ProSoft TECHNOLOGY

Where Automation Connects.

MVI46-MNETC

SLC 500 Platform

Modbus TCP/IP Communication Client Module

April 28, 2017

USER MANUAL

Your Feedback Please

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

How to Contact Us

ProSoft Technology, Inc.
9201 Camino Media, Suite 200
Bakersfield, CA 93311
+1 (661) 716-5100
+1 (661) 716-5101 (Fax)
www.prosoft-technology.com
support@prosoft-technology.com

MVI46-MNETC User Manual
Rev 1.3.0

April 28, 2017

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 at our website: www.prosoft-technology.com

Important Installation Instructions

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

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

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

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

THIS DEVICE SHALL BE POWERED BY CLASS 2 OUTPUTS ONLY.

MVI (Multi Vendor Interface) Modules

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

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

Warnings

North America Warnings

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

ATEX Warnings and Conditions of Safe Usage:

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

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

Warning: This module is not hot-swappable! Always remove power from the rack before inserting or removing this module, or damage may result to the module, the processor, or other connected devices.

Battery Life Advisory

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

Note: The battery is not user replaceable.

Markings

Electrical Ratings

- Backplane Current Load: 800 mA @ 5 Vdc
- Operating Temperature: 0°C to 60°C (32°F to 140°F)
- Storage Temperature: -40°C to 85°C (-40°F to 185°F)
- Shock: 30 g operational, 50 g non-operational; Vibration: 5 g from 10 Hz to 150 Hz
- Relative Humidity 5% to 95% (with no condensation)
- All phase conductor sizes must be at least 1.3 mm(squared) and all earth ground conductors must be at least 4mm(squared).

Label Markings

Agency Approvals and Certifications

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

243333

Contents

1 Start Here

In This Chapter

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

- **Rockwell Automation® RSLogix™ software:** launch the program, configure ladder logic, and transfer the ladder logic to the processor
- **Microsoft Windows®:** install and launch programs, execute menu commands, navigate dialog boxes, and enter data
- **Hardware installation and wiring:** install the module, and safely connect MNETC and SLC devices to a power source and to the MVI46-MNETC's application port(s)

1.1 System Requirements

The MVI46-MNETC module requires the following minimum hardware and software components:

- Rockwell Automation SLC 5/02 M0/M1 capable processors (or newer), with compatible power supply and one free slot in the rack, for the MVI46-MNETC module. The module requires 800mA of available power.
- Rockwell Automation RSLogix 500 programming software.
- Rockwell Automation RSLinx communication software
- Pentium® II 500 MHz minimum. Pentium III 733 MHz (or better) recommended
- Supported operating systems:
  - Microsoft® Windows 98
  - Windows NT® (version 4 with SP4 or higher)
  - Windows 2000
  - Windows XP

- 32 Mbytes of RAM minimum, 64 Mbytes of RAM recommended
- 50 Mbytes of free hard disk space (or more based on application requirements)
- 16-color VGA graphics adapter, 640 x 480 minimum resolution (256 Color 800 × 600 recommended)

1.2 Package Contents

The following components are included with your MVI46-MNETC 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	MVI46-MNETC Module	MVI46-MNETC	Modbus TCP/IP Communication Client Module
1	Cable	Cable #15 - RS232 Null Modem	For RS232 between a Personal Computer (PC) and the CFG port of the module
1	Cable	Cable #14 - RJ45 to DB9 Male Adapter	For connecting the module's port to Cable #15 for RS-232 connections

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

1.3 Installing ProSoft Configuration Builder Software

You must install the *ProSoft Configuration Builder* (PCB) software to configure the module. You can always get the newest version of ProSoft Configuration Builder from the ProSoft Technology website (www.prosoft-technology.com). The filename contains the version of PCB. For example, **PCB_4.4.3.4.0245.exe**.

To install ProSoft Configuration Builder from the ProSoft Technology website

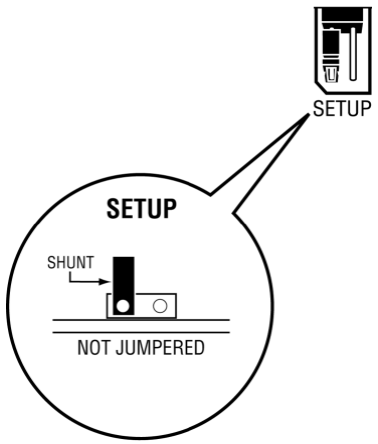
1. Open your web browser and navigate to www.prosoft-technology.com/PCB
2. Click the link at the *Current Release Version* section to download the latest version of *ProSoft Configuration Builder*.
3. Choose **SAVE** or **SAVE FILE** when prompted.
4. Save the file to your *Windows Desktop*, so that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

Note: To use the RSLogix 500 under the Windows 7 OS, you must be sure to install it using the *Run as Administrator* option. To find this option, right-click the Setup.exe program icon, and then click **RUN AS ADMINISTRATOR** on the context menu. You must install using this option even if you are already logged in as an Administrator on your network or personal computer (PC). Using the Run as Administrator option allows the installation program to create folders and files on your PC with proper permissions and security. If you do not use the Run as Administrator option, the RSLogix 500 may appear to install correctly, but you will receive multiple file access errors whenever the RSLogix 500 is running, especially when changing configuration screens. If this happens, you must completely uninstall the RSLogix 500 and then re-install using the Run as Administrator option to eliminate the errors.

1.4 Setting Jumpers

The Setup Jumper acts as "write protection" for the module's firmware. In "write protected" mode, the Setup pins are not connected, and the module's firmware cannot be overwritten. The module is shipped with the Setup jumper OFF. Do not jumper the Setup pins together unless you are directed to do so by ProSoft Technical Support (or you want to update the module firmware).

The following illustration shows the MVI46-MNETC jumper configuration with the Setup Jumper OFF.



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

1.5 Installing the Module in the Rack

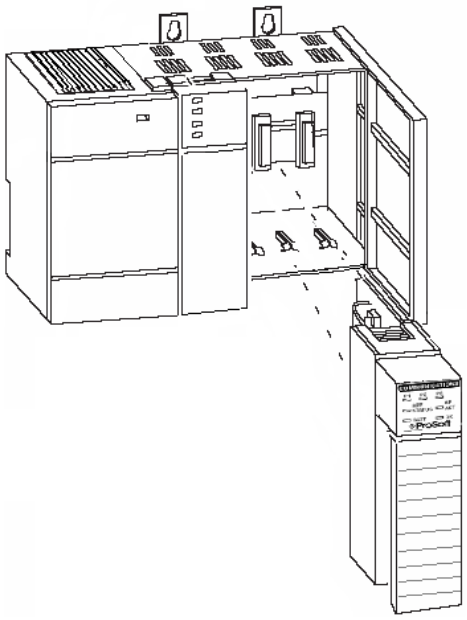
If you have not already installed and configured your SLC processor and power supply, please do so before installing the MVI46-MNETC module. Refer to your Rockwell Automation product documentation for installation instructions.

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 MVI46-MNETC into the SLC™ chassis. Use the same technique recommended by Rockwell Automation to remove and install SLC™ modules.

Warning: This module is not hot-swappable! Always remove power from the rack before inserting or removing this module, or damage may result to the module, the processor, or other connected devices.

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.

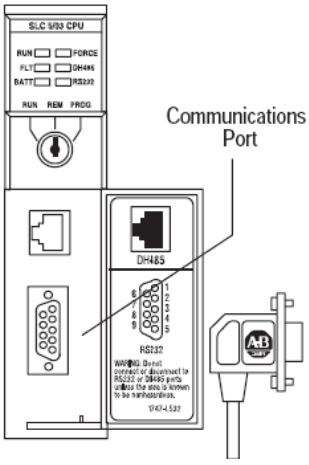


3 With a firm but steady push, snap the module into place.
4 Check that the holding clips on the top and bottom of the module are securely in the locking holes of the rack.
5 Make a note of the slot location. You will need to identify the slot in which the module is installed in order for the sample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

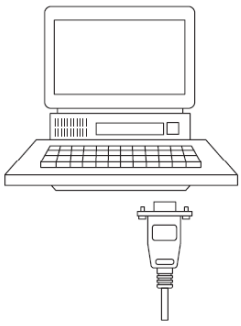
Note: If you insert the module improperly, the system may stop working, or may behave unpredictably.

1.6 Connecting Your PC to the Processor

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



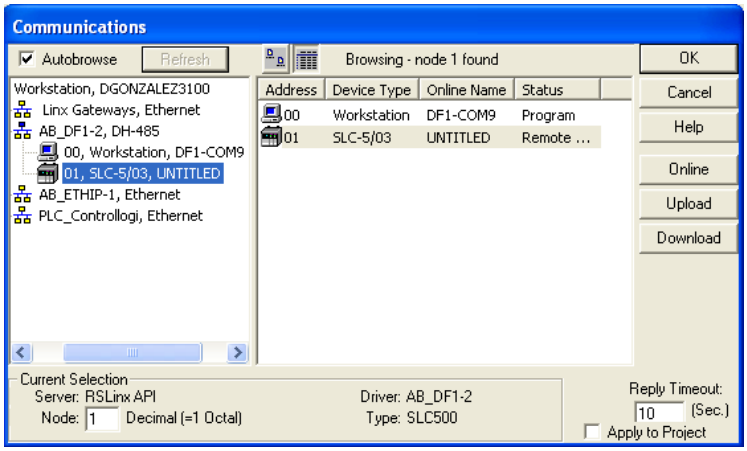
2 Connect the straight connector end of the cable to the serial port on your computer.



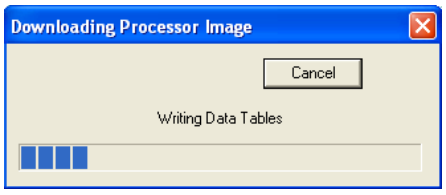
1.7 Downloading the Sample Program to the Processor

Note: The key switch on the front of the SLC processor must be in the REM position.

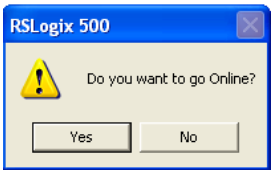
1 If you are not already online to the processor, open the **COMMUNICATIONS** menu, and then choose **DOWNLOAD**. RSLogix will establish communication with the processor.



2 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 **YES** to switch the processor from Program mode to Run mode.

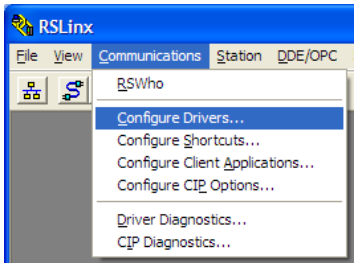


Note: If you receive an error message during these steps, refer to your RSLogix documentation to interpret and correct the error.

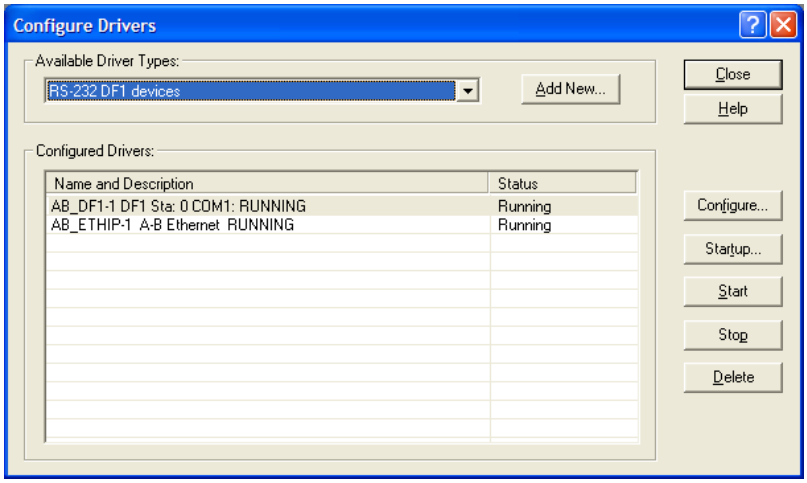
1.7.1 Configuring the RSLinx Driver for the PC COM Port

When trying to connect serially, if RSLogix is unable to establish communication with the processor, follow these steps.

1 Open *RSLinx*.
2 Open the **COMMUNICATIONS** menu, and click **CONFIGURE DRIVERS**.

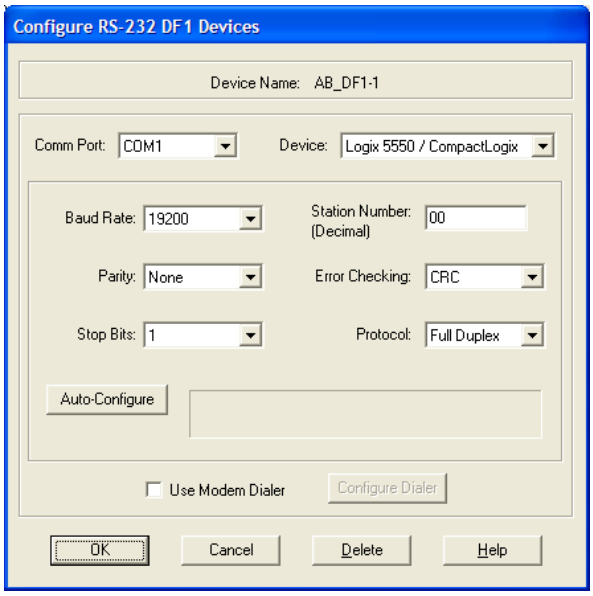


This action opens the *Configure Drivers* dialog box.



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

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



4 Click the **AUTO-CONFIGURE** button. RSLinx will attempt to configure your serial port to work with the selected driver.

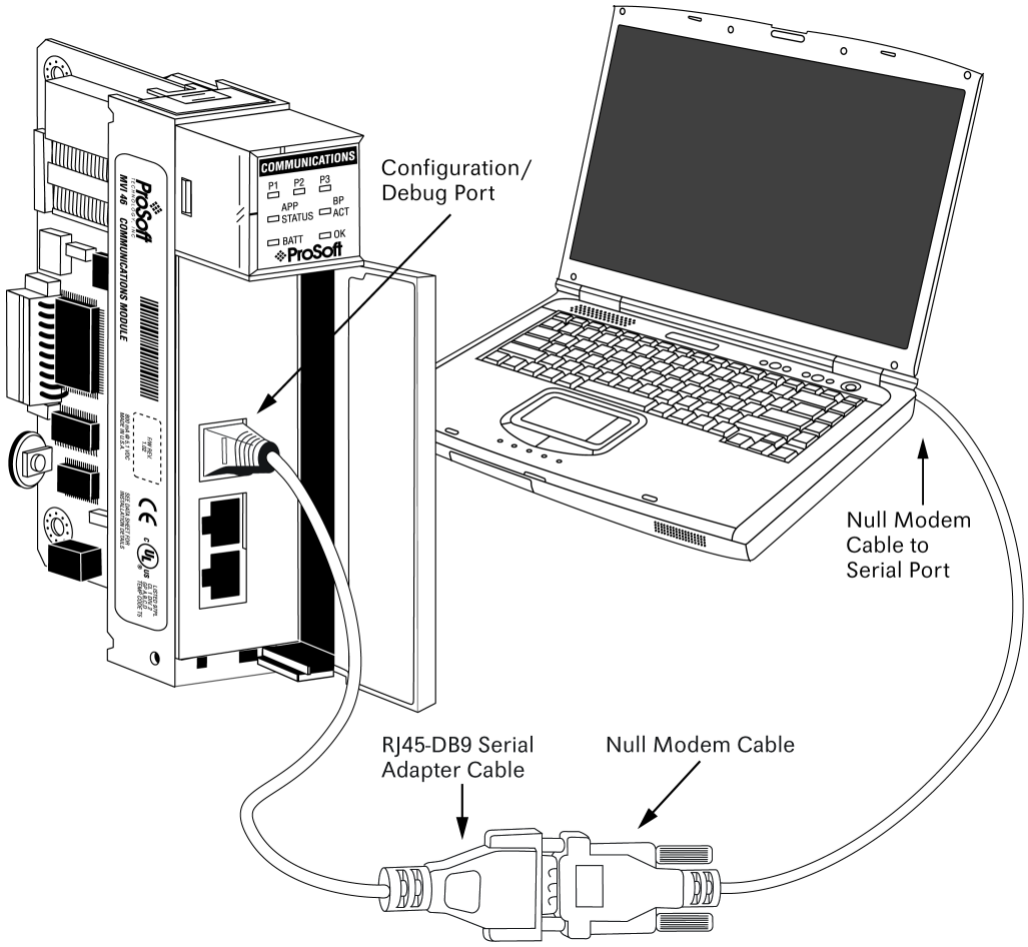
5 When you see the message *Auto Configuration Successful*, click the **OK** button to dismiss the dialog box.

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

1.8 Connecting Your PC to the Module

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

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



2 MVI46-MNETC Configuration

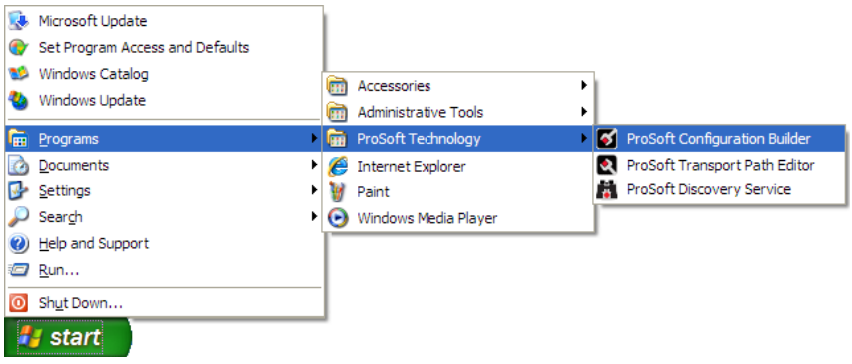
In This Chapter

2.1 Configuring the MVI46-MNETC using PCB

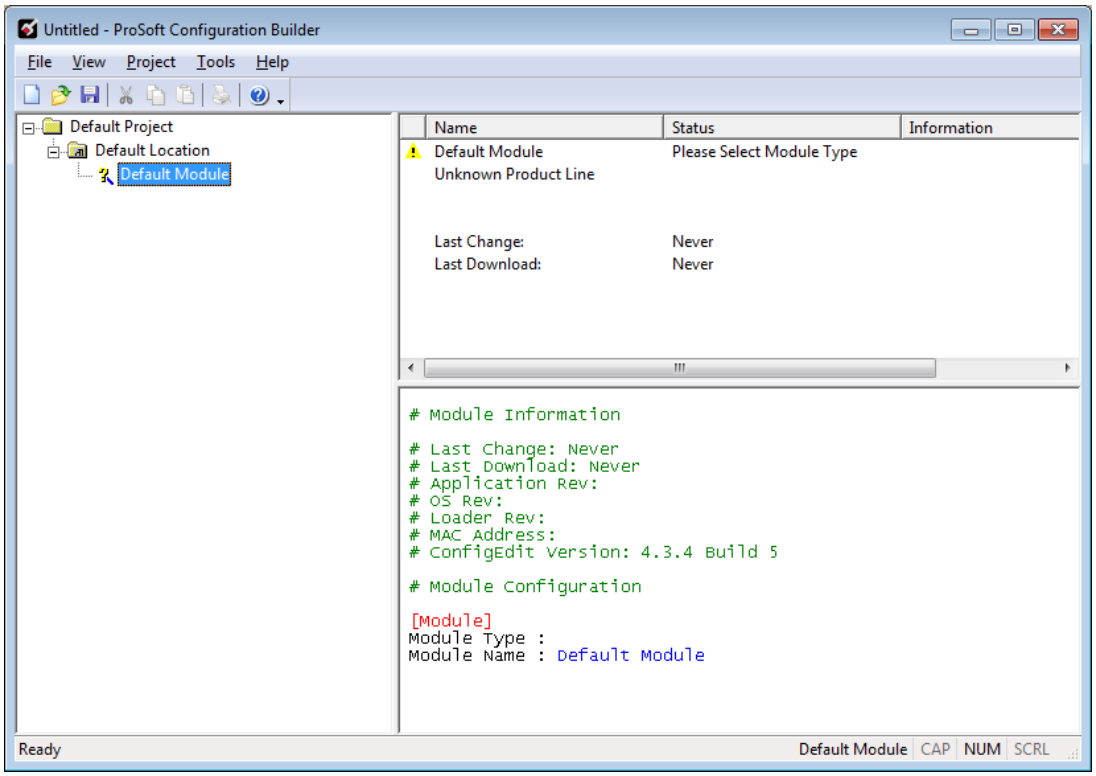
ProSoft Configuration Builder (PCB) provides a quick and easy way to manage module configuration files customized to meet your application needs. PCB is not only a powerful solution for new configuration files, but also allows you to import information from previously installed (known working) configurations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

2.1.1 Setting Up the Project

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

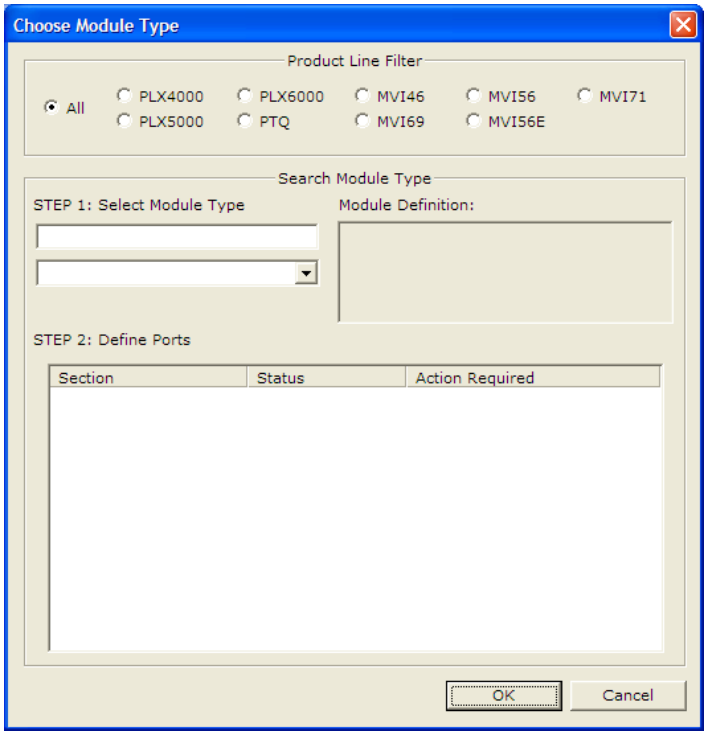


If you have used other Windows configuration tools before, you will find the screen layout familiar. PCB's window consists of a tree view on the left, and an information pane and a configuration pane on the right side of the window. When you first start *PCB*, the tree view consists of folders for **DEFAULT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



Your first task is to add the MVI46-MNETC 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 MVI46.
4. In the **SELECT MODULE TYPE** dropdown list, select MVI46-MNETC, 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.

2.1.2 Renaming PCB Objects

You can rename objects such as the *Default Project* and *Default Location* folders in the tree view. You can also rename the Module icon to customize the project.

1. Right-click the object you want to rename and then choose **RENAME.**
2. Type the new name for the object and press **Enter**.

Configuring Module Parameters

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

Printing a Configuration File

1. In the main PCB window, right-click the **MVI46-MNETC MODULE** icon and then choose **VIEW CONFIGURATION**.
2. In the *View Configuration* dialog box, click the **FILE** menu and click **PRINT**.
3. In the *Print* dialog box, choose the printer to use from the drop-down list, select the printing options, and click **OK**.

2.2 Module Configuration

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 is displayed in the PCB window.

2.2.1 Module

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

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 error/status data.

M1 Write Size

0 to 4000

This parameter limits the M1 data transferred from the processor to the module. The module application automatically adjusts the size to an even 50-word boundary as this is the minimum data transfer size for the application. For example, a value of 199 would automatically be adjusted to 200. This feature improves the transfer of data from the processor to the module.

Failure Flag Count

0 to **65535**

This parameter specifies the number of successive transfer errors that must occur before halting communication on the application port(s). If the parameter is set to **0**, the application port(s) will continue to operate under all conditions. If the value is set larger than **0** (**1** to **65535**), communications will cease if the specified number of failures occur.

Initialize Output Data

0 = No, **1** = Yes

This parameter is used to determine 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.

Duplex/Speed Code

0, **1**, **2**, **3** or **4**

This parameter allows you to cause the module to use a specific duplex and speed setting.

- Value = **1**: Half duplex, 10 MB speed
- Value = **2**: Full duplex, 10 MB speed
- Value = **3**: Half duplex, 100 MB speed
- Value = **4**: Full duplex, 100 MB speed
- Value = **0**: Auto-negotiate

Auto-negotiate is the default value for backward compatibility. This feature is not implemented in older software revisions.

2.2.2 Static ARP Table

The Static ARP Table defines a list of static IP addresses that the module will use when an ARP (Address Resolution Protocol) is required. The module will accept up to 40 static IP/MAC address data sets.

Use the Static ARP table to reduce the amount of network traffic by specifying IP addresses and their associated MAC (hardware) addresses that the MVI46-MNETC module will be communicating with regularly.

Important: If the device in the field is changed, this table must be updated to contain the new MAC address for the device and downloaded to the module. If the MAC is not changed, no communications with the module will be provided.

IP Address

Dotted notation

This table contains the static IP addresses that the module will use when an ARP is required. The module will accept up to 40 static IP/MAC address data sets.

Important: If the device in the field is changed, this table must be updated to contain the new MAC address for the device and downloaded to the module. If the MAC is not changed, no communications with the module will occur.

Hardware MAC Address

Hex value

This table contains the static MAC addresses that the module will use when an ARP is required. The module will accept up to 40 static IP/MAC address data sets.

Important: If the device in the field is changed, this table must be updated to contain the new MAC address for the device and downloaded to the module. If the MAC is not changed, no communications with the module will occur.

2.2.3 MNET Client x

This section defines general configuration for the MNET Client (Master).

Error/Status Pointer

-1 to **4990**

Starting register location in virtual 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.

Command Error Pointer

-1 to **4999**

This parameter sets the address in the internal database where the Command Error List data will be placed. If you want the Command Error List data to be moved to the processor and placed into the *ReadData* array, the value entered should be a module memory address in the Read Data area. If the value is set to **-1**, the Command Error List data will not be stored in the module's internal database and will not be transferred to the processor's *ReadData* array.

Minimum Command Delay

0 to **65535** milliseconds

This parameter specifies the number of milliseconds to wait between the initial issuances of a command. This parameter can be used to delay all commands sent to Servers 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.

Response Timeout

0 to **65535** milliseconds

This is the time in milliseconds that a Client will wait before re-transmitting a command if no response is received from the addressed server. The value to use depends on the type of 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.

Enron-Daniels

YES or **NO**

This flag specifies how the Slave driver will respond to Function Code 3, 6, and 16 commands (read and write Holding Registers) from a remote master when it is moving 32-bit floating-point data.

If the remote master expects to receive or will send one complete 32-bit floating-point value for each count of one (1), then set this parameter to **YES**. When set to **YES**, the Slave driver will return values from two consecutive 16-bit internal memory registers (32 total bits) for each count in the read command, or receive 32-bits per count from the master for write commands. Example: Count = **10**, Slave driver will send 20 16-bit registers for 10 total 32-bit floating-point values.

If, however, the remote master sends a count of two (2) for each 32-bit floating-point value it expects to receive or send, or, if you do not plan to use floating-point data in your application, then set this parameter to **NO**, which is the default setting.

You will also need to set the *Enron-Daniels Float Start* and *Enron-Daniels Float Offset* parameters to appropriate values whenever the *Enron-Daniels* parameter is set to **YES**.

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.

Enron-Daniels 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 *Enron-Daniels* is enabled. For example, if the *Enron-Daniels Float Offset* value is set to **3000** and the *Enron-Daniels Float Start* parameter is set to **7000**, data requests for register 7000 will use the internal Modbus register 3000.

ARP Timeout

1 to **60**

This parameter specifies the number of seconds to wait for an ARP reply after a request is issued.

Command Error Delay

0 to **300**

This parameter specifies the number of 100 millisecond intervals to turn off a command in the error list after an error is recognized for the command. If this parameter is set to **0**, there will be no delay.

MBAP Port Override

YES or **NO**

If this parameter is set to **YES**, all messages generated by the Client driver will be MBAP format messages to all Service Port values.

If this parameter is set to **No** (default value), or is omitted from the configuration file, all messages sent to Service Port 502 will be MBAP format messages, and all other Service Ports values will use the encapsulated Modbus message format (MNET).

Each Client is configured independently in the configuration file.

This parameter applies to firmware version 1.05 and above. For downward compatibility, you may omit this parameter from the Client's configuration.

2.2.4 MNET Client x Commands

The *MNET Client x Commands* section of the configuration sets the Modbus TCP/IP Client command list. This command list polls Modbus TCP/IP server devices attached to the Modbus TCP/IP Client port. The module supports numerous commands. This permits the module to interface with a wide variety of Modbus TCP/IP protocol devices.

The function codes used for each command are those specified in the Modbus protocol. Each command list record has the same format. The first part of the record contains the information relating to the MVI46-MNETC communication module, and the second part contains information required to interface to the Modbus TCP/IP server device.

Command List Overview

In order to interface the 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 16 commands.

The command list is processed from top (command #1) to bottom. A poll interval parameter is associated with each command to specify a minimum delay time in tenths of a second between the issuances 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.

Commands Supported by the Module

The format of each command in the list depends on the Modbus Function Code being executed.

The following table lists 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	Force (Write) Single Coil	X	X
6	Preset (Write) Single Register	X	X
7	Read Exception Status	X	X
8	Diagnostics		X
15	Force (Write) Multiple Coils	X	X
16	Preset (Write) Multiple Registers	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.

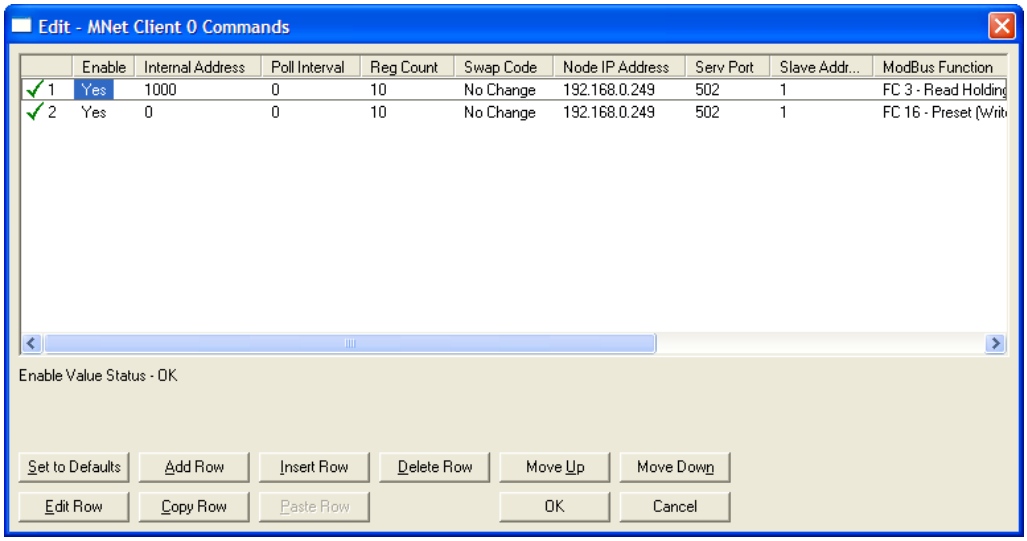
Command Entry Formats

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

1	2	3	4	5	6	7	8	9	10
Enable Code	**Internal Address**	**Poll Interval Time**	**Count**	**Swap Code**	**IP Address**	**Serv Port**	**Slave Node**	**Function Code**	**Device Modbus Address**
Code	Register (bit)	1/10th Seconds	Bit Count	0	IP Address	Port #	Address	Read Coil (0x)	Register
Code	Register (bit)	1/10th Seconds	Bit Count	0	IP Address	Port #	Address	Read Input (1x)	Register
Code	Register	1/10th Seconds	Word Count	Code	IP Address	Port #	Address	Read Holding Registers (4x)	Register
Code	Register	1/10th Seconds	Word Count	0	IP Address	Port #	Address	Read Input Registers (3x)	Register
Code	1 bit	1/10th Seconds	Bit Count	0	IP Address	Port #	Address	Force (Write) Single Coil (0x)	Register
Code	1 bit	1/10th Seconds	Word Count	0	IP Address	Port #	Address	Preset (Write) Single Register (4x)	Register
Code	Register (bit)	1/10th Seconds	Bit Count	0	IP Address	Port #	Address	Force (Write) Multiple Coil (0x)	Register
Code	Register	1/10th Seconds	Word Count	0	IP Address	Port #	Address	Preset (Write) Multiple Register (4x)	Register

The first part of the record is the module information, which relates to the MVI46-MNETC, and the second part contains information required to interface to the server device.

Command list example:



Enable

NO (0), **YES** (1), or **CONDITIONAL** (2)

This field defines whether or not the command is to be executed.

Value	Description
NO (0)	The command is disabled and will not be executed in the normal polling sequence.
YES (1)	The command is executed each scan of the command list if the Poll Interval Time is set to zero (**0**). If the Poll Interval time is set, the command will be executed when the interval timer expires.
CONDITIONAL (2)	The command executes only if the internal data associated with the command changes.

Important: The commands must also be enabled in the ladder logic in order for them to be executed. The *MNETC.CONTROL.CmdControl.WriteCmdBits[x]* controller tag array holds 16-command bit arrays for each Client. If a bit for a specific command is set to zero (**0**) in the *WriteCmdBits[x]* controller tag, the command will not be executed, regardless of its enabled or disabled state in the configuration. For more information, see Command Control Blocks.

Internal Address

0 to **65535** (for bit-level addressing)

or

0 to **4999** (for word-level addressing)

This field specifies the database address in the module's internal database to use as the destination for data brought in by a read command or as the source for data to be sent out by a write command. The database address is interpreted as a bit address or a 16-bit word (register) address, depending on the Modbus Function Code used in the command.

- For Modbus functions 1, 2, 5, and 15, this parameter is interpreted as a bit-level address.
- For Modbus functions 3, 4, 6, and 16, this parameter is interpreted as a word-level or register-level address.

Note: When using a bit level command, you must define this field at the bit level. For example, when using function codes 1 or 2 for a Read command, you must have a enter of 16000 to place the data in MNET.DATA.ReadData[0] (The module's internal register 1000 * 16 bits per register = 16000). Use this formula for function codes 5 or 15 for writing bits also.

This controller tag is a 16bit signed integer. This means you can only enter values of -32768 to 32767 in the tag. If a value to be entered is above the 32767 (but below 65535) threshold, it will display as a negative value in the tag. Simply subtract 65536 from the value to get the 'acceptable' value to enter into the tag.

Example: You need to use an Internal bit Address of 48000, but you cannot enter '48000' into the tag because it causes an error. 48000 - 65536 = -17536 You need to enter -17536 in the Internal Address parameter for this command.

Poll Interval

0 to **65535**

This parameter specifies the minimum interval between issuances of a command during continuous command execution (*Enable* code of **1**). The parameter is entered in tenths of a second. Therefore, if a value of **100** is entered for a command, the command executes no more frequently than every 10 seconds.

Reg Count

Regs: **1** to **125**

Coils: **1** to **800**

This parameter specifies the number of 16-bit registers or binary bits to be transferred by the command.

- Functions 5 and 6 ignore this field as they apply only to a single data point.
- For functions 1, 2, and 15, this parameter sets the number of bits (inputs or coils) to be transferred by the command.
- For functions 3, 4, and 16, this parameter sets the number of registers to be transferred by the command.

Swap Code

NONE

SWAP WORDS

SWAP WORDS & BYTES

SWAP BYTES

This parameter defines if and how the order of bytes in data received or sent is to be rearranged. This option exists to allow for the fact that different manufacturers store and transmit multi-byte data in different combinations. This parameter is helpful when dealing with floating-point or other multi-byte values, as there is no one standard method of storing these data types. The parameter can be set to rearrange the byte order of data received or sent into an order more useful or convenient for other applications. The following table defines the valid *Swap Code* values and the effect they have on the byte-order of the data.

Swap Code	Description
NONE	No change is made in the byte ordering (1234 = 1234)
SWAP WORDS	The words are swapped (1234=3412)
SWAP WORDS & BYTES	The words are swapped, then the bytes in each word are swapped (1234=4321)
SWAP BYTES	The bytes in each word are swapped (1234=2143)

These swap operations affect 4-byte (or 2-word) groups of data. Therefore, data swapping using these *Swap Codes* should be done only when using an even number of words, such as when 32-bit integer or floating-point data is involved.

Node IP Address

xxx.xxx.xxx.xxx

The IP address of the device being addressed by the command.

Service Port

502 or other port numbers supported on a server

This 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.

Slave Address

0 - Broadcast to all nodes

1 to **255**

Use this parameter to specify the slave address of a remote Modbus Serial device through a Modbus Ethernet to Serial converter.

Note: Use the *Node IP Address* parameter to address commands to a remote Modbus TCP/IP device. See Node IP Address (page 30).

Note: Most Modbus devices accept an address in the range of only 1 to 247, so check with the slave device manufacturer to see if a particular slave can use addresses 248 to 255.

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 node address 0 for **read** operations.

Modbus Function

1, **2**, **3**, **4**, **5**, **6**, **15**, or **16**

This parameter specifies the Modbus Function Code to be executed by the command. These function codes are defined in the Modbus protocol. The following table lists the purpose of each function supported by the module. More information on the protocol is available from www.modbus.org.

Modbus Function Code	Description
1	Read Coil Status
2	Read Input Status
3	Read Holding Registers
4	Read Input Registers
5	Force (Write) Single Coil
6	Preset (Write) Single Register
15	Force Multiple Coils
16	Preset Multiple Registers

MB Address in Device

This parameter specifies the starting Modbus register or bit address in the Server to be used by the command. Refer to the documentation of each Modbus Server device for the register and bit address assignments valid for that device.

The Modbus Function Code determines whether the address will be a register-level or bit-level OFFSET address into a given data type range. The offset will be the target data address in the Server minus the base address for that data type. Base addresses for the different data types are:

- 00001 or 000001 (0x0001) for bit-level Coil data (Function Codes 1, 5, and 15).
- 10001 or 100001 (1x0001) for bit-level Input Status data (Function Code 2)
- 30001 or 300001 (3x0001) for Input Register data (Function Code 4)
- 40001 or 400001 (4x0001) for Holding Register data (Function Codes 3, 6, and 16).

Address calculation examples:

- For bit-level Coil commands (FC 1, 5, or 15) to read or write a Coil 0X address 00001, specify a value of 0 (00001 - 00001 = 0).
- For Coil address 00115, specify 114

  (00115 - 00001 = 114)
- For register read or write commands (FC 3, 6, or 16) 4X range, for 40001, specify a value of 0

  (40001 - 40001 = 0).
- For 01101, 11101, 31101 or 41101, specify a value of 1100.

  (01101 - 00001 = 1100)
  (11101 -10001 = 1100)
  (31101 - 30001 = 1100)
  (41101 - 40001 = 1100)

Note: If the documentation for a particular Modbus Server device lists data addresses in hexadecimal (base16) notation, you will need to convert the hexadecimal value to a decimal value to enter in this parameter. In such cases, it is not usually necessary to subtract 1 from the converted decimal number, as this addressing scheme typically uses the exact offset address expressed as a hexadecimal number.

Comment

0 to 32 alphanumeric characters

2.2.5 Ethernet Configuration

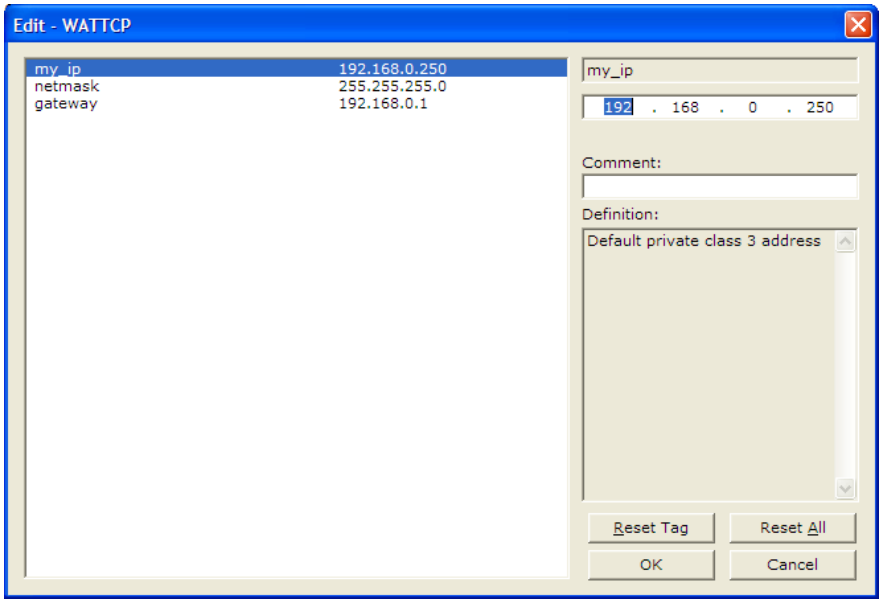
Use this procedure to configure the Ethernet settings for your module. You must assign an IP address, subnet mask and gateway address. After you complete this step, you can connect to the module with an Ethernet cable.

1 Determine the network settings for your module, with the help of your network administrator if necessary. You will need the following information:

- IP address (fixed IP required) _____ . _____ . _____ . _____
- Subnet mask _____ . _____ . _____ . _____
- Gateway address _____ . _____ . _____ . _____

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

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



3 Edit the values for *my_ip*, *netmask* (subnet mask) and *gateway* (default gateway).

4 When you are finished editing, click **OK** to save your changes and return to the *ProSoft Configuration Builder* window.

2.3 Setting Command Control Bits

Toggle the CMD_BITS_CNTRL bit in the ladder logic (Rung 6 of LAD4 - CNTRL in the sample program) to execute the Modbus Commands as structured in the MNET.CFG file. This will cause continuous traffic on the Ethernet network with data flow as configured in MNET.CFG file.

This rung also selects the Commands to enable as set by the Enabled bits in each word N50:21 through N50:50 representing each client.

- Client-0 enable Command bits are in N50:21, MSB xxxx xxxx xxxx xxxx LSB.
- Client-1 enable Command bits are in N50:22
- Client-29 enable Command bits are in N50:50

To change the set of Clients and their Commands to execute, change the bit setting from 0 to 1 or from 1 to 0 then toggle the CMD_BITS_CNTRL bit.

Example: Set Client-1 in N50:22 as follows: MSB 0001 0000 0100 0100 LSB. Bits 2 and 6 and 12 have been enabled. This means that Client-1 commands indexed as 2, 6 and 12 will be executed.

N50:20 contains Value 6000 to trigger the Block 6000, the Command Control Bits function.

2.4 Configuring the Floating Point Data Transfer

A common question when using the module as a Modbus TCP/IP client is how floating point data is handled. This really depends on the server device and how it addresses this application.

Just because your application is reading/writing floating point data, does not mean that you must configure the Float Flag, Float Start parameters within the module.

These parameters are only used to support what is typically referred to as Enron or Daniel Modbus, where one register address must have 32 bits, or one floating point value. Below is an example:

Example #1

Modbus Address	Data Type	Parameter
47101	32 bit REAL	TEMP Pump #1
47102	32 bit REAL	Pressure Pump #1
47103	32 bit REAL	TEMP Pump #2
47104	32 bit REAL	Pressure Pump #2

With the module configured as a master, you only need to enable these parameters to support a write to this type of addressing (Modbus FC 6 or 16).

If the slave device shows addressing as shown in Example #2, then you need not do anything with the Float Flag, Float Start parameters, as they use two Modbus addresses to represent one floating point value:

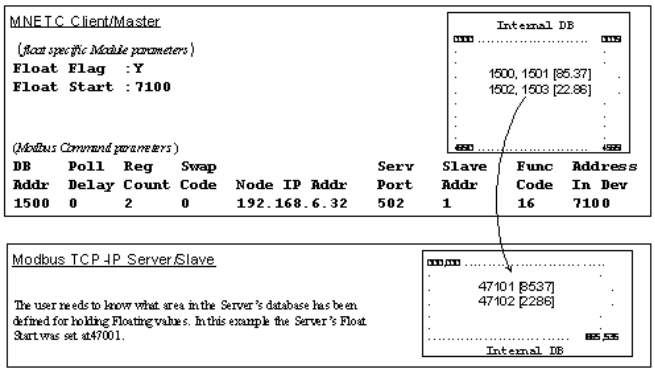
Example #2

Modbus Address	Data Type	Parameter
47101	32 bit REAL	TEMP Pump #1
47103	32 bit REAL	Pressure Pump #1
47105	32 bit REAL	TEMP Pump #2
47107	32 bit REAL	Pressure Pump #2

Because each 32 bit REAL value is represented by two Modbus Addresses (example 47101 and 47102 represent TEMP Pump #1), then you need not set the Float Flag, or Float Start for the module for Modbus FC 6 or 16 commands being written to the slave.

Below are specific examples:

Master is issuing Modbus command with FC 16 (with Float Flag: Yes) to transfer Float data to Server.



(Float specific module parameters)

Float Flag: "Y" tells the Client to consider the data values that need to be sent to the Server as floating point data where each data value is composed of 2 words (4 bytes or 32 bits).

Float Start: Tells the Client that if this address number is <= the address number in "Addr in Dev" parameter to double the byte count quantity to be included in the Command FC6 or FC16 to be issued to the Server. Otherwise the Client will ignore the "Float Flag: Y" and treat data as composed of 1 word, 2 bytes.

(Modbus Command Parameters)

DB Addr - Tells the Client where in its data memory is the beginning of data to obtain and write out to the Server (slave) device.

Reg Count - Tells the Client how many data points to send to the Server. Two counts will mean two floating points with Float Flag: Y and the "Addr in Dev" => the "Float Start" Parameter.

Swap Code - Tells the Client how to orient the Byte and Word structure of the data value. This is device dependent. Check Command Entry formats Section.

Func Code - Tells the Client to write the float values to the Server. FC16.

Addr in Dev - Tells the Client where in the Server's database to locate the data.

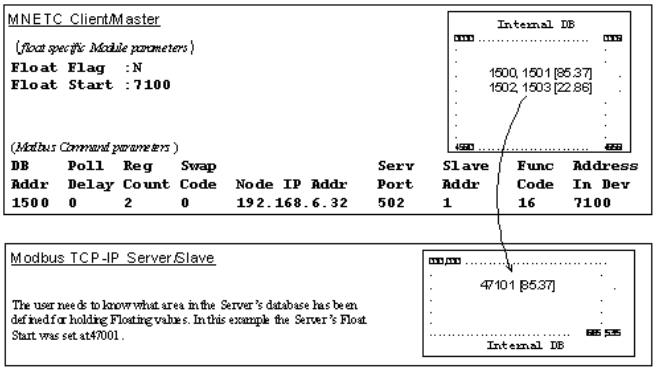
In the above example, the Client's Modbus command to transmit inside the Modbus/TCP packet will be as follows.

	Slave address	Function Code	Address in Device	Reg count	Byte Count	Data	
DEC	01	16	7100	2	8	85.37	22.86
HEX	01	10	1B BC	00 02	08	BD 71 42 AA	E1 48 41 B6

In conclusion

The Client's Modbus TCP/IP packet contains the data byte and data word counts that have been doubled from the amount specified by Reg Count due to the Float flag set to Y. Some Servers look for the byte count in the data packet to know the length of the data to read from the wire. Other servers know at which byte the data begins and read from the wire the remaining bytes in the packet as the data the Client is sending.

Client is issuing Modbus command with FC 16 (with Float Flag: No) to transfer Float data.



Float Flag: "N" tells the Client to ignore the floating values and treat each register data as a data point composed of 1 word, 2 bytes or 16 bits.

Float Start: Ignored.

DB Addr - same as when Float Flag: Y.

Reg Count - Tells the Client how many data points to send to the Server.

Swap Code - same as when Float Flag: Y.

Func Code - same as when Float Flag: Y.

Addr in Dev - same as when Float Flag: Y as long as the Server's Float Flag = Y.

In the above example, the Client's Modbus command to transmit inside the Modbus/TCP pocket will be as follows.

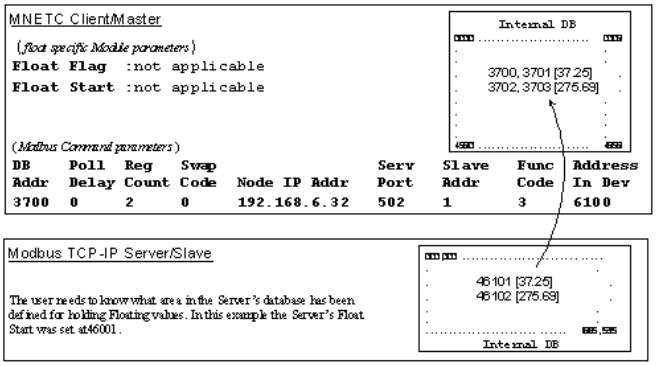
	Slave address	Function Code	Address in Device	Reg count	Byte Count	Data
DEC	01	16	7100	2	4	85.37

	Slave address	Function Code	Address in Device	Reg count	Byte Count	Data
HEX	01	10	1B BC	00 02	04	BD 71 42 AA

In conclusion

The Client's Modbus TCP/IP packet contains the data byte and data word counts that have NOT been doubled from the amount specified by Reg Count due to the Float Flag set to N. The Server looks for the byte count in the data packet to know the length of the data to read from the wire. Because of insufficient byte count, some servers will read only half the data from the Client's transmission. Other servers will read all 8 bytes in this example because they will know where in the packet the data starts and ignore the byte count parameter inside the Modbus TCP/IP packet.

Client is issuing Modbus command with FC 3 to transfer Float data from Server.



Float Flag: Not applicable with Modbus Function Code 3.

Float Start: Not applicable with Modbus Function Code 3.

DB Addr - Tells the Client where in its data memory to store the data obtained from the Server.

Reg Count - Tells the Client how many registers to request from the Server.

Swap Code - same as above.

Func Code - Tells the Client to read the register values from the Server. FC3.

Addr in Dev - Tells the Client where in the Server's database to obtain the data.

In the above example, the Client's Modbus command to transmit inside the Modbus/TCP pocket will be as follows.

	Slave address	Function Code	Address in Device	Reg count
DEC	01	3	6100	2
HEX	01	03	17 D4	00 02

In the above example the (Enron/Daniel supporting) Slave's Modbus command to transmit inside the Modbus/TCP packet will be as follows.

	Slave address	Function Code	Byte Count	Data

	Slave address	Function Code	Byte Count	Data	
DEC	01	3	8	32.75	275.69
HEX	01	03	08	00 00 42 03	D8 52 43 89

In the above example the (a NON-Enron/Daniel supporting) Slave's Modbus command that will be transmitted inside the Modbus/TCP packet will be as follows.

	Slave address	Function Code	Byte Count	Data
DEC	01	3	4	32.75
HEX	01	03	04	00 00 42 03

Note: You can use the Debug window to view the floating points in the Module's database. First, you must copy the registers in the F8-FLOAT file in the SLC to the N7-INTEGER file registers, and then swap the two words that make up a floating point. Then, copy the registers to the module's M1 file.

For example:

Copy F8:0 to N7:0 [length 2].

Next two move functions perform swapping.

Move N7:0 to N7:11

Move N7:1 to N7:10

The Copy function transfers the two words to the Module.

Copy N7:10 to M1:1.0 [length 2]

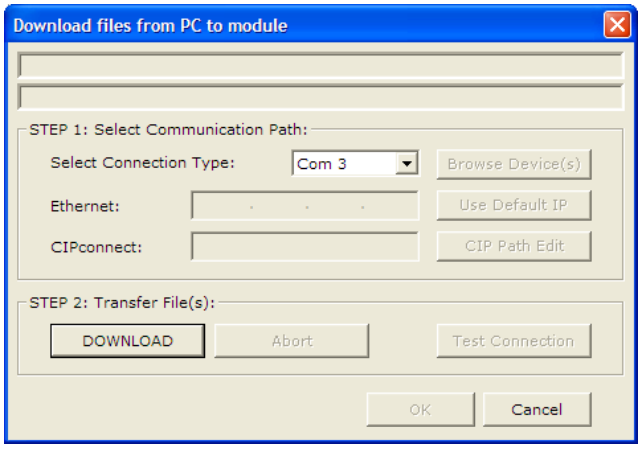
2.5 Downloading the Configuration to the Module Using Serial

For the module to use the settings you configured, you must download (copy) the updated *Project* file from your PC to the module. See Connecting the PC to the Module's Ethernet Port.

Note: The first time you download the project to the module, you must use the serial COM port to download the project, including the IP address. After that, you can use the Ethernet port to communicate with the module.

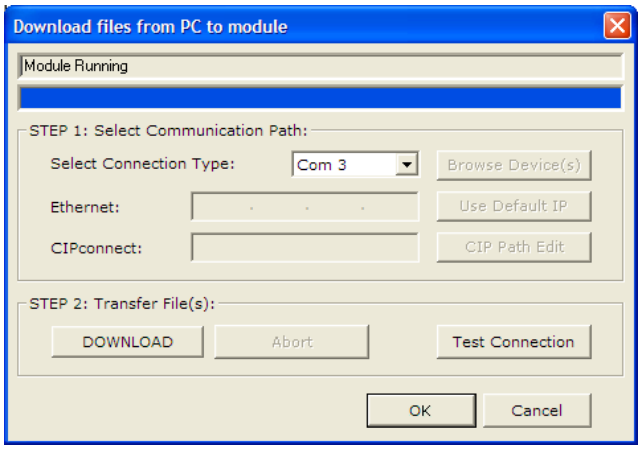
To download the project file

1 In the tree view in *RSLogix 500*, right-click the module icon, and choose **DOWNLOAD FROM PC TO DEVICE**. The program scans your PC for a valid com port (this may take a few seconds). When the *RSLogix 500* finds a valid COM port, it opens the *Download files from PC to module* dialog box.



2 Choose the COM port to use from the dropdown list, and then click **DOWNLOAD**.

The module performs a platform check to read and load its new settings. When the platform check is complete, the status bar in the *Download files* dialog box displays the message *Module Running*.



3 Ladder Logic

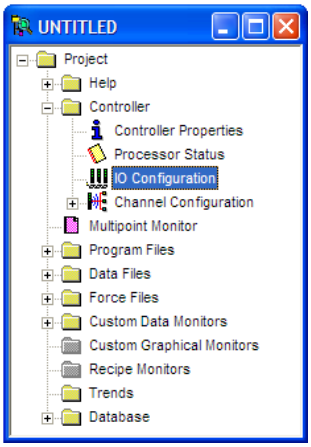
In This Chapter

Ladder logic is required for the MVI46-MNETC module to work. Tasks that must be handled by the ladder logic are module data transfer, special block handling, and status data receipt. Additionally, a power-up handler may be needed to handle the initialization of the module's data and to clear any processor fault conditions.

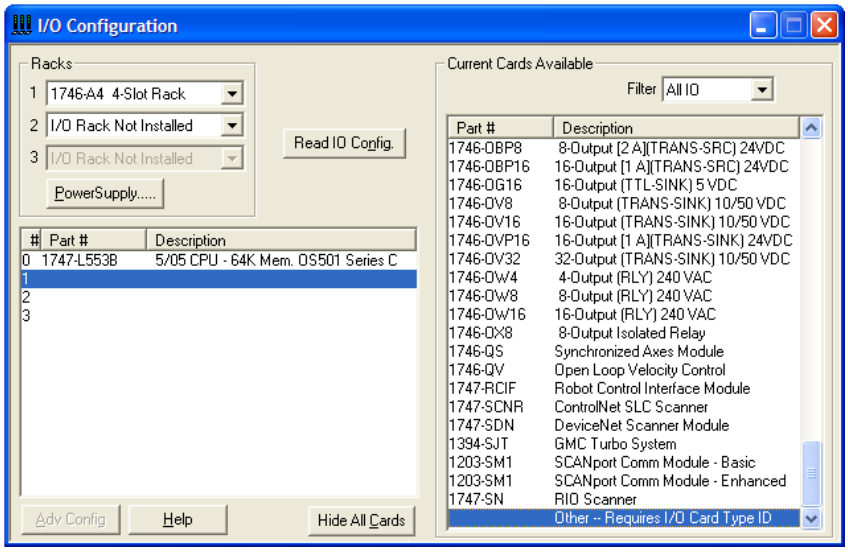
The sample ladder logic is extensively commented, to provide information on the purpose and function of each rung. For most applications, the sample ladder will work without modification.

3.1 Adding the Module to an Existing Project

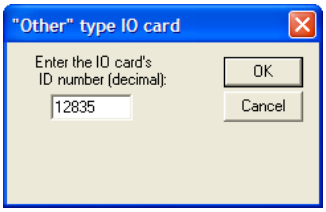
1 Double-click **I/O CONFIGURATION** in the *Controller Organization* window.



2 This action opens the I/O *Configuration* dialog box. Select an empty slot in the left pane, and then scroll to the bottom of the right pane.

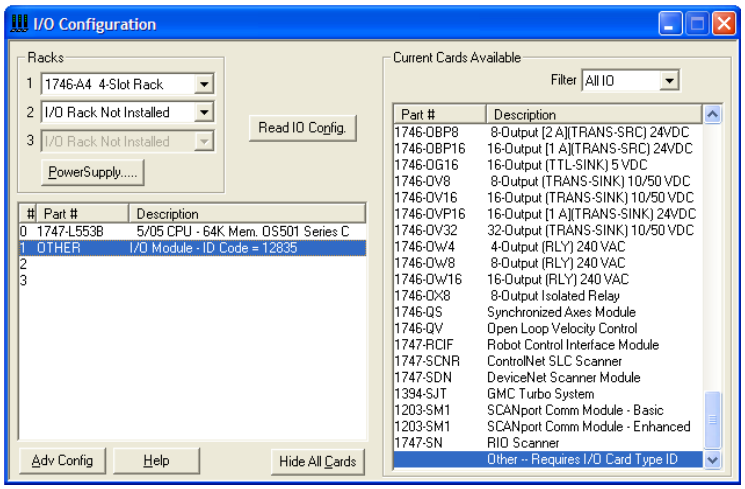


3 In the right pane, double-click **OTHER -- REQUIRES I/O CARD TYPE ID**. This action opens the **"OTHER" TYPE IO CARD** dialog box.

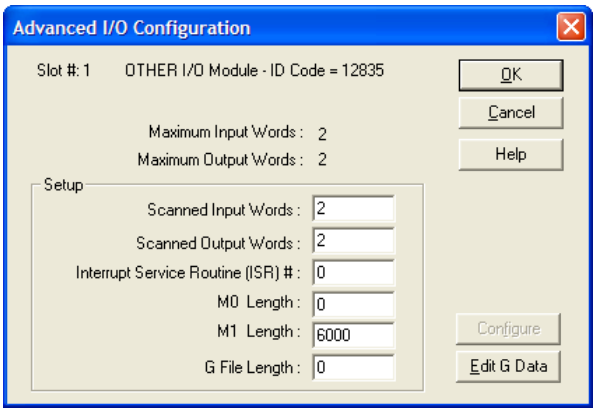


4 The module's I/O card ID number is 12835. Enter that value in the ID number field, and then click **OK** to dismiss the dialog box.

5 Observe that the module you selected is now in the list in the left pane of the *I/O Configuration* dialog box.



6 Select and double-click the new module in the left pane. This action opens the *Advanced I/O Configuration* dialog box. Fill in the dialog box with the values shown in the following illustration.



Field	Value
Scanned Input Words	2
Scanned Output Words	2
Interrupt Service Routine (ISR)#	0
M0 Length	0
M1 Length	6000
G File Length	0

7 Click OK to save your configuration.

8 Copy the ladder logic and data files from the sample program and paste them into your existing program.

Important: Take care not to overwrite existing data files in your application with data files in the sample application. Rename either the source or the destination data files, and then search and replace references in the ladder for instances of any renamed files.

9 Save the project. It is now ready to download to the processor.

4 Diagnostics and Troubleshooting

In This Chapter

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

- LED status indicators on the front of the module provide general information on the module's status.
- You can view status data contained in the module through the Configuration/Debug port or the Ethernet port, using the troubleshooting and diagnostic capabilities of *ProSoft Configuration Builder (PCB)*.
- You can transfer status data values from the module to processor memory and can monitor them in the processor manually or by customer-created logic. For details on Status Data values, see Error Status Table.

4.1 Reading Status Data from the Module

The MVI46-MNETC module returns status data words 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 can be transferred to the SLC processor continuously or at interval set by the user in ladder logic.

The Configuration/Debug port provides the following functionality:

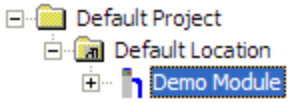
- Full view of the module's configuration data
- View of the module's status data
- Complete display of the module's internal database (registers 0 to 3999)
- Version Information
- Control over the module (warm boot, cold boot, transfer configuration)
- Facility to upload and download the module's configuration file

4.1.1 Using ProSoft Configuration Builder (PCB) for Diagnostics

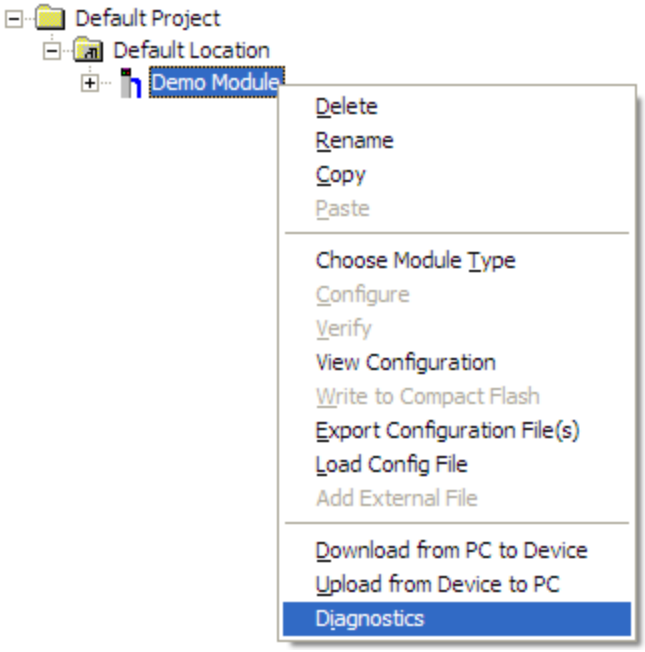
Using the Diagnostic Window in ProSoft Configuration Builder

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

1 Start *PCB*, and then right-click the module icon.

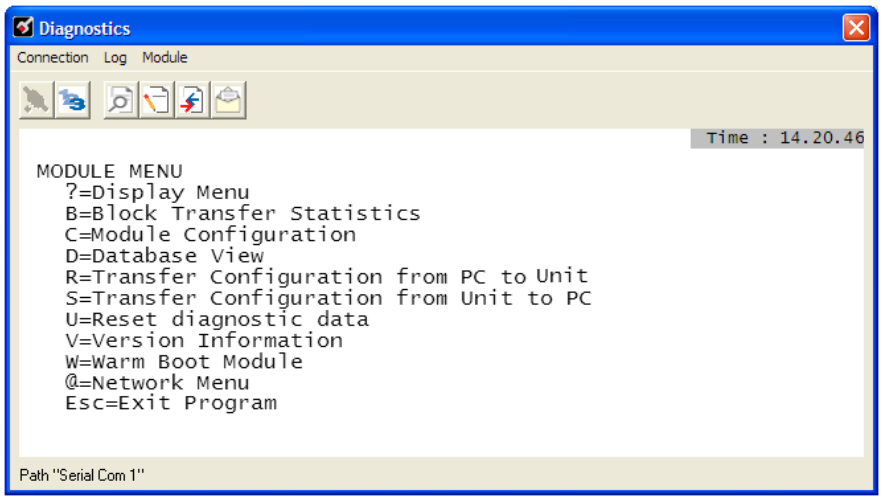


2 On the shortcut menu, choose **DIAGNOSTICS**.



This action opens the *Diagnostics* dialog box.

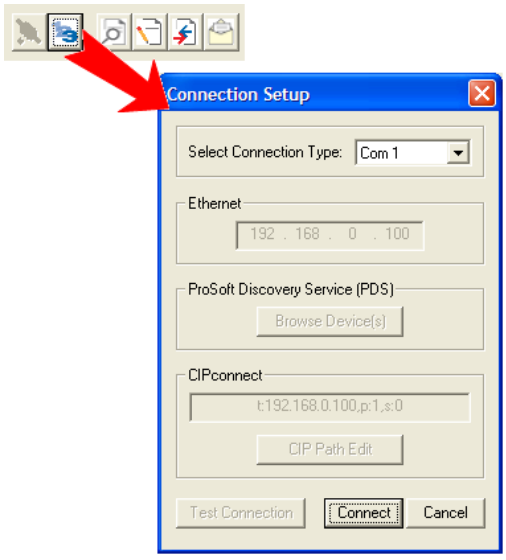
3 Press **[?]** to open the *Main* menu.



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



2 For a serial connection, verify that the null modem cable is connected properly between your computer's serial port and the module. A regular serial cable will not work.

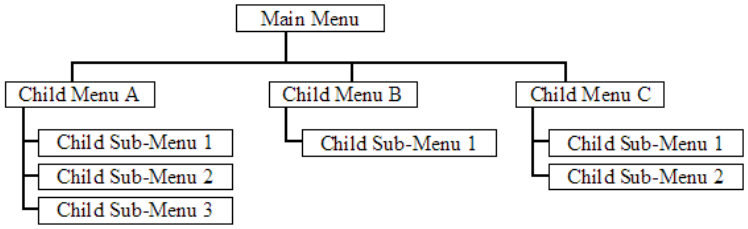
3 On computers with more than one serial port, verify that your communication program is connected to the same port that is connected to the module.

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

Navigation

All of the submenus in *ProSoft Configuration Builder* for this module contain commands to redisplay the menu or return to the previous menu. You can always return from a submenu to the next higher menu by pressing **[M]** on your keyboard.

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



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

Keystrokes

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

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

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

4.1.2 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.

```
MVI46-MNETC COMMUNICATION MODULE MENU
  ?=Display Menu
  B=Block Transfer Statistics
  C=Module Configuration
  D=Modbus Database View
  E=Client Command List Errors
  I=Client Command List
  R=Transfer Configuration from PC to MVI Unit
  S=Transfer Configuration from MVI Unit to PC
  U=Reset diagnostic data
  V=Version Information
  W=Warm Boot Module
  0=Client Communication Status
  5=Client Configuration
  @=Network Menu          Esc=Exit Program
```

Caution: Some of the commands available to you from this menu are designed for advanced debugging and system testing only, and can cause the module to stop communicating with the processor or with other devices, resulting in potential data loss or other failures. Use these commands only if you fully understand their potential effects, or 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 number of seconds passed between the two readings.

Viewing Module Configuration

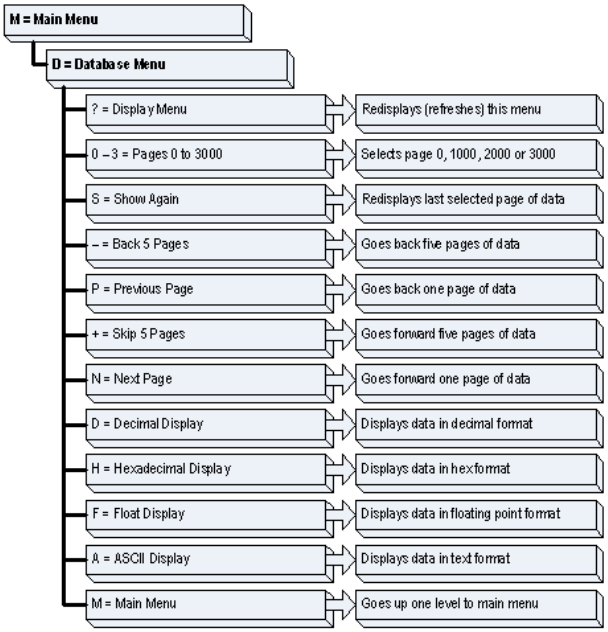
Press **[C]** to view the *Module Configuration* screen.

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

Opening the Database View Menu

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

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



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 to the PC. For more information on receiving and sending configuration files, refer to Downloading the PCB File to the Module.

Sending the Configuration File

Press **[S]** to upload (send) a configuration file from the module to your PC. For more information on receiving and sending configuration files, refer to Downloading the PCB File to the Module.

Resetting Diagnostic Data

Press **[U]** to reset the status counters for the Client and/or server(s) in the module.

Viewing Version Information

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

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

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

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

Warm Booting the Module

Caution: Some of the commands available to you from this menu are designed for advanced debugging and system testing only, and can cause the module to stop communicating with the processor or with other devices, resulting in potential data loss or other failures. Use these commands only if you fully understand their potential effects, or 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 causes the program to exit and reload, refreshing configuration parameters that must be set on program initialization. Only use this command if you must force the module to reboot.

Viewing Client Status

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

Viewing Client Configuration

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

Exiting the Program

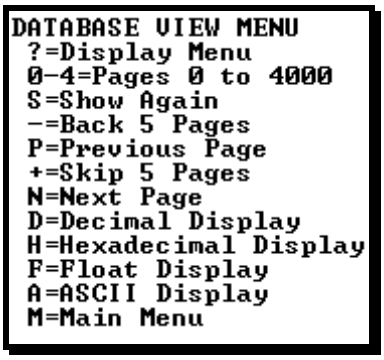
Caution: Some of the commands available to you from this menu are designed for advanced debugging and system testing only, and can cause the module to stop communicating with the processor or with other devices, resulting in potential data loss or other failures. Use these commands only if you fully understand their potential effects, or if you are specifically directed to do so by ProSoft Technology Technical Support staff.

Some of these command keys are not listed on the menu, but are active nevertheless. Please be careful when pressing keys so that you do not accidentally execute an unwanted command.

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

4.1.3 Modbus Database View Menu

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 the 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 five pages back in the database to see the previous 100 registers of data starting 500 registers before the currently displayed page.

Viewing the Previous Page of Registers

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

Moving Forward Through 5 Pages of Registers

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

Viewing the Next Page of Registers

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

Viewing Data in Decimal Format

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

Viewing Data in Hexadecimal Format

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

Viewing Data in Floating-Point Format

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

Viewing Data in ASCII (Text) Format

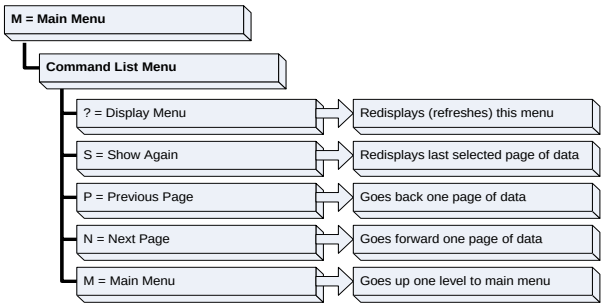
Press **[A]** from the *Database View* menu to display the data on the current page in ASCII format. This is useful for regions of the database that contain ASCII data.

Returning to the Main Menu

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

4.1.4 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

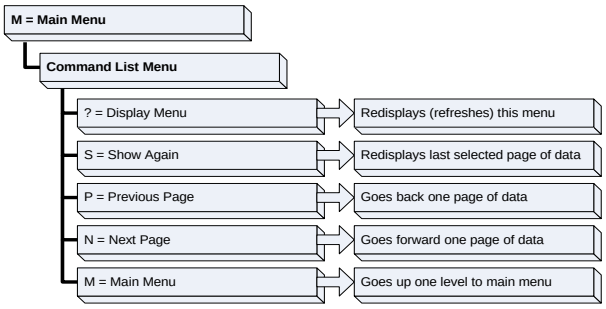
Press **[N]** to display the next page of commands.

Returning to the Main Menu

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

4.1.5 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

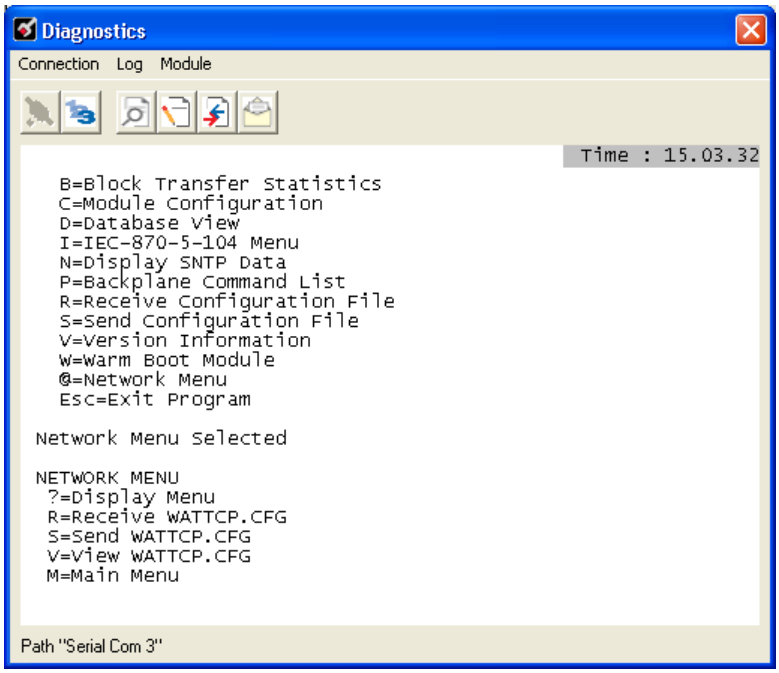
Press **[N]** to display the next page of commands.

Returning to the Main Menu

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

4.1.6 Network Menu

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



Transferring WATTCP.CFG to the Module

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

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

Transferring WATTCP.CFG to the PC

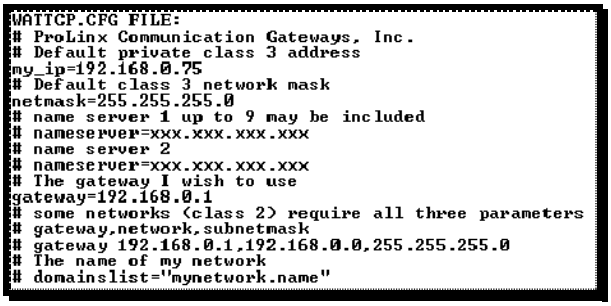
Press **[S]** to transfer the WATTCP.CFG file from the module to your PC.

Press **[Y]** to confirm the file transfer, and then follow the instructions on the computer 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

ress **[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:

LED	Color	Status	Indication
CFG	Green	On	Data is being transferred between the module and a remote terminal using the Configuration/Debug port.
		Off	No data is being transferred on the Configuration/Debug port.
P1	Green	On	Port not used
		Off	Port not used
P2	Green	On	Port not used
		Off	Port not used
APP	Amber	Off	The MVI46-MNETC is working normally.
		On	The MVI46-MNETC program has recognized a communication error.
BP ACT	Amber	On	On when the module is performing a write operation on the backplane.
		Off	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 MVI46-MNETC 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	Not used	0x0001
1	Not used	0x0002
2	Not used	0x0004
3	Not used	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	APP	0x0100
9	CMD ERR	0x0200
10	Not used	0x0400
11	Not used	0x0800
12	Not used	0x1000
13	Not used	0x2000
14	Not used	0x4000
15	Not used	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.

Clearing a Fault Condition

Typically, if the OK LED on the front of the module becomes illuminated red for over ten seconds, a hardware problem has been detected in the module or the program has exited. To attempt to clear the condition:

1. Turn the power to the rack off
2. Remove the card from the rack
3. Make certain the Compact Flash is installed and all jumpers are set correctly
4. Re-insert the card in the rack and turn the power back on
5. Verify the configuration data being transferred to the module from the SLC processor

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

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.

Problem Description	Steps to take
Processor Fault	Confirm that the module is plugged into the slot that has been configured for the MVI46-MNETC module. Confirm that the slot in the rack configuration has been set up correctly.
Processor I/O LED flashes	This indicates there is a problem with backplane communications. Verify that this and all modules in the rack are configured in the processor.
BP ACT LED remains off or blinks slowly	This indicates that backplane transfer operations are failing. Use the Configuration/Debug port facility to check this. To establish backplane communications, verify the following items: ▪ 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.

4.2.1 Ethernet LED Indicators

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

5 Reference

In This Chapter

5.1 Product Specifications

The MVI46-MNETC allows Rockwell Automation® SLC® processors to interface easily with other Modbus compatible devices.

Compatible devices include Modicon PACs, as well as a wide variety of instruments and devices. The module acts as an input/output module between the Modbus network and the Rockwell Automation backplane. The data transfer from the processor is asynchronous from the actions on the Modbus Client-controlled network. A 5000-word register space in the module exchanges data between the processor and the Modbus network.

5.1.1 General Specifications

- Single Slot - 1746 backplane compatible (local or extended I/O rack only - remote rack not supported)
- The module is recognized as an Input/Output module and has access to processor memory for data transfer between processor and module using M0/M1 files
- 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

5.1.2 Modbus TCP/IP

- 10/100 MB Ethernet port
- Module I/O data memory mapping supports up to 5000 registers and is user definable
- ProSoft Configuration Builder (PCB) software supported, a Windows-based graphical user interface providing simple product and network configuration
- Sample Ladder Logic and Add-On Instructions (AOI) are used for data transfer between module and processor and module configuration
- Personality Module (non-volatile CF card) used to store configuration allowing for quick in-the-field product replacement.

5.1.3 Functional Specifications

The MVI46-MNETC will operate on a local or remote rack. This module was created to improve the performance when servers are not needed on a Modbus TCP/IP network. The module supports up to 30 Clients with up to 16 commands for each Client, making it easy to enable/disable the commands from the ladder logic. The Client command control word contains one bit for each command. This module does not support the conditional write command.

- 10/100 MB Ethernet Application port
- Supports Enron version of Modbus protocol for floating-point data transactions
- PCB includes a powerful Modbus network analyzer
- Special functions (Event Commands, Command Control, status, etc.) are supported by message transfer (unscheduled) using the MSG instruction
- Configurable parameters for the Client including a minimum response delay of 0 to 65535 ms and floating-point support
- Supports up to 30 Clients with up to 16 commands for each Client
- All data mapping begins at Modbus register 40001.
- Error codes, network error counters, and port status data available in user data memory

Client Specifications

A port configured as a virtual Modbus Client device on the MVI46-MNETC module actively issues Modbus commands to other nodes on the Modbus network. Additionally, the Client ports have an optimized polling characteristic that polls servers with communication problems less frequently. The SLC processor can be programmed to control the activity on the port by actively selecting commands from the command list to execute or issuing commands directly from the ladder logic.

5.1.4 Hardware Specifications

Specification	Description
Backplane Current Load	800 mA @ 5 Vdc (from backplane)
Operating Temperature	0°C to 60°C (32°F to 140°F)
Storage Temperature	-40°C to 85°C (-40°F to 185°F)
Relative Humidity	5% to 95% (with no condensation)
Shock	30 g operational, 50 g non-operational
Vibration	5 g from 10150 Hz
Processor	Compatible with Rockwell Automation SLC 5/02 M0/M1 capable processors or newer
LED indicators	Module status, Backplane transfer status, Application status, Serial activity (debug port), Ethernet link and activity, and error LED status

Specification	Description
Debug/Configuration port (CFG)	
CFG Port (CFG)	RJ45 (DB-9M with supplied cable) RS-232 only No hardware handshaking
Configuration Connector	RJ45 RS-232 Connector (RJ45 to DB-9 cable shipped with unit)
Application Ports	
Ethernet Port (Ethernet Modules)	RJ45 Connector Link and activity LED indicators Electrical Isolation 1500 Vrms at 50 Hz to 60 Hz for 60 seconds, 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

5.2 Functional Overview

5.2.1 General Concepts

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

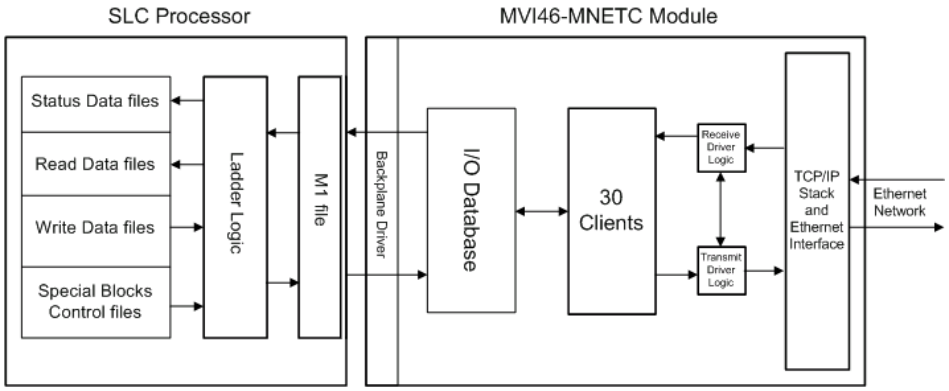
5.2.2 Backplane Data Transfer

The MVI46-MNETC module communicates directly over the SLC backplane. Data is paged between the module and the SLC processor across the backplane using the module's input and output images. 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 M1 file in the processor by the ladder logic.

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 following illustration shows the data transfer method used to move data between the SLC processor, the MVI46-MNETC module and the Modbus TCP/IP Network.



All data transferred between the module and the processor over the backplane is through the M1 file. Ladder logic in the SLC must interface the input and output image data with data defined in the SLC data files. 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

Data contained in this database is paged through the input and output images by coordination of the SLC ladder logic and the MVI46-MNETC module's program. Up to 128 words of data can be transferred between the module and the processor per ladder logic scan. Each image has a defined structure depending on the data content and the function of the data transfer. The following table lists the block numbers used by the module.

Block Range	Descriptions
1000 to 1001	Output Initialization Blocks
2000 to 2029	Controls Modbus Commands as configured in the ladder logic.
5001 to 5016	Controls the Modbus Commands as listed in configuration file.
6000	Controls the selection of individual commands in the list, through set bits.
9990	Set new module IP address
9991	Get module IP address
9998	Warm-boot control block
9999	Cold-boot control block

The Block numbers represent data codes that the Module will recognize as instruction codes for performing a specific function.

The following topics describe these groups of blocks in more detail.

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 reading and writing of the M1 files. No special block number is required to perform the normal data transfer.

The following topics describe the structure and function of each block.

5.2.3 Special Function Blocks

Special function blocks are optional blocks used to request special tasks from the module. The current version of the software supports the following special function blocks:

- Initialize Output Data block
- Event Command block
- Command Control blocks
- Command Control Bits
- Set new module IP address
- Get module IP address
- Warm-boot
- Cold-boot

Initialize Output Data Blocks (1000 and 1001)

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

Block Request from Module to Processor

Word Offset	Description	Length
5000	Value 1000. The module is requesting initialization data.	1

The Ladder logic in the processor must recognize these blocks and place the correct information in the output image to return to the module. The following table describes the format of the returned write block.

Block Response from Processor to Module

Word Offset	Description	Length
5000	Value 1001. The SLC is responding with initialization data.	1

Event Command Blocks (2000 to 2029)

Use blocks 2000 to 2029 to modify the Modbus Command parameters in the SLC's register files dynamically without the need to modify and reload the Configuration file MNET.CFG.

Event Request description (Write Block)

Event Command blocks send Modbus TCP/IP commands directly from the ladder logic to one of the Clients on the module. The following table describes the format for these blocks.

Block Request from Processor to Module

Word Offset	Description	Length
5000	2000 to 2029 (last digits indicate which client to utilize)	1
5001 to 5004	IP Address	4
5005	Service Port	1
5006	Slave Address	1
5007	Internal DB Address	1
5008	Point Count	1
5009	Swap Code	1
5010	Modbus Function Code	1
5011	Device Database Address	1

Use the parameters passed with the block to 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 (Modbus API for network communications) 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 containing the data to transfer.
- The **Point Count** parameter defines the quantity of points or registers to transfer.
- 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 server device containing the data to transfer.

Event Response description (Read Block)

When the module receives the block, it will process it and place it in the command queue. The third word 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 client is full (16 commands for each queue).

The following table describes the format for this block.

Block Response from Module to Processor

Word Offset	Description
5000	0 when Block execution is completed
5001	2000 to 2029. The last two digits indicate which Client was used.
5002	0 = Event Request Failed. 1 = Event Request Succeeded.

Modbus Command Control Blocks (5001 to 5016)

Blocks 5001 to 5016 enable the user to list as many Modbus Commands in the configuration file MNET.CFG as desired (maximum quantity = 16 per Client), then dynamically set the command execution quantity per client.

Command Request description (Write Block)

Command Control blocks place commands in the command list into the command queue. The Client has a command queue of up to 16 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.

Note: The Enable parameter in the configuration file's Modbus Command list must be set to either 0 or 1 and no other value. It does not matter if the value is set to 0 or 1 because the execution of the command is controlled by Blocks 5001 to 5016 and by Block 6000 in ladder logic.

With this Command Control block, the commands may be placed in the command queue with an Enable parameter set to zero. One to 16 commands can be placed in the command queue with a single request. The following table describes the format for this block.

Block Request from Processor to Module

Word Offset	Description	Length
5000	5001 to 5016	1
5001	Client index to utilize [00 to 29]	1
5002	Command index [00]	1
5003	Command index [01]	1

Word Offset	Description	Length
5004	Command index [02]	1
5005	Command index [03]	1
5006	Command index [04]	1
5007	Command index [05]	1
5008	Command index [06]	1
5009	Command index [07]	1
5010	Command index [08]	1
5011	Command index [09]	1
5012	Command index [10]	1
5013	Command index [11]	1
5014	Command index [12]	1
5015	Command index [13]	1
5016	Command index [14]	1
5017	Command index [15]	1

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 15 and correspond to the module's command list entries.

Command Response description (Read Block)

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.

Block Response from Module to Processor

Word Offset	Description	Length
5000	0 when Block execution is completed	1
5001	5001 to 5016. The last two digits indicate which Modbus Commands were executed.	1
5002	0 = Event Request Failed. 1 = Event Request Succeeded.	1

Command Control Bits Block (6000)

Use block 6000 to list up to 16 Modbus Commands in the configuration file MNET.CFG, then dynamically set the specific commands to execute per Client. The Modbus command execution will be continuous and sequential by the Client order and within each Client by the Command sequence order as listed in the configuration file. To dynamically modify the Client number and/or the Modbus Commands to execute from the list, modify the bit structure in the Client assigned registers and then re-activate the Block 6000.

Command Control Bits Request description (Write Block)

The Command Control Bits block places commands from the command list into the command queue. The Client has a command queue of up to 16 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 Modbus Command List section of the configuration file.

Note: The Enable parameter in the Modbus Command List section of the configuration file must be set to either 0 or 1 and no other value, otherwise the configuration file will not be properly read when the file is transferred to the module. The value can be either 0 or 1 because the execution of the command is controlled by Blocks 5001 to 5016 and by Block 6000.

The following table describes the format for this block.

Block Request from Processor to Module

Word Offset	Description	Length
5000	6000	1
5001 to 5030	Command Control Bits for clients [00 to 29]	30

Words 5001 to 5030 each contain set bits that correspond to the Command sequence position for each client in the configuration file.

For example, when Client 17 has in its configuration list all 16 Modbus commands and it is desired to execute only commands indexed 1, 8, 10 and 13, then word offset 5018 will have the following bits set:(MSB 0010 0101 0000 0010 LSB).

Command Control Bits Response description (Read Block)

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.

Block Response from Module to Processor

Word Offset	Description	Length
5000	0 when Block execution is completed	1
5001	6000 when Block execution is activated.	1
5002 to 5031	Command Control Bits for clients [00 to 29] that have been executed.	30

Set New Module IP Address Block (9990)

Block Request from Processor to Module

Word Offset	Description	Length
5000	9990	1
5001	First digit of dotted IP address	1
5002	Second digit of dotted IP address	1
5003	Third digit of dotted IP address	1
5004	Last digit of dotted IP address	1

Block Response from Module to Processor

Word Offset	Description	Length
5000	0 when completed.	1
5001	Write Block ID [9990]	1
5002	First digit of dotted IP address	1
5003	Second digit of dotted IP address	1
5004	Third digit of dotted IP address	1
5005	Last digit of dotted IP address	1

Get Module IP Address Block (9991)

Block Request from Processor to Module

Word Offset	Description	Length
5000	9991	1

Block Response from Module to Processor

Word Offset	Description	Length
5000	0 when complete.	1
5001	Write Block ID [9991]	1
5002	First digit of dotted IP address	1
5003	Second digit of dotted IP address	1
5004	Third digit of dotted IP address	1
5005	Last digit of dotted IP address	1

Warm Boot Block (9998)

This block is sent from the SLC processor to the module when the module is required to perform a warm-boot operation. This block is commonly sent to the module whenever data modifications are made in the data file register values. This will cause the module to read the new data file values and to restart. The following table describes the structure of the Warm Boot block.

Block Request from Processor to Module

Word Offset	Description	Length
5000	9998	1

Cold Boot Block (9999)

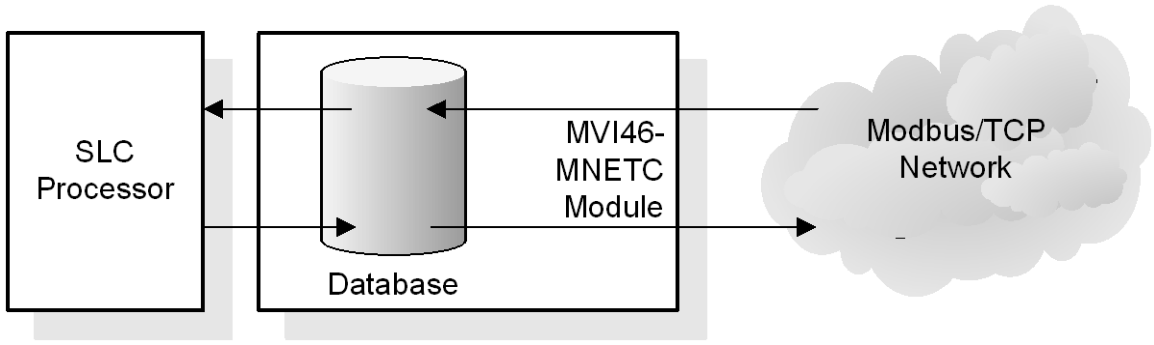
This block is sent from the SLC processor to the module when the module is required to perform the cold boot (hardware reset) operation. This block is sent to the module when a hardware problem is detected by the ladder logic that requires a hardware reset. The following table describes the structure of the Cold Boot block.

Block Request from Processor to Module

Word Offset	Description	Length
5000	9999	1

5.2.4 Data Flow between MVI46-MNETC Module and SLC Processor

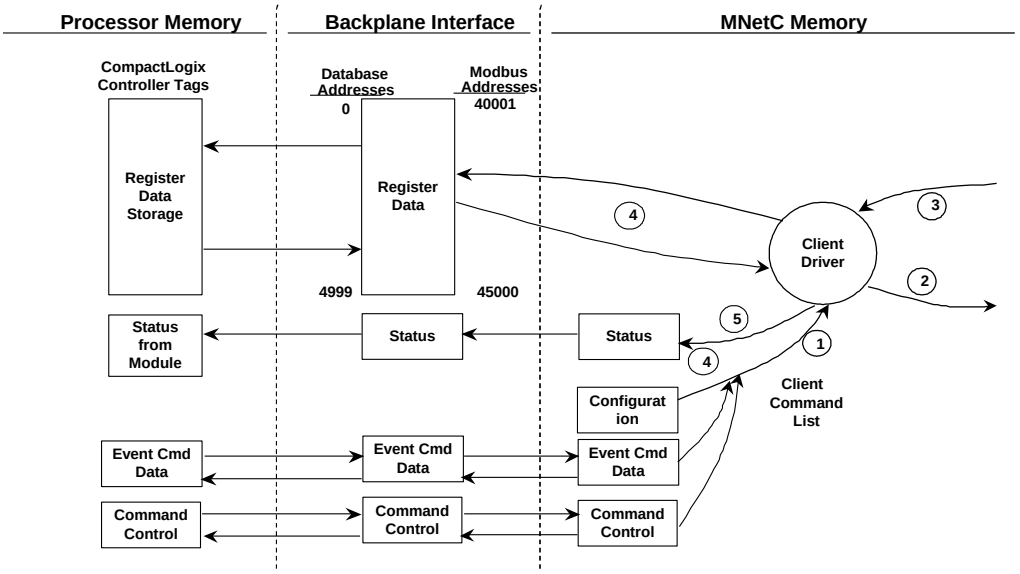
The following topics describe the flow of data between the two pieces of hardware (SLC processor and MVI46-MNETC module) and other nodes on the Modbus TCP/IP network. The module contains up to 30 clients, which can generate either MBAP (Modbus API for network communications) or MNET requests dependent on the service port selected in the command.



The following topics discuss the operation of the client drivers.

Client Driver

In the client driver, the MVI46-MNETC module issues read or write commands to servers on the Modbus TCP/IP network using 30 simulated clients. These commands are user configured in the module via the Client Command List for each client received from the module's configuration file (MNET.CFG) or issued directly from the SLC 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 in the module's configuration file. 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 Modbus Command List. These values are used by the driver to determine the type of commands to issue 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 SLC 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 16 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 [controlled by SLC Data files exclusively]).
- IP address and service port to connect to on the remote server
- Slave Node Address
- Command Type - Read or Write up to 125 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 125
- Poll Delay - 1/10th seconds

Client Command Errors

You can use the Client Command Error Pointer, which is configured for each client in the MNET.CFG file. This pointer references the 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
4110	Client 0 Command Error
4140	Client 1 Command Error
4170	Client 2 Command 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 MVI46-MNETC module has the following functional communication connections installed:

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

5.3.1 Ethernet Connection

The MVI46-MNETC 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 MVI46-MNETC module is NOT compatible with Power Over Ethernet (IEEE802.3af / IEEE802.3at) networks. Do NOT connect the module to Ethernet devices, hubs, switches or networks that supply AC or DC power over the Ethernet cable. Failure to observe this precaution may result in damage to hardware, or injury to personnel.

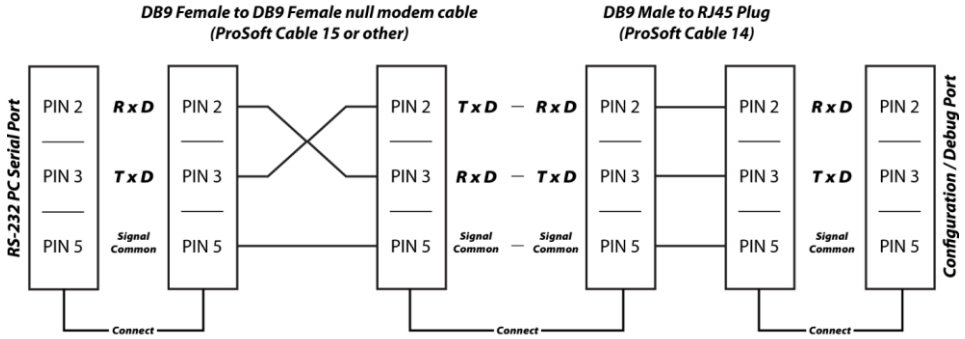
Important: The module requires a static (fixed) IP address that is not shared with any other device on the Ethernet network. Obtain a list of suitable IP addresses from your network administrator BEFORE configuring the Ethernet port on this module.

Ethernet Port Configuration - wattcp.cfg

The wattcp.cfg file must be set up properly in order to use a TCP/IP network connection. You can view the current network configuration in *Viewing the WATTCP.CFG File on the module (page 57) from the Diagnostics menu. From the Main menu, select* **[@]** (Network Menu) and **[V]** (View) options when connected to the Ethernet or Configuration/Debug port. For more information on serial port access, see the chapter on Diagnostics and Troubleshooting (page 44).

5.3.2 RS-232 Configuration/Debug Port

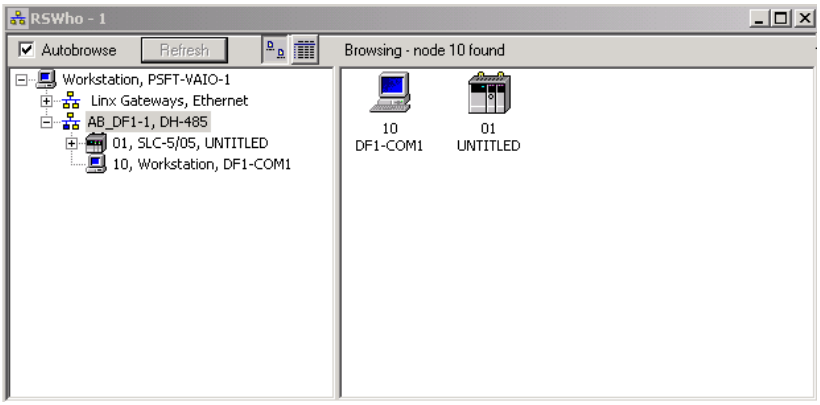
This port is physically an RJ45 connection. An RJ45 to DB-9 adapter cable is included with the module. This port permits a PC-based terminal emulation program to view configuration and status data in the module and to control the module. The cable pinout for communications on this port is shown in the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



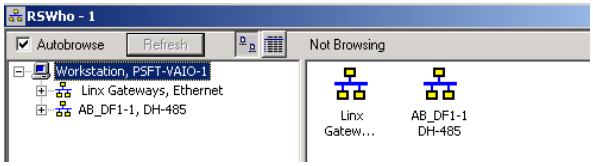
Disabling the RSLinx Driver for the Com Port on the PC

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

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



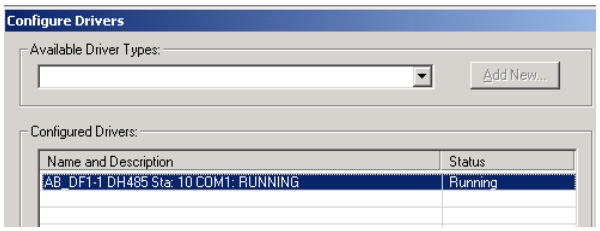
3 Notice how the DF1 driver is opened, and the driver is looking for a processor on node 1. If the network is being browsed, then you will not be able to stop this driver. To stop the driver your *RSWho* screen should look like this:



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

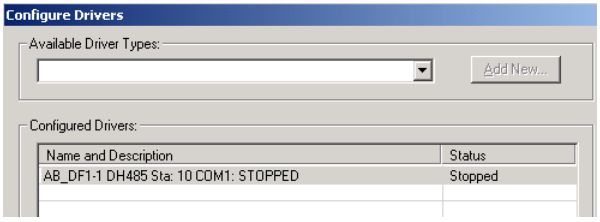
4 When you have verified that the driver is not being browsed, go to **COMMUNICATIONS** > **CONFIGURE DRIVERS**.

You may see something like this:



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

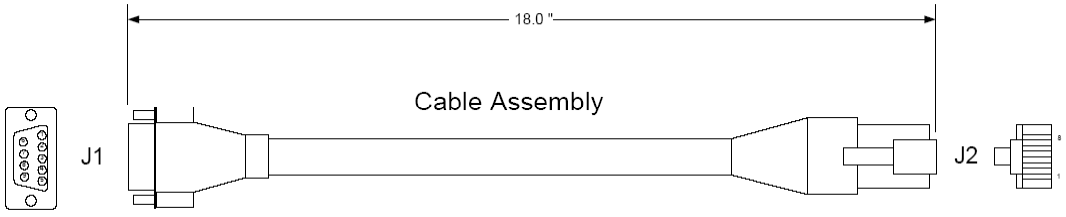
5 After you have stopped the driver you will see the following.

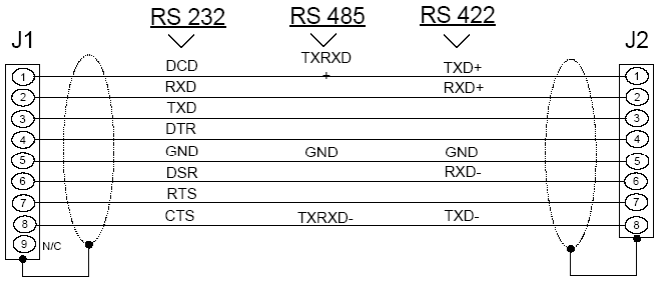


6 You may now use the com port to connect to the debug port of the module.

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

5.3.3 DB9 to RJ45 Adaptor (Cable 14)





Wiring Diagram

5.4 MVI46-MNETC Status Data Definition

This section contains a description of the members present in the status data object. This data is transferred from the module to the processor as part of the read data area when the block transfer interface is used.

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.

5.4.1 MNETC Client x Error/Status Data

Offset	Content	Description
0	Client Cmd Request	This value is incremented each time a command request is issued.
1	Client Cmd Response	This value is incremented each time a command response is received.
2	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.
3	Client Request Count	This value is incremented each time a request message is issued.
4	Client Response Count	This value is incremented each time a response message is received.
5	Client Error Sent Count	This value is incremented each time an error is sent from the Client.
6	Client Error Received Count	This value is incremented each time an error is received from a remote unit.
7	Client Cfg Error Word	This word contains a bitmap that defines configuration errors in the configuration file for the client.
8	Client Current Error Code	This value corresponds to the current error code for the Client.
9	Client Last Error Code	This value corresponds to the last error code recorded for the Client.

6 Support, Service & Warranty

In This Chapter

6.1 Contacting Technical Support

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

1 Product Version Number
2 System architecture
3 Network details

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

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

Note: *For technical support calls within the United States, ProSoft's 24/7 after-hours phone support is available for urgent plant-down issues. Detailed contact information for all our worldwide locations is available on the following page.*

Internet	Website: www.prosoft-technology.com E-mail address: support@prosoft-technology.com
Asia Pacific (location in Malaysia)	Tel: +603.7724.2080 E-mail: asiapc@prosoft-technology.com Languages spoken include: Chinese, English
Asia Pacific (location in China)	Tel: +86.21.5187.7337 x888 E-mail: asiapc@prosoft-technology.com Languages spoken include: Chinese, English
Europe (location in Toulouse, France)	Tel: +33 (0) 5.34.36.87.20 E-mail: support.EMEA@prosoft-technology.com Languages spoken include: French, English
Europe (location in Dubai, UAE)	Tel: +971-4-214-6911 E-mail: mea@prosoft-technology.com Languages spoken include: English, Hindi
North America (location in California)	Tel: +1.661.716.5100 E-mail: support@prosoft-technology.com Languages spoken include: English, Spanish
Latin America (Oficina Regional)	Tel: +1-281-2989109 E-Mail: latinam@prosoft-technology.com Languages spoken include: Spanish, English
Latin America (location in Puebla, Mexico)	Tel: +52-222-3-99-6565 E-mail: soporte@prosoft-technology.com Languages spoken include: Spanish
Brasil (location in Sao Paulo)	Tel: +55-11-5083-3776 E-mail: brasil@prosoft-technology.com Languages spoken include: Portuguese, English

6.2 Warranty Information

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

Documentation is subject to change without notice.

Index