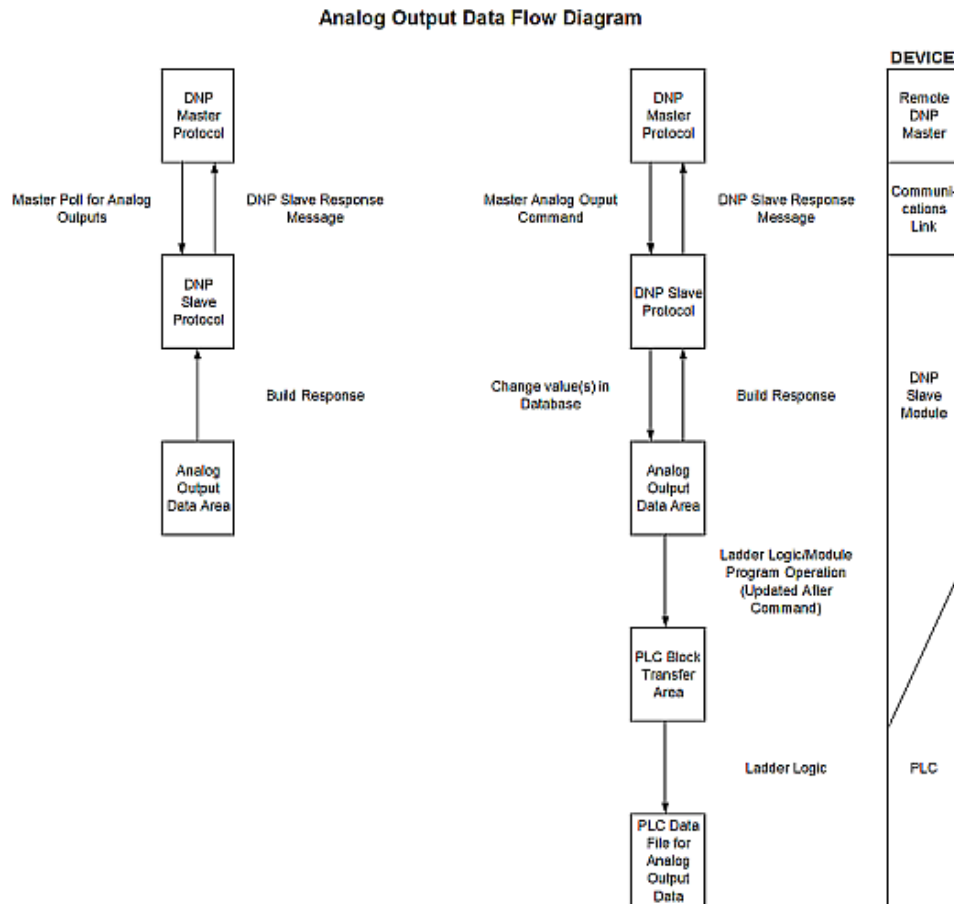
This data type stores analog data with a data range of 0 to 65535 or -32768 to 32767. The size of this data area is determined from the configuration parameter Analog Inputs. These data are transferred to the module from the PLC using the read operation. Therefore, these data are read-only for the module 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.



DNP Analog Output Data

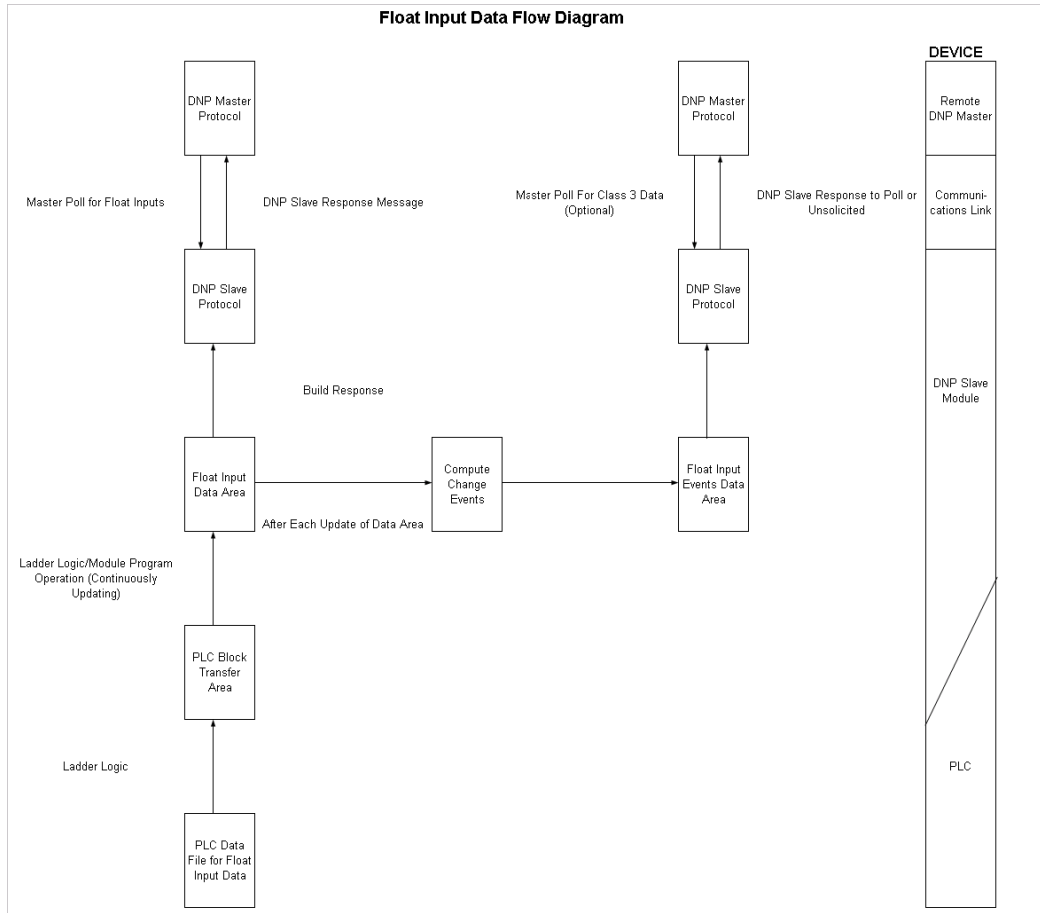
This data type stores analog values sent from the DNP master unit to the module and PLC with a data range of 0 to 65535 or -32768 to 32767. The size of this data area is determined from the configuration parameter Analog Outputs. These data are transferred from the module to the PLC using the write operation. 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.



**DNP Float Input Data**

This data type stores float data. The size of this data area is determined from the configuration parameter float 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.

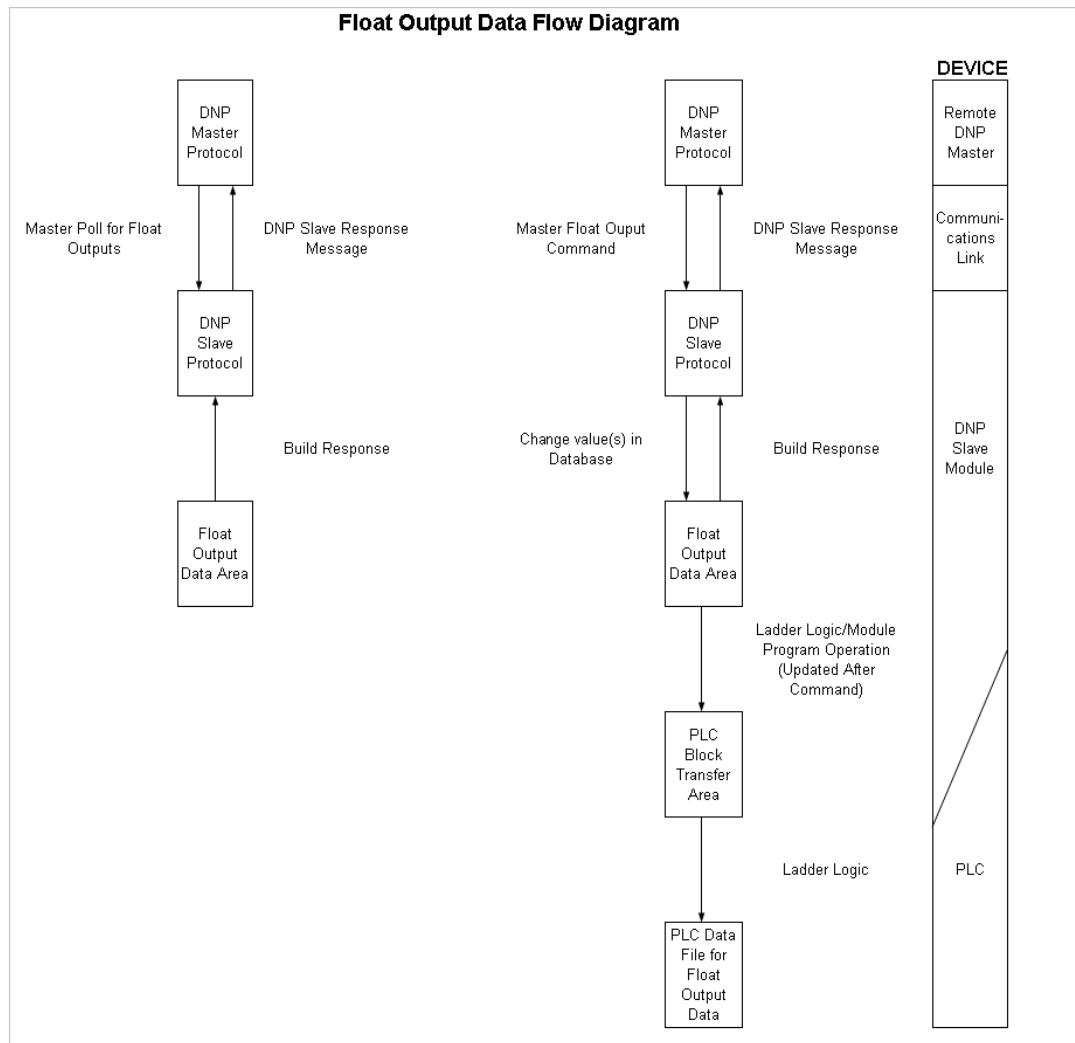
The DNP master unit can read the current value data and the event data from the module. Event messages generated by the module can be retrieved using a poll for Class 3 data, as all float input events are considered a Class 3 data type. If unsolicited message generation is enabled in the application, 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 float input data is shown in the following figure.





**DNP Float Output Data**

This data type stores Float values sent from the DNP master unit to the module and PLC. The size of this data area is determined from the configuration parameter Float 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 3405.000 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 float output data is shown in the following figure.



### **5.3.2 Communication Parameters**

This phase of design defines the communication parameters required for successful communications between the module and DNP master and slave units over the Ethernet network. Determine the IP address for the module and the list of IP addresses that can connect to the unit if this feature is enabled. Consult with the MIS person in charge of assigning these addresses and setting up the network configuration. The Reference chapter contains a form to aid in setting these parameters. Fill out this form before attempting to configure the module. You must also determine if the UDP or the TCP protocol or both will be used in your application. The module supports a single connection for the TCP protocol. The UDP server supports receipt of messages from multiple clients. Access to both servers can be limited by using the IP address list filtering.

### **5.3.3 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 the **Module Setup** section.

### **5.3.4 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.

If this feature is implemented, the module will request the data from the processor. Ladder logic must handle the blocks requested by the module (1000 to 1044) based on the modules configuration values for the write block data. When the block is requested, the module must place the correct data in the block and return the block to the module. The module will receive the data and initialize the output values. Each block required by the module for initialization will be requested.

### **5.3.5 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 (DNPSNET.CFG on the module's Compact Flash Disk). 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 this option is enabled, the module will request the binary and analog output from the PLC. This is done using blocks 1000 to 1044. Ladder logic must transfer the data for this feature to operate.

After the successful initialization of the module, the program will start the normal data transfer between the module and the ControlLogix processor. The program will send a read block first and then wait for a write block to receive data from the PLC. This alternating sequence of read and write will continue as long as the program is running. The program will update the DNP memory areas in the module with the new data and generate events for digital and analog input status changes.

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 for a complete discussion on the use of this important feature.

## 5.4 Cable Connections

The MVI56-DNPSNET module has the following functional communication connections installed:

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

### 5.4.1 Ethernet Connection

The MVI56-DNPSNET 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.

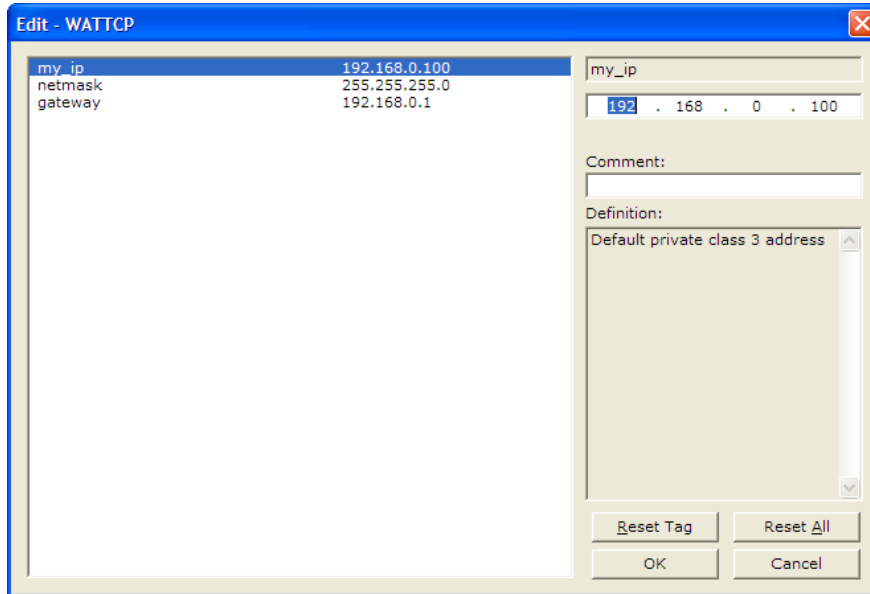
**Note:** Depending on hardware configuration, you may see more than one RJ45 port on the module. The Ethernet port is labeled *Ethernet*.

**Warning:** The MVI56-DNPSNET 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

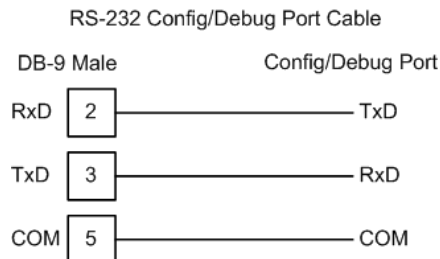
The wattcp.cfg file must be set up properly in order to use a TCP/IP network connection. You can view the current network configuration in *ProSoft Configuration Builder (PCB)*, as shown:



You may also view the network configuration using a PC serial port connection and an ASCII terminal program (like Windows HyperTerminal) by selecting **[@]** (Network Menu) and **[V]** (View) options when connected to the Debug port. For more information on serial port access, see the chapter on Diagnostics and Troubleshooting (page 51).

### **5.4.2 RS-232 Configuration/Debug Port**

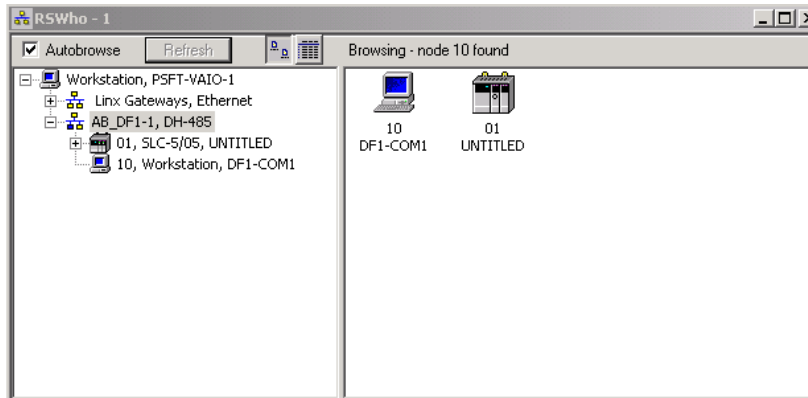
This port is physically an RJ45 connection. An RJ45 to DB-9 adapter cable is included with the module. This port permits a PC-based terminal emulation program to view configuration and status data in the module and to control the module. The cable pinout for communications on this port is shown in the following diagram.



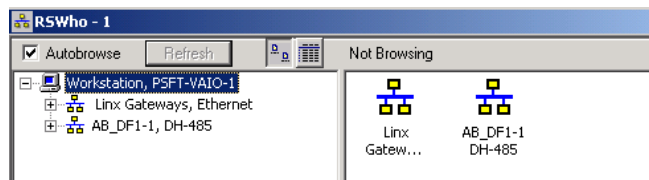
Disabling the RSLinx Driver for the Com Port on the PC



The communication port driver in *RSLinx* can occasionally prevent other applications from using the PC's COM port. If you are not able to connect to the module's configuration/debug port using *ProSoft Configuration Builder (PCB)*, *HyperTerminal* or another terminal emulator, follow these steps to disable the *RSLinx* driver.

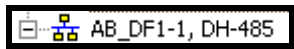
- 1 Open *RSLinx* and go to **COMMUNICATIONS > RSWHO**.
- 2 Make sure that you are not actively browsing using the driver that you wish to stop. The following shows an actively browsed network.



- 3 Notice how the DF1 driver is opened, and the driver is looking for a processor on node 1. If the network is being browsed, then you will not be able to stop this driver. To stop the driver your *RSWho* screen should look like this:

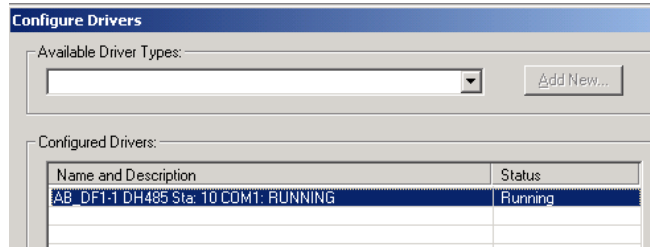


Branches are displayed or hidden by clicking on the  or the  icons.

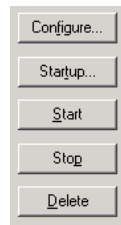


- 4 When you have verified that the driver is not being browsed, go to **COMMUNICATIONS > CONFIGURE DRIVERS**.

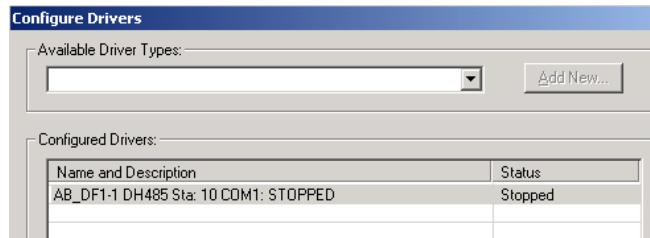
You may see something like this:



If you see the status as running, you will not be able to use this com port for anything other than communication to the processor. To stop the driver press the **STOP** button on the side of the window:



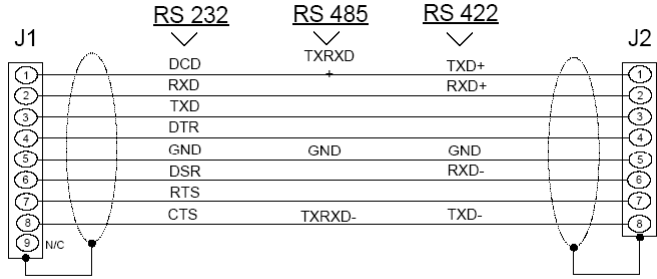
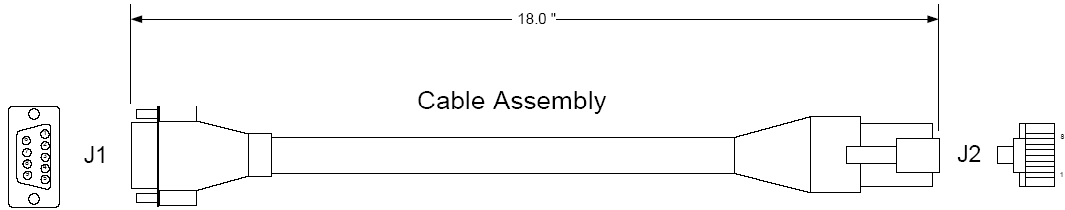
5 After you have stopped the driver you will see the following.



6 You may now use the com port to connect to the debug port of the module.

**Note:** You may need to shut down and restart your PC before it will allow you to stop the driver (usually only on *Windows NT* machines). If you have followed all of the above steps, and it will not stop the driver, then make sure you do not have *RSLogix* open. If *RSLogix* is not open, and you still cannot stop the driver, then reboot your PC.

### 5.4.3 DB9 to RJ45 Adaptor (Cable 14)



Wiring Diagram



## 5.5 Configuration Data

This section contains listings of the MVI56-DNPSNET module's configuration contained in the DNPSNET.CFG file.

[Section]/Item	Value	Range	Description
[Backplane Configuration]			Backplane transfer parameters
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.
Read Register Start:		0 to 8999	This parameter specifies the starting register in the module where data will be transferred from the module to the processor. Valid range for this parameter is 0 to 8999.
Read Register Count:		0 to 9000	This parameter specifies the number of registers to be transferred from the module to the processor. Valid entry for this parameter is 0 to 9000.
Write Register Start:		0 to 8999	This parameter specifies the starting register in the module where the data transferred from the processor will be placed. Valid range for this parameter is 0 to 8999.
Write Register Count:		0 to 9000	This parameter specifies the number of registers to transfer from the processor to the module. Valid entry for this parameter is 0 to 9000.
Failure Flag Count:		0 to 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, the module will stop communications when the preset value is reached based on successive failures.
Error Offset:		0 to 8964	This 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 8964, the data will be placed in the modules database.
Initialize Output Data:		Y or N	This parameter determines if the output data for the module should be initialized with values from the processor. If the value is set to N, the output data will be initialized to 0. If the value is set to Y, the data will be initialized with data from the processor.
[Section]/Item			
[DNP ENET Slave]			Server and protocol parameters
Internal Slave ID:		0 to 65534	This is the DNP address for the module. All messages with this address from the master will be processed by the module.

<b>[Section]/Item</b>	<b>Value</b>	<b>Range</b>	<b>Description</b>
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.
Binary Inputs:		0 to 500 words	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
Analog Inputs:		0 to 500 points	Number of analog input points to configure in the DNP slave device. Each point will occupy a one word area in the module memory.
Float Inputs		0 to 150	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.
Counters:		0 to 250 points	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.
Binary Outputs:		0 to 500 words	Number of digital output 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 outputs will be defined for the application.
Analog Outputs:		0 to 500 points	Number of analog output points to configure in the DNP slave device. Each point will occupy a one word area in the module memory.
Float Outputs		0 to 150	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.
BI Class		0=disable, else 1 to 3	This parameter specifies the default class to be utilized for all the binary input points in the DNP database that are not defined in the override list section.
AI Class		0=disable, else 1 to 3	This parameter specifies the default class to be utilized for all the analog input points in the DNP database that are not defined in the override list section.
Float Class		0=disable, else 1 to 3	This parameter specifies the default class to be utilized for all the floating-point input points in the DNP database that are not defined in the override list section.
AI Deadband:		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.
Float Deadband		0 to 32767 data units	This value sets the global deadband for all float input points. When the current value for a float input point is not within the deadband limit set based on the last event for the point, an event will be generated.

<b>[Section]/Item</b>	<b>Value</b>	<b>Range</b>	<b>Description</b>
Select/Operate Arm Time:		1 to 65535 milliseconds	Time period after select command received in which operate command will be performed. After the select command is received, the operate command will only be honored if it arrives within this period of time.
Write Time Interval:		0 to 1440 minutes	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.
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.
Unsolicited Response:		Y or N	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.
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:		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.
Uresp Master Address:		0 to 65534	DNP destination address where unsolicited response messages are sent.
BI With Flag		Y or N	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.
BI Events without time		Y or N	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.
AI With Flag		Y or N	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.
AI Events with time:		Y or N	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.
BO Without Flag		Y or N	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.

<b>[Section]/Item</b>	<b>Value</b>	<b>Range</b>	<b>Description</b>
Counter with Flag		Y or N	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.
Frozen Counter with Flag		Y or N	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.
Time Sync Before Events:		Y or N	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.
Use Trip/Close Single Point		Y or N	This Parameter allows Trip/Close commands to be sent to the module, for Single point control. Each DNP Trip/Close command will occupy 1 bit within the module memory.

<b>[Section]/Item</b>	<b>Value</b>	<b>Range</b>	<b>Description</b>
[DNP ENET IP Addresses]			List of valid IP addresses accepted by module
START			
# Insert the list of IP addresses for the host(s) to connect to this unit. Only used if Use IP List set to 1.			
END			

## 5.6 MVI56-DNPSNET Status Data

This section contains a listing of the MVI56-DNPSNET module's status data area.

Word	Variable Name	Description
0	Scan Counter	Program scan counter incremented each time the program loop is executed.
1-2	Product Name (ASCII)	These two words contain the product name of the module in ASCII format.
3-4	Revision (ASCII)	These two words contain the product revision level of the firmware in ASCII format.
5-6	Operating System Revision (ASCII)	These two words contain the module's internal operating system revision level in ASCII format.
7-8	Production Run Number (ASCII)	These two words contain the production "batch" number for the particular chip in the module in ASCII format.
9	Read Block Count	Total number of blocks transferred from the module to the processor.
10	Write Block Count	Total number of blocks transferred from the processor to the module.
11	Parse Block Count	Total number of blocks parsed by the module that were received from the processor.
12	Block number error	Number of BTW requests that resulted in an incorrect BTW identification code.
13	DNP Slave Port total number of message frames received by slave	This value represents the total number of message frames that have matched this slaves address on this port. This count includes message frames which the slave may or may not be able to parse and respond.
14	DNP Slave Port total number of response message frames sent from slave	This value represents the number of good (non-error) responses that the slave has sent to the master on this port. The presumption is that if the slave is responding, the message was good. Note: This is a frame count.
15	DNP Slave Port total number of message frames seen by slave	This value represents the total number of message frames received by the slave, regardless of the slave address.
16	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.
17	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.

Word	Variable Name	Description
18	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.
19	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.
20	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.
21	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.
22	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.
23	DNP Slave Binary Input Event count	This value contains the total number of binary input events which have occurred.
24	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.
25	DNP Slave Analog Input Event count	This value contains the total number of analog input events which have occurred.
26	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.
27	DNP Slave Float Input Event count in buffer	This value represents the number of float input events which are waiting to be sent to the master.
28	Reserved	Future Use
29	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.
30	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.
31	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.
32	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.
33	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.
34	UDP Receive Count	Number of UDP messages received
35	UDP Transmit Count	Number of UDP messages transmitted

<b>Word</b>	<b>Variable Name</b>	<b>Description</b>
36	Unsolicited Error Count	Number of failures when trying to send unsolicited event data
37	State Value	This variable has a value of 0 if there is not a message being sent and 1 if a message is being sent.
38	TCP Socket State Value	State machine value for the TCP socket
39	UDP Socket State Value	State machine value for the UDP socket
40	DNP Busy With Message State	Socket busy state -1=TCP socket not connected, 0=TCP or UDP not processing message, 1 or 3 = TCP processing message, and 2=UDP socket processing message.
41	Application Fragment	Application fragmentation flag/counter
42	Transmit Frame State	Transmit Frame State
43	TCP Message Length	Bytes Received on the TCP port for the current message.
44	UDP Message Length	Bytes received on the UDP port for the current message.
45	Port TX State	This variable has a value of 0 if there is not a message being sent and 1 if a message is being sent.
46	Free Memory LSB	Free memory in module
47	Free Memory MSB	

## 5.7 Internal Indication Bits (IIN Bits) for DNP Server

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

### 5.7.1 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 the master station know that the broadcast was 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 the master. The master should write the date and time when the 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.

### 5.7.2 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 the requested operation is already executing. The slave will never set this bit.
5	Not used.
6	Reserved. Always 0.
7	Reserved. Always 0.



## 5.8 DNP Subset Definition

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

OBJECT			REQUEST		RESPONSE			NOTES
Obj	Var	Description	Func Codes	Qual Codes (hex)	Func Codes	Qual Codes (hex)	Data Size (bits)	
1	0	Binary Input: All Variations	1	06			1	Slave will return variation 1 data
	1	Binary Input	1	06	129, 130	00, 01	1	Slave will return this variation
	2	Binary Input with Status			129, 130	00, 01	8	Slave will return Unknown Object to this request
2	0	Binary Input Change: All Variations	1	06, 07, 08			56	Slave will return variation 2 data
	1	Binary Input Change Without Time	1	06, 07, 08	129, 130	17, 28	8	Slave will return this variation
	2	Binary Input Change With Time	1	06, 07, 08	129, 130	17, 28	56	Slave will return this variation
	3	Binary Input Change With Relative Time	1	06, 07, 08	129, 130	17, 28	24	Slave will parse this message and return no data
10	0	Binary Output: All Variations	1	06			8	Slave will return variation 2 data
	1	Binary Output					1	Slave will return Unknown Object to this request
	2	Binary Output Status	1	06	129, 130	00, 01	8	Slave will return this variation
12	0	Control Block: All Variations					88	Slave will use variation 1 control
	1	Control Relay Output Block	3, 4, 5, 6	17, 28	129	Echo of request	88	Slave will respond correctly to this variation
	2	Pattern Control Block					88	Slave will return Unknown Object to this request
	3	Pattern Mask					16	Slave will return Unknown Object to this request
20	0	Binary Counter: All Variations	1, 7, 8, 9, 10	06			32	Slave will return variation 5 data
	1	32-Bit Binary Counter			129, 130	00, 01	40	Slave will return Unknown Object to this request
	2	16-Bit Binary Counter			129, 130	00, 01	24	Slave will return Unknown Object to this request
	3	32-Bit Delta Counter			129, 130	00, 01	40	Slave will return Unknown Object to this request
	4	16-Bit Delta Counter			129, 130	00, 01	24	Slave will return Unknown Object to this request
	5	32-Bit Binary Counter Without Flag	1, 7, 8, 9, 10	06		129, 130	00, 01	32

	6	16-Bit Binary Counter Without Flag	1, 7, 8, 9, 10	06	129, 130	00, 01	16	Slave will return this variation (counter upper 16-bits removed)
	7	32-Bit Delta Counter Without Flag			129, 130	00, 01	32	Slave will return Unknown Object to this request
	8	16-Bit Delta Counter Without Flag			129, 130	00, 01	16	Slave will return Unknown Object to this request
21	0	Frozen Counter: All Variations	1	06			32	Slave will return variation 9 data
	1	32-Bit Frozen Counter			129, 130	00, 01	40	Slave will return Unknown Object to this request
	2	16-Bit Frozen Counter			129, 130	00, 01	24	Slave will return Unknown Object to this request
	3	32-Bit Frozen Delta Counter					40	Slave will return Unknown Object to this request
	4	16-Bit Frozen Delta Counter					24	Slave will return Unknown Object to this request
	5	32-Bit Frozen Counter With Time Of Freeze					88	Slave will return Unknown Object to this request
	6	16-Bit Frozen Counter With Time Of Freeze					72	Slave will return Unknown Object to this request
	7	32-Bit Frozen Delta Counter With Time Of Freeze					88	Slave will return Unknown Object to this request
	8	16-Bit Frozen Delta Counter With Time Of Freeze					72	Slave will return Unknown Object to this request
	9	32-Bit Frozen Counter Without Flag	1	06	129, 130	00, 01	32	Slave will return this variation
	10	16-Bit Frozen Counter Without Flag	1	06	129, 130	00, 01	16	Slave will return this variation (counter upper 16-bits removed)
	11	32-Bit Frozen Delta Counter Without Flag					32	Slave will return Unknown Object to this request
12	16-Bit Frozen Delta Counter Without Flag					16	Slave will return Unknown Object to this request	
22	0	Counter Change Event: All Variations	1	06, 07, 08				Slave will parse this request and return no data
	1	32-Bit Counter Change Event Without Time			129, 130	17, 28	40	Slave will return Unknown Object to this request
	2	16-Bit Counter Change Event Without Time			129, 130	17, 28	24	Slave will return Unknown Object to this request
	3	32-Bit Delta Counter Change Event Without Time					40	Slave will return Unknown Object to this request
	4	16-Bit Delta Counter Change Event Without Time					24	Slave will return Unknown Object to this request
	5	32-Bit Counter Change Event With Time					88	Slave will return Unknown Object to this request

	6	16-Bit Counter Change Event With Time			72	Slave will return Unknown Object to this request
	7	32-Bit Delta Counter Change Event With Time			88	Slave will return Unknown Object to this request
	8	16-Bit Delta Counter Change Event With Time			72	Slave will return Unknown Object to this request
23	0	Frozen Counter Event: All Variations				Slave will return Unknown Object to this request
	1	32-Bit Frozen Counter Event Without Time			40	Slave will return Unknown Object to this request
	2	16-Bit Frozen Counter Event Without Time			24	Slave will return Unknown Object to this request
	3	32-Bit Frozen Delta Counter Event Without Time			40	Slave will return Unknown Object to this request
	4	16-Bit Frozen Delta Counter Event Without Time			24	Slave will return Unknown Object to this request
	5	32-Bit Frozen Counter Event With Time			88	Slave will return Unknown Object to this request
	6	16-Bit Frozen Counter Event With Time			72	Slave will return Unknown Object to this request
	7	32-Bit Frozen Delta Counter Event With Time			88	Slave will return Unknown Object to this request
	8	16-Bit Frozen Delta Counter Event With Time			72	Slave will return Unknown Object to this request
30	0	Analog Input: All Variations	1	06		16 Slave will respond with variation 4 data
	1	32-Bit Analog Input	1	06	129, 130 00, 01	40 Slave will return this variation (Note: Data will only be 16-bit)
	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
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
32	0	Analog Change Event: All Variations	1	06, 07, 08			24	Slave will return variation 2 data
	1	32-Bit Analog Change Event Without Time	1	06, 07, 08	129, 130	17, 28	40	Slave will return this variation (Note: Data only 16-bit)
	2	16-Bit Analog Change Event Without Time	1	06, 07, 08	129, 130	17, 28	24	Slave will return this variation
	3	32-Bit Analog Change Event With Time	1	06, 07, 08	129, 130	17, 28	88	Slave will return this variation (Note: Data only 16-bit)
	4	16-Bit Analog Change Event With Time	1	06, 07, 08	129, 130	17, 28	72	Slave will return this variation
	5	Short Floating Point Analog Input	1	06	129, 130	00, 01	40	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
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
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 Status	1	06	129, 130	00, 01	40	Slave will return this variation
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
	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
	3	Date and Time at Last Recorded Time	2		48	Slave will process the data in this object for time synchronization.
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	
	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	
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
	24			Slave supports use of this new time synchronization function. Used with Obj 52, Var 3.

## 5.9 Device Profile

<b>DNP V3.00 DEVICE PROFILE DOCUMENT</b>	
Vendor Name: ProSoft Technology, Inc.	
Device Name: MVI56-DNPSNET (VERSION 1.00)	
Highest DNP Level Supported : For Request: L2 For Responses: L2	Device Function: Slave (TCP/IP Server (Data Provider))
<p>Notable objects, functions, and/or qualifiers supported in addition to the highest DNP level stated above (see attached table for complete list):</p> <p>Definition of selected IIN bits:</p> <p>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.</p> <p>The following features are configurable on the module: Time sync before events are generated and default analog input events, Obj32V4 or O32V2, select option.</p> <p>Counter Freeze with reset will not zero values in the processor. Therefore, this function should not be utilized.</p> <p>Module will not generate events until Restart IIN bit is cleared by DNP master.</p>	
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

<b>DNP V3.00</b>	
<b>DEVICE PROFILE DOCUMENT</b>	
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	



## 5.10 Event Size Computation

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

```
((number of static points)*(rate per second scan of change function)) / (rate per second of master event data poll)
```

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

```
(51 * 2) / .02 = 510 events
```

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

```
(number of points that change per change function * rate per second of scan of change function)*(number of seconds between master event data poll)
```

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

```
(5 * 2) * 5 = 50 events required
```

The number of events that can be defined in the system is limited to 400. The event buffer will overflow in systems which are very dynamic unless one of the following conditions exist:

- The master frequently polls the slave device for events to keep the buffer empty.

OR

- The slave is configured to send unsolicited messages to the master station. This method requires full-duplex operation of the network because the slave may be sending a message during a request from the master station.

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

## 6 Support, Service & Warranty

### 6.1 Contacting Technical Support

ProSoft Technology, Inc. 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 associated ladder files, if any
- 2 Module operation and any unusual behavior
- 3 Configuration/Debug status information
- 4 LED patterns
- 5 Details about the interfaced serial, Ethernet or Fieldbus devices

**Note:** For technical support calls within the United States, ProSoft's 24/7 after-hours phone support is available for urgent plant-down issues.

North America (Corporate Location)	Europe / Middle East / Africa Regional Office
Phone: +1.661.716.5100 info@prosoft-technology.com Languages spoken: English, Spanish REGIONAL TECH SUPPORT support@prosoft-technology.com	Phone: +33.(0)5.34.36.87.20 france@prosoft-technology.com Languages spoken: French, English REGIONAL TECH SUPPORT support.emea@prosoft-technology.com
Latin America Regional Office	Asia Pacific Regional Office
Phone: +52.222.264.1814 latinam@prosoft-technology.com Languages spoken: Spanish, English REGIONAL TECH SUPPORT support.la@prosoft-technology.com	Phone: +60.3.2247.1898 asiapc@prosoft-technology.com Languages spoken: Bahasa, Chinese, English, Japanese, Korean REGIONAL TECH SUPPORT support.ap@prosoft-technology.com

For additional ProSoft Technology contacts in your area, please visit:  
<https://www.prosoft-technology.com/About-Us/Contact-Us>.

### 6.2 Warranty Information

For complete details regarding ProSoft Technology's TERMS & CONDITIONS OF SALE, WARRANTY, SUPPORT, SERVICE AND RETURN MATERIAL AUTHORIZATION INSTRUCTIONS, please see the documents at:  
[www.prosoft-technology.com/legal](http://www.prosoft-technology.com/legal)

## Index

### I

[Backplane Configuration] • 26  
[DNP ENET IP ADDRESSES] • 39  
[DNP ENET Slave] • 28  
[DNP Slave Analog Inputs] • 37  
[DNP Slave Binary Inputs] • 36  
[DNP Slave Float Inputs] • 38

### A

Adding the Module to an Existing Project • 48  
AI Class • 31  
AI Deadband • 32  
AI Events with Time • 33  
AI with Flag • 33  
AI\_Events • 47  
Analog Inputs • 30  
Analog Outputs • 31  
App Layer Confirm Tout • 32

### B

Backplane Data Transfer • 72  
Battery Life Advisory • 3  
BI Class • 31  
BI Events Without Time • 34  
BI with Flag • 34  
BI\_Events • 46  
Binary Inputs • 30  
Binary Outputs • 31  
Block 9958  
    Binary Input Event • 76  
Block 9959  
    Analog Input Event • 78  
Block 9961  
    SOE input events • 79, 81  
Block 9970  
    Set Processor Time • 83  
Block 9971  
    Set Module Time • 84  
Block 9998  
    Warm Boot • 84  
Block 9999  
    Cold Boot • 84  
Block Format for Read • 81, 83  
Block Format for Write • 80, 82  
BO Without Flag • 34  
BP (Backplane) • 46

### C

Cable Connections • 100  
Class • 36, 37, 38  
Class 1 Unsol Resp Min • 33

Class 2 Unsol Resp Min • 33  
Class 3 Unsol Resp Min • 33  
Clearing a Fault Condition • 53  
Clock • 45  
CMDcontrolbits • 45  
Communication Parameters • 98  
Configuration Data • 105  
Configuring Module Parameters • 24  
Configuring the MVI56-DNPSNET Module • 21  
Configuring the RSLinx Driver for the PC COM Port • 17  
Connecting Your PC to the ControlLogix Processor • 13  
Connecting your PC to the Module • 20  
Contacting Technical Support • 122  
Counter with Flag • 34  
Counters • 31  
Creating Optional Comment Entries • 25

### D

Data • 45  
Data Flow Between MVI56-DNPSNET Module and the ControlLogix Processor • 85  
Data Requirements • 87  
Data Transfer at Startup • 98  
Data Transfer Interface • 89  
Database View Menu • 60, 63  
DB9 to RJ45 Adaptor (Cable 14) • 104  
Deadband • 37, 38  
Designing the system • 86  
Determining the Firmware Version of Your Processor • 14  
Device Profile • 119  
Diagnostics and Troubleshooting • 7, 51, 101  
Disabling the RSLinx Driver for the Com Port on the PC • 102  
Displaying the Current Page of Registers Again • 63  
DNP Analog Input Data • 94  
DNP Analog Output Data • 95  
DNP Counter Data • 93  
DNP Digital Input Data • 91  
DNP Digital Output Data • 92  
DNP Float Input Data • 96  
DNP Float Output Data • 97  
DNP Menu • 60, 61  
DNP Subset Definition • 113  
DNPMModuleDef Object • 43  
Downloading the Project to the Module Using a Serial COM Port • 41  
Downloading the Sample Program to the Processor • 19

### E

Error Offset • 27  
Error Status Table • 54  
Ethernet Configuration • 40  
Ethernet Connection • 100  
Ethernet LED Indicators • 53  
Ethernet Port Configuration - wattcp.cfg • 101

Event Size Computation • 121  
Exiting the Program • 61

## F

Failure Flag Count • 26  
First Byte • 112  
Float Class • 31  
Float Deadband • 32  
Float Inputs • 30  
Float Outputs • 31  
Frozen Counter with Flag • 35  
Functional Overview • 72  
Functional Specifications • 71  
Functionality • 98

## G

General Concepts • 72  
General Specifications • 69  
Guide to the MVI56-DNPSNET User Manual • 7

## H

Hardware Specifications • 70  
How to Contact Us • 2

## I

Important Installation Instructions • 3  
Initializing Output Data • 27  
Installing ProSoft Configuration Builder Software • 11  
Installing the Module in the Rack • 12  
Internal Indication Bits (IIN Bits) for DNP Server • 112  
Internal Slave ID • 28

## K

Keystrokes • 58

## L

Ladder Logic • 42  
LED Status Indicators • 52

## M

Main Menu • 59  
Module Data Objects • 43  
Module Name • 26  
Module Operation • 99  
Moving Back Through 5 Pages of Registers • 64  
Moving Forward (Skipping) Through 5 Pages of Registers • 64  
MVI (Multi Vendor Interface) Modules • 3  
MVI56-DNPSNET Application Design • 86  
MVI56-DNPSNET Status Data • 109

## N

Navigation • 58  
Network Menu • 60, 65  
Normal Data Transfer • 74

## O

Opening the Database View Menu • 59  
Opening the DNP Database View Menu • 61  
Opening the DNP Menu • 60  
Opening the Network Menu • 60  
Opening the Sample Ladder Logic • 14

## P

Package Contents • 10  
Pinouts • 3, 100, 104  
Point # • 36, 37, 38  
Printing a Configuration File • 25  
Product Specifications • 7, 69

## R

Read Block • 74  
Read Register Count • 26  
Read Register Start • 26  
Reading Status Data from the Module • 67  
Receiving the Configuration File • 60  
Reference • 7, 68  
Renaming PCB Objects • 24  
Returning to the Main Menu • 61, 64, 66  
RS-232 Configuration/Debug Port • 101

## S

Second Byte • 112  
Select/Operate Arm Time • 32  
Selecting the Slot Number for the Module • 16  
Sending the Configuration File • 60  
Setting Jumpers • 11  
Setting Up the Project • 22  
Special Function Blocks • 76  
Special Objects • 46  
Start Here • 7, 8  
Status • 44  
Support, Service & Warranty • 7, 122  
System Requirements • 9

## T

Time Sync Before Events • 35  
Transferring WATTCP.CFG to the Module • 65  
Transferring WATTCP.CFG to the PC • 65  
Trip/Close • 75  
Troubleshooting • 53

## U

Unsol Resp Delay • 33  
Unsolicted Response • 32  
Uresp Master Address • 33  
Use IP List • 28  
Use Trip/Close Single Point • 29  
Using ProSoft Configuration Builder • 22  
Using ProSoft Configuration Builder (PCB) for Diagnostics • 56  
Using the Diagnostic Window in ProSoft Configuration Builder • 56

## V

- Viewing a List of Valid Hosts • 61
- Viewing Block Transfer Statistics • 59
- Viewing Data in ASCII (Text) Format • 64
- Viewing Data in Decimal Format • 64
- Viewing Data in Floating-Point Format • 64
- Viewing Data in Hexadecimal Format • 64
- Viewing DNP Communication Status • 61
- Viewing DNP Configuration • 61
- Viewing DNP Set Up & Pointers • 61
- Viewing Module Configuration • 59
- Viewing Register Pages • 63
- Viewing TCP Socket Status • 62
- Viewing the Next Page of Registers • 64
- Viewing the Previous Page of Registers • 64
- Viewing the WATTCP.CFG File on the module • 66
- Viewing UDP Socket Status • 62
- Viewing Version Information • 60

## W

- Warm Booting the Module • 60
- Warranty Information • 122
- Write Block • 75
- Write Register Count • 26
- Write Register Start • 26
- Write Time Interval • 32

## Y

- Your Feedback Please • 2