ProSoft TECHNOLOGY

Where Automation Connects.

inRAx®

MVI56-MNETR

ControlLogix Platform

Modbus TCP/IP Interface Module with Reduced Data Block

September 19, 2008

USER MANUAL

Please Read This Notice

Successful application of this module requires a reasonable working knowledge of the Rockwell Automation ControlLogix hardware, the MVI56-MNETR Module and the application in which the combination is to be used. For this reason, it is important that those responsible for implementation satisfy themselves that the combination will meet the needs of the application without exposing personnel or equipment to unsafe or inappropriate working conditions.

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

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

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

Warnings

UL Warnings

A Warning - Explosion Hazard - Substitution of components may impair suitability for Class I, Division 2.
B Warning - Explosion Hazard - When in Hazardous Locations, turn off power before replacing or rewiring modules.
Warning - Explosion Hazard - Do not disconnect equipment unless power has been switched off or the area is known to be nonhazardous.
C Suitable for use in Class I, division 2 Groups A, B, C and D Hazardous Locations or Non-Hazardous Locations.

ATEX Warnings and Conditions of Safe Usage:

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

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

Electrical Ratings

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

Markings:

 II 3 G 0C <=Ta<= 60C EEx nA IIC T4 DEMKO 07ATEX0710717X

Battery Life Advisory

All modules in the MVI series use a rechargeable Lithium Vanadium Pentoxide battery to backup the 512K SRAM memory, 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 it becomes fully charged. After it is fully charged, the battery provides backup power for the CMOS setup and configuration data, the real-time clock, and the 512K SRAM memory for approximately 21 days.
Before you remove a module from its power source, ensure that the battery within the module is fully charged. A fully charged battery will hold the BIOS settings (after being removed from its power source) for a limited number of days. When the battery is fully discharged, the module will revert to the default BIOS settings.

Note: The battery is not user replaceable.

ProSoft® Product Documentation

In an effort to conserve paper, ProSoft Technology no longer includes printed manuals with our product shipments. User Manuals, Datasheets, Sample Ladder Files, and Configuration Files are provided on the enclosed CD and are available at no charge from our web site: http://www.prosoft-technology.com
Printed documentation is available for purchase. Contact ProSoft Technology for pricing and availability.
Asia Pacific: +603.7724.2080
Europe, Middle East, Africa: +33.5.34.36.87.20
Latin America: +1.281.298.9109
North America: +1.661.716.5100

Your Feedback Please

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

ProSoft Technology
1675 Chester Avenue, Fourth Floor
Bakersfield, CA 93301
+1 (661) 716-5100
+1 (661) 716-5101 (Fax)
http://www.prosoft-technology.com

MVI56-MNETR User Manual
September 19, 2008

Contents

Guide to the MVI56-MNETR User Manual

Function		Section to Read	Details
Introduction (Must Do)	→	Start Here (page 9)	This Section introduces the customer to the module. Included are: package contents, system requirements, hardware installation, and basic configuration.
Verify Communication, Diagnostic and Troubleshooting	→	Verifying Communication (page 66) Diagnostics and Troubleshooting (page 55)	This section describes how to verify communications with the network. Diagnostic and Troubleshooting procedures.
Reference Product Specifications Functional Overview Glossary	→	Reference (page 69) Functional Overview (page 71) Product Specifications (page 69)	These sections contain general references associated with this product, Specifications, and the Functional Overview.
Support, Service, and Warranty Index	→	Support, Service and Warranty (page 97)	This section contains Support, Service and Warranty information. Index of chapters.

1 Start Here

In This Chapter

Installing the MVI56-MNETR module requires a reasonable working knowledge of the Rockwell Automation hardware, the MVI56-MNETR Module and the application in which they will be used.

Caution: It is important that those responsible for implementation can complete the application without exposing personnel, or equipment, to unsafe or inappropriate working conditions. Safety, quality and experience are key factors in a successful installation.

1.1 System Requirements

The MVI56-MNETR 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-MNETR module. The module requires 800mA 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 XP Professional with Service Pack 1 or 2
  - 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)
- CD-ROM drive
- 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-MNETR 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-MNETR Module	MVI56-MNETR	Modbus TCP/IP Interface Module with Reduced Data Block
1	Cable	Cable #15, RS232 Null Modem	For RS232 Connection to the CFG Port
1	Cable	RJ45 to DB9 Male Adapter	For DB9 Connection to Module's Port
1	inRAx Solutions CD		Contains sample programs, utilities and documentation for the MVI56-MNETR module.

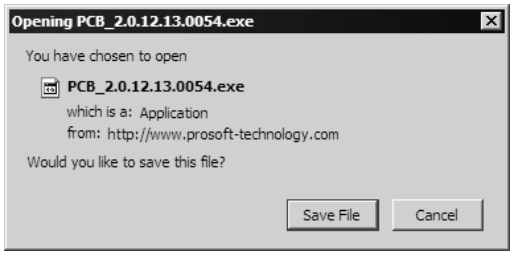
If any of these components are missing, please contact ProSoft Technology Support for replacement parts.

1.3 Install ProSoft Configuration Builder Software

You must install the ProSoft Configuration Builder (PCB) software in order to configure the MVI56-MNETR module. You can always get the newest version of ProSoft Configuration Builder from the ProSoft Technology web site.

To install ProSoft Configuration Builder from the ProSoft Web Site

1. Open your web browser and navigate to http://www.prosoft-technology.com/pcb
2. Click the **Download Here** link to download the latest version of ProSoft Configuration Builder.
3. Choose "Save" or "Save File" when prompted. The following illustrations show the file download prompt for two of the most common web browsers.



4. Make a note of the location where you saved the file, for example "Desktop", or "My Documents", so you can start the installation program.
5. When the download is complete, locate and open the file, and then follow the instructions on your screen to install the program.

If you do not have access to the Internet, you can install ProSoft Configuration Builder from the ProSoft Solutions CD-ROM, included in the package with your MVI56-MNETR module.

To install ProSoft Configuration Builder from the CD-ROM

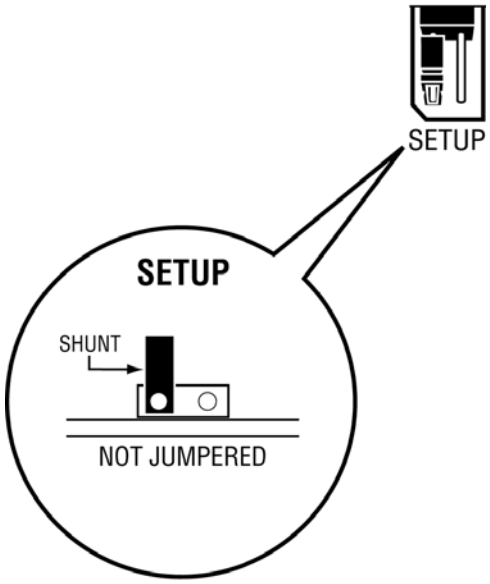
1. Insert the ProSoft Solutions CD-ROM into the CD drive of your PC. Wait for the startup screen to appear.
2. On the startup screen, click *Product Documentation*. This action opens an explorer window.

3 Click to open the *Utilities* folder. This folder contains all of the applications and files you will need to set up and configure your module.
4 Double-click the *ProSoft Configuration Builder Setup* program and follow the instructions on your screen to install the software on your PC.

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

1.4 Setting Jumpers

Note: 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.



1.5 Install the Module in the Rack

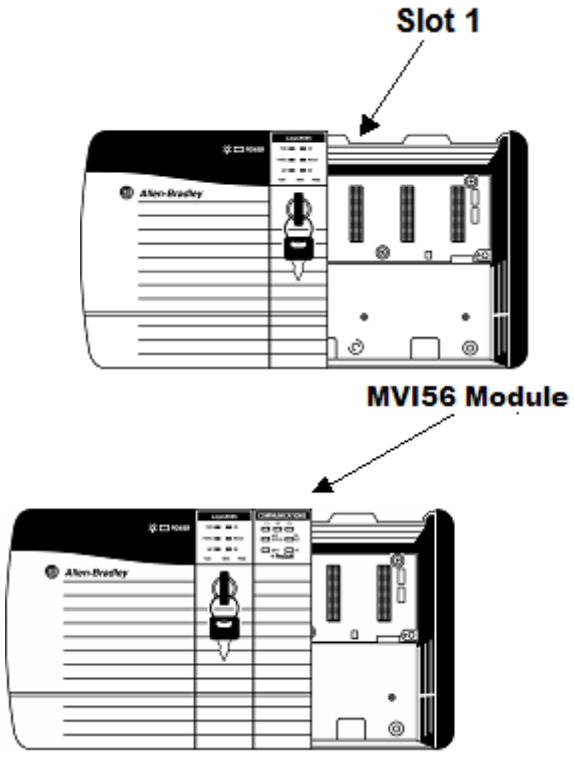
If you have not already installed and configured your ControlLogix processor and power supply, please do so before installing the MVI56-MNETR module. Refer to your Rockwell Automation product documentation for installation instructions.

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-MNETR 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.



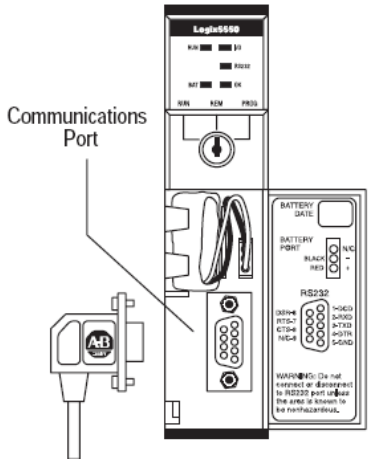
3. With a firm but steady push, snap the module into place.

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 will need to 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.

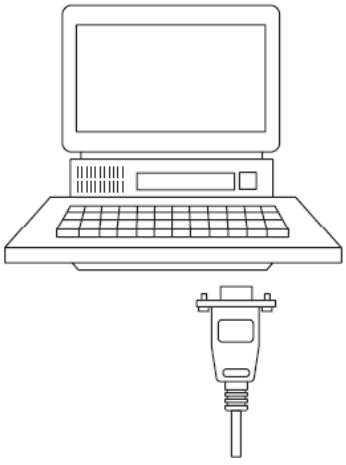
Note: If you insert the module improperly, the system may stop working, or may behave unpredictably.
Note: If you are installing MVI56-MNETR with other modules connected to the PCI bus, the peripheral modules will not have holding clips. Make sure all of the modules are aligned with their respective slots before you snap them into place.

1.6 Connect your PC to the Processor

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 Open the Sample Ladder Logic

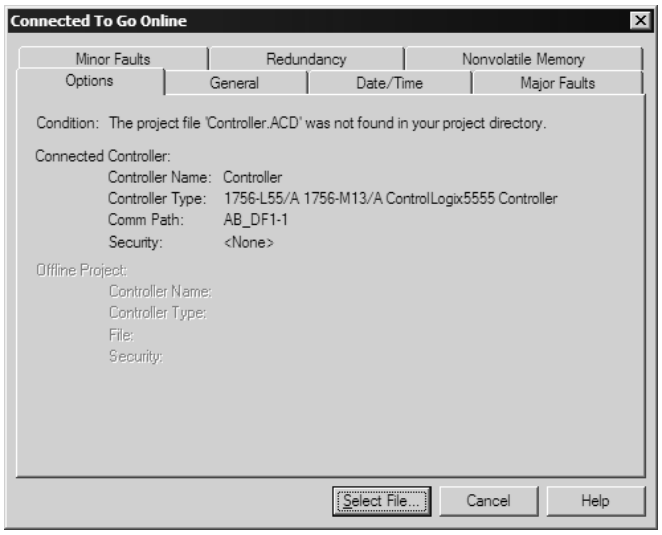
The sample program for your MVI56-MNETR module includes custom tags, data types and ladder logic for data I/O and status monitoring. For most applications, you can run the sample ladder program without modification, or, for advanced applications, you can incorporate the sample program into your existing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

The inRAx Solutions CD provides one or more versions of the sample ladder logic. 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.

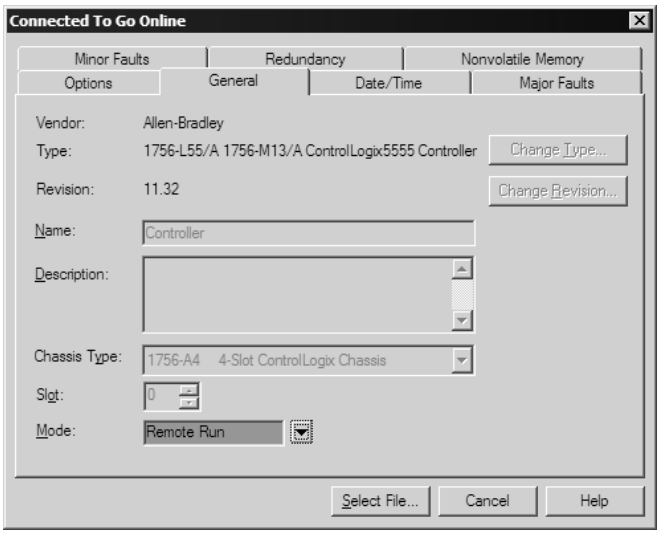
1.7.1 To Determine the Firmware Version of your Processor

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 On 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 11.32

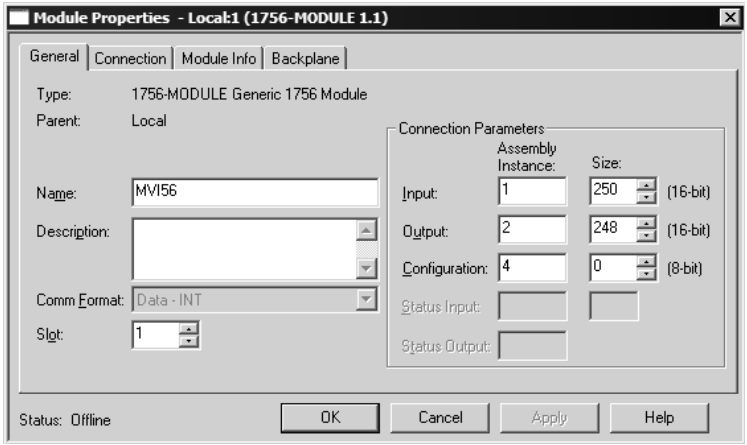


1.7.2 Select the Slot Number for the Module

The 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 [1] 1756-MODULE MVI56, 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.



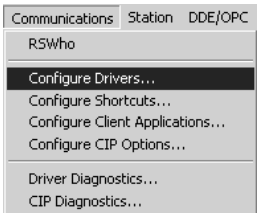
3 In the **Slot:** field, use the spinners on the right side of the field to select the slot number where the module will reside in the rack, and then click OK.

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

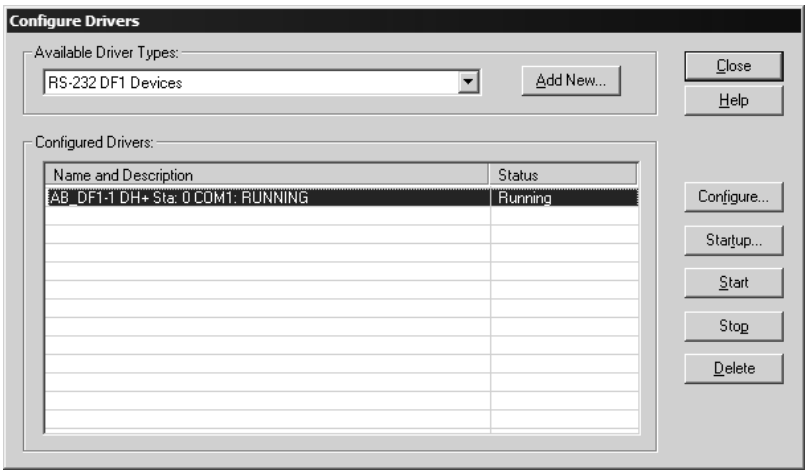
1.7.3 Configuring RSLinx

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

1 Open RSLinx.
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 Allen-Bradley DF1 Communications Device 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.

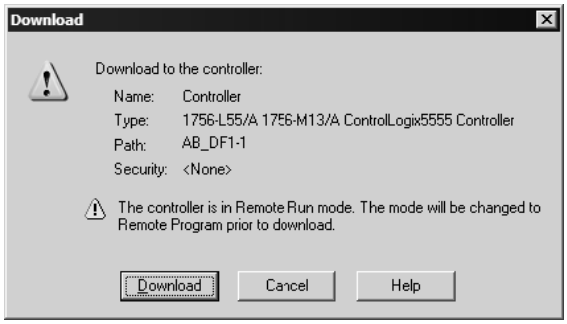
Note: If the auto-configuration procedure fails, verify that the cables are connected correctly between the processor and the serial port on your computer, and then try again. If you are still unable to auto-configure the port, refer to your RSLinx documentation for further troubleshooting steps.

1.8 Download the Sample Program to the Processor

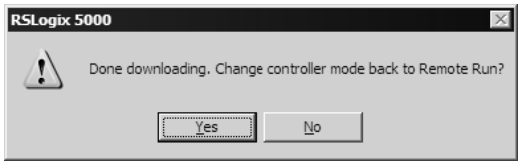
Note: The key switch on the front of the ControlLogix module must be in the REM position.

To download the sample program from RSLogix 5000 to the ControlLogix processor

1. If you are not already online to the processor, open the Communications menu, and then choose Download. RSLogix will establish communication with the processor.
2. When communication is established, RSLogix will open a confirmation dialog box. Click the Download button to transfer the sample program to the processor.



3. RSLogix will compile the program and transfer it to the processor. This process may take a few minutes.
4. When the download is complete, RSLogix will open another confirmation dialog box. Click OK to switch the processor from Program mode to Run mode.

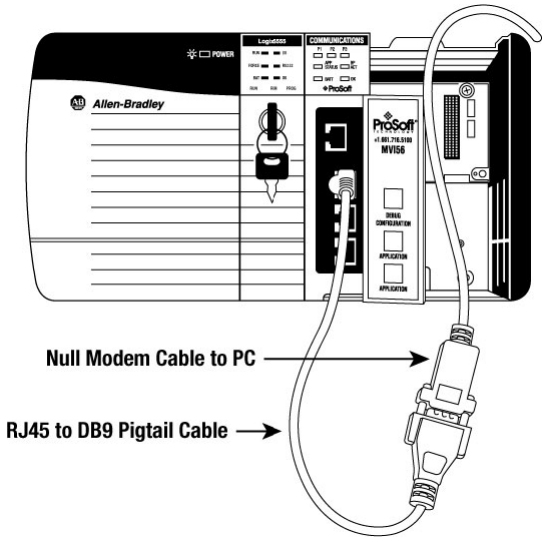


Note: If you receive an error message during these steps, refer to your RSLogix documentation to interpret and correct the error.

1.9 Connect 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 or laptop.



2 Module Configuration

In This Chapter

This section contains the setup procedure, data, and ladder logic requirements for successful application of the MVI56-MNETR module. Each step in the setup procedure is defined in order to simplify the use of the module.

2.1 MVI56-MNETR Sample Add-On Instruction Import Procedure

Note: this section only applies if you are using RSLogix5000 version 16 or higher. If you are configuring the MVI56-MNETR module with an earlier version of RSLogix 5000, please refer to Installing and Configuring the Module (page 92).

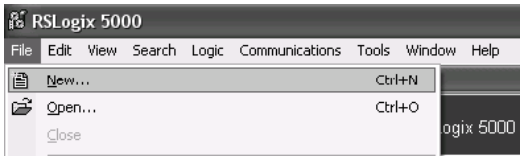
Before You Begin

The following file is required before you start this procedure. Copy the file from the ProSoft Solutions CD-ROM, or download it from http://www.prosoft-technology.com.

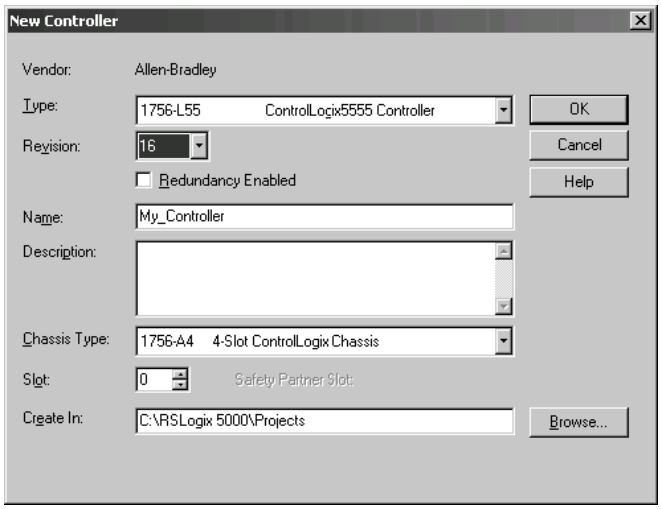
File Name	Description
MVI56MNETR.L5X	L5X file containing Add-On instruction, user defined data types, data objects and ladder logic required to setup the MVI56-MNETR module

2.1.1 Create a new RSLogix5000 project

1 Open the File menu, and then choose New…

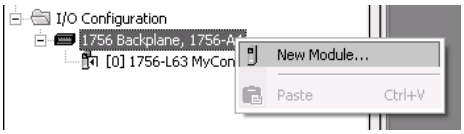


2 Select Revision 16

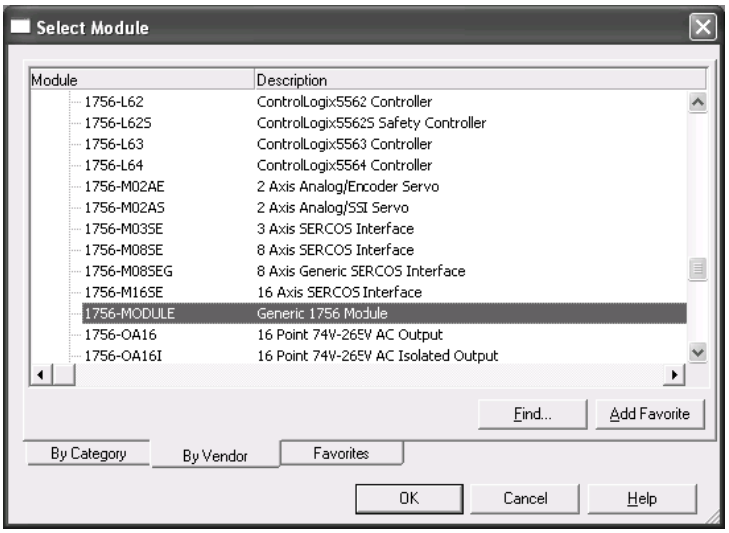


2.1.2 Create the Module

1 Right-click I/O Configuration and choose New Module…

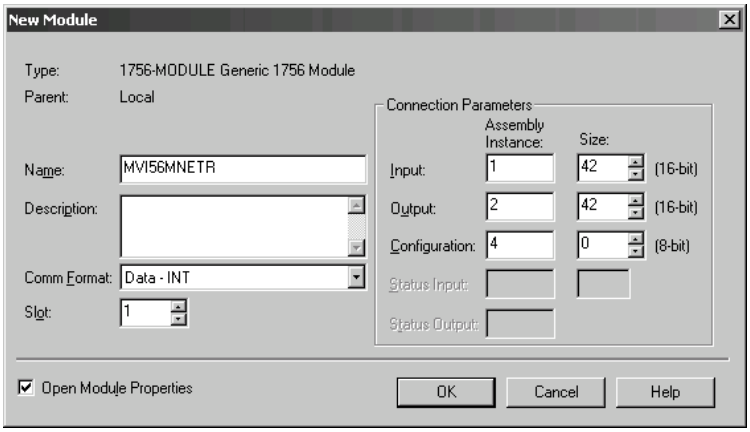


2 Select 1756-MODULE

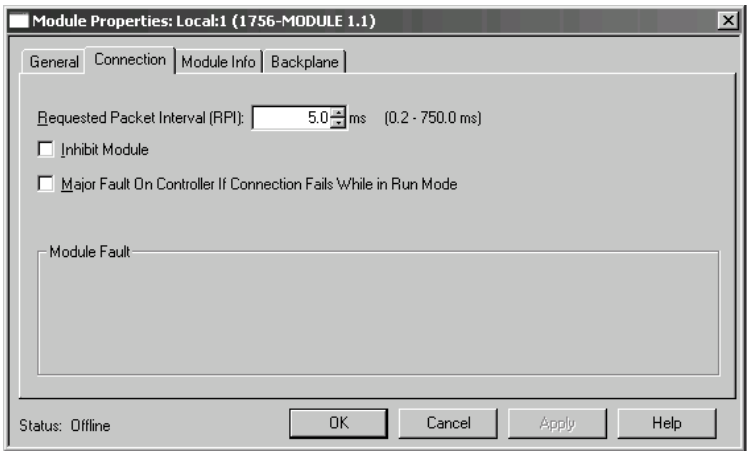


Set the Module Properties values as follows:

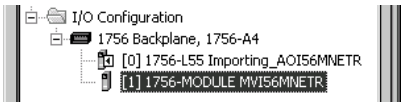
Parameter	Value
Name	Enter a module identification string. Example: MVI56MNETR
Description	Enter a description for the module. Example: ProSoft communication module for ModbusTCP/IP.
Comm. Format	Select DATA-INT
Slot	Enter the slot number in the rack where the MVI56-MNETR module is or will be located.
Input Assembly Instance	1
Input Size	42
Output Assembly Instance	2
Output Size	42
Configuration Assembly Instance	4
Configuration Size	0



3 On the Connection tab, set the RPI value for your project. Click OK to confirm.

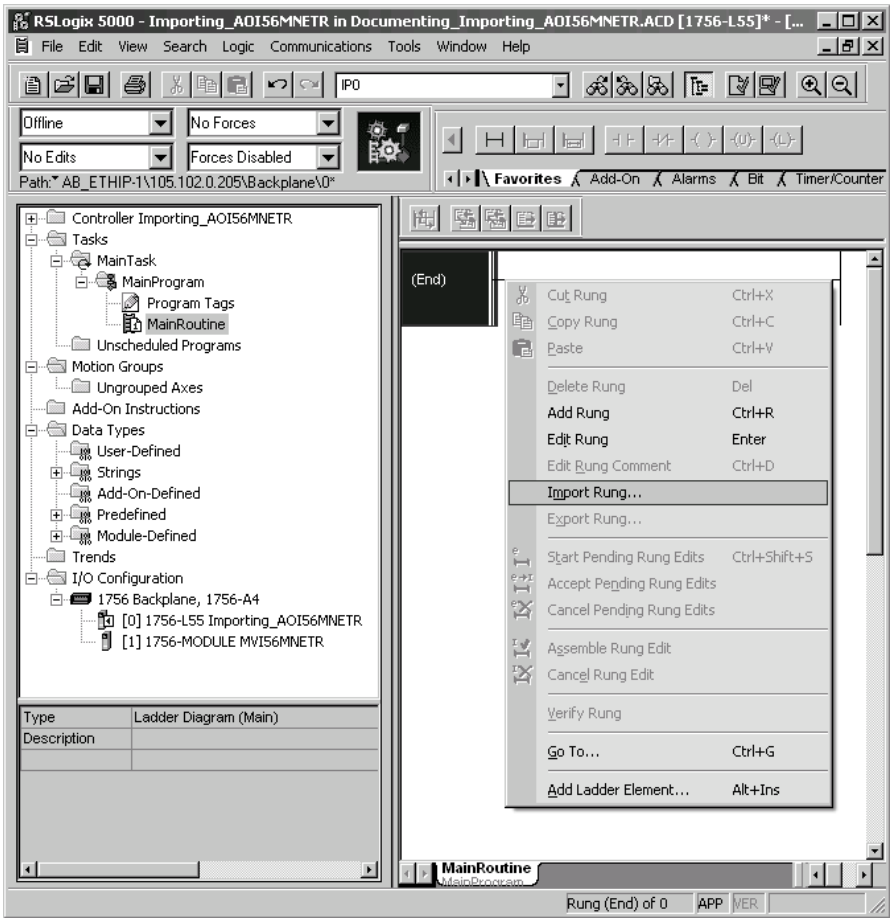


Now the MVI56-MNETR module will be visible at the I/O Configuration section

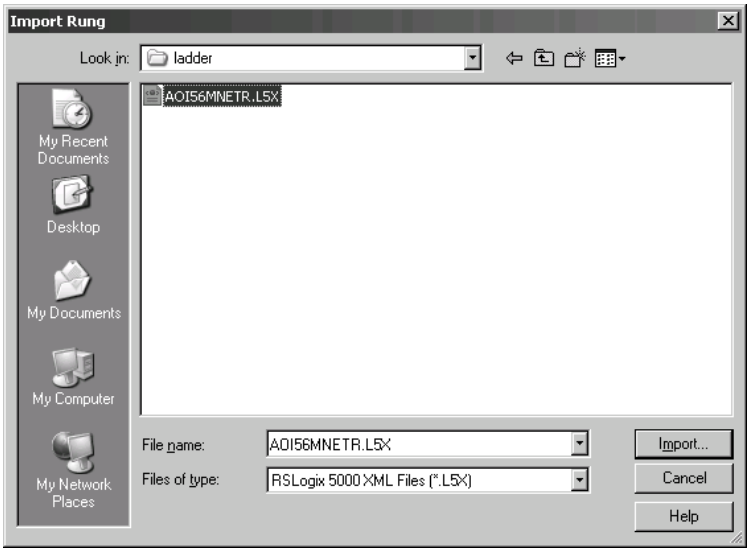


2.1.3 Import the Ladder Rung

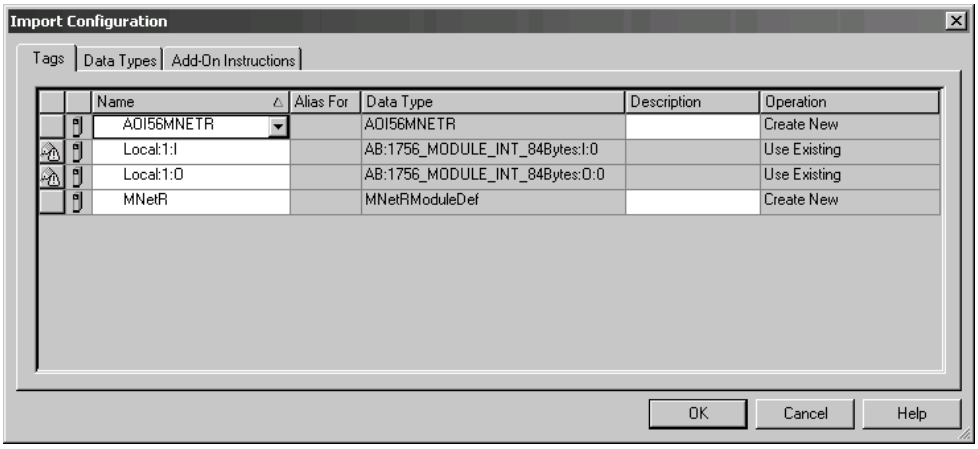
1 Open your application in RSLogix 5000.
2 Expand the Tasks folder, and then expand the Main Task folder.
3 On the Main Program folder, click the right mouse button to open a shortcut menu. On the shortcut menu, choose New Routine.
4 In the New Routine dialog box, enter the name and description of your routine, and then click OK.
5 Select an empty rung in the new routine, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose "Import Rung…".



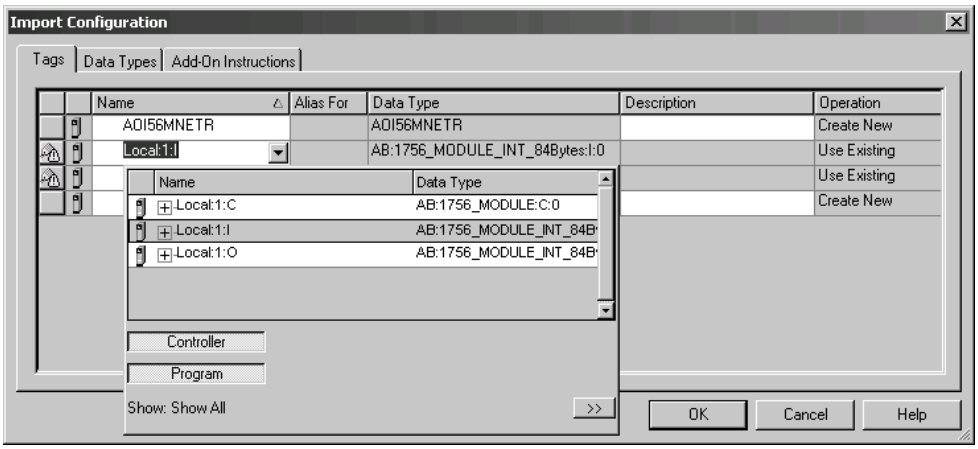
6 Select the *AOI56MNETR.L5X* file



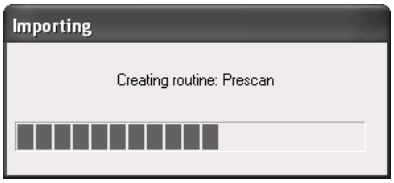
7 The following window will be displayed showing the controller tags to be created during the import procedure:



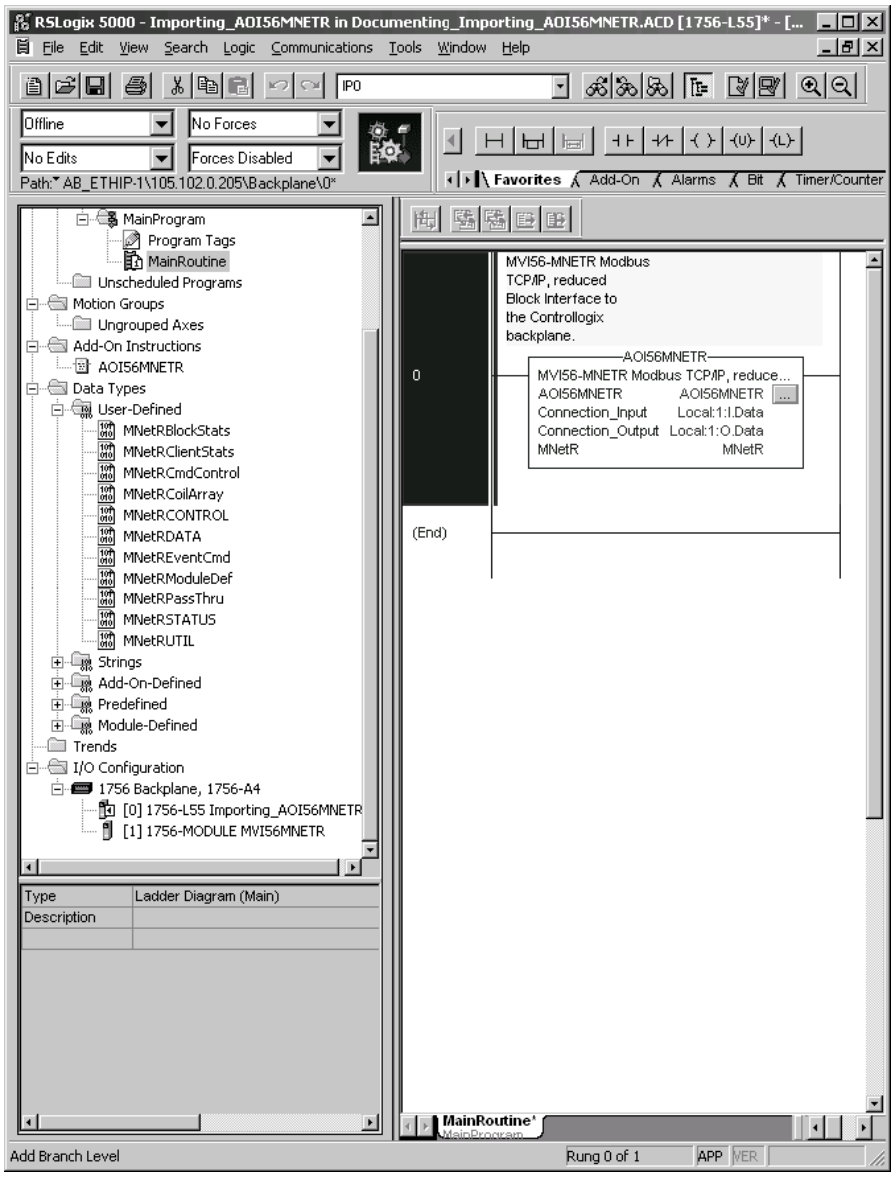
8 If you are using the module in a different slot (or remote rack) select the correct connection input and output variables associated to the module. If your module is located in slot 1 of the local rack this step is not required.



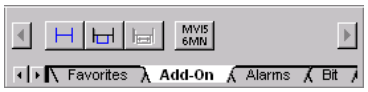
9 Click OK to confirm the import. RSLogix will indicate that the import is under progress:



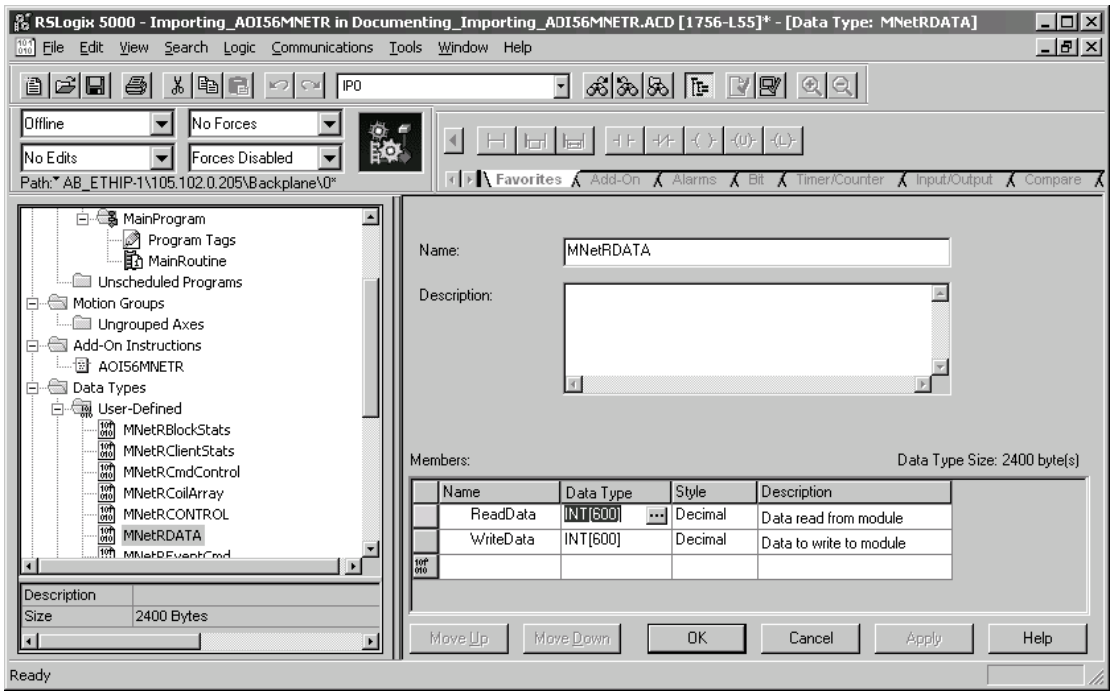
When the import is completed, the new rung with the Add-On instruction will be visible as shown in the following illustration.



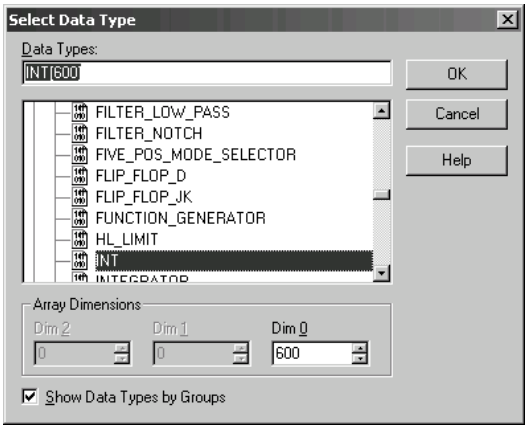
The procedure has also imported new user defined data types, data objects and the Add-On instruction to be used at your project.



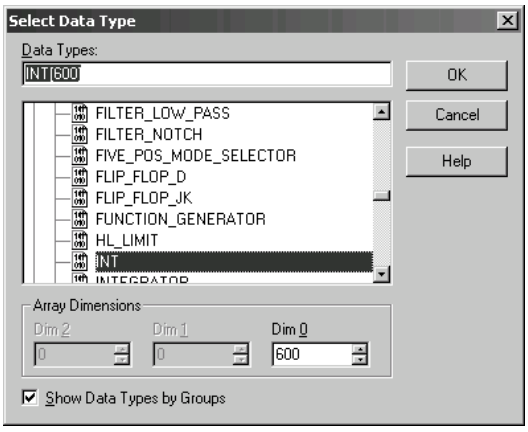
10 The Add-On instruction may come with User-Defined Tag arrays ReadData and WriteData set to the factory default values of 600. When your module's requirements are different, edit these array sizes to the values indicated in the Module's configuration file's section [MODULE] parameter "Read Register Count" and "Write Register Count" values.



- Example: if "Read Register Count : 1000" then set ReadData tag array size to INT[1000].



- Example: if "Write Register Count : 2000" then set WriteData tag array size to INT[2000].



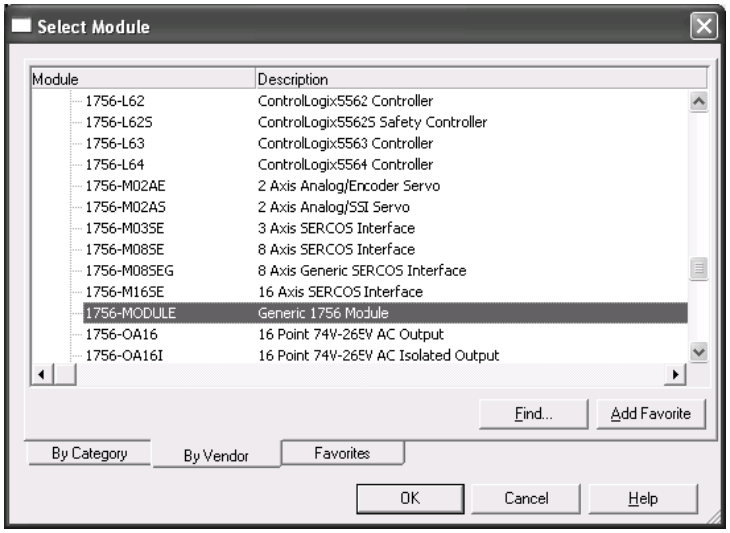
1 At the conclusion of the modifications of ReaData and WriteData array sizes, save the application and proceed to download the ladder logic into the processor.

2.1.4 Adding Multiple Modules (Optional)

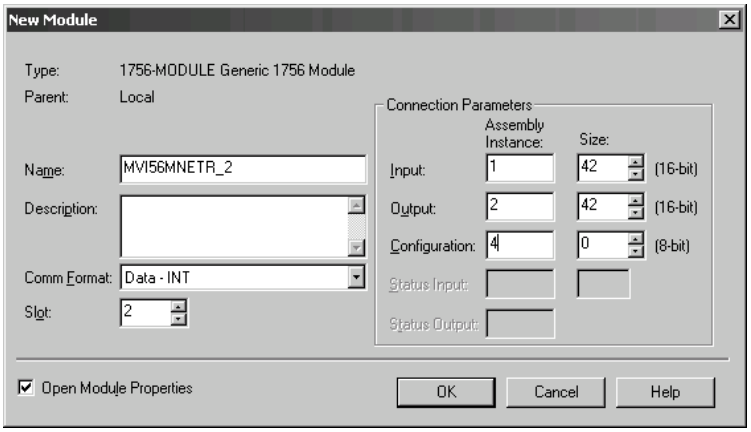
Important: If your application requires more than one MVI56-MNETR module into the same project, follow the steps below:

1 In the I/O Configuration folder, click the right mouse button to open a shortcut menu, and then choose New Module.

2 Select 1756-MODULE

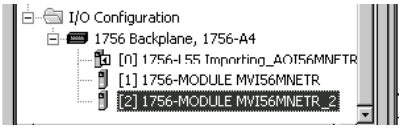


3 Fill in the module properties as follows:

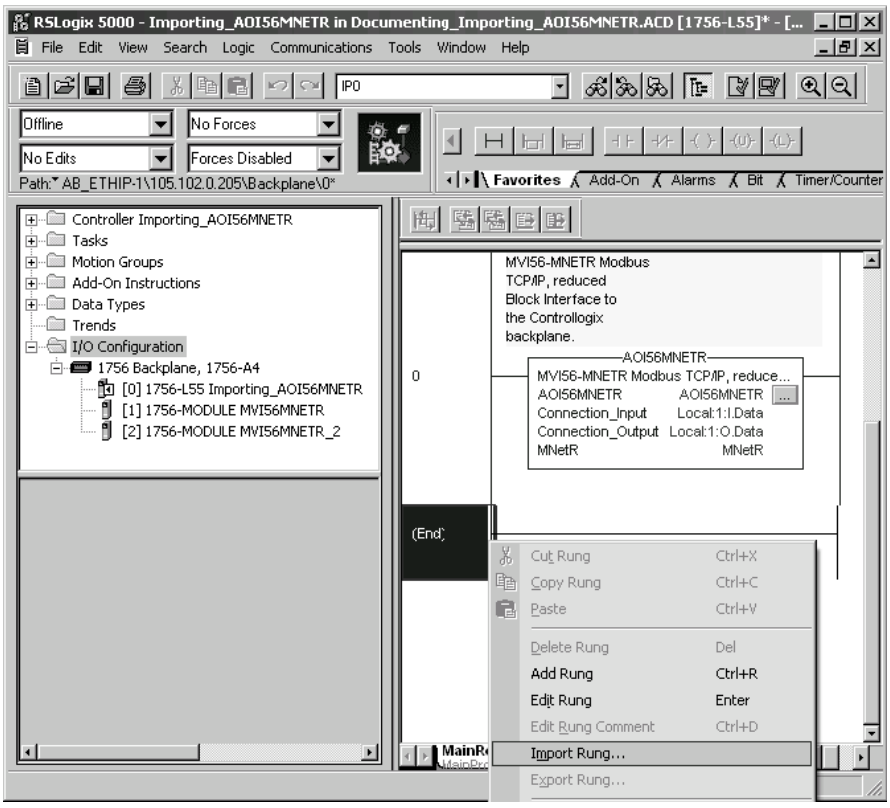


Parameter	Value
Name	Enter a module identification string. Example: MVI56MNETR_2
Description	Enter a description for the module. Example: ProSoft communication module for ModbusTCP/IP communication.
Comm Format	Select DATA-INT
Slot	Enter the slot number in the rack where the MVI56-MNETR module is located.
Input Assembly Instance	1
Input Size	42
Output Assembly Instance	2
Output Size	42
Configuration Assembly Instance	4
Configuration Size	0

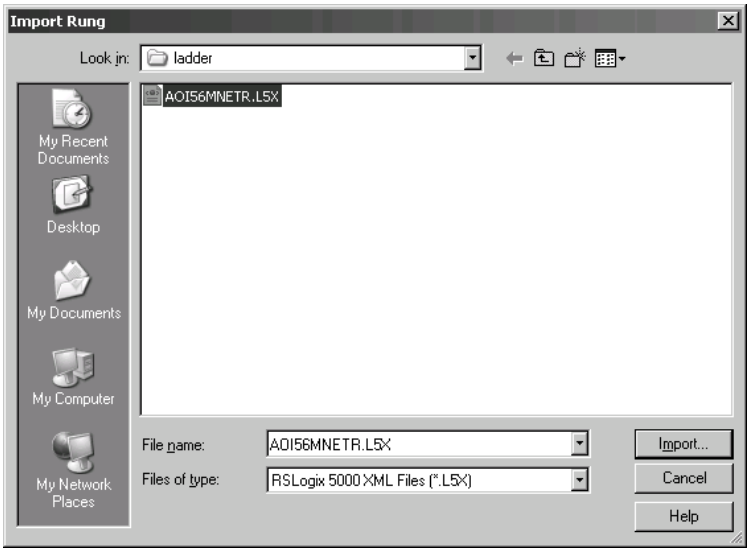
4 Click OK to confirm. The new module is now visible:



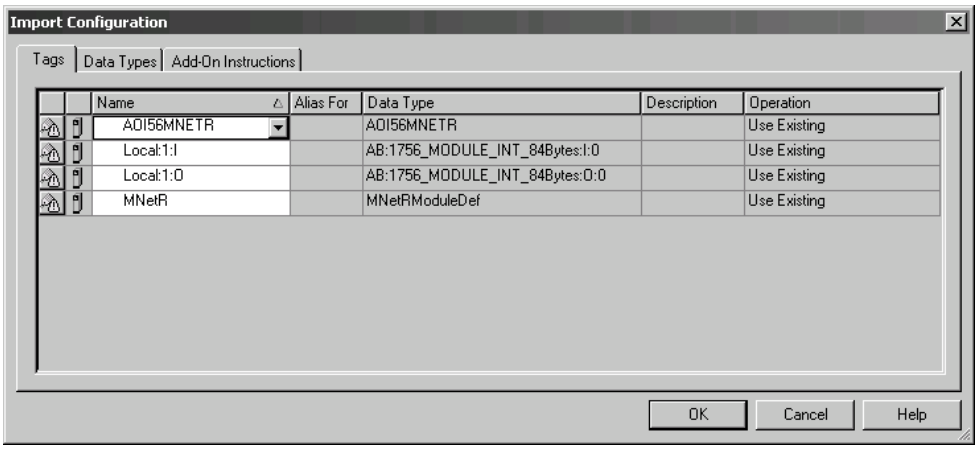
5 Expand the Tasks folder, and then expand the MainTask folder.
6 On the MainProgram folder, click the right mouse button to open a shortcut menu. On the shortcut menu, choose New Routine.
7 In the New Routine dialog box, enter the name and description of your routine, and then click OK.
8 Select an empty rung in the new routine, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose "Import Rung…".



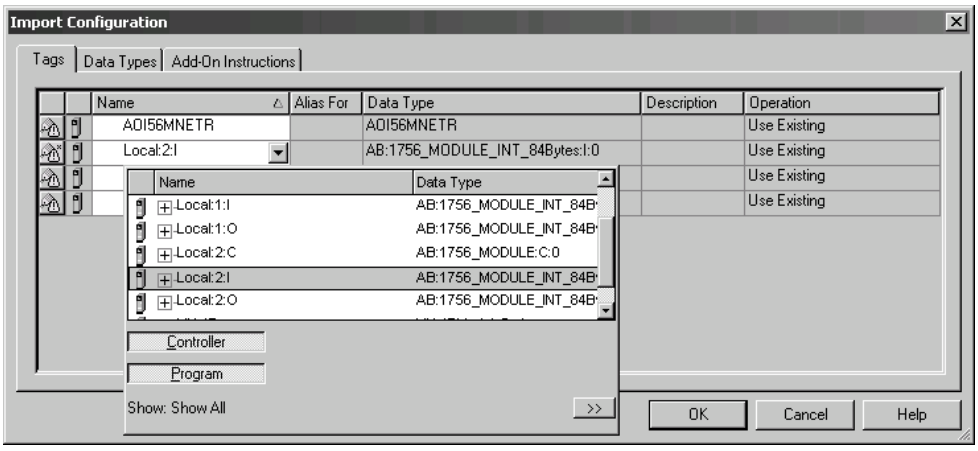
9 Select the file AOI56MNETR.L5X

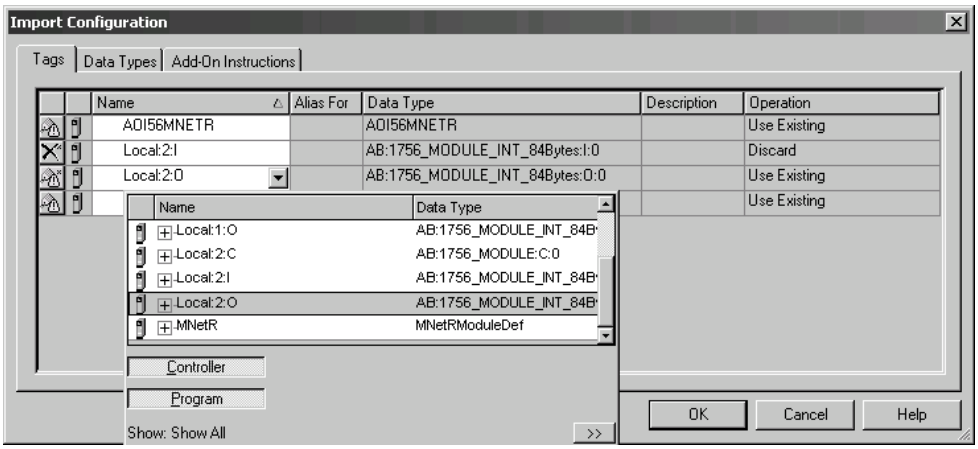


10 The following window will be displayed showing the tags to be imported:

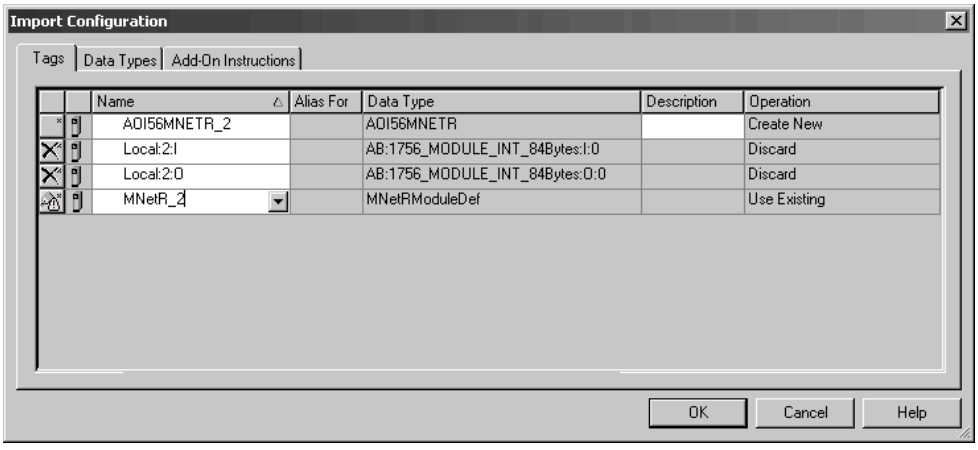


11 Associate the I/O connection variables to the correct module. The default values are Local:1:I and Local:1:O so these require change.

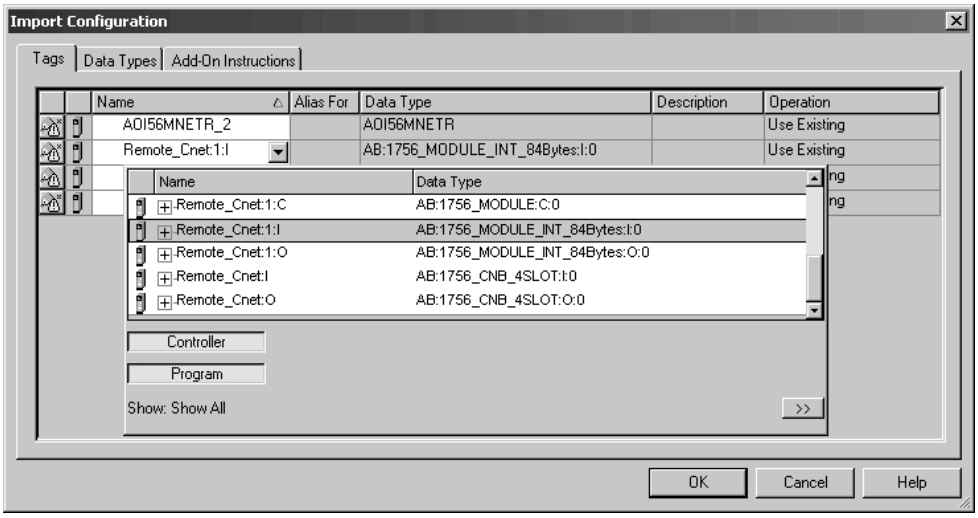


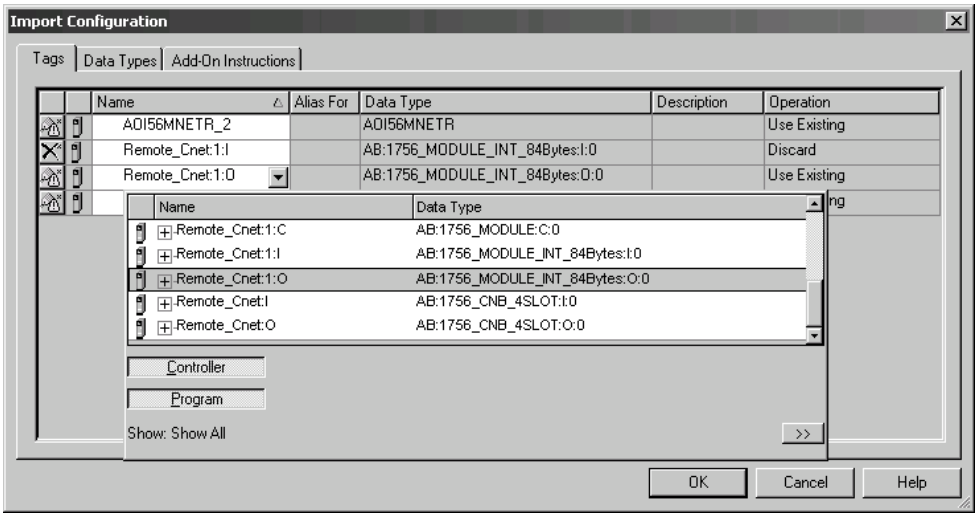


Change the default tags MNETR and AOI56MNETR to avoid conflict with existing tags. This procedure will append the string "_2" as follows:

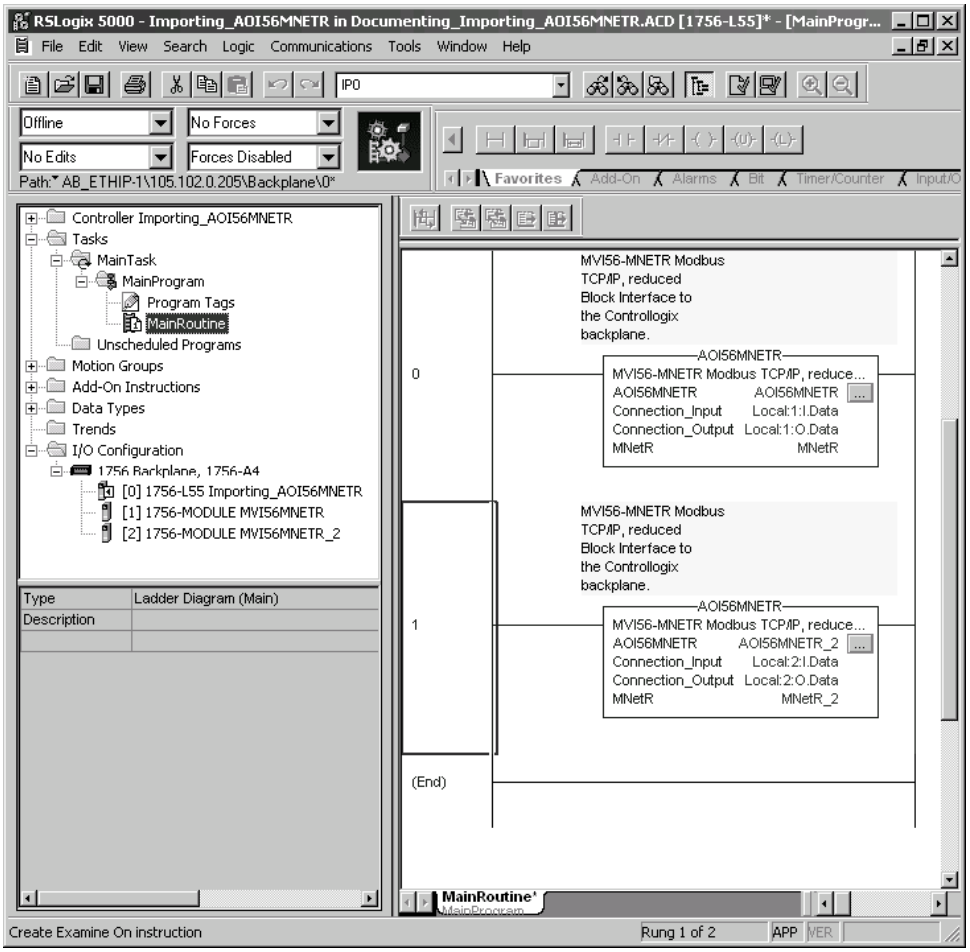


Or, in a Remote Rack application…

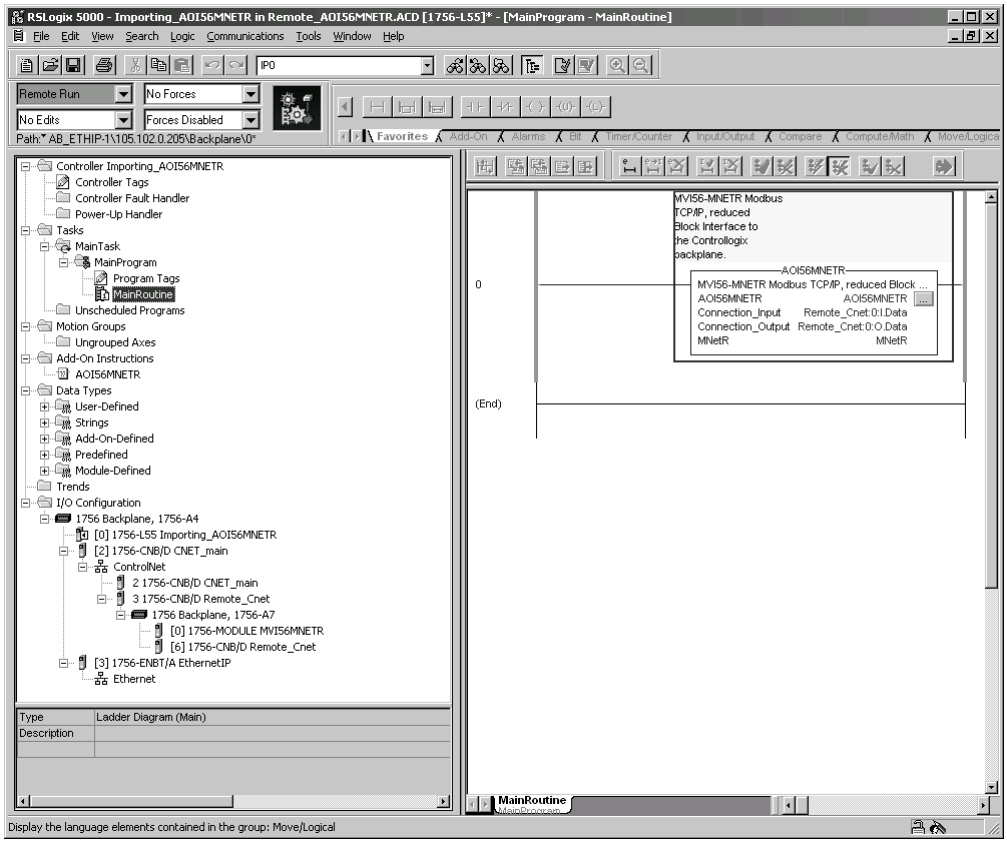


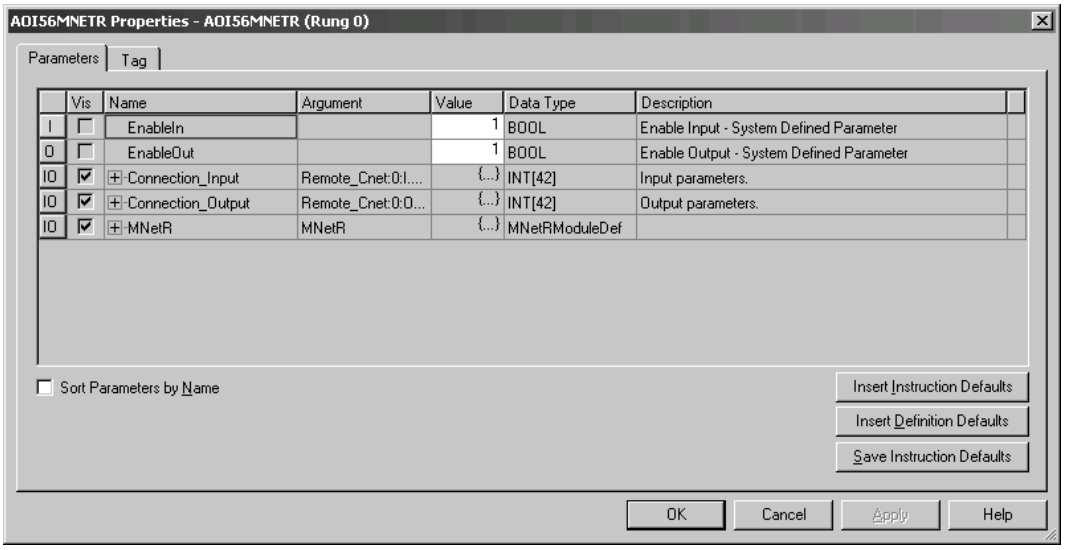


1 Click OK to confirm.



Or, in a Remote Rack application…





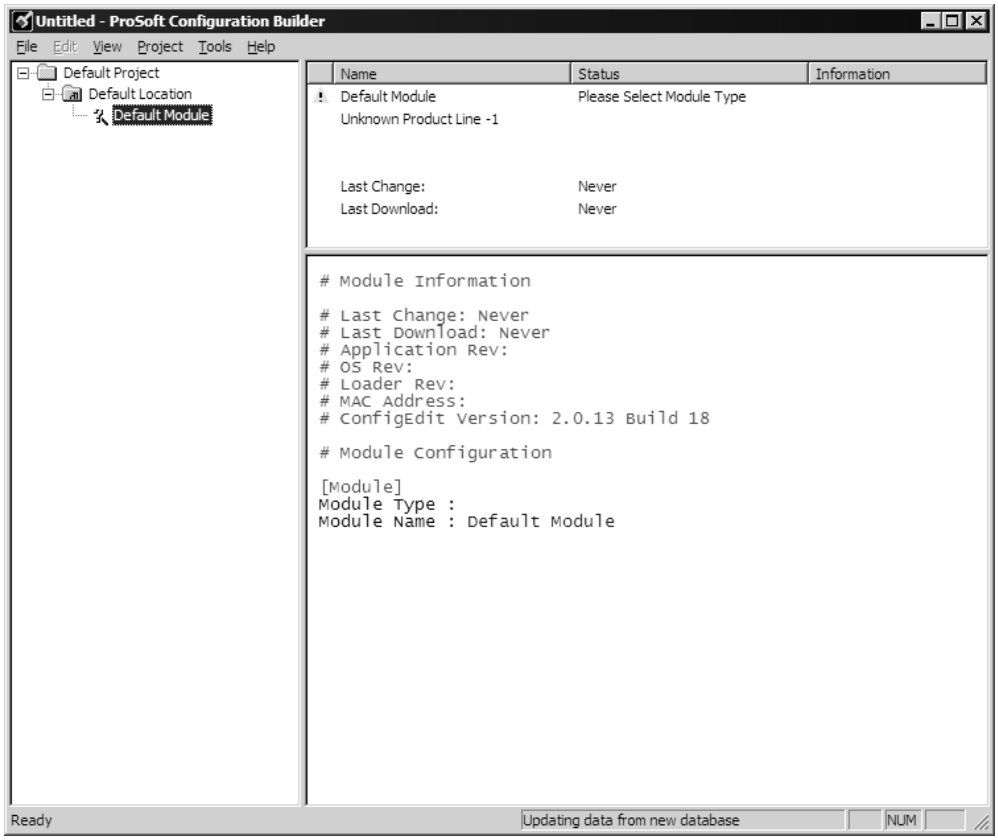
The setup procedure is now complete. Save the project and download the application to your ControlLogix processor.

2.2 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.

2.2.1 Set Up the Project

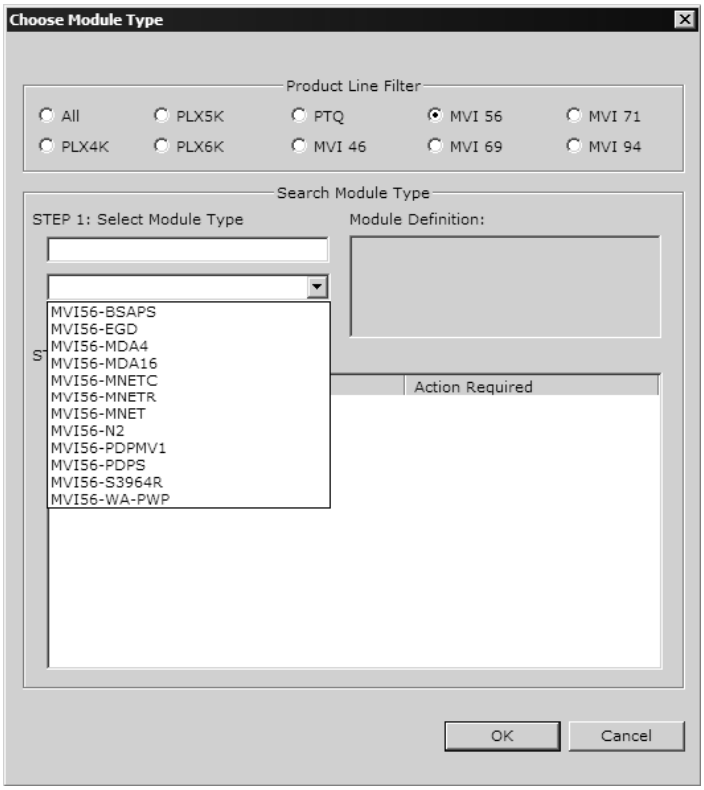
To begin, start ProSoft Configuration Builder. If you have used other Windows configuration tools before, you will find the screen layout familiar. ProSoft Configuration Builder's window consists of a tree view on the left, an information pane and a configuration pane on the right side of the window.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



Your first task is to add the MVI56-MNETR 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-MNETR, and then click OK to save your settings and return to the ProSoft Configuration Builder window.

The next task is to set the module parameters.

Adding a Project

To add a project to an existing project file:

1 Select the Default Project icon.
2 Choose Project from the Project menu, then choose Add Project. A new project folder appears.

Adding a Module

To add a module to your project:

1 Double-click the Default Module icon to open the Choose Module Type dialog box.
2 On the Choose Module Type dialog box, select the module type.

Or

1 Open the Project menu and choose Location.
2 On the Location menu, choose Add Module.

To add a module to a different location:

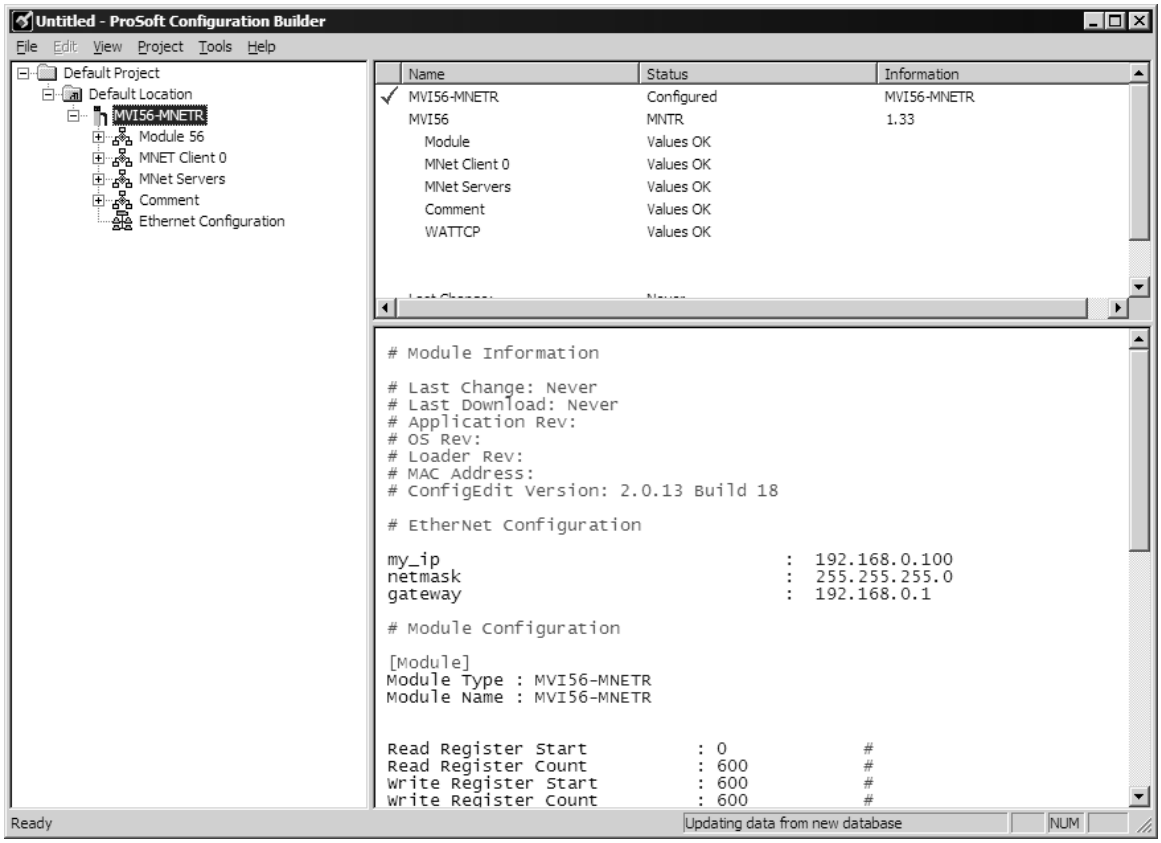
1 Right-click the Location folder and choose Add Module. A new module icon appears.

Or

1 Select the Location icon.
2 From the Project menu, select Location, then select Add Module.

2.2.2 Set Module Parameters

Notice that the contents of the information pane and the configuration pane changed when you added the MVI56-MNETR module to the project.



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

To rename an object:

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

Module Entries

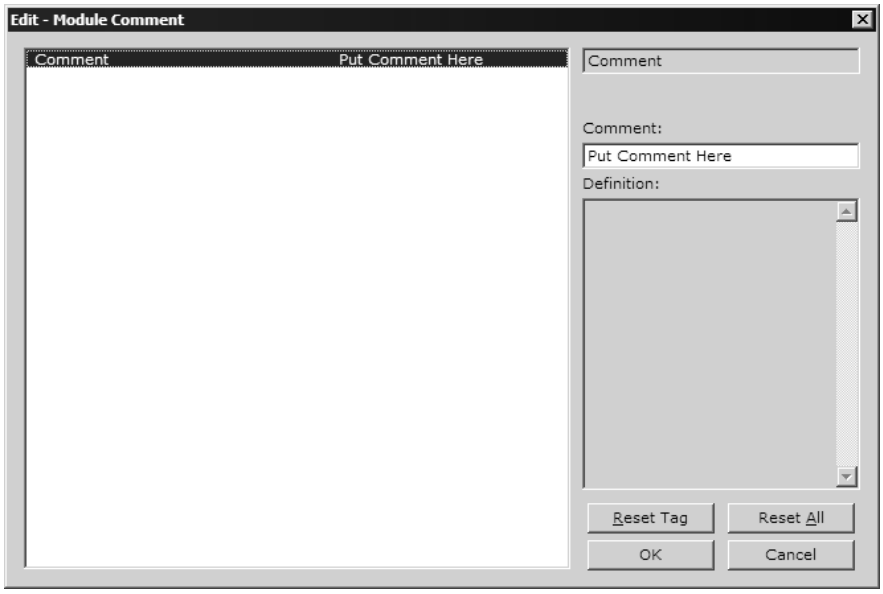
To configure module parameters

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

Comment Entries

To add comments to your configuration file:

1. Click the plus sign to the left of the Comment icon to expand the Module Comments.
2. Double-click the Module Comment icon. The Edit - Module Comment dialog appears.



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

Printing a Configuration File

To print a configuration file:

1. Select the Module icon, and then click the right mouse button to open a shortcut menu.
2. On the shortcut menu, choose View Configuration. This action opens the View Configuration window.
3. On the View Configuration window, open the File menu, and choose Print. This action opens the Print dialog box.
4. On the Print dialog box, choose the printer to use from the dropdown list, select printing options, and then click OK.

2.3 Configuration Data

This section contains a listing of the parameters and their definitions for the MVI56-MNETR module configuration.

MVI56-MNETR COMMUNICATION MODULE CONFIGURATION			
[Section]/Item	**Value**	**Range**	**Description**
[MODULE]			Configuration header for general module information
Module Name:		Up to 80 chars	Name of the module for use on reports. Use this parameter to identify your module in your system.
Error/Status Pointer:		-1 to 4955	Starting register location in virtual Modbus database for the error/status table. If a value of -1 is entered, the error/status data will not be placed in the database. All other valid values determine the starting location of the data. This data area includes the module version information and all server error/status data.
Write Register Start:		0 to 4999	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 4999.
Write Register Count:		0 to 5000	This parameter specifies the number of registers to transfer from the processor to the module. Valid entry for this parameter is 0 to 5000.
Read Register Start:		0 to 4999	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 4999.
Read Register Count:		0 to 5000	This parameter specifies the number of registers to be transferred from the module to the processor. Valid entry for this parameter is 0 to 5000.
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 (1 to 65535), communications will cease if the specified number of failures occur.
Initialize Output Data:		0 or 1	This parameter determines if the output data for the module should be initialized with values from the processor. If the value is set to 0, the output data will be initialized to 0. If the value is set to 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.
Pass-Through Mode:		0, 1, 2 or 3	This parameter specifies the pass-through mode for write messages received by the MNET and MBAP server ports. If the parameter is set to 0, all write messages will be placed in the module's virtual database. If a value of 1 is entered, write messages received will be sent to the processor as unformatted messages. If a value of 2 is entered, write messages received will be sent to the processor as formatted messages. If a value of 3 is entered, write messages received will be sent to the processor with the bytes swapped in a formatted message.

[Section]/Item	Value	Range	Description
[MNET CLIENT 0]			**Start header for Client 0**
Error/Status Pointer:		-1 to 4990	Starting register location in virtual Modbus database for the error/status table for this client. If a value of -1 is entered, the error/status data will not be placed in the database. All other valid values determine the starting location of the data.
Minimum Command Delay:		0 to 65535	This parameter specifies the number of milliseconds to wait between the initial issuance of a command. This parameter can be used to delay all commands sent to slaves to avoid "flooding" commands on the network. This parameter does not affect retries of a command as they will be issued when failure is recognized.
Command Error Pointer:		-1 to 4999	This parameter sets the address in the internal database where the command error data will be placed. If the value is set to -1, the data will not be transferred to the database.
Response Timeout:		0 to 65535	This parameter represents the message response timeout period in 1-ms increments. This is the time that a client will wait before re-transmitting a command if no response is received from the addressed slave. The value is set depending upon the communication network used and the expected response time of the slowest device on the network.
Retry Count:		0 to 10	This parameter specifies the number of times a command will be retried if it fails.

[Section]/Item	Value	Range	Description
[MNET Servers]			Start header for Client 0
Float Flag:		Yes or No	This flag specifies if the floating-point data access functionality is to be implemented. If the float flag is set to Yes, Modbus functions 3, 6 and 16 will interpret floating point values for registers as specified by the two following parameters.
Float Start:		0 to 65535	This parameter defines the first register of floating-point data. All requests with register values greater-than or equal to this value will be considered floating-point data requests. This parameter is only used if the Float Flag is enabled. For example, if a value of 7000 is entered, all requests for registers 7000 and above will be considered as floating-point data.
Float Offset:		0 to 9999	This parameter defines the start register for floating-point data in the internal database. This parameter is used only if the Float Flag is enabled. For example, if the Float Offset value is set to 3000 and the float start parameter is set to 7000, data requests for register 7000 will use the internal Modbus register 3000.

[Section]/Item	Value	Range	Description
Output Offset:		0 to 3999	This parameter defines the start register for Modbus Command data in the internal database. This parameter is enabled when a value greater than 0 is set. For example, if the Output Offset value is set to 3000, data requests for Modbus Coil register address 00001 will use the internal database register 3000, bit 0. If the Output Offset value is set to 3000, data requests for Modbus Coil register address 00016 will use the internal database register 3000, bit 15. Function codes affected are 1, 5, and 15.
Bit Input Offset		0 to 3999	This parameter defines the start register for Modbus Command data in the internal database. This parameter is enabled when a value greater than 0 is set. For example, if the Bit Input Offset value is set to 3000, data requests for Modbus Input register 10001 will use the internal database register 3000, bit 0. If the Bit Input Offset value is set to 3000, data requests for Modbus Coil register address 10016 will use the internal database register 3000, bit 15. Function code affected is 2.
Holding Register Offset		0 to 4999	This parameter defines the start register for Modbus Command data in the internal database. This parameter is enabled when a value greater than 0 is set. For example, if the Holding Register Offset value is set to 4000, data requests for Modbus Word register address 40001 will use the internal database register 4000. Function codes affected are 3, 6, 16, and 23.
Word Input Offset		0 to 4999	This parameter defines the start register for Modbus command data in the internal database. This parameter is enabled when a value greater than 0 is set. For example, if the word Input Offset value is set to 4000, data requests for Modbus Word register address 30001 will use the internal database register 4000. Function code affected is 4.

2.4 Command List Overview

In order to interface the MVI56-MNETR module with Modbus TCP/IP Server devices, you must construct a command list. The commands in the list specify the Server device to be addressed, the function to be performed (read or write), the data area in the device to interface with and the registers in the internal database to be associated with the device data. The Client command list supports up to 100 commands.

The command list is processed from top (command #0) to bottom. A poll interval parameter is associated with each command to specify a minimum delay time in tenths of a second between the issuance of a command. If the user specifies a value of 10 for the parameter, the command will be executed no more frequently than every 1 second.

Write commands have a special feature, as they can be set to execute only if the data in the write command changes. If the register data values in the command have not changed since the command was last issued, the command will not be executed.

If the data in the command has changed since the command was last issued, the command will be executed. Use of this feature can lighten the load on the network. In order to implement this feature; set the enable code for the command to a value of 2.

2.5 Commands Supported by the Module

The format of each command in the list is dependent on the Modbus Function Code being executed. The tables below list the functions supported by the module:

Function Code	Definition	Supported in Client	Supported in Server
1	Read Coil Status	X	X
2	Read Input Status	X	X
3	Read Holding Registers	X	X
4	Read Input Registers	X	X
5	Set Single Coil	X	X
6	Single Register Write	X	X
7	Read Exception Status		X
15	Multiple Coil Write	X	X
16	Multiple Register Write	X	X
22	Mask Write 4X		X
23	Read/Write		X

Each command list record has the same general format. The first part of the record contains the information relating to the communication module and the second part contains information required to interface to the Modbus TCP/IP Server device.

2.6 Command Entry Formats

The following table shows the structure of the configuration data necessary for each of the supported commands.

2.6.1 MNET MODBUS Command Structure

Column #	1	2	3	4	5	6	7	8	9	10
Function Code	Enable Code	Internal Address	Poll Interval Time	Count	Swap Code	IP Address	Serv Port	Slave Node	Function Code	Device Modbus Address
fc1	Code	Register	1/10th Seconds	Count	0	IP Address	Port #	Address	1	Register
fc2	Code	Register	1/10th Seconds	Count	0	IP Address	Port #	Address	2	Register
fc3	Code	Register	1/10th Seconds	Count	Code	IP Address	Port #	Address	3	Register
fc4	Code	Register	1/10th Seconds	Count	Code	IP Address	Port #	Address	4	Register
fc5	Code	Register	1/10th Seconds	Count	0	IP Address	Port #	Address	5	Register
fc6	Code	Register	1/10th Seconds	Count	0	IP Address	Port #	Address	6	Register
fc15	Code	Register	1/10th Seconds	Count	0	IP Address	Port #	Address	15	Register
fc16	Code	Register	1/10th Seconds	Count	Code	IP Address	Port #	Address	16	Register

The first part of the record is the Module Information, which relates to the ProLinx module and the second part contains information required to interface to the Server device.

Command list example:

```
[MNet Client 0 Commands]
#
# The file contains examples for a Modbus TCP/IP control using MBAP (port 502)
# and MNET (port 2000) service ports.
#
#   1     2      3      4     5  6                   7     8    9      10
#        DB   Poll    Reg  Swap                    Serv Slave Func Address
#Enab  Addr  Delay  Count  Code  Node IP Address   Port  Addr Code  In Dev
START
    1    30      0     10     0  192.168.0.57       502     1    3      10
    0     0      0     10     0  192.168.0.57       502     1   16      10
END
```

<table>
<tr><th>Parameter</th><th>Range</th><th>Description</th></tr>
<tr><td>Enable</td><td>0,1,2</td><td>This field defines whether or not the command is to be executed and under what conditions.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>The command is disabled and will not be executed in the normal polling sequence.</td></tr>
<tr><td>1</td><td>The command is executed each scan of the command list if the Poll Interval Time is set to zero. If the Poll Interval time is set, the command will be executed, when the interval timer expires.</td></tr>
<tr><td>2</td><td>The command will execute only if the internal data associated with the command changes. This value is valid only for write commands.</td></tr>
</table></td></tr>
<tr><td>Internal Address</td><td>0 to 4999</td><td>This field specifies the internal database register to be associated with the command.
- If the command is a read function, the data read from the Server device will be placed starting at the register value entered in this field.
- If the command is a write function, the data written to the Server device will be sourced from the address specified.</td></tr>
<tr><td>Poll Interval</td><td>0 to 65535</td><td>This parameter specifies the minimum interval to execute continuous commands (Enable code of 1). The parameter is entered in units of 1/10th seconds. Therefore, if a value of 10 is entered for a command, the command will execute no more frequently than every 1 second.</td></tr>
<tr><td>Count</td><td>Regs
1 to 125
Coils
1 to 2000</td><td>This parameter specifies the number of registers or digital points to be associated with the command.
Functions 5 and 6 ignore this field as they only apply to a single data point.
For functions 1, 2 and 15, this parameter sets the number of digital points (inputs or coils) to be associated with the command.
For functions 3, 4 and 16, this parameter sets the number of registers to be associated with the command.</td></tr>
<tr><td>Swap Code</td><td>0,1,2,3</td><td>This parameter is used only for functions 3, 4, 6 and 16 to define if the data received (or sent) from the module is to be ordered differently than data received from the server device. This parameter is helpful when dealing with floating-point or other multi-register values, as there is no standard method of storage of these data types in Server devices. This parameter can be set to order the register data received in an order useful by other applications. The following table defines the values and their associated operations:
<table>
<tr><th>Swap Code</th><th>Description</th></tr>
<tr><td>0</td><td>None - No Change is made in the byte ordering (1234 = 1234)</td></tr>
<tr><td>1</td><td>Words - The words are swapped (1234=3412)</td></tr>
<tr><td>2</td><td>Words & Bytes - The words are swapped then the bytes in each word are swapped (1234=4321)</td></tr>
<tr><td>3</td><td>Bytes - The bytes in each word are swapped (1234=2143)</td></tr>
</table>
When swapping words, make sure you are using an even value in the Count Field. Odd values may generate unexpected results.</td></tr>
<tr><td>Node IP Address</td><td>xxx.xxx.xxx.xxx</td><td>The IP address of the device being addressed by the command.</td></tr>
<tr><td>Service Port</td><td>502
or other supported ports on server</td><td>Use a value of 502 when addressing Modbus TCP/IP servers which are compatible with the Schneider Electric MBAP (Modbus API for network communications) specifications (this will be most devices). If a server implementation supports another service port, enter the value here.</td></tr>
</table>

<table>
<tr><th>Parameter</th><th>Range</th><th>Description</th></tr>
<tr><td>Slave Node</td><td>1 to 255 (0 is a broadcast)</td><td>This parameter specifies the Modbus slave node address on the network to be considered. Values of 1 to 255 are permitted. Most Modbus devices only accept an address in the range of 1 to 247 so be careful. If the value is set to zero, the command will be a broadcast message on the network. The Modbus protocol permits broadcast commands for write operations. Do not use this node address for read operations.</td></tr>
<tr><td>Function Code</td><td>1,2,3,4,5,6,
15,16</td><td>This parameter specifies the Modbus function to be executed by the command. These function codes are defined in the Modbus protocol. The following table defines the purpose of each function supported by the module. More information on the protocol is available from the Schneider Electric web site (www.modicon.com).
<table>
<tr><th>Modbus Function Code</th><th>Description</th></tr>
<tr><td>1</td><td>Read Coil Status</td></tr>
<tr><td>2</td><td>Read Input Status</td></tr>
<tr><td>3</td><td>Read Holding Registers</td></tr>
<tr><td>4</td><td>Read Input Registers</td></tr>
<tr><td>5</td><td>Single Coil Write</td></tr>
<tr><td>6</td><td>Single Register Write</td></tr>
<tr><td>15</td><td>Multiple Coil Write</td></tr>
<tr><td>16</td><td>Multiple Register Write</td></tr>
</table></td></tr>
<tr><td>Device Address</td><td></td><td>This parameter specifies the starting Modbus register or digital point address to be considered by the command in the Modbus slave device. Refer to the documentation of each Modbus slave device on the network for their register and digital point address assignments.

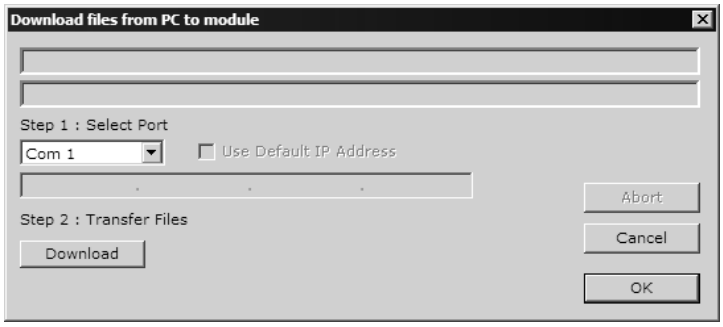
The FC determines the addresses range and that this value will be the register or bit OFFSET into a given data range. For instance, if the command is to be a bit command (FC 1, 2, 5, or 15) to Read/Write a Coil 0X address 00001, then the value to enter here would be 0. For Coil address 00110, the value here would be 109. For register Read/Write commands (FC 3, 4, 6, or 16) in the 3X (FC4) or 4X (FC3), say 30001 or 40001, the value here would, again be 0. For 31101 or 41101, the value to enter for this parameter would be 1100.</td></tr>
</table>

2.7 Download the Project to the Module

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

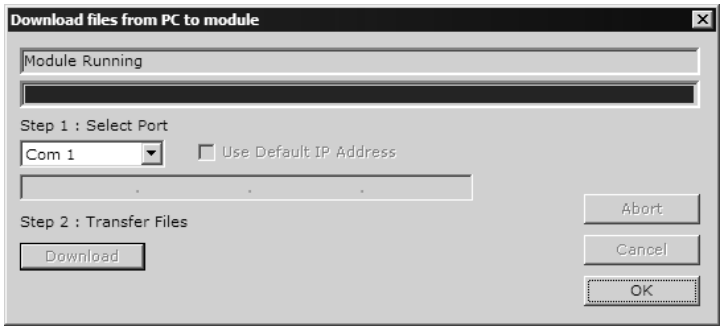
To Download the Project File

1. In the tree view in ProSoft Configuration Builder, click once to select the MVI56-MNETR module.
2. Open the **Project menu,** and then choose **Module / Download**. The program will scan your PC for a valid com port (this may take a few seconds). When PCB has found a valid com port, the following 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 ProSoft Configuration Builder will be updated with the message *"Module Running*".



3 Ladder Logic

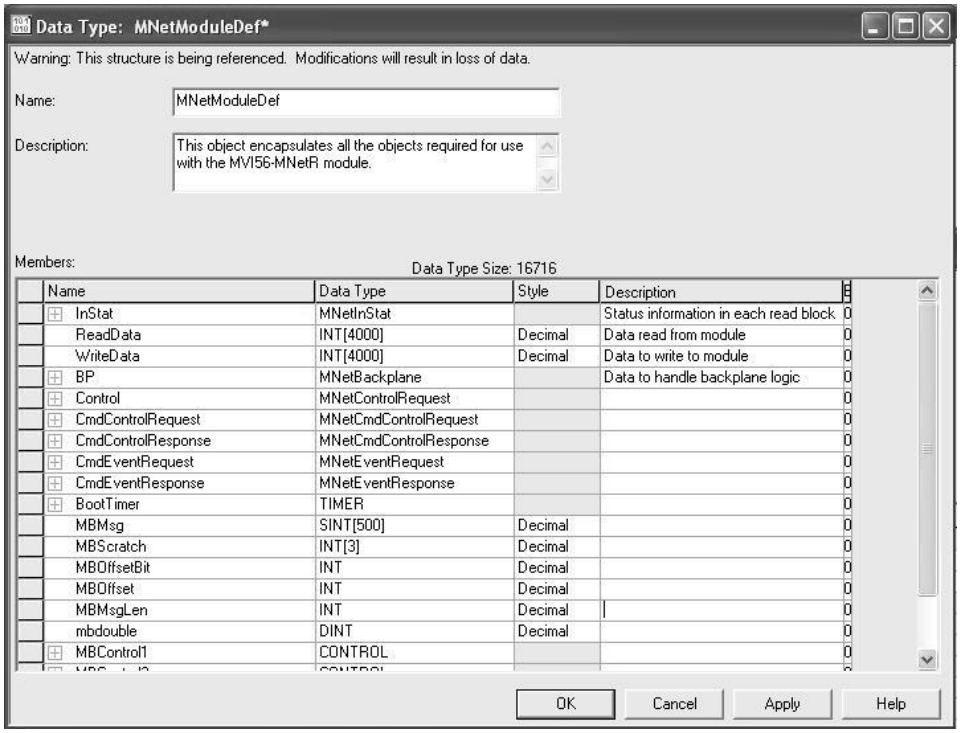
In This Chapter

Ladder logic is required for application of the MVI56-MNETR 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, on the ProSoft Solutions CD-ROM, 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 Object (MNETModuleDef)

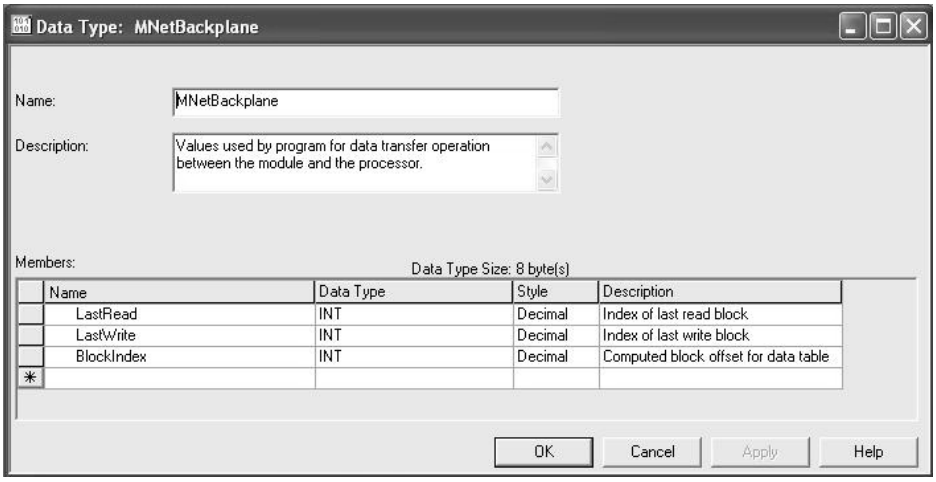
All data related to the MVI56-MNETR is stored in a user defined data type. An instance of the data type is required before the module can be used. This is done by declaring a variable of the data type in the Controller Tags Edit Tags dialog box. The structure of the object is displayed in the following illustration:



This object contains objects that define the configuration, user data, status and command control data related to the module. Each of these object types is discussed in the following topics of the document.

3.1.1 Backplane Control Object

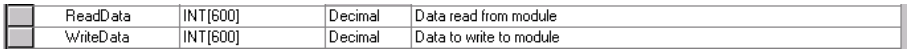
This data object stores the variables required for the data transfer between the processor and the MVI56-MNETR module. The structure of the object is shown below:



The LastRead tag stores the latest Read Block ID received from the module. The LastWrite tag stores the latest Write Block ID to be sent to the module. The Block Index tag is an intermediate variable used during the block calculation.

3.1.2 User Data Objects

These objects hold data to be transferred between the processor and the MVI56-MNETR module. The user data is the read and write data transferred between the processor and the module as "pages" of data up to 40 words long.

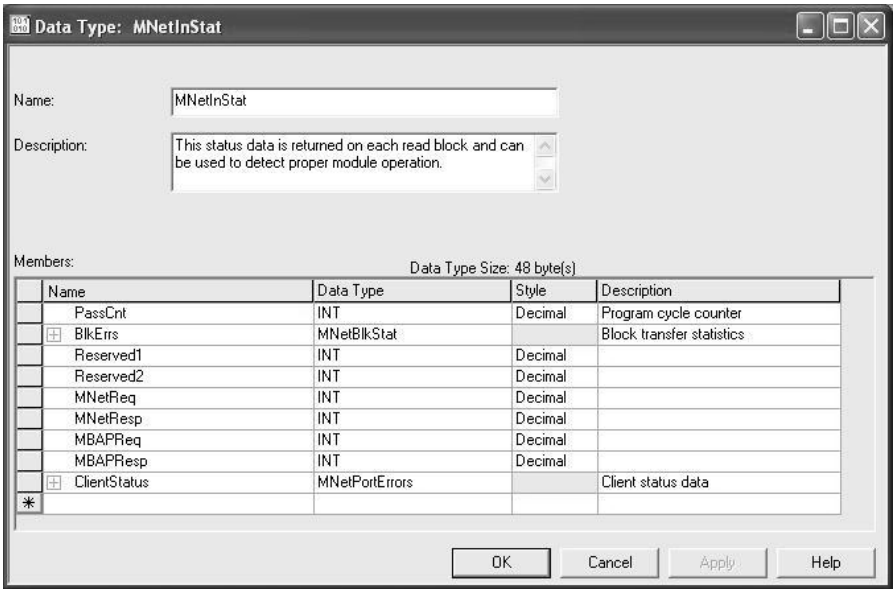


The read data (**ReadData**) is an array set to match the value entered in the **Read Register Count** parameter of the MNET.CFG file. For ease of use, this array should be dimensioned as an even increment of 40 words. This data is paged up to 50 words at a time from the module to the processor. The ReadData task places the data received into the proper position in the read data array. Use this data for status and control in the ladder logic of the processor.

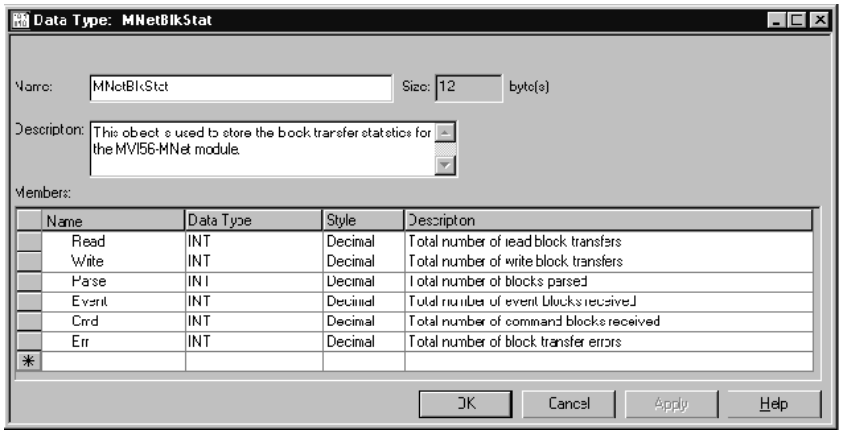
The write data (**WriteData**) is an array set to match the value entered in the **Write Register Count** parameter of the MNET.CFG file. For ease of use, this array should be dimensioned as even increments of 40 words. This data is paged up to 40 words at a time from the processor to the module. The WriteData task places the write data into the output image for transfer to the module. This data is passed from the processor to the module for status and control information for use in other nodes on the network.

3.1.3 Status Data

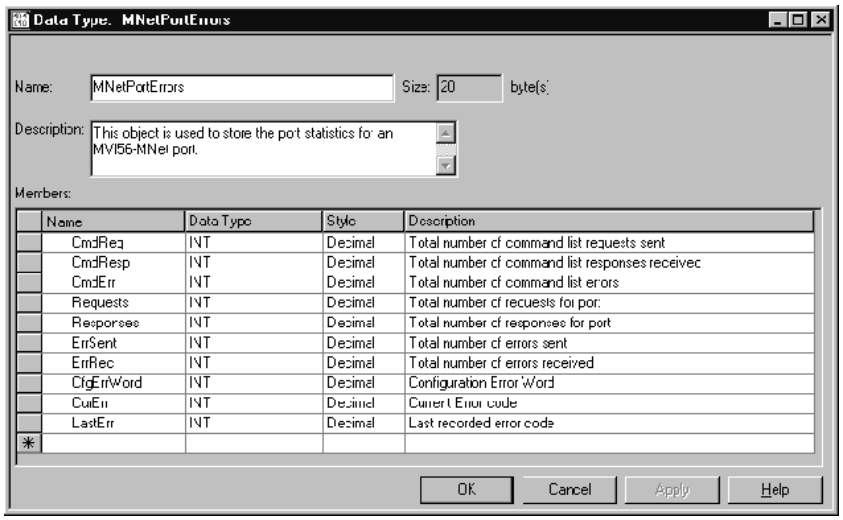
This object views the status of the module. The **MNetInStat** object shown below is updated each time a read block is received by the processor. Use this data to monitor the state of the module at a "real-time rate".



This object contains a structure that includes the status information for the data transfer operations between the processor and the module. The structure of this object is displayed below:



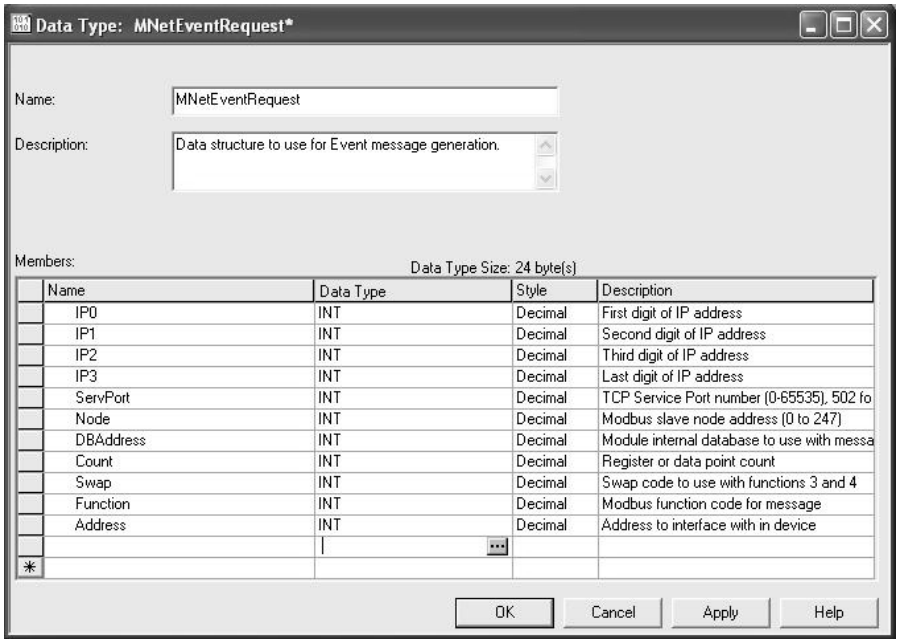
Additionally, the status object contains a structure for each of the servers in the module. The structure used for each server is shown below:

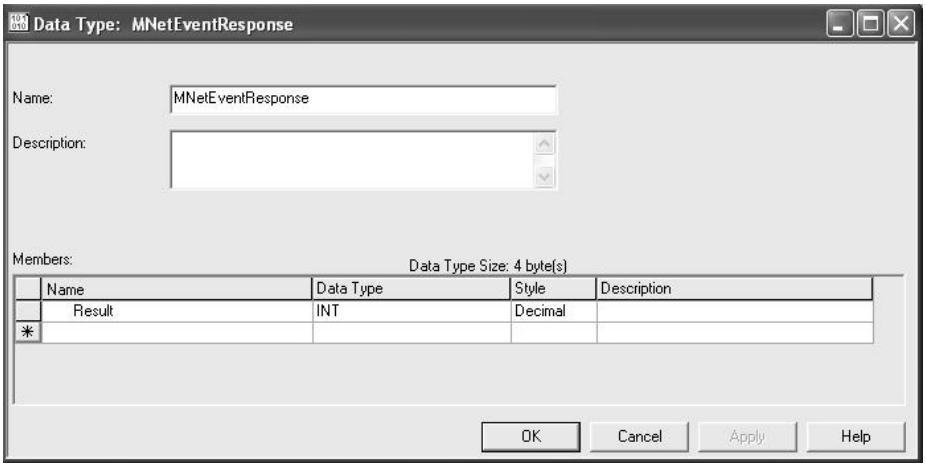


Refer to MVI56-MNETR Status Data Definition for a complete listing of the data stored in status object.

3.1.4 Event Command Data

The MNetEventRequest and MNetEventResponse structures hold the information required for an event command. An array of these objects should be defined and hold the event command set to be employed in the application. The structure of the object is shown below:





3.1.5 Modbus Message Data

During pass-through operation, write messages received at the MVI56-MNETR server write messages through to the processor. It is the responsibility of the ladder logic to process the message received using this feature. Two data objects are required for this mode: a variable to hold the length of the message and a buffer to hold the message.

This information is passed from the module to the processor using a block identification code of 9996 if the unformatted pass-through mode (code 1) is selected as the pass through mode in the configuration file. Word one of this block contains the length of the message and the message starts at word 3. Other controller tags are required to store the controlled values contained in these messages. The Modbus protocol supports control of binary output (coils - functions 5 and 15) and registers (functions 6 and 16).

Additionally, formatted message blocks can be sent from the module to the processor when the pass-through option is selected using the format selection (codes 2 or 3 in the MNET.CFG file). These blocks require less decoding than the unformatted blocks. Refer to the user manual for a full discussion on utilizing the pass-through option in an application.

4 Diagnostics and Troubleshooting

In This Chapter

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

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

4.1 Reading Status Data from the Module

The MVI56-MNETR module returns a 47-word Status Data block that can be used to determine the module's operating status. This data is located in the module's database at a user set location. This data is transferred to the ControlLogix processor continuously. For a complete listing of the status data object, refer to the Reference chapter.

The Configuration/Debug port provides the following functionality:

- Full view of the module's configuration data
- View of the module's status data
- Complete display of the module's internal database (registers 0 to 4999)
- Version Information
- Control over the module (warm boot, cold boot, transfer configuration)
- Facility to upload and download the module's configuration file

4.1.1 Required Hardware

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

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

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

4.1.2 The Configuration/Debug Menu

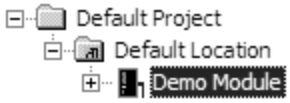
The Configuration and Debug menu for this module is arranged as a tree structure, with the Main Menu at the top of the tree, and one or more sub-menus for each menu command. The first menu you see when you connect to the module is the Main menu.

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

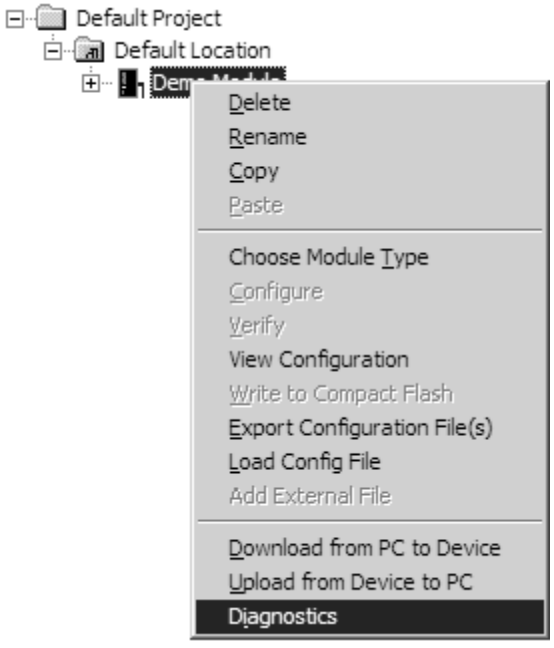
Using the Diagnostic Window in ProSoft Configuration Builder

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

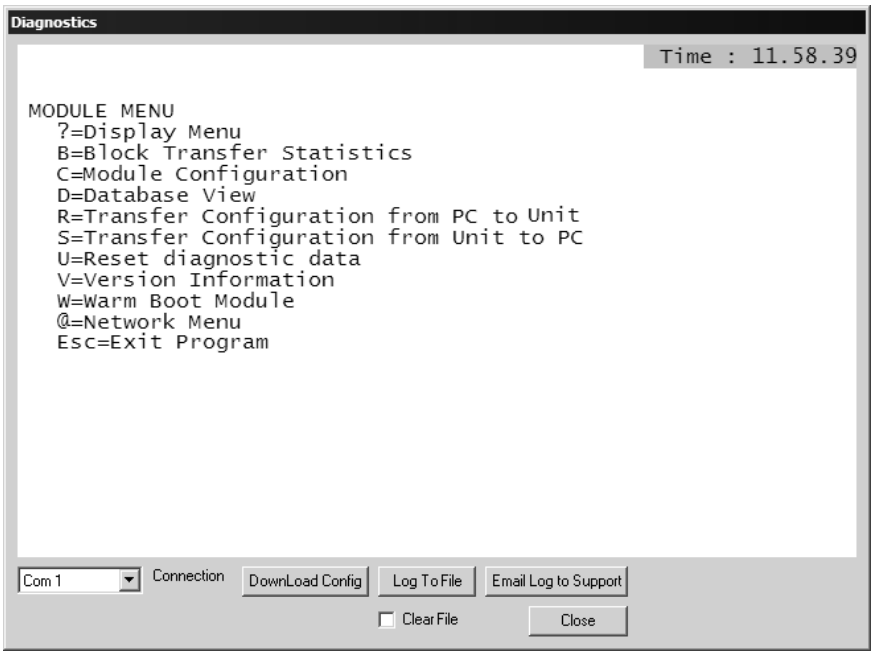
1 Start PCB program with the application file to be tested. Right click over the module icon.



2 On the shortcut menu, choose Diagnostics.



3 This action opens the Diagnostics dialog box. Press "?" to display the Main Menu.



Important: The illustrations of configuration/debug menus in this section are intended as a general guide, and may not exactly match the configuration/debug menus in your own module.

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

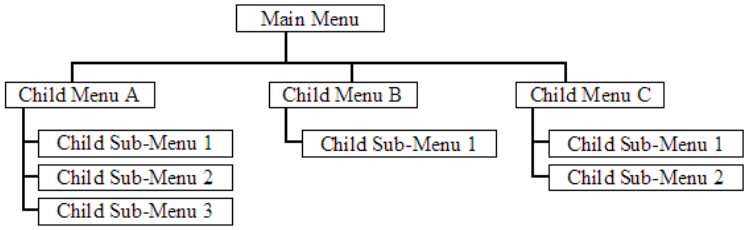
1 Verify that the null modem cable is connected properly between your computer's serial port and the module. A regular serial cable will not work.
2 On computers with more than one serial port, verify that your communication program is connected to the same port that is connected to the module.

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

Navigation

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

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



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

Keystrokes

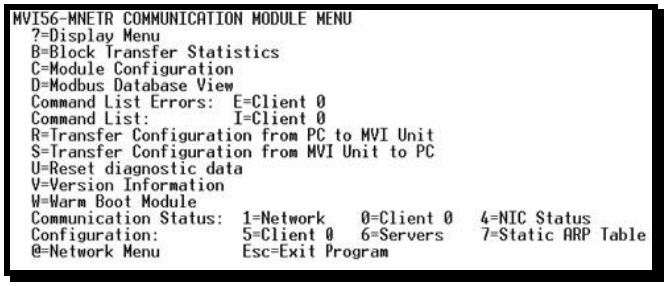
The keyboard commands on these menus are almost always non-case sensitive. You can enter most commands in lower case or capital letters.

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

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

4.1.3 Main Menu

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



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 Menu

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

Opening the Command List Menu

Press **[L]** to open the Command List menu. Use this command to view the configured command list for the module.

Opening the Command Error List Menu

Press **[I]** to open the Command Error List. This list consists of multiple pages of command list error/status data. Press **[?]** to view a list of commands available on this menu.

Receiving the Configuration File

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

Sending the Configuration File

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

Resetting diagnostic data

Press **[U]** to reset the status counters for the client and/or servers in the module.

Viewing Version Information

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

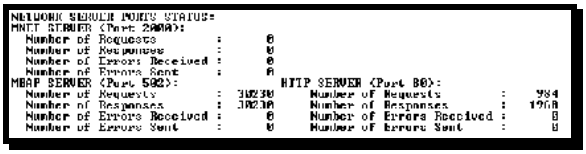
Use this command to view the current version of the software for the module, as well as other important values. You may be asked to provide this information when calling for technical support on the product.

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.

Viewing Network Status

Press **[1]** to display the statistics of the network server ports. After selecting the option, the following information will be displayed:



Viewing Client Status

Press **[0]** (zero) to display the statistics of the client.

Viewing NIC Status

Press **[4]** to view NIC status. Use this command to view the communication status for the Network Interface Card.

Viewing Client Configuration

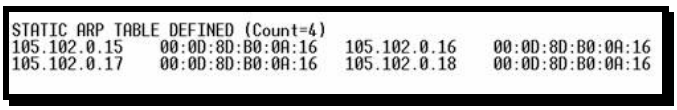
Press **[5]** to display the configuration information for the client.

Viewing Server Configuration

Press **[6]** to display the configuration information for the servers.

Viewing the Static ARP Table

Press [7] to view the Static ARP Table. Use this command to view the list of IP and MAC addresses that are configured not to receive ARP messages from the module.



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. You can find more information about the commands on this menu in the Network Menu (page 64) section.

Warm Booting the Module

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

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

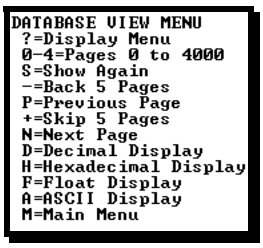
Exiting the Program

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

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

4.1.4 Modbus Database View

Press **[D]** to open the Modbus Database View menu. Use this command to view the module's internal database values. Press **[?]** to view a list of commands on this menu.



All data contained in the module's database is available for viewing using the commands. Refer to Modbus Protocol Specification for information on the structure of Modbus messages. Each option available on the menu is discussed in the following topics.

Viewing Register Pages

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

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

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

Redisplaying the Current Page

Press **[S]** to display the current page of data.

Moving Back Through 5 Pages of Registers

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

Viewing the Previous 100 Registers of Data

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

Skipping 500 Registers of Data

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

Viewing the Next 100 Registers of Data

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

Viewing Data in Decimal Format

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

Viewing Data in Hexadecimal Format

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

Viewing Data in Floating Point Format

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

Viewing Data in ASCII (Text) Format

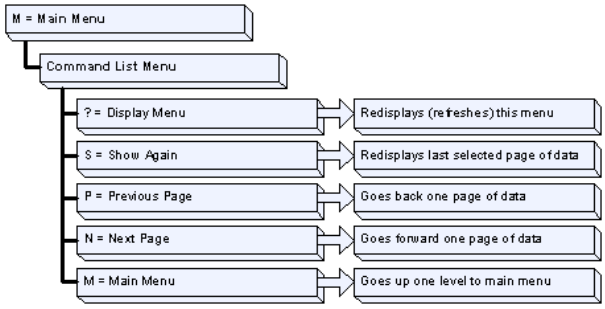
Press **[A]** to display the data on the current page in ASCII format. This is useful for regions of the database that contain ASCII data.

Returning to the Main Menu

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

4.1.5 Command List Menu

Use this menu to view the configured command list for the module. Press **[?]** to view a list of commands available on this menu.



Redisplaying the Menu

Press **[?]** to display the current menu. Use this command when you are looking at a screen of data, and want to view the menu choices available to you.

Viewing the Previous Page of Commands

Press **[P]** to display the previous page of commands.

Viewing the Next Page of Commands

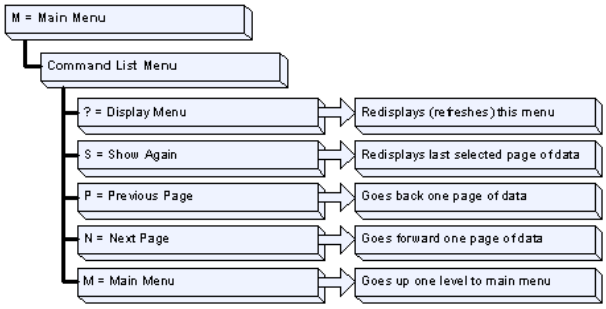
Press **[N]** to display the next page of commands.

Returning to the Main Menu

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

4.1.6 Master Command Error List Menu

Use this menu to view the command error list for the module. Press **[?]** to view a list of commands available on this menu.



Redisplaying the Menu

Press **[?]** to display the current menu. Use this command when you are looking at a screen of data, and want to view the menu choices available to you.

Viewing the Previous Page of Commands

Press **[P]** to display the previous page of commands.

Viewing the Next Page of Commands

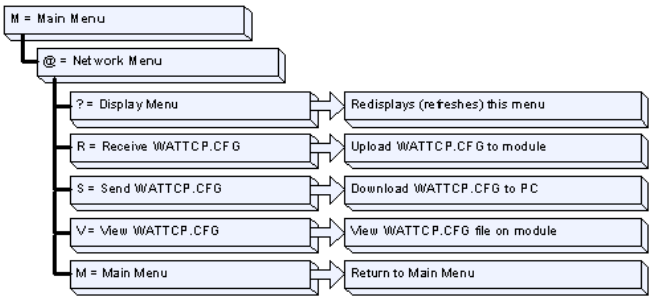
Press **[N]** to display the next page of commands.

Returning to the Main Menu

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

4.1.7 Network Menu

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

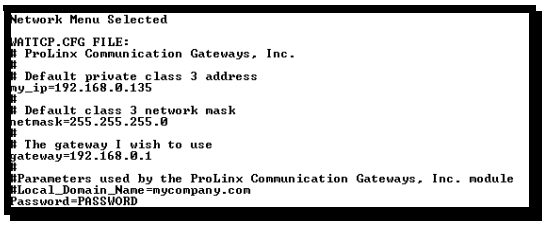
Press **[S]** to transfer the WATTCP.CFG file from the module to your PC.

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.



Returning to the Main Menu

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

4.2 LED Status Indicators

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

ProSoft Module	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
		Off	Port not used
P2	Green	On	Port not used
		Off	Port not used
APP	Amber	Off	The MVI56-MNETR is working normally.
		On	The MVI56-MNETR module program has recognized a communication error.
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 a configuration error is found for the client, the client configuration error word will have a value other than zero. The configuration error word bits have the following definitions:

Bit	Description	Value
0		0x0001
1		0x0002
2		0x0004
3		0x0008
4	Invalid retry count parameter	0x0010
5	The float flag parameter is not valid.	0x0020
6	The float start parameter is not valid.	0x0040
7	The float offset parameter is not valid.	0x0080
8		0x0100
9		0x0200
10		0x0400

Bit	Description	Value
11		0x0800
12		0x1000
13		0x2000
14		0x4000
15		0x8000

Correct any invalid data in the configuration for proper module operation. When the configuration contains a valid parameter set, all the bits in the configuration word will be clear. This does not indicate that the configuration is valid for the user application. Make sure each parameter is set correctly for the specific application.

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.

4.2.1 Ethernet LED Indicators

LED	State	Description
Data	Off	No activity on the port.
	Green Flash	The port is either actively transmitting or receiving data.
Link	Off	No connection to hub or network is detected.
	Green Solid	Connected to hub or network correctly. This is the normal operating state.

4.2.2 Clearing a Fault Condition

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

To clear the condition, follow these steps:

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

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

4.2.3 Troubleshooting

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

Processor Errors

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

Module Errors

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

5 Reference

In This Chapter

5.1 Product Specifications

The MVI56 Modbus TCP/IP Communication Module with Reduced Data Block allows ControlLogix I/O compatible processors to interface easily with other Modbus TCP/IP protocol compatible devices.

Compatible devices include not only Modicon processors (which support the Modbus TCP/IP protocol) but also a wide assortment of other clients and server devices.

This module uses a small I/O data area for data transfer between the module and the ControlLogix processor, making it ideal for ControlNet or Ethernet applications with the module in a remote rack.

5.1.1 Features and Benefits

The MVI56-MNETR module is a single slot solution that provides a powerful connection between a ControlLogix processor and Modbus TCP/IP network applications.

The Modbus TCP/IP network applications include those networks hosted by Modicon Quantum processors, networks controlled by operator interface software applications, and the growing number of manufactured devices that support this protocol. The MVI56-MNETR module acts as an input/output module between the Modbus TCP/IP network and the ControlLogix processor. The data transfer from the processor is asynchronous from the actions on the Modbus TCP/IP network. A 5000-word register space in the module exchanges data between the processor and the Modbus TCP/IP network.

- Support for the storage and transfer of up to 5000 registers to/from the ControlLogix processor
- User-definable module memory usage
- 10/100 Base-T Ethernet compatible interface
- Configurable parameters for the client including a minimum response delay of 0 to 65535 milliseconds and floating point support

5.1.2 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
- This module uses a small I/O data area for 40 words data block transfer between the module and the ControlLogix processor, for applications with the module in a remote rack.

5.1.3 Hardware Specifications

Specification	Description
Backplane Current Load	800 mA @ 5 V DC 3mA @ 24V DC
Operating Temperature	0 to 60°C (32 to 140°F)
Storage Temperature	-40 to 85°C (-40 to 185°F)
Shock	30g Operational
	50g non-operational
	Vibration: 5 g from 10 to 150 Hz
Relative Humidity	5% to 95% (non-condensing)
LED Indicators	Module Status
	Backplane Transfer Status
	Application Status
	Serial Activity
Application port (Ethernet)	
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
Debug/Configuration port (CFG)	
CFG Port (CFG)	RJ45 (DB-9M with supplied cable)
	RS-232 only
	No hardware handshaking

5.1.4 Functional Specifications

- Support for the storage and transfer of up to 5000 registers to/from the ControlLogix processor's controller tags
- User-definable module memory usage
- 10/100 Base-T Ethernet compatible interface
- Configurable parameters for the client include
  - Minimum Command Delay
  - Pass-Through Mode
- The ControlLogix processor can be programmed to control the activity on the client by actively selecting commands from the command list to execute or issue commands directly from the ladder logic
- The module supports 10 servers for Modbus TCP/IP (Port 502) and 10 servers for MNET (Port 2000)
- One hundred commands are supported on each port
- A client configured as a virtual Modbus master device on the MVI56-MNETR module will actively issue Modbus TCP/IP commands to other nodes on the Modbus TCP/IP network
- The servers permit remote clients to interact with all data contained in the module. This data can be derived from other Modbus clients on the network through the client on the module or from the ControlLogix processor
- The module can be configured to pass write commands (functions 5, 6, 15 and 16) directly from the remote host to the processor
- Accepts Modbus function code commands of 1, 2, 3, 4, 5, 6, 7, 15, 16, 22 and 23 from an attached Modbus TCP/IP client

5.2 Functional Overview

This section provides an overview of how the MVI56-MNETR module transfers data using the MNET protocol. You should understand the important concepts in this chapter before you begin installing and configuring the module.

5.2.1 General Concepts

The following discussion explains several concepts that are important for understanding the operation of the MVI56-MNETR module.

About the MODBUS TCP/IP Protocol

MODBUS is a widely-used protocol originally developed by Modicon in 1978. Since that time, the protocol has been adopted as a standard throughout the automation industry.

The original MODBUS specification uses a serial connection to communicate commands and data between client and server devices on a network. Later enhancements to the protocol allow communication over Ethernet networks using TCP/IP as a "wrapper" for the MODBUS protocol. This protocol is known as MODBUS TCP/IP.

MODBUS TCP/IP is a client/server protocol. The client establishes a connection to the remote server. When the connection is established, the client sends the MODBUS TCP/IP commands to the server. The MVI56-MNETR module works both as a client and as a server.

Aside from the benefits of Ethernet versus serial communications (including performance, distance and flexibility) for industrial networks, the MODBUS TCP/IP protocol allows for remote administration and control of devices over a TCP/IP network. The efficiency, scalability and low cost of a MODBUS TCP/IP network make this an ideal solution for industrial applications.

The MVI56-MNETR module acts as an input/output module between devices on a MODBUS TCP/IP network and the Rockwell Automation backplane. The module uses an internal database to pass data and commands between the processor and the client and server devices on the MODBUS TCP/IP network.

Module Power Up

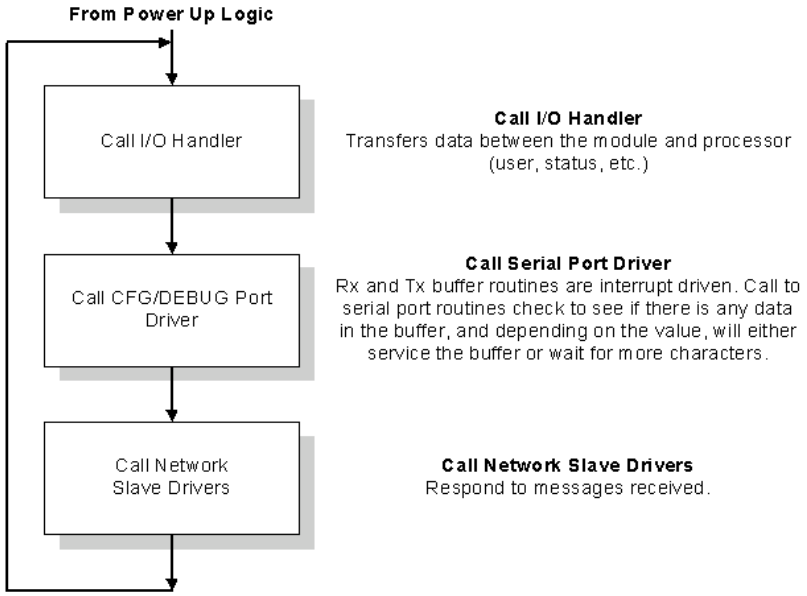
On power up the module begins performing the following logical functions:

1. Initialize hardware components
   - Initialize ControlLogix backplane driver
   - Test and Clear all RAM
2. Read configuration for module from MNET.CFG file on Compact Flash Disk
3. Initialize Module Register space
4. Enable Server Drivers
5. Enable Client Driver

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

Main Logic Loop

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



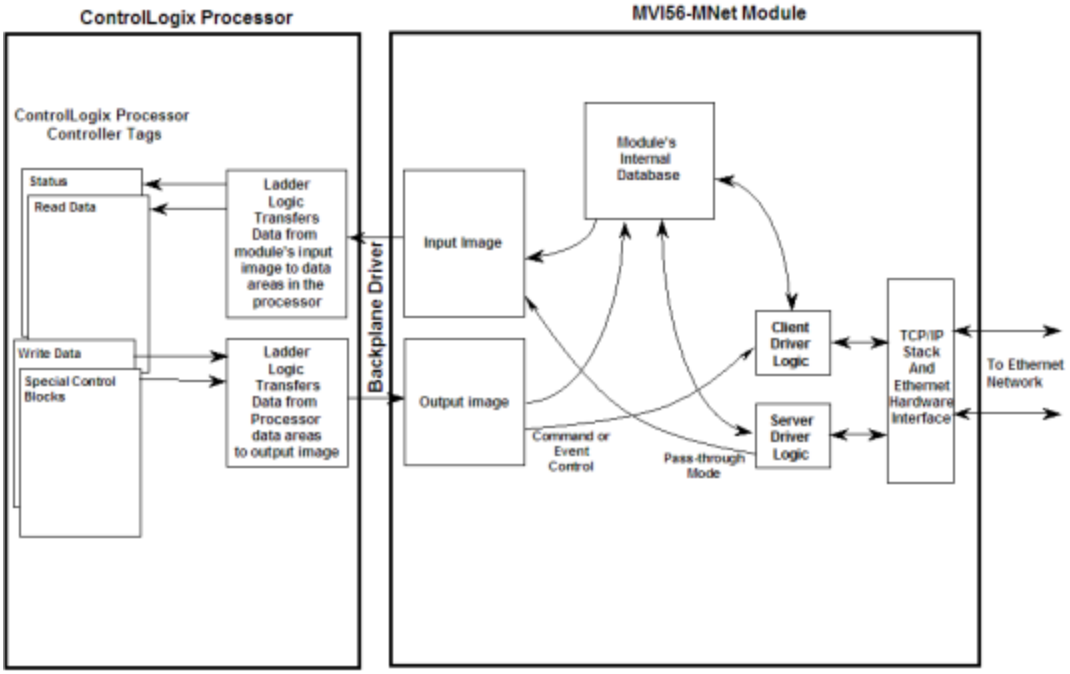
Backplane Data Transfer

The MVI56-MNETR 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 42 words. This data is transferred in the scheduled I/O timeslot.

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 42 words. This data is transferred in the scheduled I/O timeslot.

The following illustration shows the data transfer method used to move data between the ControlLogix processor, the MVI56-MNETR module and the Modbus TCP/IP 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 database. This database is defined as a virtual Modbus data table with addresses from 0 (40001 Modbus) to 4999 (45000 Modbus). The following illustration shows the layout of the database:

Module's Internal Database Structure

5000 registers for user data

	0
Register Data	
	4999

Data contained in this database is paged through the input and output images by coordination of the ControlLogix ladder logic and the MVI56-MNETR module's program. Up to 40 words of data can be transferred from the module to the processor at a time. Up to 40 words of data can be transferred from the processor to the module. Each image has a defined structure depending on the data content and the function of the data transfer. The module uses the following block numbers:

Block Range	Descriptions
-1	Status block
0	Status block
1 to 125	Read or write data
1000 to 1124	Output Initialization Blocks
2000	Event Command Block
5001 to 5006	Command Control
9956	Formatted pass-through block from function 6 or 16 with word data.
9957	Formatted pass-through block from function 6 or 16 with floating-point data.
9958	Formatted pass-through block from function 5.
9959	Formatted pass-through block from function 15.
9960	Formatted pass-through block from function 22.
9961	Formatted pass-through block from function 23.
9970	Function 99 indication block.
9996	Unformatted Pass-through block with raw Modbus message.
9998	Warm-boot control block
9999	Cold-boot control block

These block identification codes can be broken down into a few groups: Normal data transfer blocks (-1 to 125), Initialization blocks (1000 to 1124), Command control blocks (2000, 5001 to 5006, 9998 and 9999) and pass-through function blocks (9956 to 9961, 9970 and 9996).

Normal Data Transfer

Normal data transfer includes the paging of the user data found in the module's internal database in registers 0 to 4999 and the status data. These data are transferred through read (input image) and write (output image) blocks. The structure and function of each block is discussed below:

Status Read Data Block

This block is automatically copied from the module and contains status information about the module.

Offset	Description	Length
0	Write Block ID	1
1	Program Scan Counter	1
2 to 7	Block Transfer Status	6
8 to 9	Reserved Server Status	2
10 to 11	MNet Server Status	2

Offset	Description	Length
12 to 13	MBAP Server Status	2
14 to 23	MNet Client Status	10
24 to 40	Reserved	17
41	Read Block ID (-1 or 0)	1

MVI56-MNETR Client Status

Offset	Client Status
3	Total number of command list requests
4	Total number of command list responses
5	Total number of command list errors
6	Total number of requests of slave
7	Total number of responses
8	Total number of errors sent
9	Total number of errors received
10	Total number of Configuration Errors
11	Total number of Current Errors
12	Last Error

Read Block

These blocks of data transfer information from the module to the ControlLogix processor. The structure of the input image used to transfer this data is shown in the following table.

Offset	Description	Length
0	Write Block ID	1
1 to 40	Read Data	40
41	Read Block ID	1

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 40 words (block offsets 1 to 40) of data. In addition to moving user data, the block also contains status data for the module.

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 three read and two write blocks are used with the application, the sequence will be as follows:

R1W1→R2W2→R3W1→R1W2→R2W1→R3W2→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 Modbus 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 structure of the output image used to transfer this data is shown in the following table.

Offset	Description	Length
0	Write Block ID	1
1 to 40	Write Data	40
41	Spare	1

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 40 words (block offsets 1 to 40) of data.

Initialize Output Data

When the module performs a restart operation, it will request blocks of output data from the processor to initialize the module's output data (Read Data Area). Use the **Initialize Output Data** parameter in the configuration file to bring the module to a known state after a restart operation. The structure of the block used to request the data is displayed in the following table.

Offset	Description	Length
0	1000 to 1124	1
1 to 40	Spare	40
41	1000 to 1124	1

The block number in word 20 of the block determines the data set of up to 40 output words to transfer from the processor. Ladder logic in the processor must recognize these blocks and place the correct information in the output image to be returned to the module. The format of the returned write block is shown below:

Offset	Description	Length
0	1000 to 1124	1
1 to 40	Output Data to preset in module.	40
41	Spare	1

Command Control Blocks

Command Control Blocks are special optional blocks used to request special tasks from the module. The current version of the software supports four command control blocks; event command control, command control, warm boot, and cold boot.

Note: Event Commands and Command Control are not needed for normal Modbus command list polling operations, and are needed only occasionally for special circumstances.

Event Command

Event command control blocks send Modbus TCP/IP commands directly from the ladder logic to one of the clients on the module. The format for these blocks is displayed in the following table.

Offset	Description	Length
0	2000	1
1 to 4	IP Address	4
5	Service Port	1
6	Slave Address	1
7	Internal DB Address	1
8	Point Count	1
9	Swap Code	1
10	Modbus Function Code	1
11	Device Database Address	1
12 to 41	Spare	30

The parameters passed with the block construct the command. The **IP Address** for the node to reach on the network is entered in four registers (1 to 4). Each digit of the IP address is entered in the appropriate register.

For example, to interface with node 192.168.0.100, enter the values 192, 168, 0 and 100 in registers 1 to 4. The **Service Port** field selects the TCP service port on the server to connect. If the parameter is set to 502, a standard MBAP message will be generated. All other service port values will generate a Modbus command message encapsulated in a TCP/IP packet.

The **Internal DB Address** parameter specifies the module's database location to associate with the command. The **Point Count** parameter defines the number of points or registers for the command. The **Swap Code** is used with Modbus functions 3 and 4 requests to change the word or byte order. The **Modbus Function Code** has one of the following values 1, 2, 3, 4, 5, 6, 15 or 16. The **Device Database Address** is the Modbus register or point in the remote slave device to be associated with the command.

When the module receives the block, it will process it and place it in the command queue. A detailed description of the block is presented in the following diagram:

Word	Description
0	This word contains the block 2000 identification code to indicate that this block contains a command to execute by the Client Driver.
1 to 4	These words contain the IP address for the server the message is intended. Each digit (0 to 255) of the IP address is placed in one of the four registers. For example, to reach IP address 192.168.0.100, enter the following values in words 1 to 4 → 192, 168, 0 and 100. The module will construct the normal dotted IP address from the values entered. The values entered will be anded with the mask 0x00ff to insure the values are in the range of 0 to 255.
5	This word contains the TCP service port the message will be interfaced. For example, to interface with a MBAP device, the word should contain a value of 502. To interface with a MNET device, a value of 2000 should be utilized. Any value from 0 to 65535 is permitted. A value of 502 will cause a MBAP formatted message to be generated. All other values will generate an encapsulated Modbus message.
6	This word contains the Modbus node address to use with the message. This field should have a value from 0 to 41.

Word	Description
7	This word contains the internal Modbus address in the module to be used with the command. This word can contain a value from 0 to 4999.
8	This word contains the count parameter that determines the number of digital points or registers to associate with the command.
9	The parameter specifies the swap type for the data. This function is only valid for function codes 3 and 4.
10	This word contains the Modbus function code to be used with the command.
11	This word contains the Modbus address in the slave device to be associated with the command.
12 to 41	Spare

The module will respond to each event command block with a read block with the following format:

Offset	Description	Length
0	Write Block ID	1
1	0=Fail, 1=Success	1
2 to 40	Spare	39
41	2000	1

Word two of the block can be used by the ladder logic to determine if the command was added to the command queue of the module. The command will only fail if the command queue for the port is full (100 commands for each queue).

Command Control

Command control blocks place commands in the command list into the command queue. The client has a command queue of up to 100 commands. The module services commands in the queue before the user defined command list. This gives high priority to commands in the queue. Commands placed in the queue through this mechanism must be defined in the module's command list. Under normal command list execution, the module will only execute commands with the Enable parameter set to one or two. If the value is set to zero, the command is skipped. Commands may be placed in the command queue with an Enable parameter set to zero using this feature. These commands can then be executed using the command control blocks.

One to six commands can be placed in the command queue with a single request. The following table describes the format for this block.

Offset	Description	Length
0	5001 to 5006	1
1	Command index	1
2	Command index	1
3	Command index	1
4	Command index	1
5	Command index	1
6	Command index	1
7 to 41	Spare	35

The last digit in the block code defines the number of commands to process in the block. For example, a block code of 5003 contains 3 command indexes that are to be placed in the command queue. The Command index parameters in the block have a range of 0 to 99 and correspond to the module's command list entries.

The module responds to a command control block with a block containing the number of commands added to the command queue for the port. The following table describes the format for this block.

Offset	Description	Length
0	Write Block ID	1
1	Number of commands added to command queue	1
2 to 40	Spare	39
41	5001 to 5006	1

Warm Boot

This block is sent from the ControlLogix processor to the module (output image) when the module is required to perform a warm-boot (software reset) operation. The structure of the control block is shown below:

Offset	Description	Length
0	9998	1
1 to 41	Spare	41

Cold Boot

This block is sent from the ControlLogix processor to the module (output image) when the module is required to perform the cold boot (hardware reset) operation. This block is sent to the module when a hardware problem is detected by the ladder logic that requires a hardware reset. The structure of the control block is shown in the following table.

Offset	Description	Length
0	9999	1
1 to 41	Spare	41

Pass-Through Control Blocks

If the module is set for pass-through operation by placing a value of 1 to 3 in the configuration file parameter **Pass-Through** Mode, the module will send special blocks to the module when a write request is received from a client. Any Modbus function 5, 6, 15 or 16 commands will be passed from the server to the processor using this block identification numbers 9956 to 9961, 9970 and 9996. Ladder logic must handle the receipt of these blocks and to place the enclosed data into the proper controller tags in the module.

There are two basic modes of operation when the pass-through feature is utilized: Unformatted (code 1) and Formatted (code 2 or 3). The unformatted mode will pass the message received on the server directly to the processor without any processing. The format of the read block transferred in this mode is shown in the following table.

Unformatted

Unformatted Pass-Through Command (Read Block)

Offset	Description	Length
0	9996	1
1	Number of bytes in Modbus msg	1
2	Reserved (always 0)	1
3 to 40	Modbus message received	38
41	9996	1

The ladder logic should copy and parse the received message and control the processor as expected by the master device. The processor must respond to the pass-through control block with a write block with the following format.

Unformatted Pass-Through Command (Write Block)

Offset	Description	Length
0	9996	1
1 to 41	Spare	41

This informs the module that the command has been processed and can be cleared from the pass-through queue.

In formatted pass-through mode, the module processes the received write request and generates a special block dependent on the function received. There are two modes of operation when the formatted pass-through mode is selected. If code 2 is utilized (no swap), the data received in the message is presented in the order received by the module. If code 3 is utilized (swap mode), the bytes in the data area of the message will be swapped. This selection is applied to all received write requests. The block identification code used with the request is dependent on the Modbus function requested. Block 9956 passes word type data for functions 6 and 16. Block 9957 passes a floating-point message for functions 6 and 16. Block 9958 is utilized when Modbus function 5 data is received. Block 9959 is employed when function 15 is recognized. Block 9960 is used for function 22 and Block 9961 is used for function 23 requests. Block 9970 is utilized for function 99. The format for the read blocks is shown in the following table.

Formatted

Formatted Pass-Through Command Blocks (Read Block)

Offset	Description	Length
0	9956, 9957, 9958, 9960 or 9961	1
1	Number of word registers in Modbus data set	1
2	Starting address for Modbus data set	1
3 to 40	Modbus data set	38
41	9956, 9957, 9958, 9960 or 9961	1

Formatted Pass-Through Command Blocks (Read Block)

Offset	Description	Length
0	9959	1
1	Number of word registers in Modbus data set	1
2	Starting word address for Modbus data set	1
3 to 21	Modbus data set	19
22 to 40	Bit mask to use with the data set. Each bit to be considered with the data set will have a value of 1 in the mask. Bits to ignore in the data set will have a value of 0 in the mask.	19
41	9959	1

Formatted Pass-Through Command Blocks (Read Block)

Offset	Description	Length
0	9970	1
1	1	1
2	0	1
3 to 40	Spare data area	38
41	9996	1

The ladder logic should copy and parse the received message and control the processor as expected by the master device. The processor must respond to the formatted pass-through control blocks with a write block with one of the following formats:

Formatted Pass-Through Response (Write Block)

Offset	Description	Length
0	9956, 9957, 9958, 9960 or 9961	1
1 to 41	Spare data area	41

Formatted Pass-Through Response (Write Block)

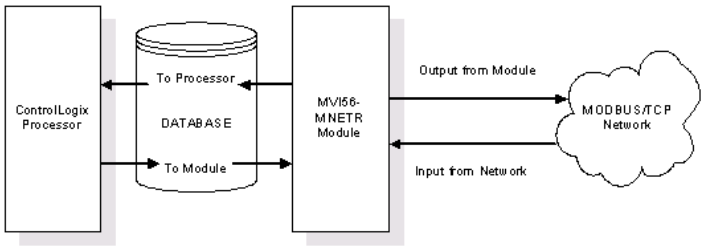
Offset	Description	Length
0	9959	1
1 to 41	Spare data area	41

Formatted Pass-Through Response (Write Block)

Offset	Description	Length
0	9970	1
1 to 41	Spare data area	41

5.2.2 Data Flow between MVI56-MNETR Module and ControlLogix Processor

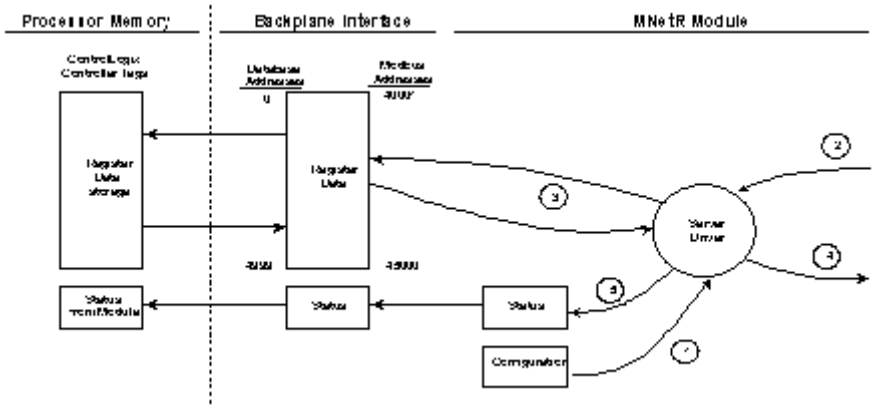
The following topics describe the flow of data between the two pieces of hardware (ControlLogix processor and MVI56-MNETR module) and other nodes on the Modbus TCP/IP network under the module's different operating modes. The module contains both servers and a client. The servers accept TCP/IP connections on service ports 502 (MBAP) (10 servers) and 2000 (MNET) (10 servers). The client can generate either MBAP or MNET requests dependent on the service port selected in the command.



The following topics discuss the operation of the server and client drivers.

Server Driver

The Server Driver allows the MVI56-MNETR module to respond to data read and write commands issued by clients on the Modbus TCP/IP network. The following illustration and associated table describe the flow of data into and out of the module.

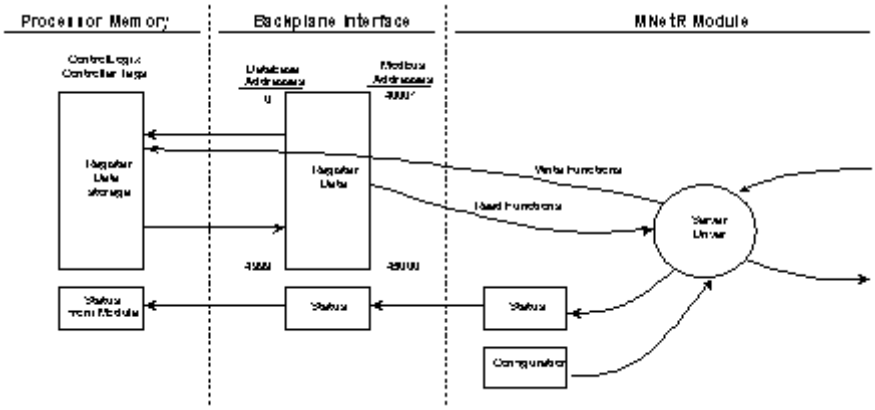


Step	*Description*
1	The server driver receives the configuration information from the configuration file on the Compact Flash Disk, and the module initializes the servers.
2	A Host device, such as a Modicon PLC or an HMI application issues a read or write command to the module's node address. The server driver qualifies the message before accepting it into the module.

Step	*Description*
3	When the module accepts the command, the data is immediately transferred to or from the 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. If the pass-through feature is utilized, the write message is transferred directly to the processor and is not written to the module's database.
4	When 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 Server Driver.

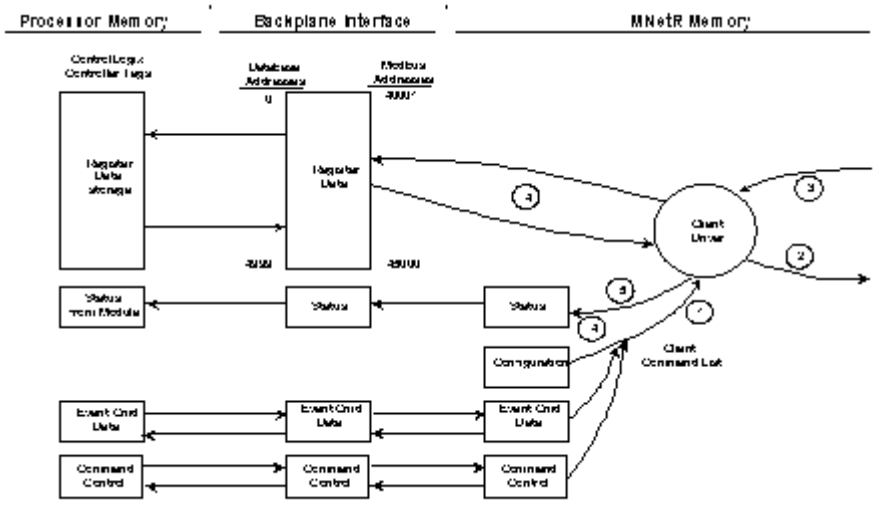
After the server socket is open, it must receive messages within a one minute period, or else it will close the socket. After closing, the socket will be reused.

An exception to this normal mode is when the pass-through mode is implemented. In this mode, all write requests will be passed directly to the processor and will not be placed in the database. This permits direct, remote control of the processor without the intermediate database. This mode is especially useful for Master devices that do not send both states of control. For example, a SCADA system may only send an on command to a digital control point and never send the clear state. The SCADA system expects the local logic to reset the control bit. Pass-through must be used to simulate this mode. The following illustration describes the data flow for a slave port with pass-through enabled:



Client Driver

In the client driver, the MVI56-MNETR module issues read or write commands to servers on the Modbus TCP/IP network. These commands are user configured in the module via the Client Command List received from the module's configuration file (MNET.CFG) or issued directly from the ControlLogix processor (event command control). Command status is returned to the processor for each individual command in the command list status block. The location of this status block in the module's internal database is user defined. The following flow chart and associated table describe the flow of data into and out of the module.



Step	Description
1	The client driver obtains configuration data from the MNET.CFG file when the module restarts. The configuration data obtained includes the timeout parameters and the Command List. These values are used by the driver to determine the type of commands to be issued to the other nodes on the Modbus TCP/IP network.
2	When configured, the client driver begins transmitting read and/or write commands to the other nodes on the network. If writing data to another node, the data for the write command is obtained from the module's internal database to build the command.
3	Presuming successful processing by the node specified in the command, a response message is received into the client driver for processing.
4	Data received from the node on the network is passed into the module's internal database, assuming a read command.
5	Status data is returned to the processor for the client and a Command List error table can be established in the module's internal database.

Client Command List

In order for the client to function, the module's Client Command List must be defined. This list contains up to 100 individual entries, with each entry containing the information required to construct a valid command. This includes the following:

- Command enable mode ((0) disabled, (1) continuous or (2) conditional)
- IP address and service port to connect to on the remote server

- Slave Node Address
- Command Type - Read or Write up to 100 words per command
- Database Source and Destination Register Address - Determines where data will be placed and/or obtained
- Count - Select the number of words to be transferred - 1 to 100
- Poll Delay - 1/10th seconds

Client Command Errors

You can use the MNET 0 Client Command Error Pointer in the MNET.CFG file to set the M1 offset register where all command error codes will be stored. This means that the first register refers to command 1 and so on.

Offset	Description
1	Command 1 Error
2	Command 2 Error
3	Command 3 Error
…	….
…	…

For every command that has an error, the module automatically sets the poll delay parameter to 30 seconds. This instructs the module to wait 30 seconds until it attempts to issue the command again.

As the list is read in from the configuration file and as the commands are processed, an error value is maintained in the module for each command. This error list can be transferred to the processor. The errors generated by the module are displayed in the following table.

Standard Modbus Protocol Errors

Code	Description
1	Illegal Function
2	Illegal Data Address
3	Illegal Data Value
4	Failure in Associated Device
5	Acknowledge
6	Busy, Rejected Message

Module Communication Error Codes

Code	Description
-1	CTS modem control line not set before transmit
-2	Timeout while transmitting message
-11	Timeout waiting for response after request
253	Incorrect slave address in response
254	Incorrect function code in response
255	Invalid CRC/LRC value in response

MNET Client Specific Errors

Code	Description
-33	Failed to connect to server specified in command
-36	MNET command response timeout
-37	TCP/IP connection ended before session finished

Command List Entry Errors

Code	Description
-40	Too few parameters
-41	Invalid enable code
-42	Internal address > maximum address
-43	Invalid node address (<0 or >255)
-44	Count parameter set to 0
-45	Invalid function code
-46	Invalid swap code
-47	ARP could not resolve MAC from IP (bad IP address, not part of a network, invalid parameter to ARP routine).
-48	Error during ARP operation: the response to the ARP request did not arrive to the module after a 5 second timeout.

Note: When the client gets error -47 or -48, it places the command offline for 30 seconds so it will not keep waiting for servers that are not present on the network, blocking the commands being executed to valid/live servers on the network.

5.3 Cable Connections

The MVI56-MNETR module has the following communication connections on the module:

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

5.3.1 Ethernet Connection

The MVI56-MNETR 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-MNETR module is NOT compatible with Power Over Ethernet (IEEE802.3af / IEEE802.3at) networks. Do NOT connect the module to Ethernet devices, hubs, switches or networks that supply AC or DC power over the Ethernet cable. Failure to observe this precaution may result in damage to hardware, or injury to personnel.
Important: The module requires a static (fixed) IP address that is not shared with any other device on the Ethernet network. Obtain a list of suitable IP addresses from your network administrator BEFORE configuring the Ethernet port on this module.

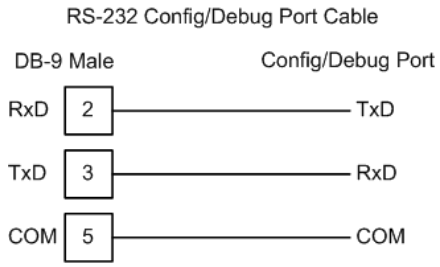
Ethernet Port Configuration - wattcp.cfg

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

```
# WATTCP.CFG FILE:
# ProSoft Technology.
  my_ip=192.168.0.100
# Default class 3 network mask
  netmask=255.255.255.0
# The gateway I wish to use
  gateway=192.168.0.1
```

5.3.2 RS-232 Configuration/Debug Port

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

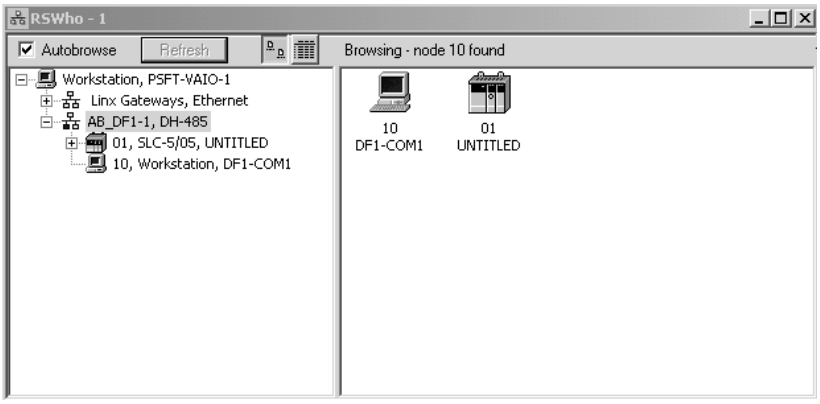


Disabling the RSLinx Driver for the Com Port on the PC

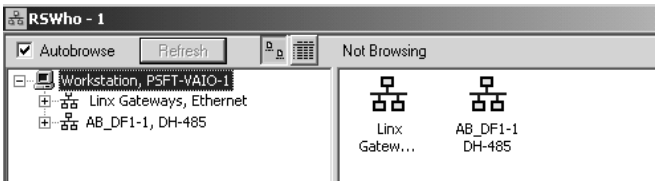
The communication port driver in RSLinx can occasionally prevent other applications from using the PC's COM port. If you are not able to connect to the module's configuration/debug port using ProSoft Configuration Builder (PCB), HyperTerminal or another terminal emulator, follow these steps to disable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

1 Open RSLinx and go to Communications>RSWho

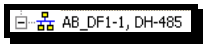
2 Make sure that you are not actively browsing using the driver that you wish to stop. The following shows an actively browsed network:



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:

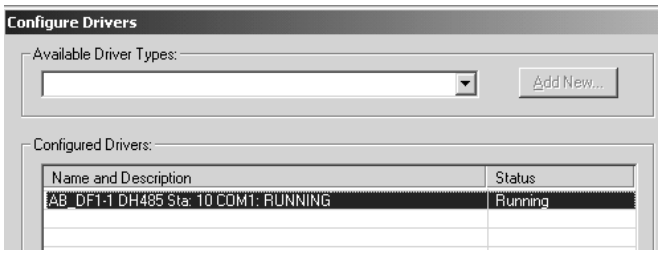


Branches are displayed or hidden by clicking on the ⊞ or the ⊟ icons.

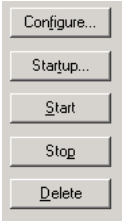


4 When you have verified that the driver is not being browsed, go to

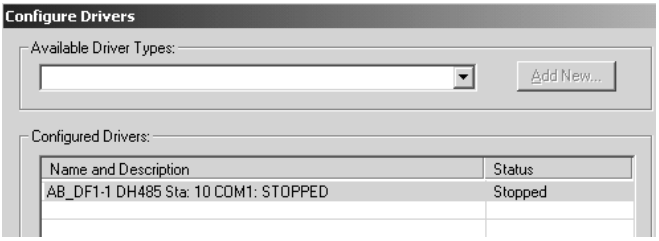
Communications>Configure Drivers
You may see something like this:



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" on the side of the window:



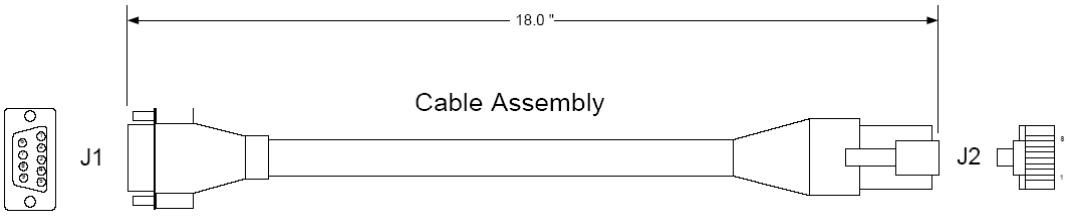
5 After you have stopped the driver you will see the following:

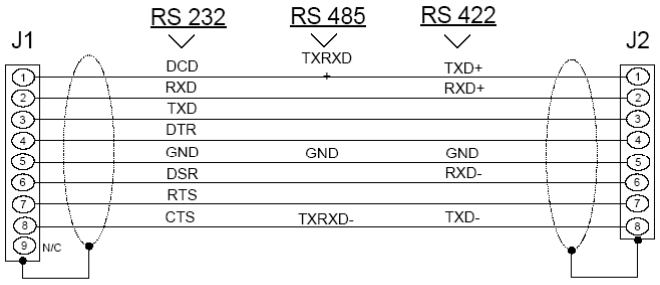


6 Upon seeing this, you may now use that com port to connect to the debug port of the module.

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.

5.3.3 DB9 to RJ45 Adaptor (Cable 14)





Wiring Diagram

5.4 MVI56-MNETR Status Data Definition

This section contains a description of the members present in the MNETInStat[0,-1] object. This data is transferred from the module to the processor as part of each read block.

Offset	Content	Description
0	Program Scan Count	This value is incremented each time a complete program cycle occurs in the module.
1	Read Block Count	This field contains the total number of read blocks transferred from the module to the processor.
2	Write Block Count	This field contains the total number of write blocks transferred from the processor to the module.
3	Parse Block Count	This field contains the total number of blocks successfully parsed that were received from the processor.
4	Command Event Block Count	This field contains the total number of command event blocks received from the processor.
5	Command Block Count	This field contains the total number of command blocks received from the processor.
6	Error Block Count	This field contains the total number of block errors recognized by the module.
7	Reserved	Not Used
8	Reserved	Not Used
9	Reserved	Not Used
10	Reserved	Not Used
11	Reserved	Not Used
12	Reserved	Not Used
13	Reserved	Not Used
14	Reserved	Not Used
15	Reserved	Not Used
16	Reserved	Not Used
17	Reserved	Not Used
18	Reserved	Not Used
19	Reserved	Not Used
20	MNet Request Count	This counter increments each time a MNet (port 2000) request is received.
21	MNet Response Count	This counter is incremented each time a MNet (port 2000) response message is sent.
22	Reserved	Not Used
23	Reserved	Not Used
24	Reserved	Not Used
25	Reserved	Not Used
26	Reserved	Not Used
27	Reserved	Not Used
28	Reserved	Not Used
29	Reserved	Not Used

Offset	Content	Description
30	MBAP Request Count	This counter increments each time a MBAP (port 502) request is received.
31	MBAP Response Count	This counter is incremented each time a MBAP (port 502) response message is sent.
32	Reserved	Not Used
33	Reserved	Not Used
34	Reserved	Not Used
36	Reserved	Not Used
36	Reserved	Not Used
239	Client Cmd Request	This value is incremented each time a command request is issued.
240	Client Cmd Response	This value is incremented each time a command response is received.
241	Client Cmd Error	This value is incremented each time an error message is received from a remote unit or a local error is generated for a command.
242	Client Request Count	This value is incremented each time a request message is issued.
243	Client Response Count	This value is incremented each time a response message is received.
244	Client Error Sent Count	This value is incremented each time an error is sent from the client.
245	Client Error Received Count	This value is incremented each time an error is received from a remote unit.
246	Client Cfg Error Word	This word contains a bit map that defines configuration errors in the configuration file for the client.
247	Client Current Error Code	This value corresponds to the current error code for the client.
248	Client Last Error Code	This value corresponds to the last error code recorded for the client.

5.5 Installing and Configuring the Module

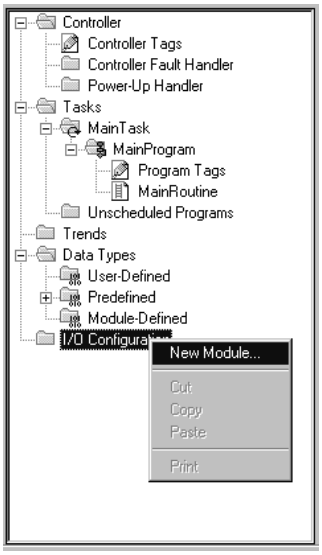
Important: The following steps describe how to install and configure the MVI56-MNETR module with RSLogix 5000 version 15 or older. If you are using RSLogix 5000 version 16, please refer to Sample Add-On Instruction Import Procedure (page 21).

The configuration process consists of the following steps.

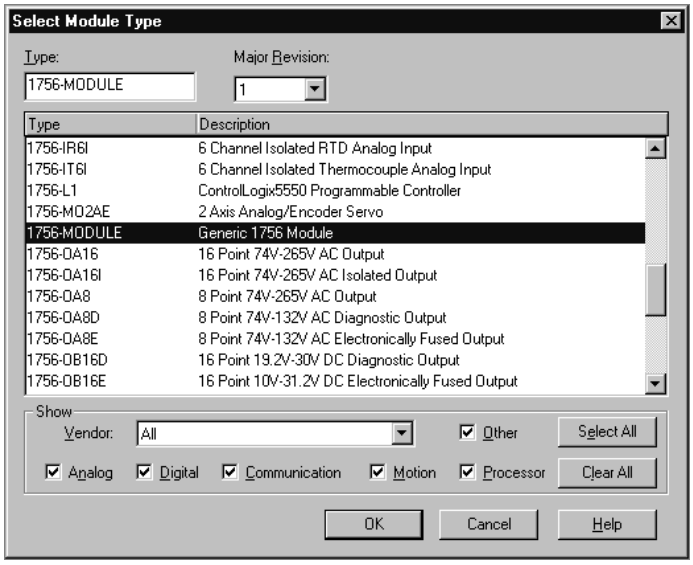
1 Download the sample program to the processor.

Note: For most applications, the sample program will work without modification. We strongly recommend setting up the module first with the sample program, before attempting to add the module to an existing application or create a custom application.
Modify the module's configuration files to meet the needs of your application, and copy the updated configuration to the module. Example configuration files are provided on the CD-ROM. Refer to Modifying the Configuration File (page 39) for more information on the configuration files.

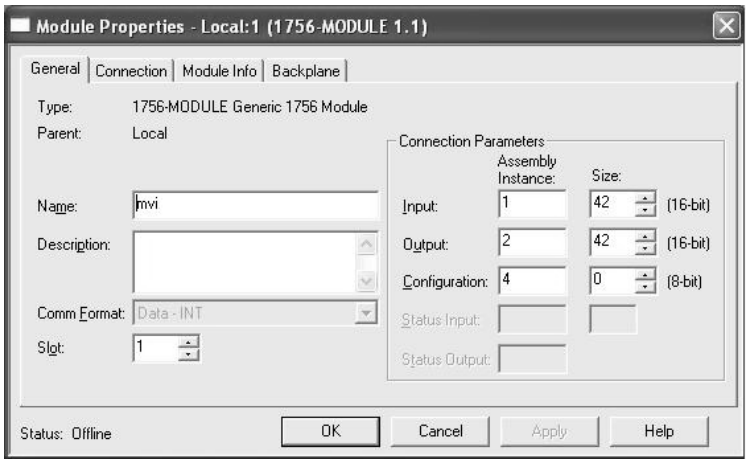
2 First, define the module to the system. In RSLogix 5000, click the right mouse button on the I/O Configuration option in the Controller Organization list to open a shortcut menu. Select the New Module… option from the I/O Configuration menu.



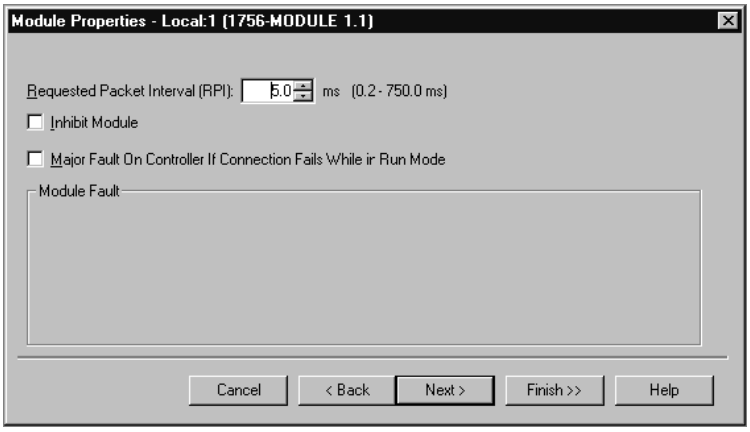
This action opens the following dialog box.



3 Select the 1756-Module (Generic 1756 Module) from the list and click OK. This action opens the Module Properties dialog box.

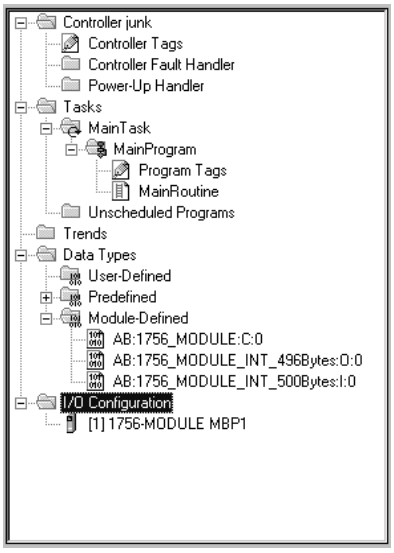


4 Fill in the fields as shown, with the Name, Description and Slot options for your application. You must select the **Comm Format** as **Data - INT** in the dialog box. Failure to set the **Assembly Instance** and **Size** values correctly will result in a module that will not communicate over the backplane of the ControlLogix rack. Click **Next** to open the next dialog box.

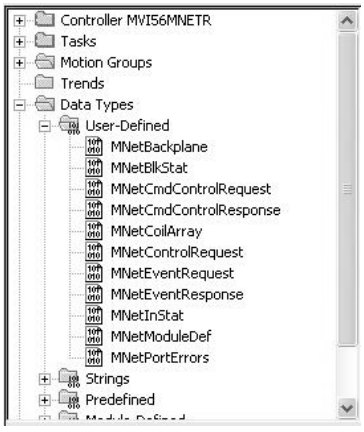


5 Select the Request Packet Interval value for scanning the I/O on the module. This value represents the minimum frequency the module will handle scheduled events. This value should not be set to less than 1 millisecond. Values between 1 and 10 milliseconds should work with most applications.

After completing the module setup, the module will appear in the Controller Organization list. The data required for the module will be defined to the application, and objects will be allocated in the Controller Tags data area. An example of the Controller Organization list is shown below.

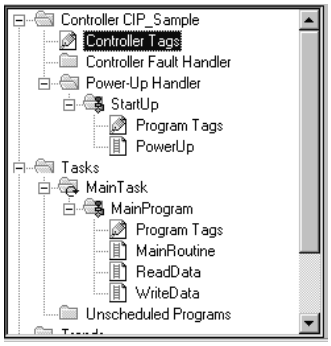


6 Next, define the User Defined Data Types to use with the module. You can copy these data types from the example ladder logic. The Controller Organization list should display the User Defined Data Types shown below:



7 Next, define the data tag to be used to interface with the module and the ladder logic. Open the Controller Tags Edit Tags dialog box and enter the values required for the application. The MVI56-MNETR module is defined in the example as MNET1. If you are using the example ladder logic, the tag name has already been performed. Otherwise, you can use any valid tag name.

8 The last step is to add the ladder logic. If you are using the example ladder logic, modify the ladder to fit your application. If you are not using the ladder example, copy the ladder logic shown in the Controller Organization list below to your application.



It is now time to edit the MNET.CFG file to set up the module for the specific application. Refer to the **Configuration File** section of this document.

The module is now set up and ready to use with your application. Insert the module in the rack and attach the serial communication cable and the network cable. Download the new application to the controller and place the processor in run mode. Download the new MNET.CFG and WATTCP.CFG files to the module using HyperTerminal program or any other terminal emulation program. If all the configuration parameters are set correctly and the module is attached to a network, the module's Application LED (APP LED) should remain off and the backplane activity LED (BP ACT) should blink very rapidly. If you encounter errors, refer to **Diagnostics and Troubleshooting** (page 55) for information on how to connect to the module's Config/Debug port to use its troubleshooting features.

6 Support, Service & Warranty

In This Chapter

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

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

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

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

6.1 How to Contact Us: Technical Support

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

Asia Pacific

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

Languages spoken include: Chinese, English

Europe (location in Toulouse, France)

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

Languages spoken include: French, English

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

+1.661.716.5100, support@prosoft-technology.com (mailto:support@prosoft-technology.com)

Languages spoken include: English, Spanish

For technical support calls within the United States, an after-hours *answering system allows pager access to one of our qualified technical and/or application support engineers at any time to answer your questions.*

Brasil (location in Sao Paulo)

+55-11-5084-5178 , eduardo@prosoft-technology.com (mailto:eduardo@prosoft-technology.com)

Languages spoken include: Portuguese, English

6.2 Return Material Authorization (RMA) Policies and Conditions

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

6.2.1 All Product Returns:

a) In order to return a Product for repair, exchange or otherwise, the Customer must obtain a Returned Material Authorization (RMA) number from ProSoft and comply with ProSoft shipping instructions.
b) In the event that the Customer experiences a problem with the Product for any reason, Customer should contact ProSoft Technical Support at one of the telephone numbers listed above (page 97). A Technical Support Engineer will request that you perform several tests in an attempt to isolate the problem. If after completing these tests, the Product is found to be the source of the problem, we will issue an RMA.
c) All returned Products must be shipped freight prepaid, in the original shipping container or equivalent, to the location specified by ProSoft, and be accompanied by proof of purchase and receipt date. The RMA number is to be prominently marked on the outside of the shipping box. Customer agrees to insure the Product or assume the risk of loss or damage in transit. Products shipped to ProSoft using a shipment method other than that specified by ProSoft or shipped without an RMA number will be returned to the Customer, freight collect. Contact ProSoft Technical Support for further information.
d) A 10% restocking fee applies to all warranty credit returns whereby a Customer has an application change, ordered too many, does not need, etc.

6.2.2 Procedures for Return of Units Under Warranty:

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

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

6.2.3 Procedures for Return of Units Out of Warranty:

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

The following is a list of non-repairable units:

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

6.2.4 Purchasing Warranty Extension:

a) ProSoft's standard warranty period is three (3) years from the date of shipment as detailed in "Limited Warranty (page 100)". The Warranty Period may be extended at the time of equipment purchase for an additional charge, as follows:

- Additional 1 year = 10% of list price
- Additional 2 years = 20% of list price
- Additional 3 years = 30% of list price

6.3 LIMITED WARRANTY

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

6.3.1 What Is Covered By This Warranty

a) *Warranty On New Products*: ProSoft warrants, to the original purchaser, that the Product that is the subject of the sale will (1) conform to and perform in accordance with published specifications prepared, approved and issued by ProSoft, and (2) will be free from defects in material or workmanship; provided these warranties only cover Product that is sold as new. This Warranty expires three years from the date of shipment (the "Warranty Period"). If the Customer discovers within the Warranty Period a failure of the Product to conform to specifications, or a defect in material or workmanship of the Product, the Customer must promptly notify ProSoft by fax, email or telephone. In no event may that notification be received by ProSoft later than 39 months. Within a reasonable time after notification, ProSoft will correct any failure of the Product to conform to specifications or any defect in material or workmanship of the Product, with either new or used replacement parts. Such repair, including both parts and labor, will be performed at ProSoft's expense. All warranty service will be performed at service centers designated by ProSoft.
b) *Warranty On Services*: Materials and labor performed by ProSoft to repair a verified malfunction or defect are warranteed in the terms specified above for new Product, provided said warranty will be for the period remaining on the original new equipment warranty or, if the original warranty is no longer in effect, for a period of 90 days from the date of repair.

6.3.2 What Is Not Covered By This Warranty

a) ProSoft makes no representation or warranty, expressed or implied, that the operation of software purchased from ProSoft will be uninterrupted or error free or that the functions contained in the software will meet or satisfy the purchaser's intended use or requirements; the Customer assumes complete responsibility for decisions made or actions taken based on information obtained using ProSoft software.

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

6.3.3 Disclaimer Regarding High Risk Activities

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

6.3.4 Intellectual Property Indemnity

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

a) Any documentation included with Product purchased from ProSoft is protected by copyright and may not be duplicated or reproduced in any form without prior written consent from ProSoft.
b) ProSoft's technical specifications and documentation that are included with the Product are subject to editing and modification without notice.
c) Transfer of title shall not operate to convey to Customer any right to make, or have made, any Product supplied by ProSoft.
d) Customer is granted no right or license to use any software or other intellectual property in any manner or for any purpose not expressly permitted by any license agreement accompanying such software or other intellectual property.
e) Customer agrees that it shall not, and shall not authorize others to, copy software provided by ProSoft (except as expressly permitted in any license agreement accompanying such software); transfer software to a third party separately from the Product; modify, alter, translate, decode, decompile, disassemble, reverse-engineer or otherwise attempt to derive the source code of the software or create derivative works based on the software; export the software or underlying technology in contravention of applicable US and international export laws and regulations; or use the software other than as authorized in connection with use of Product.
f) **Additional Restrictions Relating To Software And Other Intellectual Property**

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

6.3.5 Disclaimer of all Other Warranties

The Warranty set forth in What Is Covered By This Warranty (page 100) are in lieu of all other warranties, express or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose.

*6.3.6 Limitation of Remedies ***

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

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

6.3.7 Time Limit for Bringing Suit

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

6.3.8 No Other Warranties

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

6.3.9 Allocation of Risks

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

6.3.10 Controlling Law and Severability

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

Index