ProSoft
TECHNOLOGY

Where Automation Connects.

inRAx®

MVI56-PDPMV1
ControlLogix Platform
PROFIBUS DPV1 Master

March 25, 2011

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® Product Documentation

In an effort to conserve paper, ProSoft Technology no longer includes printed manuals with our product shipments. User Manuals, Datasheets, Sample Ladder Files, and Configuration Files are provided on the enclosed CD-ROM in Adobe® Acrobat Reader file format (.PDFs). These product documentation files may also be freely downloaded from our web site: 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.

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; 3 mA @ 24 Vdc
- Operating Temperature: 32 °F to 140 °F (0 °C to 60 °C)
- Storage Temperature: -40 °F to 185 °F (-40 °C to 85 °C)
- Shock: 30 g operational; 50 g non-operational; Vibration: 5 g from 10 Hz to 150 Hz
- Relative Humidity 5% to 95% (with no condensation)
- 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

\<Ex\>
II 3 G
Ex nA IIc T6 X
0 °C <= Tamb <= 60 °C

Agency Approvals and Certifications

CE	EMC-EN61326-1:2006; EN61000-6-4:2007
CB Safety	CA/10533/CSA IEC 61010-1 Ed. 2, CB 243333-2056722 (2090408)
GOST-R	EN61010
CSA	61010
DNV	DET NORSKE VERITAS Test 2.4
Lloyds	Lloyds Register Test Specification Number 1,2002
ATEX	EN 60079-0:July 2006; EN60079-15:Oct 2005

243333   ME06

Contents

Guide to the MVI56-PDPMV1 User Manual

Function		Section to Read	Details
Introduction (Must Do)	→	Start Here (page 11)	This section introduces the customer to the module. Included are: package contents, system requirements, hardware installation, and basic configuration.
Diagnostic and Troubleshooting	→	Diagnostics and Troubleshooting (page 177)	This section describes Diagnostic and Troubleshooting procedures.
Reference Product Specifications	→	Reference (page 197) Product Specifications (page 198)	These sections contain general references associated with this product and its Specifications..
Support, Service, and Warranty Index	→	Support, Service and Warranty (page 245) Index	This section contains Support, Service and Warranty information. Index of chapters.

1   Start Here

In This Chapter

For most applications, the installation and configuration steps described in this section will work without additional programming. ProSoft Technology strongly recommends that you complete the steps in this chapter before developing a custom application.

After you have verified that the module is installed and communicating successfully with the processor and the PROFIBUS network.

1.1   System Requirements

The MVI56-PDPMV1 module requires the following minimum hardware and software components:

- Rockwell Automation ControlLogix processor, with compatible power supply and one free slot in the rack, for the MVI56-PDPMV1 module. The module requires 800 mA of available power.
- Rockwell Automation RSLogix 5000 programming software version 2.51 or higher.
- Rockwell Automation RSLinx communication software
- Pentium® II 450 MHz minimum. Pentium III 733 MHz (or better) recommended
- Supported operating systems:
    o Microsoft Windows XP Professional with Service Pack 1 or 2
    o Microsoft Windows 2000 Professional with Service Pack 1, 2, or 3
    o Microsoft Windows Server 2003
- 128 Mbytes of RAM minimum, 256 Mbytes of RAM recommended
- 100 Mbytes of free hard disk space (or more based on application requirements)
- 256-color VGA graphics adapter, 800 x 600 minimum resolution (True Color 1024 x 768 recommended)
- CD-ROM drive
- Dial-up Internet connection (Broadband connection recommended)
- Approved PROFIBUS cabling and connectors.
- Small flat blade screwdriver to secure the PROFIBUS connector to the module

Note: You can install the module in a local or remote rack. For remote rack installation, the module requires Ethernet I/P or ControlNet access to the processor.

1.2   Package Contents

The following components are included with your MVI56-PDPMV1 module, and are all required for installation and configuration.

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

Qty.	Part Name	Part Number	Part Description
1	MVI56-PDPMV1 Module	MVI56-PDPMV1	PROFIBUS DPV1 Master
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
1	inRAx Solutions CD		Contains sample programs, utilities and documentation for the MVI56-PDPMV1 module.

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

1.3   Installing ProSoft Configuration Builder Software

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

To install ProSoft Configuration Builder from the ProSoft Technology website

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

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

To install ProSoft Configuration Builder from the Product CD-ROM

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

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

1.4   Installing the Module in the Rack

If you have not already installed and configured your ControlLogix processor and power supply, please do so before installing the MVI56-PDPMV1 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.

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

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

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

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

1.5   Choosing Sample Ladder Logic For Your Application

The sample program for your MVI56-PDPMV1 module includes custom tags, data types and ladder logic for data I/O, status and alarm monitoring, and mailbox messaging. For most applications, you can run the sample ladder program without modification, or, for advanced applications, you can incorporate the sample program into your existing application.

NOTE: ControlLogix firmware versions 15 or earlier do not support Add-On Instructions (AOIs). For these applications, you will need to use the plain ladder logic sample programs.
For applications with ControlLogix firmware version 16 or newer, you can use the supplied AOI samples, which encapsulate the ladder logic, controller tags, and user-defined data types into one convenient import file.

SPECIAL NOTE: For AOI applications created using ProSoft Configuration Builder (PCB) version 2.2.0 or later, the following process of sample ladder selection is not necessary. PCB versions 2.2.0 and up allow you to export custom-made RSLogix 5000 .L5X Rung Import files that precisely match the module's configuration. Selecting and modifying sample logic will not be required if you use PCB 2.2.0. or later and follow the procedure found in the topic, Export the Processor Files for PCB v2.2.0 or later (page 107). For more details, see the section, Using the MVI56-PDPMV1 Add-On Instructions (page 24).

The inRAx Solutions CD provides several versions of the sample ladder logic. The version number appended to the sample file name (for example, "_v13" or "_v15") indicates which ControlLogix firmware version the sample program was created to match. It is best if the ControlLogix processor firmware version and sample program version match. However, RSLogix 5000 can convert an older version sample to a later ControlLogix firmware version, if necessary.

The following criteria determine which sample program to select:

- o   Processor Firmware Version
- o   MVI56-PDPMV1 module firmware version
- o   MVI56-PDPMV1 mode used (FLEX or LEGACY)

Note: FLEX mode is available only for module firmware versions 1.21 or later.

The next few topics will help you determine which sample programs will work best for your application.

1.5.1 Determining the Firmware Version of Your Processor

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

1 Connect an RS-232 serial cable from the COM (serial) port on your PC to the communication port on the front of the processor.
2 Start RSLogix 5000 and close any existing project that may be loaded.
3 Open the **COMMUNICATIONS** menu and choose **GO ONLINE**. RSLogix will establish communication with the processor. This may take a few moments.
4 When RSLogix has established communication with the processor, the *Connected To Go Online* dialog box will open.
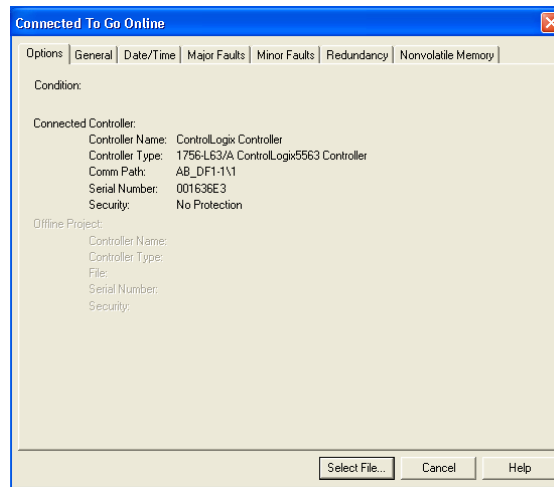
5 In the *Connected To Go Online* dialog box, click the **GENERAL** tab. This tab shows information about the processor, including the *Revision* (firmware) version. In the following illustration, the firmware version is 17.2.
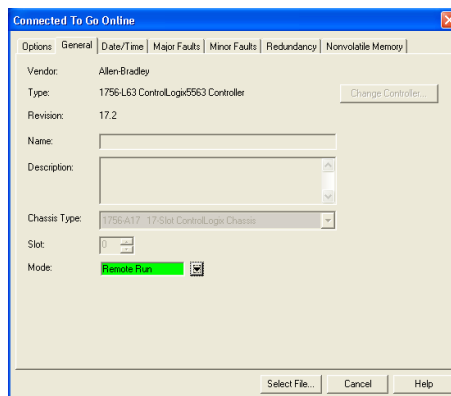
1.5.2 Determining the Firmware Version of the MVI56-PDPMV1 Module

There are two ways to verify the module firmware version:

Checking the Version During Power-Up

When the module powers up, it will search for the ControlLogix processor to establish backplane connectivity. If the correct ladder is not loaded to the processor, or if the module is not located in the configured slot, or the processor is missing, the module will not allow access to the debug menu and will print the message "Waiting for Connection to the processor…"

```
C>pb_msts1
Open Backplane Interface....
Waiting for connection to processor....Waiting for connection t
iting for connection to processor....Waiting for connection to
ing for connection to processor....Waiting for connection to pr
g for connection to processor....Waiting for connection to proc
```

Because debug menu access is not allowed at this point, view the debug menu messages that are printed during power-up for the module firmware version. Refer to the message that appears just before the message "C>pb_msts1":

If the message "MVI56-PDPMV1 Firmware Version = x.xx" appears on the screen as shown in the following illustration, the module firmware is version 1.21 or later.
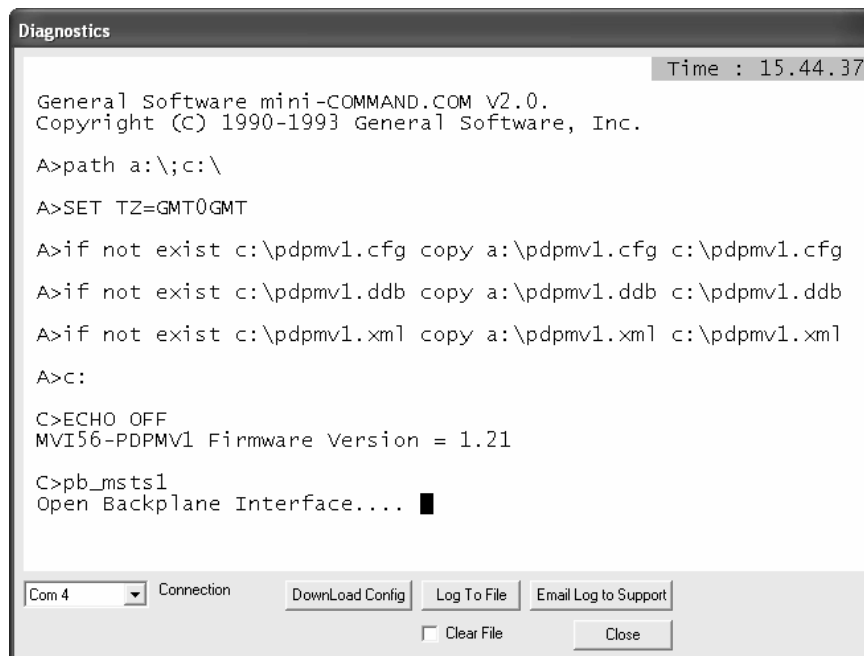
If the message "MVI56-PDPMV1 Firmware Version = x.xx" does not appear on the screen, the module firmware version is 0.30.
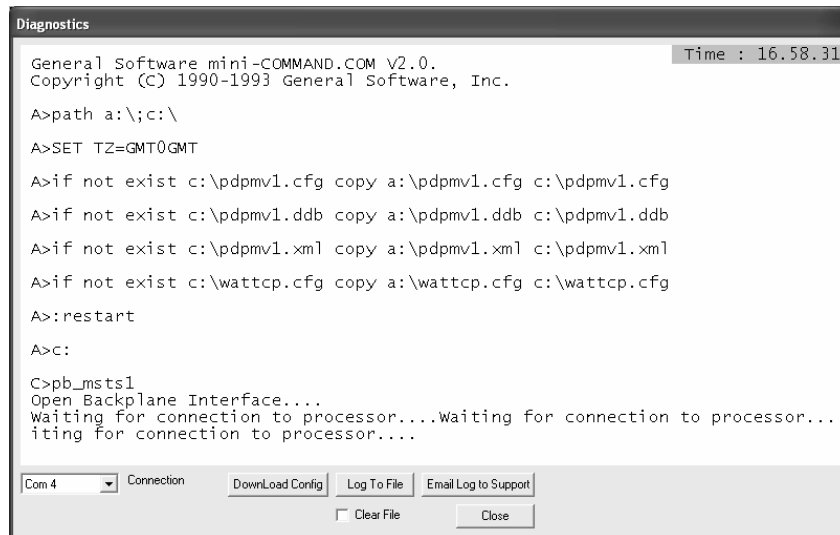
Checking the Version After Power-Up

If the correct sample program (LEGACY or FLEX) is loaded into your processor and the module is located in same slot that was configured, then after module power-up it will be possible to access the debug menu through ProSoft Configuration Builder (PCB).

1   Press **[?]** to display the main menu:
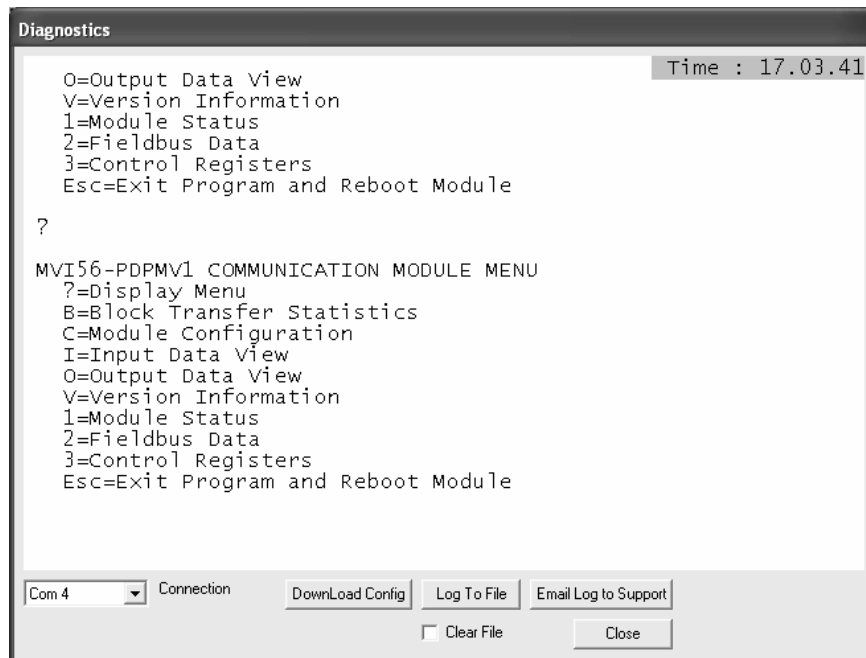
2 Press **[V]** to display the module version information. Verify "SOFTWARE REVISION LEVEL" for the module firmware version (1.21 for the example below):

```
Diagnostics
                                                        Time : 17.03.55
  V

  VERSION INFORMATION:
    MVI56-PDPMV1 COMMUNICATION MODULE
    (c) 1999-2007, ProSoft Technology, Inc.

    PRODUCT NAME CODE        : PDM5
    SOFTWARE REVISION LEVEL  : 1.21
    OPERATING SYSTEM REVISION : 0607
    RUN NUMBER               : 2801
    PROGRAM SCAN COUNTER     : 11078
    FREE MEMORY              : 372064
    BACKPLANE DRIVER VERSION : 1.08
    BACKPLANE API VERSION    : 1.08
    MODULE NAME : MVI56 ProSoft Technology, Inc.
    VENDOR ID   : 309     DEVICE TYPE    : 12
    PRODUCT CODE: 83      SERIAL NUMBER : 00006F59
    REVISION    : 1.01
  MODULE NAME:
  MVI56-PDPMV1


 Com 4        ▼  Connection    DownLoad Config   Log To File   Email Log to Support
                                        ☐ Clear File        Close
```

1.5.3 Determining the Module Mode (LEGACY or FLEX)

There are two versions of the sample program: LEGACY (fixed I/O sizes) and FLEX (variable I/O sizes) versions.

▪ MVI56-PDPMV1 firmware version 0.30 must use the LEGACY sample program and fixed I/O sizes.
▪ MVI56-PDPMV1 firmware version 1.21 or later can use either the LEGACY or FLEX mode sample programs.

For module firmware version 1.21 or later, the **USE LEGACY MODE** configuration parameter in ProSoft Configuration Builder selects which mode you will use. Setting the parameter to **YES** says to use fixed I/O sizes in LEGACY mode. Setting the parameter to **NO** says to use variable I/O sizes in FLEX mode. This step is described in Select the Connection Parameters for the Module (page 69).

LEGACY mode allows backward compatibility with firmware version 0.30 and implements the following features:

▪ Fixed backplane I/O block size (input size = 250 words, output size = 248 words)
▪ Status and Mailbox transferred through regular I/O blocks.

LEGACY mode offers these advantages:

- Works with all module firmware versions; use LEGACY mode for backward compatibility in older applications.
- Since all parameters are fixed, configuration is simpler than FLEX mode.
- Status data is bought across the backplane as I/O data without the need for additional logic to trigger MSG instructions.

FLEX mode implements the following features:

- Supports variable backplane I/O block sizes (input backplane block size can be set from 12 words to 250 words; output backplane block size can be set from 5 to 248 words).
- Status, Slave Diagnostic, and Mailbox data transferred through MSG instructions.
- Supports IOT instruction for immediate write of backplane output.

When transfer speed for smaller data configurations is a priority or when backward compatibility is not required, consider using FLEX mode, which provides these advantages:

- Makes it easier to support installing the module in a remote rack. The flexible backplane I/O block size allows you to adjust the amount of data to transfer for any specific remote rack application and optimize network bandwidth utilization.
- Improves backplane I/O block transfer performance by implementing MSG instructions to retrieve module status. This means regular I/O block transfers are not needed for status data, so process I/O data can be updated more often. The same applies to Mailbox MSG data transfers.
- Improves backplane performance by implementing the IOT instruction for Immediate Output Transfers, which overrides the module RPI time and sends data as fast as possible.

1.5.4 Sample Program Summary

Use the following table to determine which sample program to use for your application.

MVI56-PDPMV1 Version	ControlLogix Version	Use Legacy Mode parameter value	Sample Ladder	Add-On Instruction
0.30	13	-	MVI56PDPMV1_LEGACY_v13.ACD	-
0.30	15	-	MVI56PDPMV1_LEGACY_v15.ACD	-
0.30	16	-	MVI56PDPMV1_LEGACY_v16.ACD	MVI56PDPMV1_LEGACY_v1_3.L5X
1.21 or later	13	Y (LEGACY)	MVI56PDPMV1_LEGACY_v13.ACD	-
1.21 or later	15	Y (LEGACY)	MVI56PDPMV1_LEGACY_v15.ACD	-
1.21 or later	16	Y (LEGACY)	MVI56PDPMV1_LEGACY_v16.ACD	MVI56PDPMV1_LEGACY_v1_3.L5X
1.21 or later	17	Y (LEGACY)	MVI56PDPMV1_LEGACY_v17.ACD	MVI56PDPMV1_LEGACY_v1_3.L5X
1.21 or later	13	N (FLEX)	MVI56PDPMV1_FLEX_v13.ACD	-
1.21 or later	15	N (FLEX)	MVI56PDPMV1_FLEX_v15.ACD	-
1.21 or later	16	N (FLEX)	MVI56PDPMV1_FLEX_v16.ACD	MVI56PDPMV1_FLEX_v1_2.L5X
1.21 or later	17	N (FLEX)	MVI56PDPMV1_FLEX_v17.ACD	MVI56PDPMV1_FLEX_v1_2.L5X

Note: For ControlLogix version 16, you will use the Add-On Instruction (AOI) rung import (.L5X) files, which encapsulate all the required ladder logic, controller tags, and user-defined data types into a single import file; or you can use the version 16 ladder logic sample files (.ACD), which already have the AOI files imported. The plain ladder logic sample files (.ACD) for versions 13 and 15 use the same logic as the AOI versions but display them as standard ladder logic. The separate AOI .L5K file exists to make it easier to import the sample application into an existing project or new project file. You do not need to import the AOI .L5K files into any of the .ACD ladder logic sample programs. ControlLogix versions 15 and earlier do not support Add-On Instructions; so do not try to import the AOI .L5K files into those older versions of RSLogix 5000.
- For ControlLogix firmware version 16, refer to Using the MVI56-PDPMV1 Add-On Instructions (page 24).
- For ControlLogix firmware version 15 or older, Using the MVI56-PDPMV1 Sample Ladder Logic (page 23).

1.6   Using the MVI56-PDPMV1 Sample Ladder Logic

If you will be using ControlLogix firmware versions 16 or higher, you may skip to the section on Using the MVI56-PDPMV1 Add-On Instructions (page 24).

If you will be using ControlLogix firmware versions 13 or 15, then follow this procedure:

1   Copy any of the sample programs you may want to use from the distribution CD in a convenient folder on your local PC hard drive.
2   Start RSLogix 5000 and close any project or programs that may open automatically when the program starts.
3   On the **FILE** menu, click the **OPEN** button.
4   Browse to the location where you saved the sample program files on your local drive and select (left-mouse-click) the one that matches your firmware version of your ControlLogix processor firmware.
5   Left-click the **OPEN** button.
6   RSLogix will load the sample program.

The next step is to configure the correct controller type and slot number for your application. For details, skip to the section on Common Settings for All Sample Programs (page 65).

If you want to add the sample file to an existing application, do the following:

1   Open the RSLogix 5000 project .ACD file you wish to use.
2   In a separate instance of RSLogix 5000, open the sample ladder logic you wish to use.
3   Follow the procedures for adding a Module to the project and setting Connection Parameters, as found in Creating the MVI56-PDPMV1 Module Profile - Legacy (page 25), Creating the MVI56-PDPMV1 Module Profile - Flex (page 50), and Common Settings for All Sample Programs (page 65).
4   Manually copy the user-defined data types from the sample program and paste them into your existing application.
5   Manually copy the controller tags from the sample program and paste them into your existing application.
6   Manually copy the ladder logic tasks and rungs from the sample program and paste them into your existing application.

1.7 Using the MVI56-PDPMV1 Add-On Instruction

Note: This section applies only for applications with ControlLogix firmware version 16 or higher. For earlier firmware versions, refer to Using the MVI56-PDPMV1 Sample Ladder Logic (page 23).

The Add-On Instruction (AOI) is supplied already installed as part of a complete .ACD program file and also as an .L5X ladder import file that can be imported into an existing or new application. Importing the AOI reduces module setup time and simplifies your application logic. The .L5X file automatically imports the following components into an application:

- User-defined data types
- Controller tags and tag arrays
- Add-On Instruction logic
- Ladder rung required to call the Add-On Instruction

This section will show the procedure on how to import the Add-On Instruction into your existing ladder for both module modes (LEGACY or FLEX).

1.7.1 LEGACY Sample Add-On Instruction Import Procedure

Note: This section applies only if your application meets one or more of the following conditions:
- MVI56-PDPMV1 module firmware version 0.30
- MVI56-PDPMV1 module firmware version 1.21 (or later) AND configured in LEGACY mode (**USE LEGACY MODE** = **YES**)

Before You Begin

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

File Name	Description
MVI56PDPMV1_LEGACY_v*.L5X (where * is the current sample program version number)	L5X file containing Add-On Instruction, user-defined data types, controller tags and tag arrays, and ladder logic required to set up the MVI56-PDPMV1 module in Legacy mode

Creating a New RSLogix5000 Project - Legacy

1 Open the *File* menu, and then choose **NEW**.
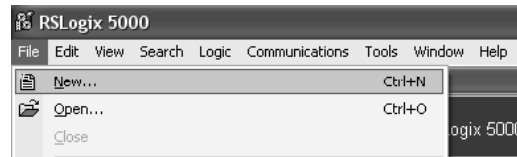
2 Select Revision **16**.
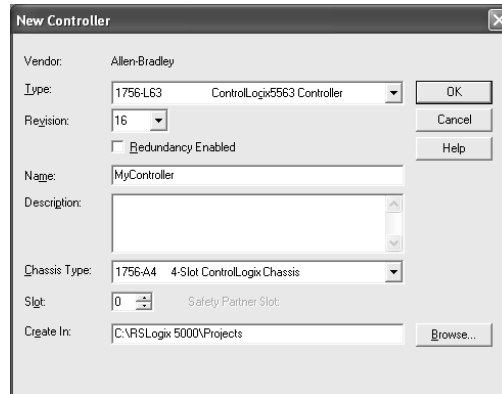
Creating the Module - Legacy
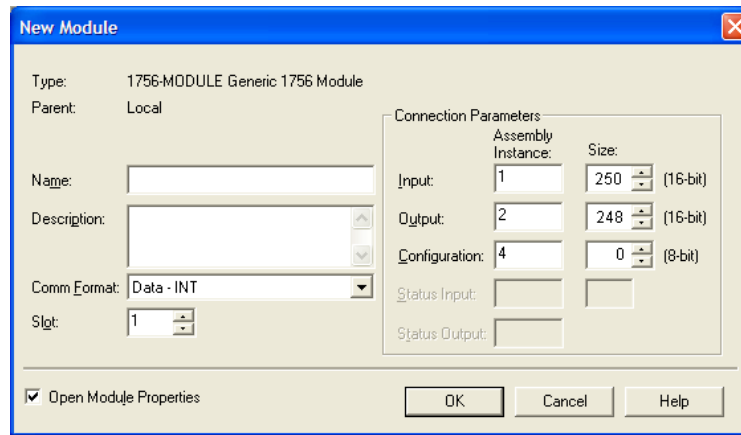
1 Add the MVI56-PDPMV1 module to the project.

In the *Controller Organization* window, select **I/O CONFIGURATION** and click the right mouse button to open a shortcut menu. On the shortcut menu, choose **NEW MODULE**.
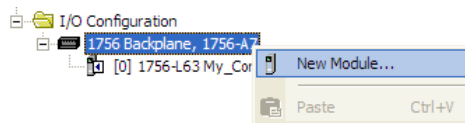
This action opens the *Select Module* dialog box.
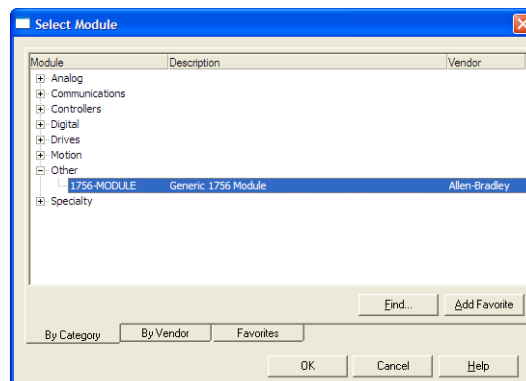
2 Select the **1756-MODULE (GENERIC 1756 MODULE)** from the list and click **OK.**
This action opens the *New Module* dialog box.

3 Set the *Module Properties* values as follows:

Parameter	Value
Name	Enter a module identification string. Example: MVI56PDPMV1
Description	Enter a description for the module. Example: ProSoft communication module for PROFIBUS communication.
Comm Format	Select **DATA-INT**
Slot	Enter the slot number in the rack where the MVI56-PDPMV1 module is located.
Input Assembly Instance	**1**
Input Size	**250**
Output Assembly Instance	**2**
Output Size	**248**
Configuration Assembly Instance	**4**
Configuration Size	**0**

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

Now the MVI56-PDPMV1 module will be visible in the *I/O Configuration* section.

Importing the Ladder Rung - LEGACY

Note: *ProSoft Configuration Builder (PCB), version 2.2.0*, or higher, will allow you to export an .L5X rung import file based on the configuration information you enter in PCB. When using LEGACY mode, these custom export files will be very similar to the provided sample LEGACY AOI Rung Import files.

For additional information on how to export custom-made .L5X files from *PCB*, please see Exporting the Processor Files (page 105).

1 Open your application in RSLogix 5000.
2 Expand the *Tasks* folder, and then expand the *MainTask* folder.
3 On the *MainProgram* folder, click the right mouse button to open a shortcut menu. On the shortcut menu, choose **NEW ROUTINE**.
4 In the *New Routine* dialog box, enter the name and description of your routine, and then click **OK**.
5 Select an empty rung in the new routine, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose **IMPORT RUNG**.
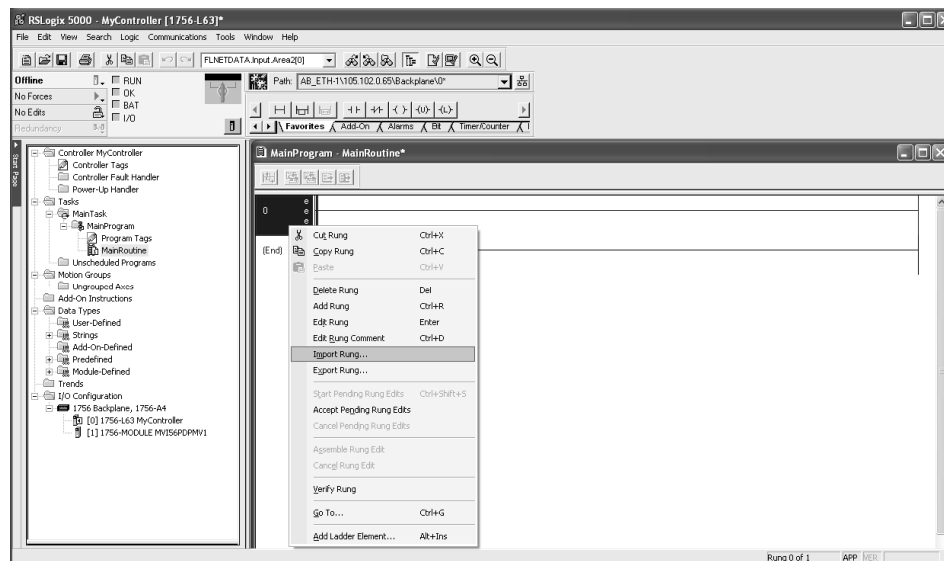
6   Select the MVI*56PDPMV1_LEGACY_v*.ACD* standard sample file, or the
MVI56PDPMV1_LEGACY_AddOn_Rung_v1_2.L5X custom file you exported
from PCB, and click the **IMPORT** button.
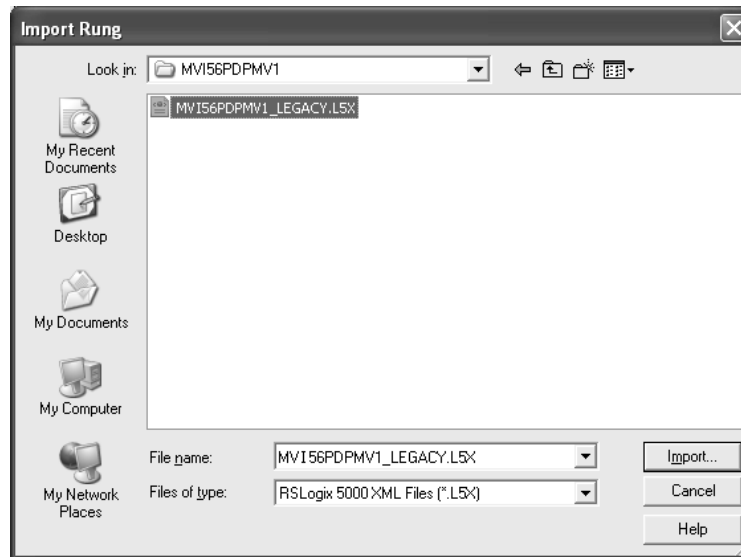
7   A window will be displayed showing the controller tags to be created during
the import procedure.
8   If you are using the module in a different slot (or remote rack) select the
correct connection input and output variables to provide the correct path to
the module. If your module is located in Slot 1 of the local rack, this step is
not required.
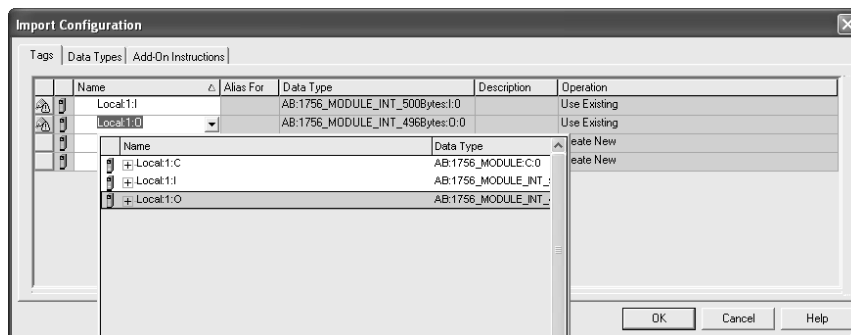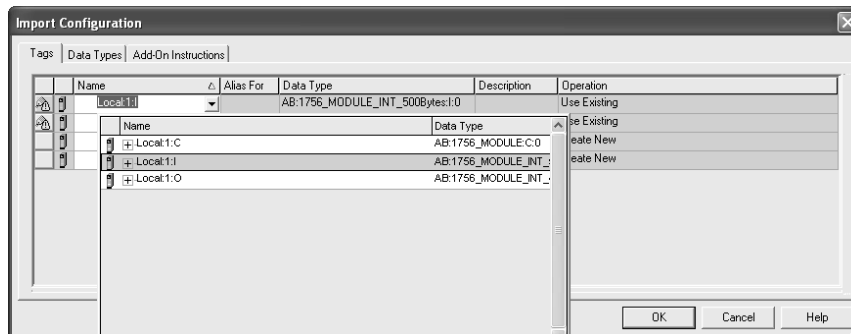
9 Click **OK** to confirm the import. RSLogix will indicate that the import is in progress:
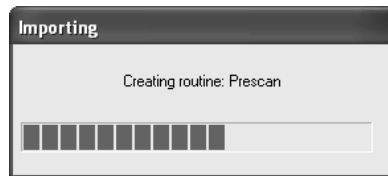
10 When the import is completed, the new rung with the Add-On Instruction will be visible as shown in the following illustration.
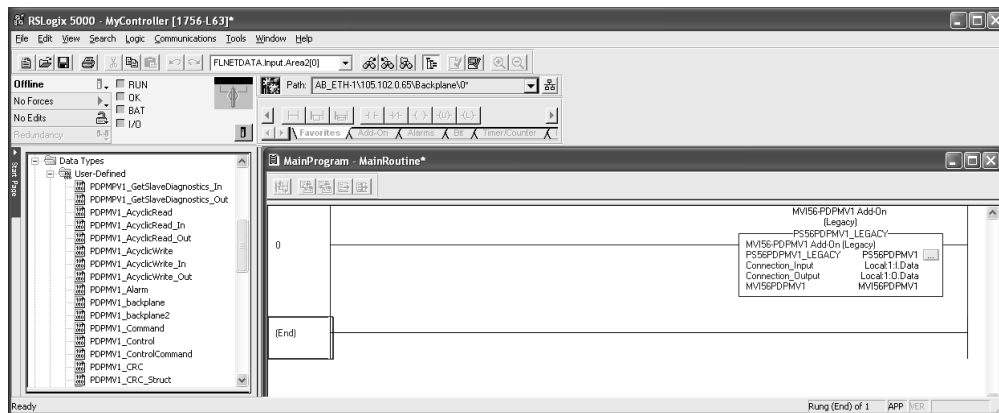
The procedure has also imported new user-defined data types, controller tags and tag arrays, and the Add-On Instruction to be used at your project.

Adding Multiple Modules (Optional) - Legacy

Important: If your application requires more than one MVI56-PDPMV1 module into the same project, follow the steps below.

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

2 Select **1756-MODULE**.

3 Fill the module properties as follows:

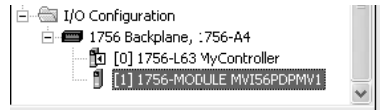| Parameter	Value
Name	Enter a module identification string. Example: MVI56PDPMV1_2
Description	Enter a description for the module. Example: ProSoft communication module for PROFIBUS communication.
Comm Format	Select **DATA-INT**
Slot	Enter the slot number in the rack where the MVI56-PDPMV1 module is located.
Input Assembly Instance	**1**
Input Size	Enter the number of words per backplane input block to copy from the module to the processor. The allowable range is from 12 to 250 words. Each input block will reserve 8 words for general purposes (CRC, block/CIP handshaking). The remainder is reserved for PROFIBUS input data.
Output Assembly Instance	**2**
Output Size	Enter the number of words per backplane output block to copy from the module to the processor. The allowable range is from 5 to 248 words. Each output block will reserve 5 words for general purposes (CRC and block handshaking). The remainder is reserved for PROFIBUS output data.
Configuration Assembly Instance	**4**
Configuration Size	**0**

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

5 Expand the *Tasks* folder, and then expand the *MainTask* folder.
6 On the *MainProgram* folder, click the right mouse button to open a shortcut menu. On the shortcut menu, choose **NEW ROUTINE**.
7 In the *New Routine* dialog box, enter the name and description of your routine, and then click **OK**.

8 Select an empty rung in the new routine, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose **IMPORT RUNG**.

9 Select the file *PS56PDPMV1_LEGACY_v*.L5X*
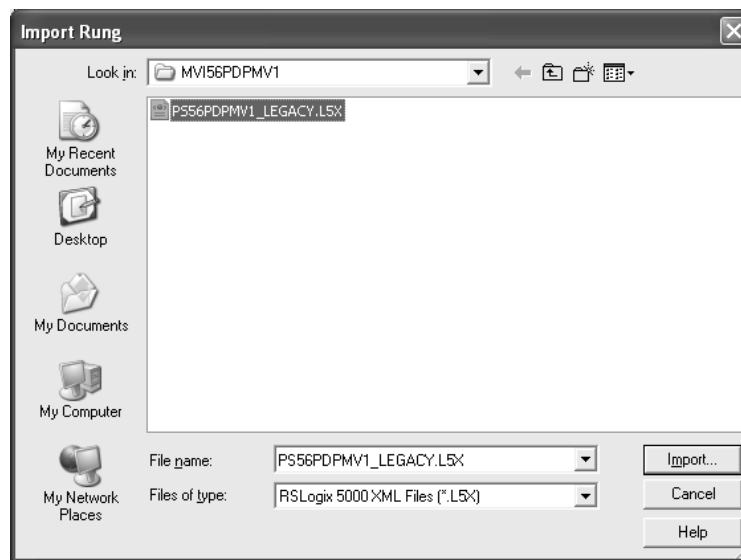
10 The following window will be displayed showing the tags to be imported:
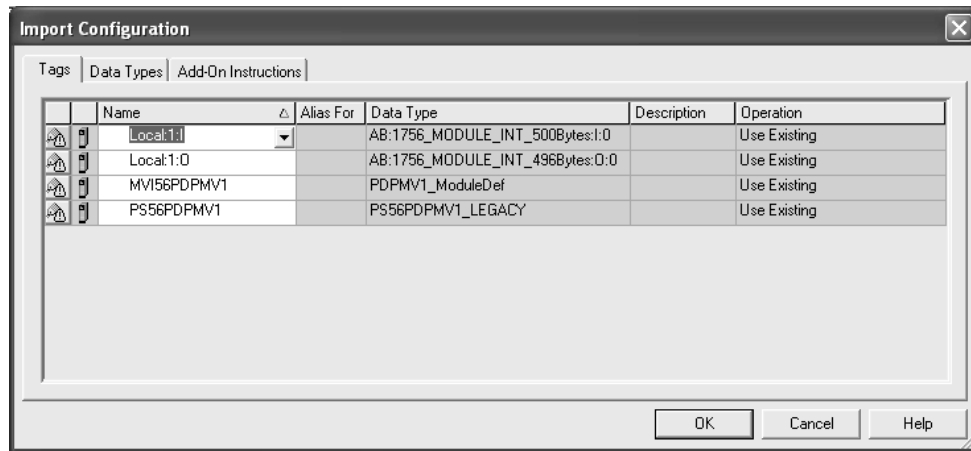
11 Associate the I/O connection variables to the correct module. The default values are *Local:1:I* and *Local:1:O* so these require change.
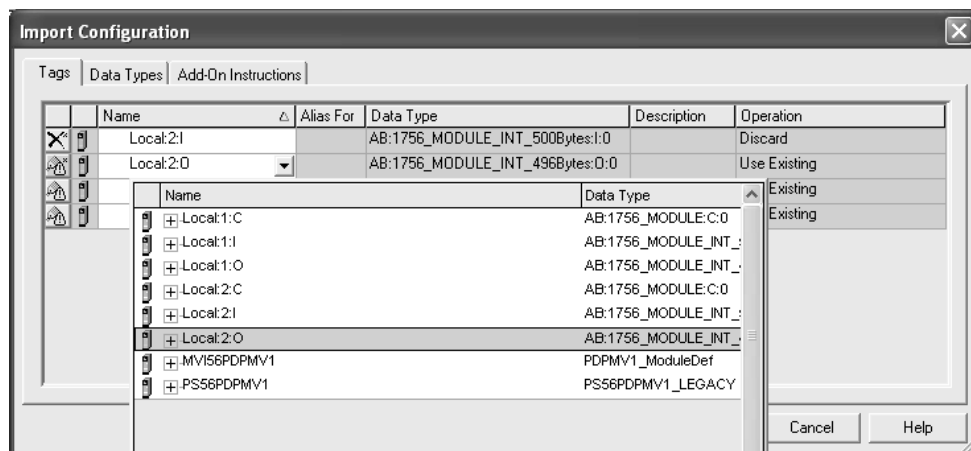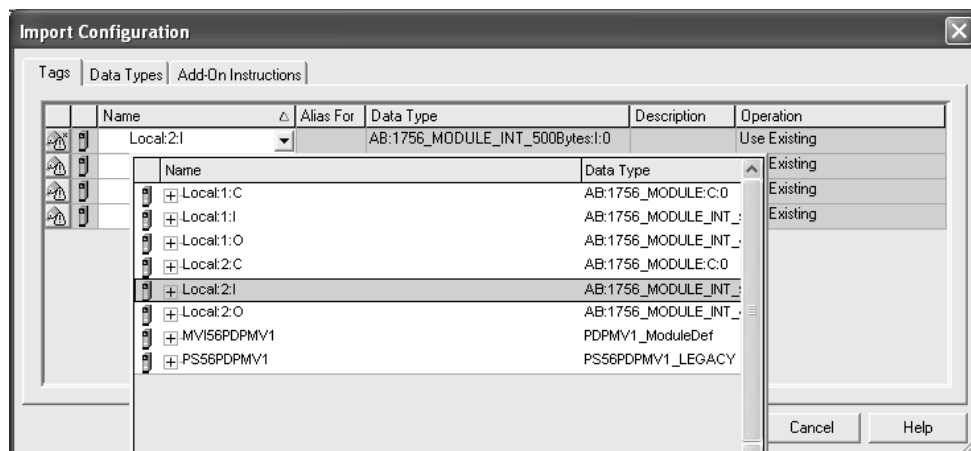
12 Change the default tags MVI56PDPMV1 and PS56PDPMV1 to avoid conflict with existing tags by append the string "_2" (or any other unique identifier string) as follows:
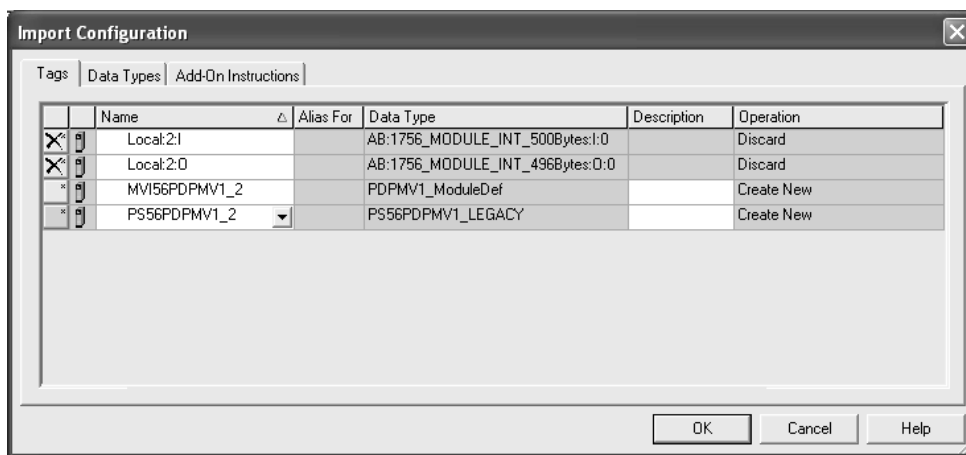
13 Click **OK** to confirm.
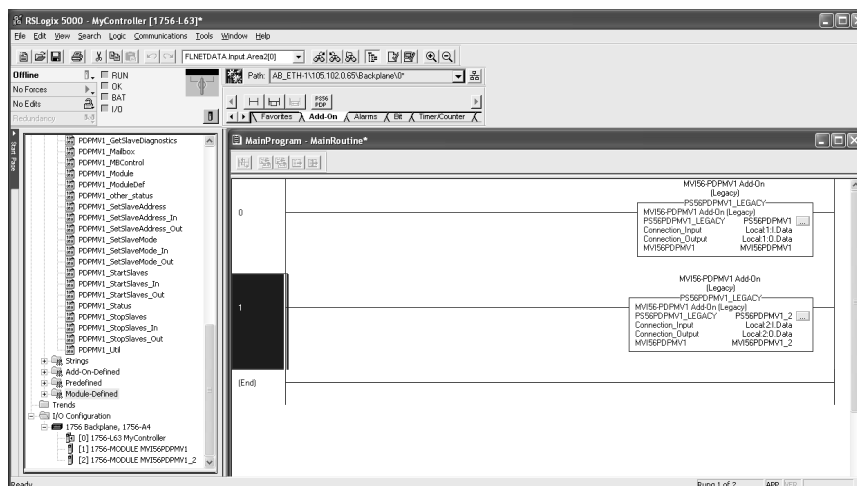
The setup procedure is now complete. Save the project and download the application to your ControlLogix processor.

Adding MVI56-PDPMV1 Logic To Existing Applications

Perform the following steps to add the sample ladder logic into your existing application. The controller tags and user-defined data types provided with the sample ladder logic are critical to incorporating the MVI56-PDPMV1 into your application.

Legacy Mode

> **Note:** You cannot perform this procedure while you are online to the controller.

1 Open your application in RSLogix 5000.
2 In the *Controller Organization* list in RSLogix 5000, click the right mouse button on the *I/O Configuration* option to open a shortcut menu. On the shortcut menu, choose **NEW MODULE**. This action opens the *Select Module Type* dialog box.
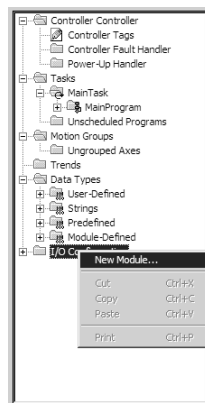
3 In the *Select Module Type* dialog box, select **1756-MODULE (GENERIC 1756 MODULE)** from the list and, and then click **OK**. This action opens the *Module Properties* dialog box.
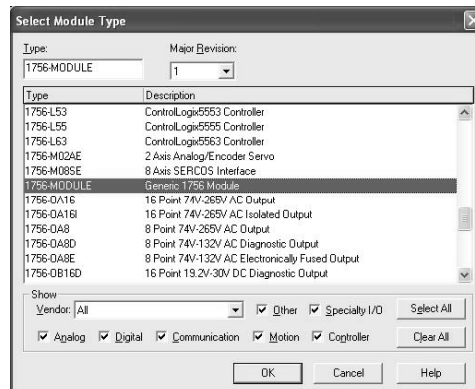
4 In the *Module Properties* dialog box, enter the **NAME, DESCRIPTION** and **SLOT** options for your application, using the examples in the following illustration. You must select the **COMM FORMAT** as **DATA - INT** in the dialog box. Failure to set the correct parameters will result in backplane communication problems between the module and the processor.
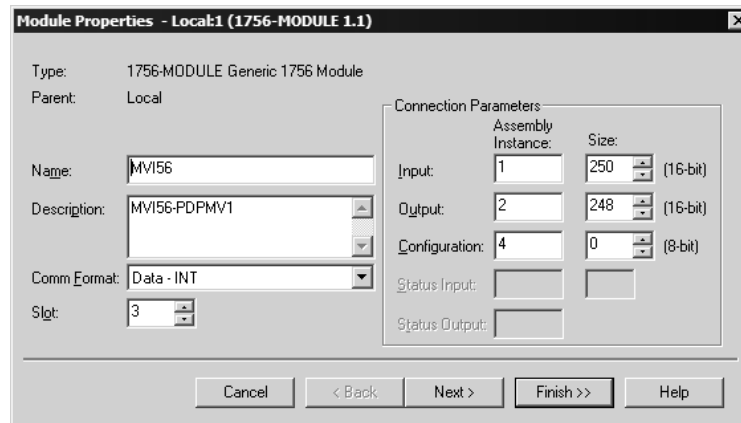
5 Click the **CONNECTION** tab and change the *Requested Packet Interval* to **5.0 MS** as shown in the following illustration.
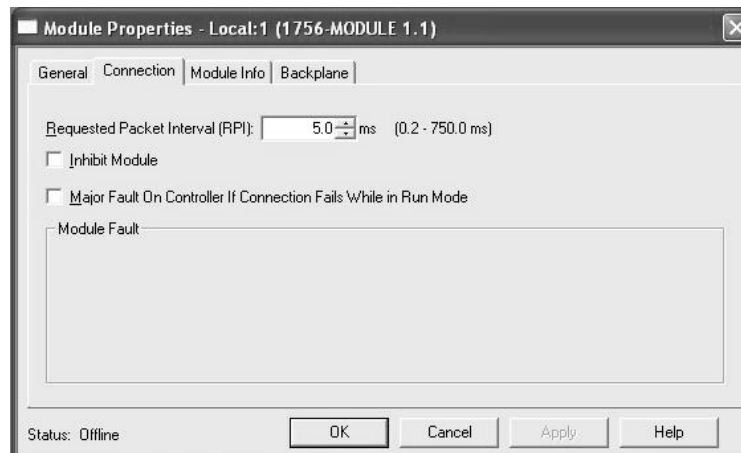
6 Click **FINISH** to save the module into your existing application.
7 Next, open the sample program in RSLogix 5000. Refer to Open the Sample Program in RSLogix (page 16) to determine the correct version of the sample program to use.

8 Copy the ladder logic routines from the sample ladder logic into your existing application. Repeat this step for each subroutine.
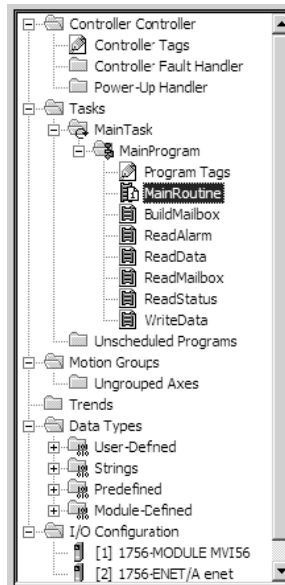
9 Copy and paste each of the user-defined types from the sample ladder logic into your existing application.
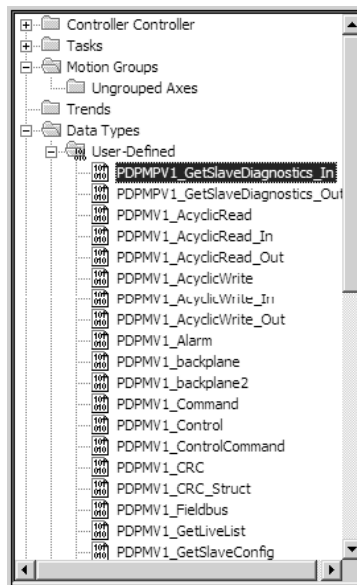
10 Export the controller tags from the sample program (Tools/Export) into a CSV file, and then import the CSV file (Tools/Import) into your existing application.

1.7.2 FLEX Sample Add-On Instruction Import Procedure

Note: This section applies only if the MVI56-PDPMV1 firmware version is 1.21 (or later) AND configured in FLEX mode (**USE LEGACY MODE** = **NO**)

Before You Begin

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

File Name	Description
PS56PDPMV1_FLEX_v*.L5X (where * is the current sample program version number)	L5X file containing Add-On Instruction, user-defined data types, controller tags, and ladder logic required to set up the MVI56-PDPMV1 module

Flexible Connection I/O Length Overview

This section describes the flexible backplane connection I/O concept.

The connection parameters may assume values within the following ranges:

	Minimum Size (16-bit)	Maximum Size (16-bit)	Maximum PROFIBUS data count per block (16-bit)	Assembly Instance
Input	**12**	**250**	[input size - 8]	**1**
Output	**5**	**248**	[bp output size - 5]	**2**
Configuration	**0**	**0**	not used	**4**

For applications where the module will be used in the local rack (the same rack as the ControlLogix processor), use the maximum backplane I/O sizes.

For applications where the module will be used in a remote rack, use the backplane I/O sizes that will provide optimal network performance (PROFIBUS and ControlNet or EtherNet/IP).

The configured sizes will have an impact on the number of blocks required to transfer the entire PROFIBUS I/O database between the ControlLogix processor and the MVI56-PDPMV1 module. Reducing the number of words per backplane block may improve the ControlNet or EtherNet/IP network performance. However, doing so will increase the number of backplane block transfers and have a negative effect on overall backplane data transfer efficiency.

IMPORTANT: ControlLogix Version 16 with FLEX Mode
If you change the connection sizes in the module profile, you must also update the following data types with the new sizes. The connection I/O sizes defined in the module profile must always match the following user-defined data types:
PDPMV1_CONNECTION_INPUT
PDPMV1_CONNECTION_OUTPUT
You will also need to adjust limits in some of the ladder logic or AOI logic. For additional details, see Adjusting the Connection Sizes for Your Application. (page 41)
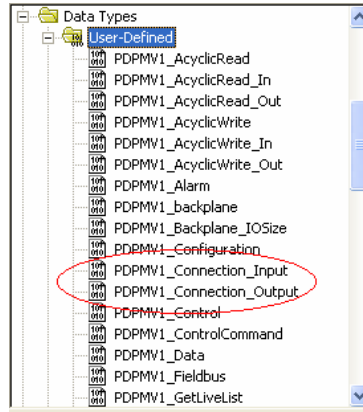
The array sizes must be modified (in the *Data Type* column) to match the connection sizes configured in the module profile. Failure to do so will prevent the module from establishing backplane communication with the ControlLogix processor.

Example 1

For the following example, if the module profile is configured as:
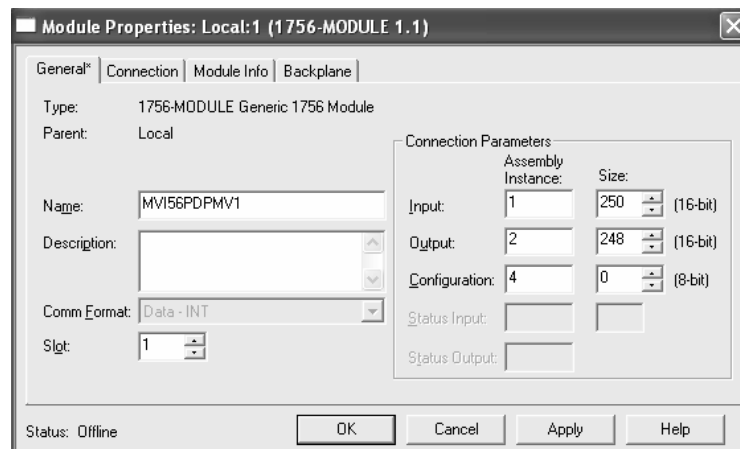
Then, the **PDPMV1_CONNECTION_INPUT** user-defined data type must have an array size of **250** words, as shown in the following illustration.

And, the **PDPMV1_CONNECTION_OUTPUT** user-defined data type must have an array size of **248** words, as shown in the following illustration.

Note: Each backplane block contains certain registers reserved for general purposes (non-PROFIBUS data transfer). The backplane input block reserves 8 words for general purposes and the backplane output block reserves 5 words. You must subtract these reserved words from the total I/O block size to calculate the actual number of PROFIBUS I/O data per backplane I/O block and where the break points between data blocks will be in the PROFIBUS data arrays.

Adjusting the Connection Sizes for Your Application

Note: *ProSoft Configuration Builder (PCB), version 2.2.0*, or higher, will automatically make the following adjustments for you based on the configuration information you enter in PCB. The newer versions of PCB allow you to export custom-made RSLogix 5000 .L5X Add-On Instruction (AOI) Rung Import files that are already tailored to your application-specific configuration. These custom files will be similar to the provided sample AOI Rung Import files but will already include the adjustments shown in the next few topics. If you use the custom import files, you will not need to do any of the following; it will be done for you already. If this is the case, all you need to do is import the custom .L5X file according to the procedure found in Importing the Ladder Rung: FLEX (page 53).

For additional information on how to export custom-made .L5X files from PCB, please see the topic, Exporting the Processor Files (page 105).

The information provided in this topic shows you what kind of modifications are being done automatically for you by PCB. This information can also help you manually edit the standard FLEX sample files to better fit your application or help you determine what kinds of modifications you might need to make to any existing applications you may already be using.

1   The AOI *Connection Input* and *Output* parameters use certain data type sizes that must be set based on how you configured the MVI56-PDPMV1 module.
2   In the **DATA TYPES**/**USER-DEFINED** folder, double-click **PDPMV1_CONNECTION_INPUT** data type.
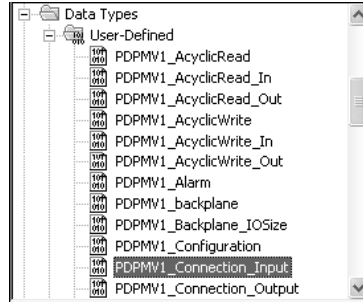
3 Adjust the array size for your application. The default array size is INT[250] (maximum size). This example is for 250 input words.
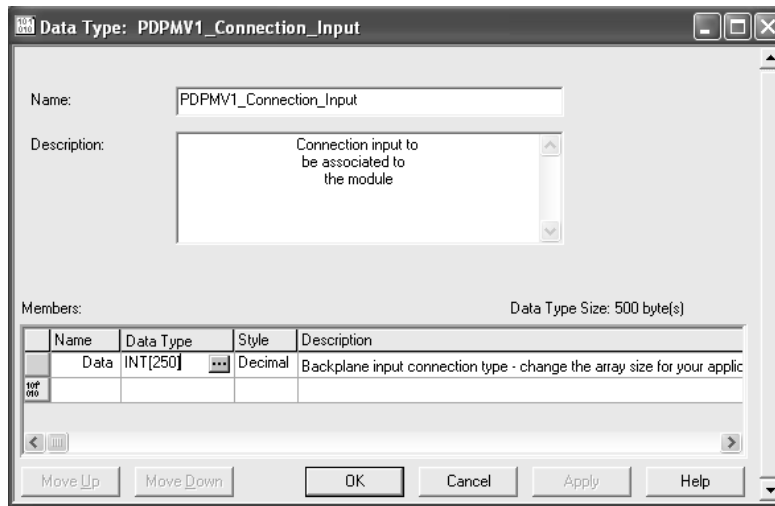
4 In the **DATA TYPES**/**USER-DEFINED** folder, double-click **PDPMV1_CONNECTION_OUTPUT** data type.
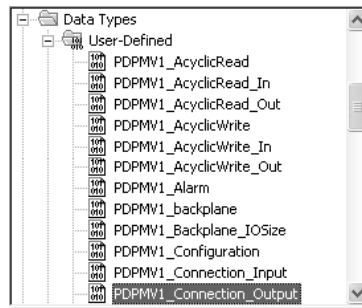
5 Adjust the array size for your application. The default array size is INT[248] (maximum size). This example is for 248 output words.
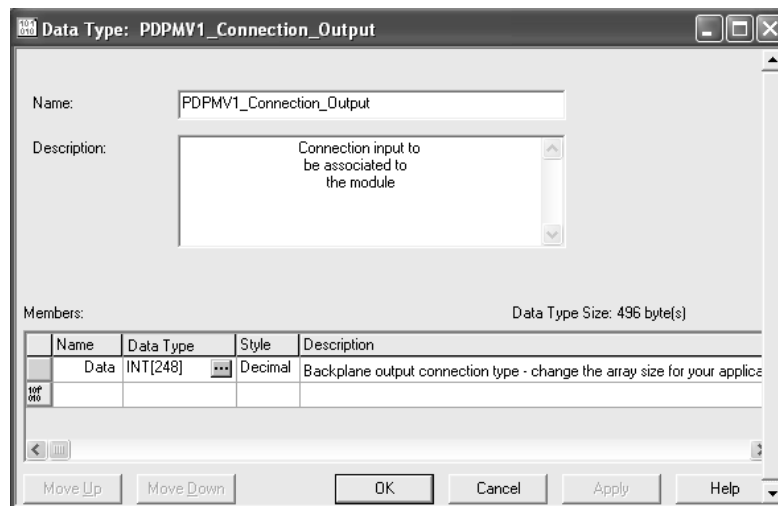
6 The imported ladder rung has also used the default I/O settings (Input Count = 250 words, Output Count = 248 words). If different values are used, make the following changes to the ladder logic:

COP Instruction (input)

Change the Length to match the number of input words configured for the module. The default uses 250 words.
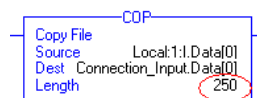
NEQ Instruction

Change the array element as the last element of the connection input array. The default refers to element 249, since this is the last element of the array of 250 words. For example, if the module connection input is set as 122 words, set the last element as 121.
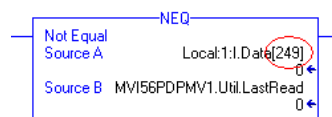
COP Instruction (output)

Change the Length to match the number of output words configured for the module. The default uses 248 words.
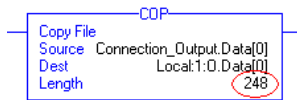
Example - FLEX application

For this example, if your application requires the PROFIBUS total data sizes configured as follows (through the ProSoft Configuration Builder):

▪ **PROFIBUS INPUT DATA SIZE: 100 WORDS**
▪ **PROFIBUS OUTPUT DATA SIZE: 100 WORDS**
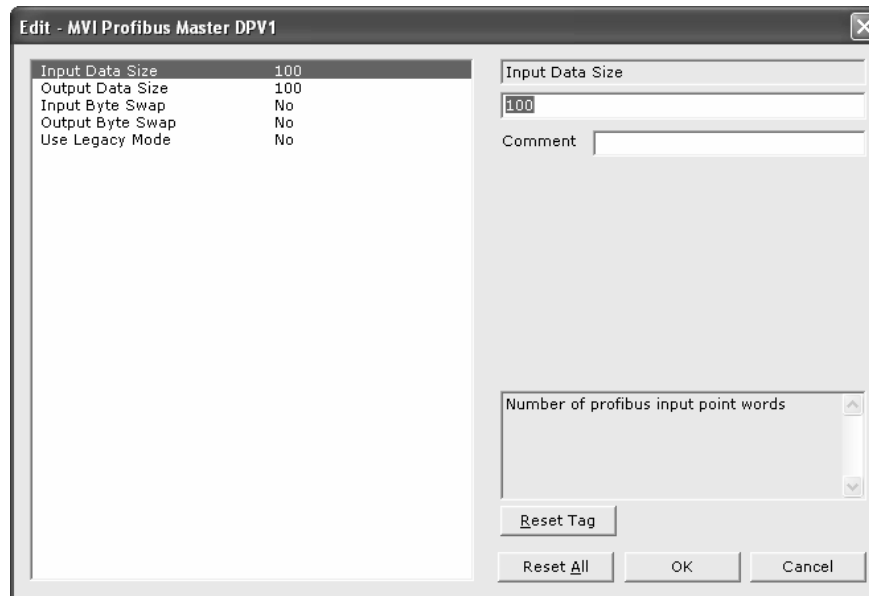
Then, the following four scenarios illustrate the flexible I/O size functionality (FLEX), using different backplane I/O sizes for the same PROFIBUS I/O size.

Scenario 1: The PROFIBUS data fits into one backplane block

This example uses the following backplane configuration:
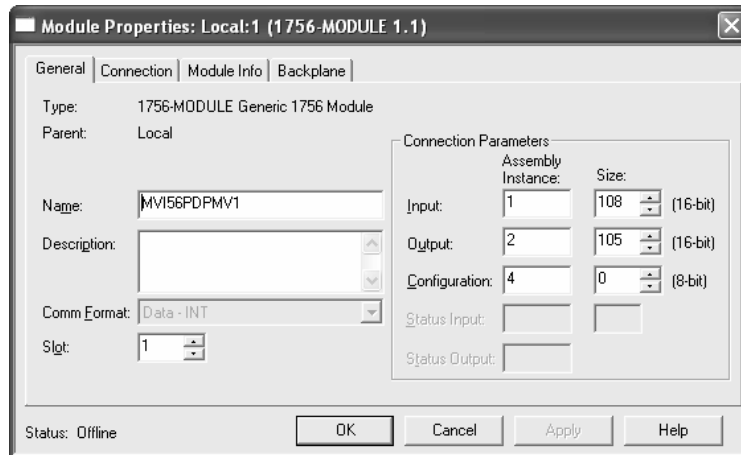
For this scenario, each backplane block reserves 100 words for PROFIBUS data, so the data fits into one backplane I/O block. The module will generate two block IDs, blocks 0 and 1, in order to force a Block ID switch, as required for the ladder logic to function correctly, but both blocks will contain the same data.

Backplane Block Overview

Block Number	Input/ Output	Total word count in backplane block	PROFIBUS Data Word Count in backplane block	Initial PROFIBUS Data Word Offset in backplane block	Last PROFIBUS Data Word Offset in backplane block
0	Input	108	100	0	99
1	Input	108	100	0	99
0	Output	105	100	0	99
1	Output	105	100	0	99

Scenario 2: The PROFIBUS data fits into two backplane blocks (different PROFIBUS data count per block)

For this example, we will subtract one word for each I/O backplane block, using the following backplane configuration:
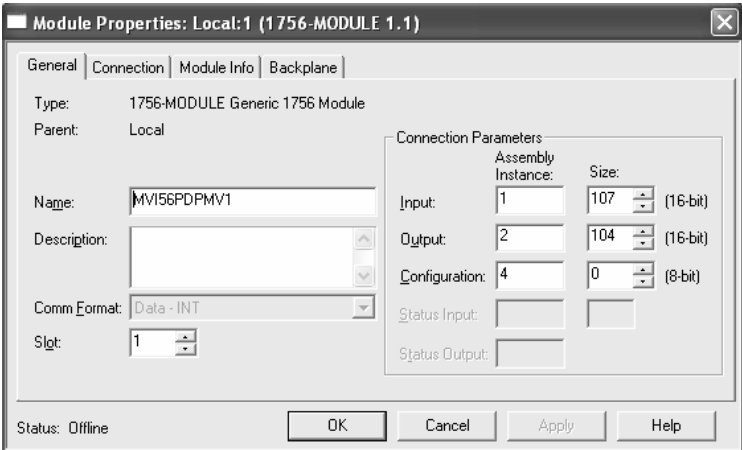
For this scenario, each backplane block reserves a maximum of 99 words for PROFIBUS I/O data. The configured PROFIBUS I/O size (100 words) will not fit into one block, so two backplane input blocks and two backplane output blocks will be required to transfer the entire PROFIBUS I/O data.

Note: For this scenario, block 0 is no longer required because there are two backplane blocks with different data.

Backplane Block Overview

Block Number	Input/ Output	Total word count in backplane block	PROFIBUS Data Word Count in backplane block	Initial PROFIBUS Data Word Offset in backplane block	Last PROFIBUS Data Word Offset in backplane block
1	Input	107	99	0	98
2	Input	107	1	99	99
1	Output	104	99	0	98
2	Output	104	1	99	99

Note: This configuration would be inefficient and is therefore not recommended. The previous example is a more efficient configuration that will accomplish the same thing. This example is here solely to illustrate how changing the I/O sizes without changing the amount of PROFIBUS data to be transferred will affect the number of I/O data blocks the module will need to transfer.

Scenario 3: The PROFIBUS data fits into two backplane blocks (same PROFIBUS data count per block)

For this example, we will split the PROFIBUS data evenly within two backplane blocks, using the following backplane configuration:

For this scenario, each backplane block reserves 50 words for PROFIBUS I/O data. The configured PROFIBUS I/O size (100 words) will not fit into one block, so two backplane input blocks and two backplane output blocks will be required.

Backplane Block Overview

Block Number	Input/ Output	Total word count in backplane block	PROFIBUS Data Word Count in backplane block	Initial PROFIBUS Data Word Offset in backplane block	Last PROFIBUS Data Word Offset in backplane block
1	Input	58	50	0	49
2	Input	58	50	50	99
1	Output	55	50	0	49
2	Output	55	50	50	99

Note: This configuration would be inefficient and is therefore not recommended. The first example (Scenario 1) is a more efficient configuration that will accomplish the same thing. This example is here solely to illustrate how changing the I/O sizes without changing the amount of PROFIBUS data to be transferred will affect the number of I/O data blocks the module will need to transfer.

Scenario 4: The PROFIBUS data fits into ten backplane blocks (same PROFIBUS data count per block)
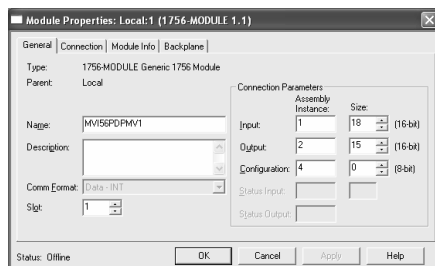
For this scenario, each backplane block reserves 10 words for PROFIBUS I/O data. The configured PROFIBUS I/O size (100 words) will not fit into one block, so ten backplane input blocks and ten backplane output blocks will be required.

Backplane Block Overview

Block Number	Input/ Output	Total word count in Backplane block	PROFIBUS Data Word Count in backplane block	Initial PROFIBUS Data Word Offset in backplane block	Last PROFIBUS Data Word Offset in backplane block
1	Input	18	10	0	9
2	Input	18	10	10	19
3	Input	18	10	20	29
4	Input	18	10	30	39
5	Input	18	10	40	49
6	Input	18	10	50	59
7	Input	18	10	60	69
8	Input	18	10	70	79
9	Input	18	10	80	89
10	Input	18	10	90	99
1	Output	15	10	0	9
2	Output	15	10	10	19
3	Output	15	10	20	29
4	Output	15	10	30	39
5	Output	15	10	40	49
6	Output	15	10	50	59
7	Output	15	10	60	69
8	Output	15	10	70	79
9	Output	15	10	80	89
10	Output	15	10	90	99

Note: This configuration would be extremely inefficient and is therefore not recommended. The first example (Scenario 1) is a more efficient configuration that will accomplish the same thing much faster with one backplane transfer. This example is here solely to illustrate how changing the I/O sizes without changing the amount of PROFIBUS data to be transferred will affect the number of I/O data blocks the module will need to transfer.

Creating a New RSLogix5000 Project

1 Open the *File* menu and then choose **NEW**.
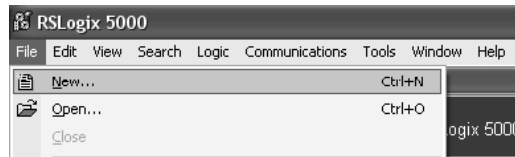
2 Select Revision **16**.
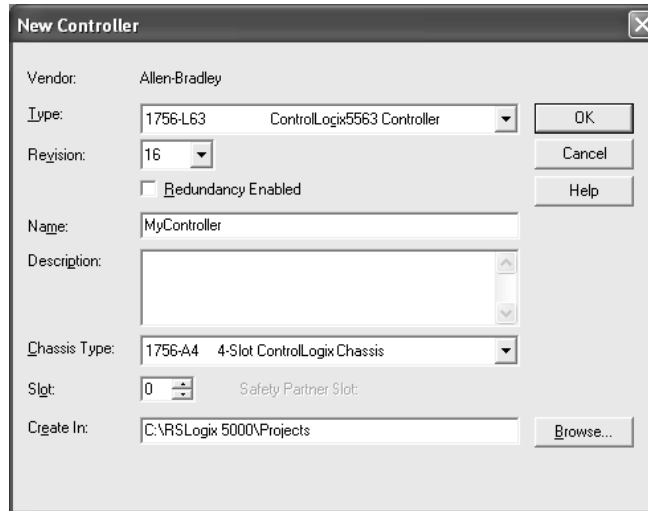
Creating the MVI56-PDPMV1 Module Profile - Flex

1 Right-click **I/O CONFIGURATION** and then choose **NEW MODULE**.

2 Select **1756-MODULE**.

3 Fill the module properties as follows:

Parameter	Value
Name	Enter a module identification string. Example: MVI56PDPMV1
Description	Enter a description for the module. Example: ProSoft communication module for PROFIBUS communication.
Comm Format	Select **DATA-INT**
Slot	Enter the slot number in the rack where the MVI56-PDPMV1 module is located.
Input Assembly Instance	**1**
Input Size	Enter the number of words per backplane input block to copy from the module to the processor. The allowable range is from 12 to 250 words. Each input block will reserve 8 words for general purposes (CRC, block/CIP handshaking). The remainder is reserved for PROFIBUS input data.
Output Assembly Instance	**2**
Output Size	Enter the number of words per backplane output block to copy from the module to the processor. The allowable range is from 5 to 248 words. Each output block will reserve 5 words for general purposes (CRC and block handshaking). The remainder is reserved for PROFIBUS output data.
Configuration Assembly Instance	**4**
Configuration Size	**0**

4 The following illustration shows an example for a local rack application where the module was set with the maximum backplane I/O sizes (250/248 words).
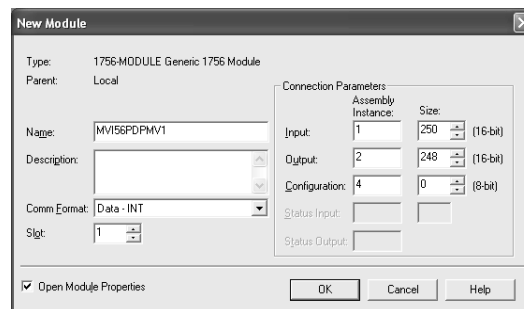
5 Click the **CONNECTION** tab, and then enter the *Requested Package Interval (RPI)* value.

   o If the module is located in the local rack (same rack where the processor is located) or in a remote rack connected through EtherNet/IP adapters, select the maximum value of **750 MS**.
   o If the module is located in a remote rack connected through ControlNet adapters, adjust the **RPI** time to be not less than **5 MS** and not less than (and, preferably, some even multiple of) the RSNetWorx for ControlNet Network Update Time (NUT).

6 Click **OK** to confirm.
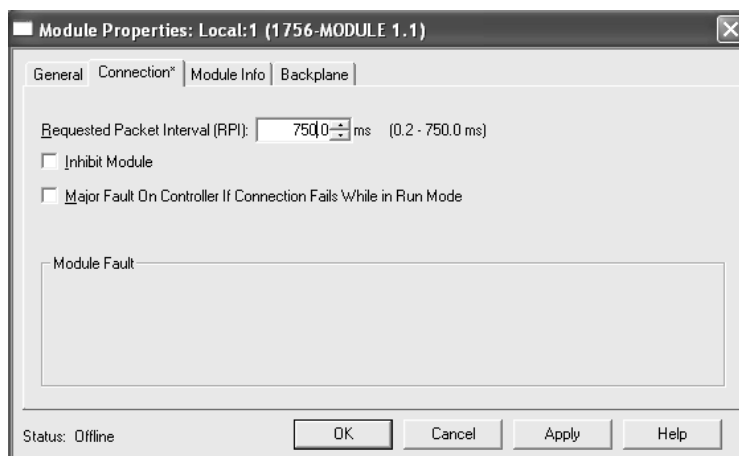
Now the MVI56-PDPMV1 module will be visible in the *I/O Configuration* section.
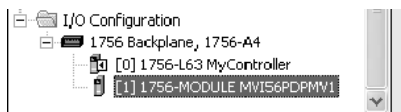
Importing the Ladder Rung - Flex

1 Open your application in RSLogix 5000.
2 Expand the *Tasks* folder, and then expand the *MainTask* folder.
3 In the *MainProgram* folder, select an empty rung, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose **IMPORT RUNG**.
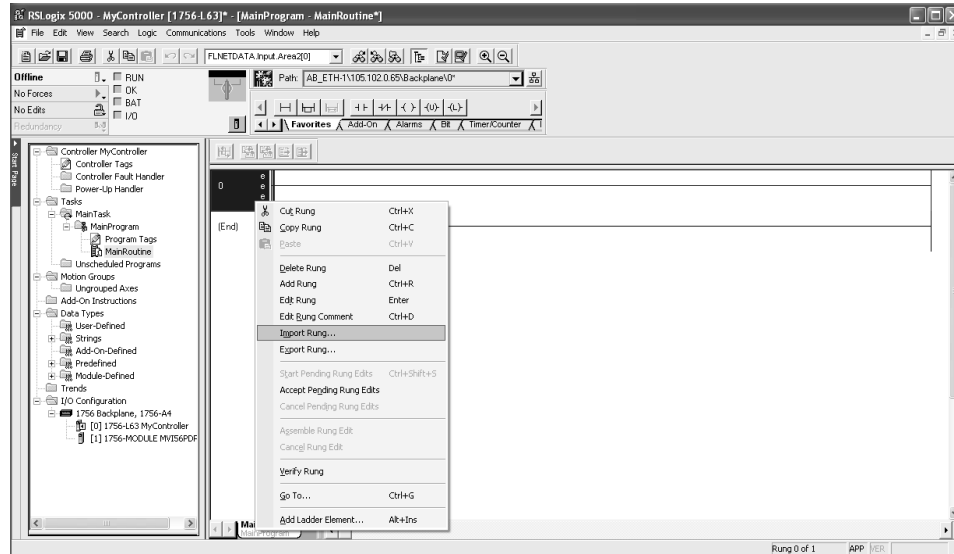
4 Select the *PS56PDPMV1_FLEX_v*.L5X* standard sample file, or the *MVI56PDPMV1_FLEX_AddOn_Rung_v1_2.L5X* custom file you exported from PCB, and click the **IMPORT** button.
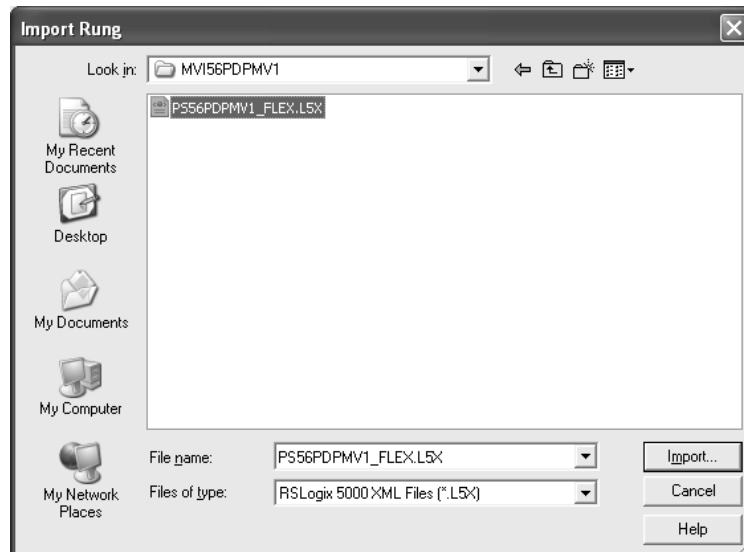
5 The following window will be displayed showing the controller tags associated with the project. The *I/O connection* parameters will be set (as default) as **LOCAL:1:I** and **LOCAL:1:O**, which assumes that the module is located at the local rack and Slot 1.
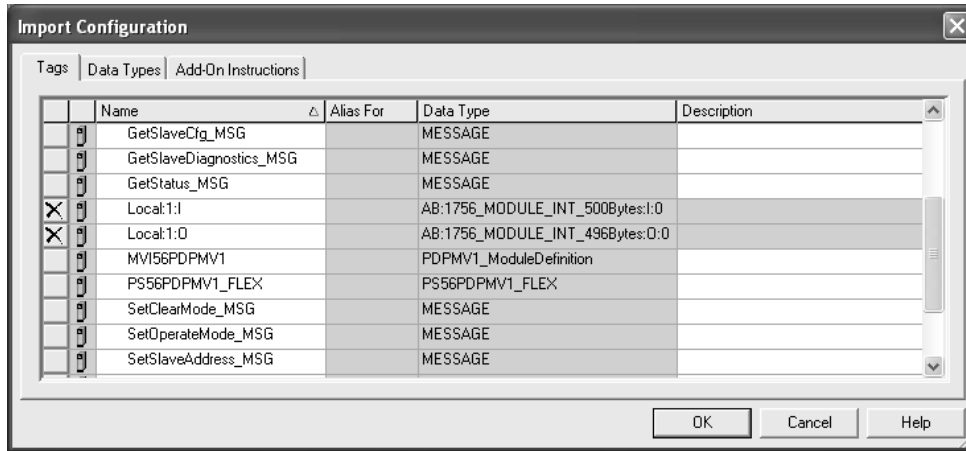
6 If your module is located in a different slot or in a remote rack, modify the parameters to set the correct path to the module, as follows:
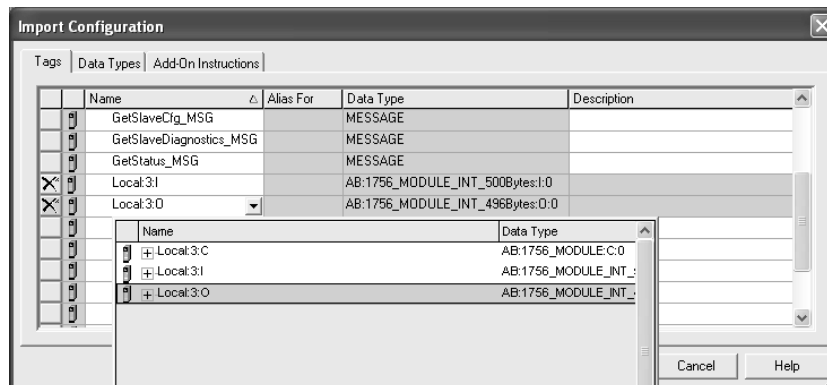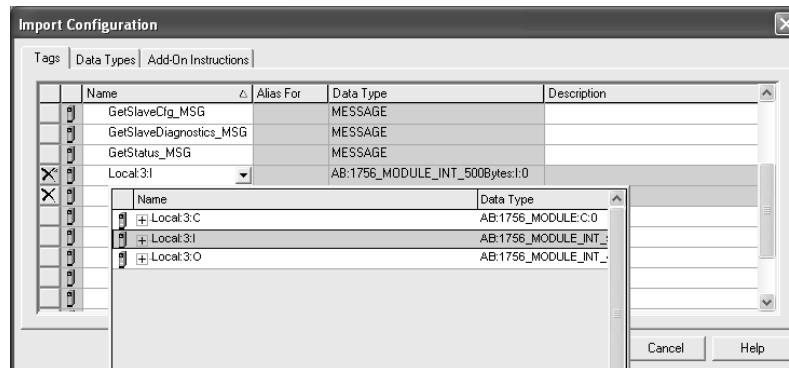
7 Click **OK** to confirm the import.
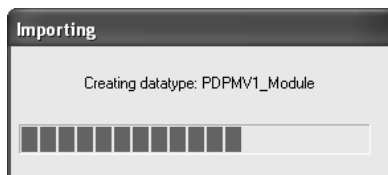
8 The new rung will be now visible in the ladder logic showing also the PS56PDPMV1_FLEX Add-On Instruction.
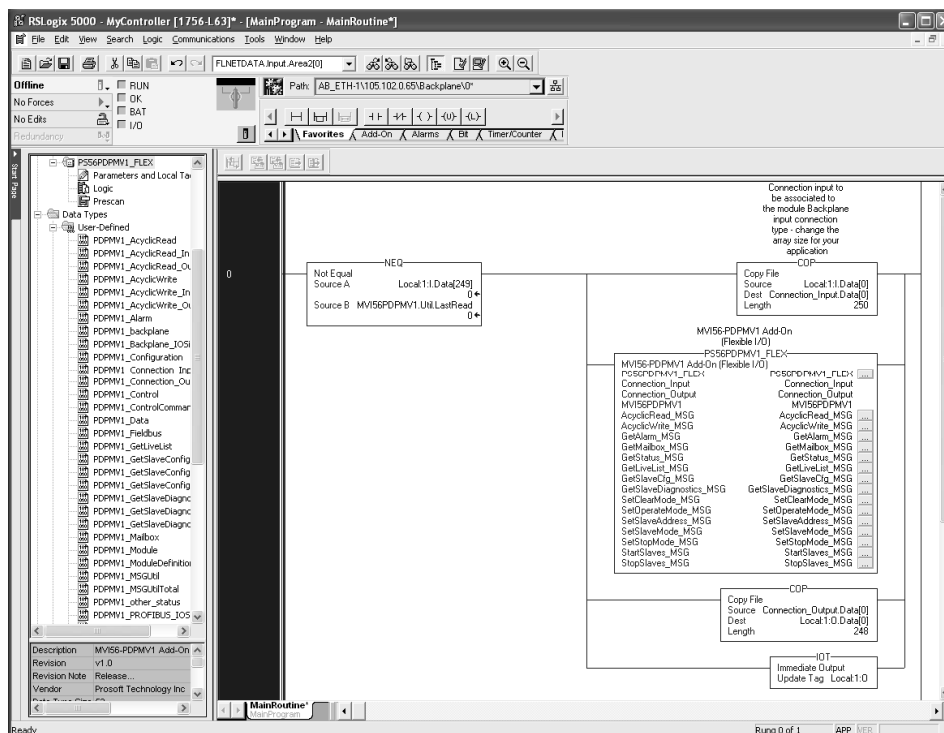
9 The procedure has also imported new user-defined data types, controller tags, and controller tag arrays. The new Add-On Instruction can also be used through the new toolbar button (at Add-On tab).

Adding Multiple Modules (Optional) - Flex

Important: If your application requires more than one MVI56-PDPMV1 module into the same project, follow the steps below.

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

2 Select **1756-MODULE**.
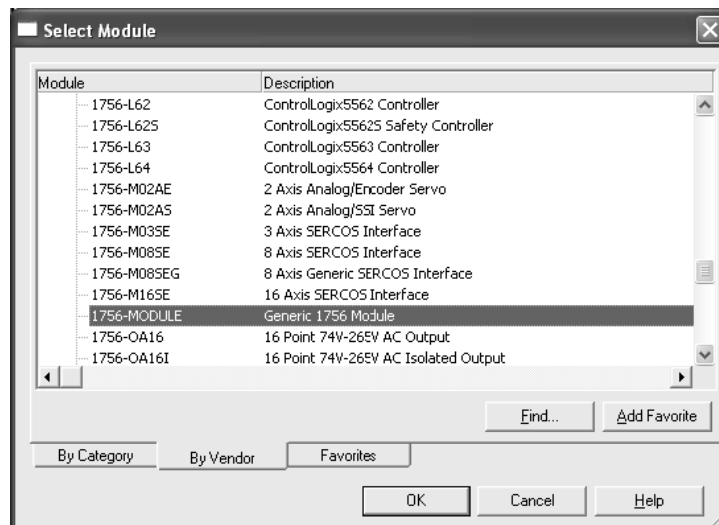
3 Fill the module properties as required:

Parameter	Value
Name	Enter a module identification string. Example: MVI56PDPMV1_2
Description	Enter a description for the module. Example: ProSoft communication module for PROFIBUS communication.
Comm Format	Select **DATA-INT**
Slot	Enter the slot number in the rack where the MVI56-PDPMV1 module is located.
Input Assembly Instance	**1**
Input Size	Enter the number of words per backplane input block to copy from the module to the processor. The allowable range is from **12 TO 250** words. Each input block will reserve 8 words for general purposes (CRC, block/CIP handshaking). The remainder is reserved for PROFIBUS input data.
Output Assembly Instance	**2**
Output Size	Enter the number of words per backplane output block to copy from the module to the processor. The allowable range is from **5 TO 248** words. Each output block will reserve 5 words for general purposes (CRC and block handshaking). The remainder is reserved for PROFIBUS output data.
Configuration Assembly Instance	**4**
Configuration Size	**0**

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

5 Expand the *Tasks* folder, and then expand the *MainTask* folder.
6 On the *MainProgram* folder, click the right mouse button to open a shortcut menu. On the shortcut menu, choose **NEW ROUTINE**.
7 In the *New Routine* dialog box, enter the name and description of your routine, and then click **OK**.

8 Select an empty rung in the new routine, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose **IMPORT RUNG**.

9 Select the *PS56PDPMV1_FLEX_v*.L5X* standard sample file, or the *MVI56PDPMV1_FLEX_AddOn_Rung_v1_2.L5X* custom file you exported from PCB, and click the **IMPORT** button.
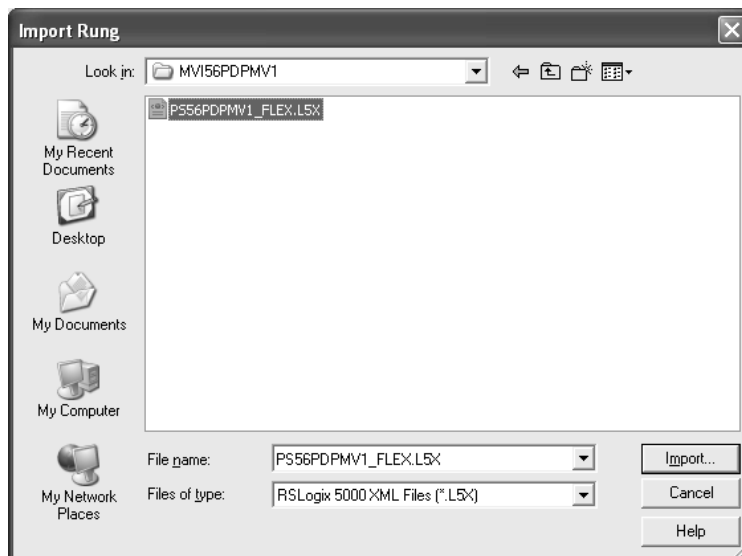
10 The following window will be displayed showing the tags to be imported:
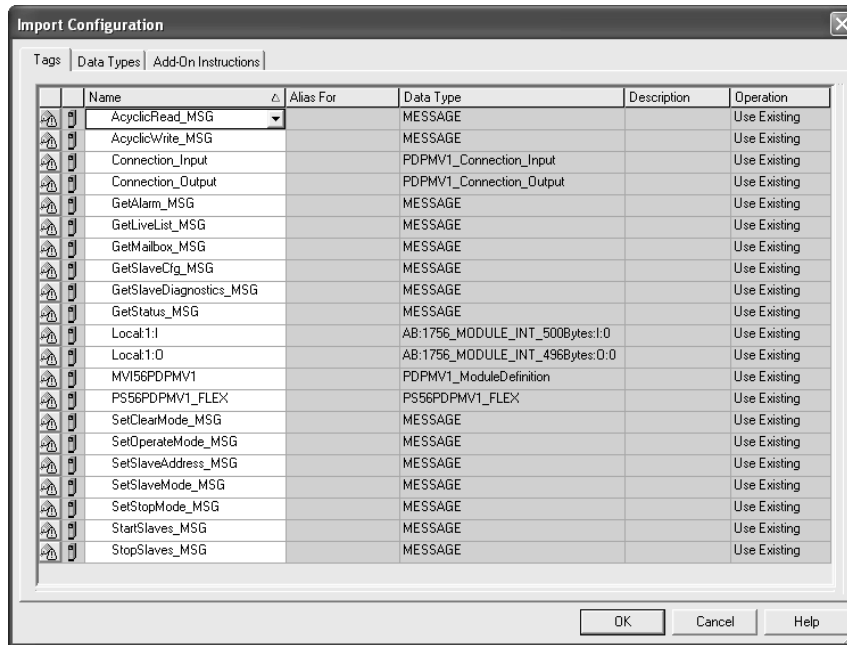
11 Change the names of all controller tags to avoid a conflict with the existing names. In this step, you should append a string to the default tag names, such as "_2", as shown in the following illustration.
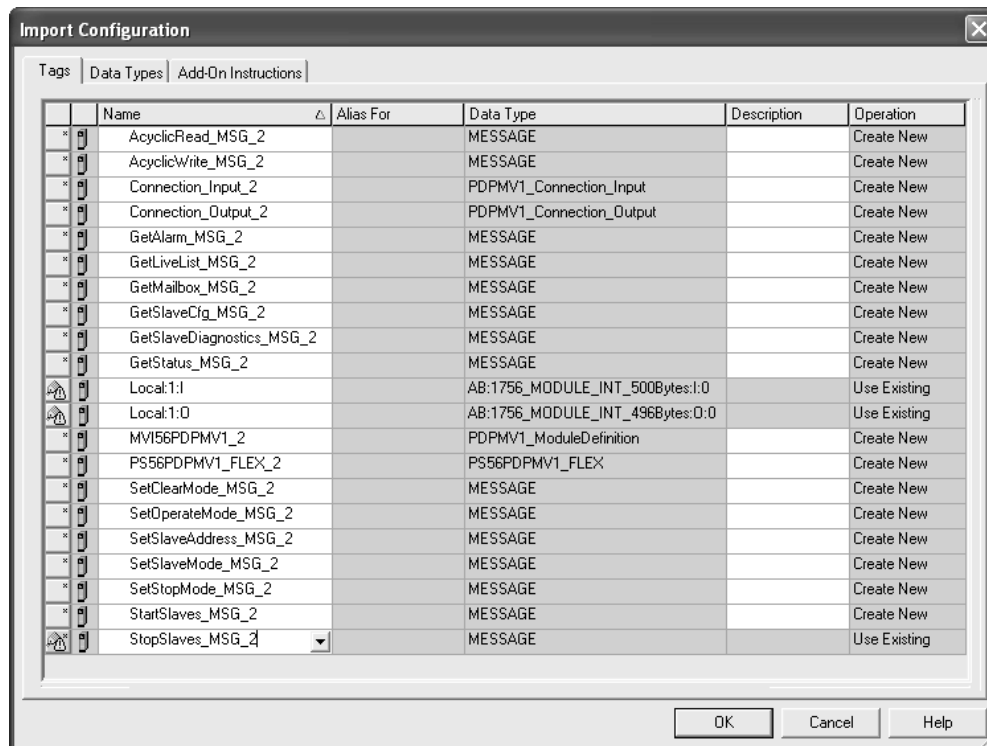
12 For the connection I/O variables, select the variables associated to the correct module as follows:

Input Connection Variable
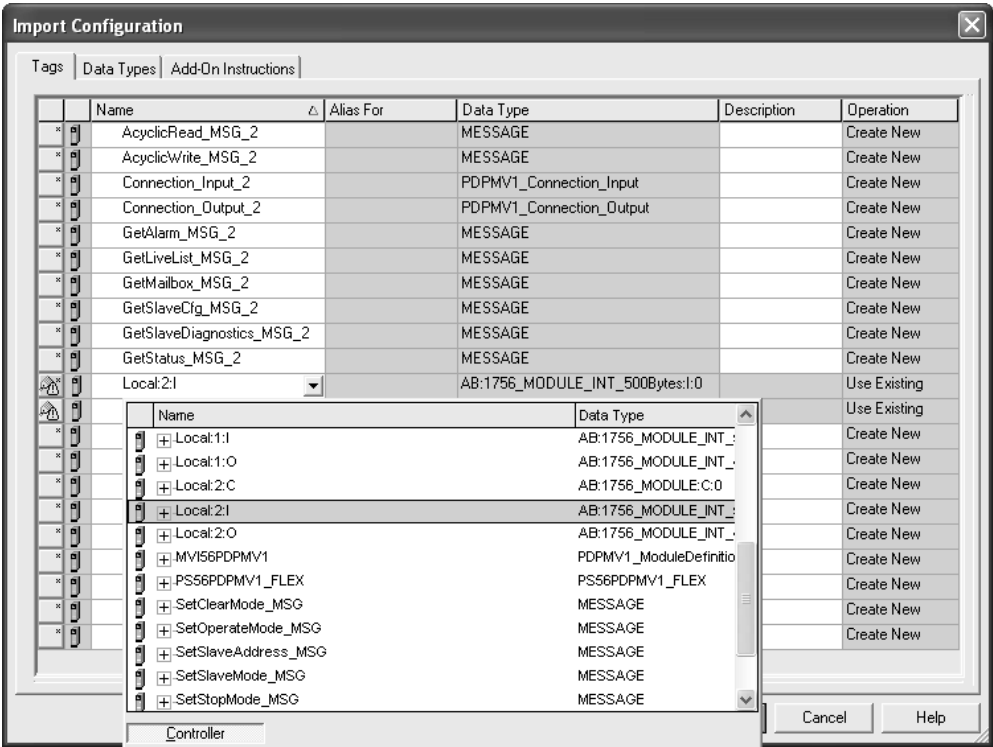
Output Connection Variable
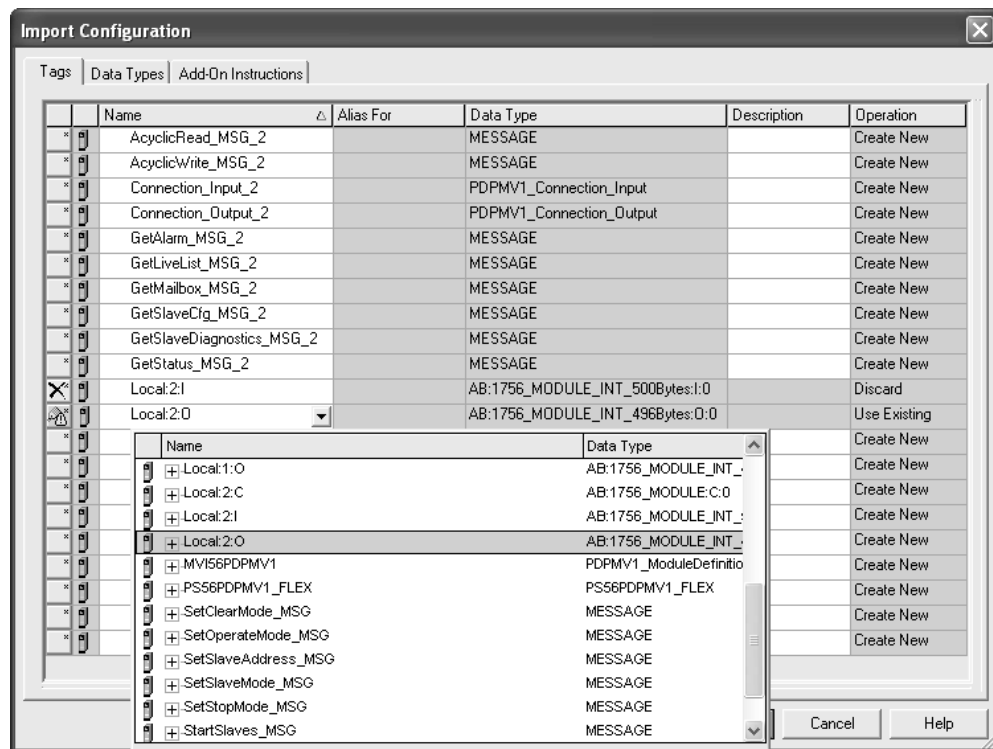
13 Click **OK** to confirm the import procedure.
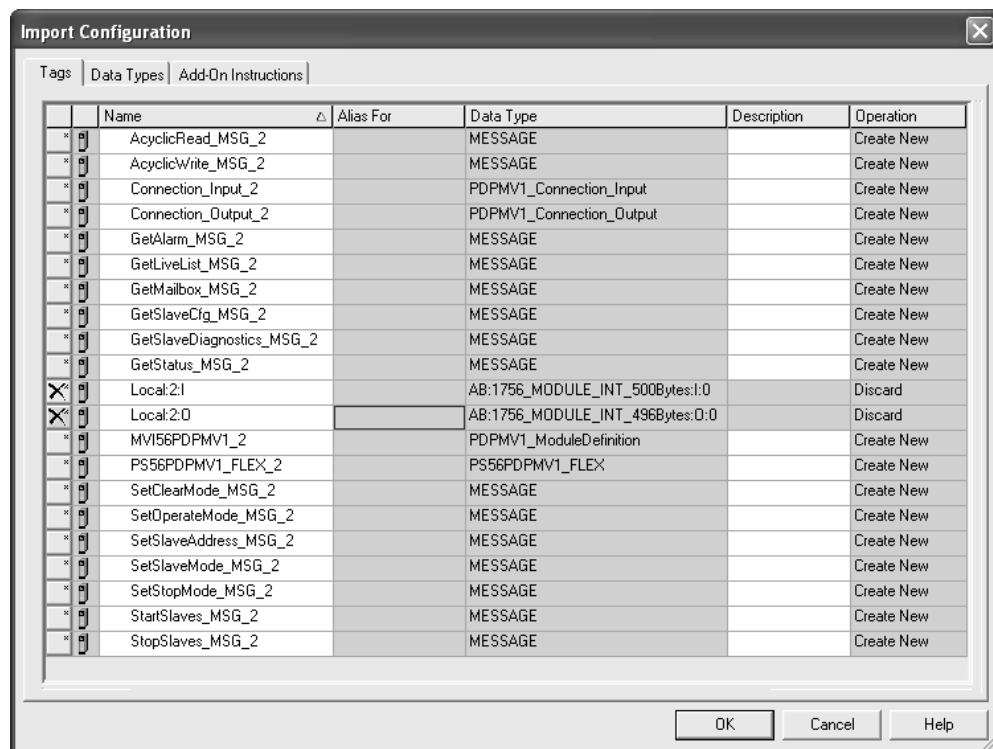
14 The new rung will now be visible in the ladder logic.
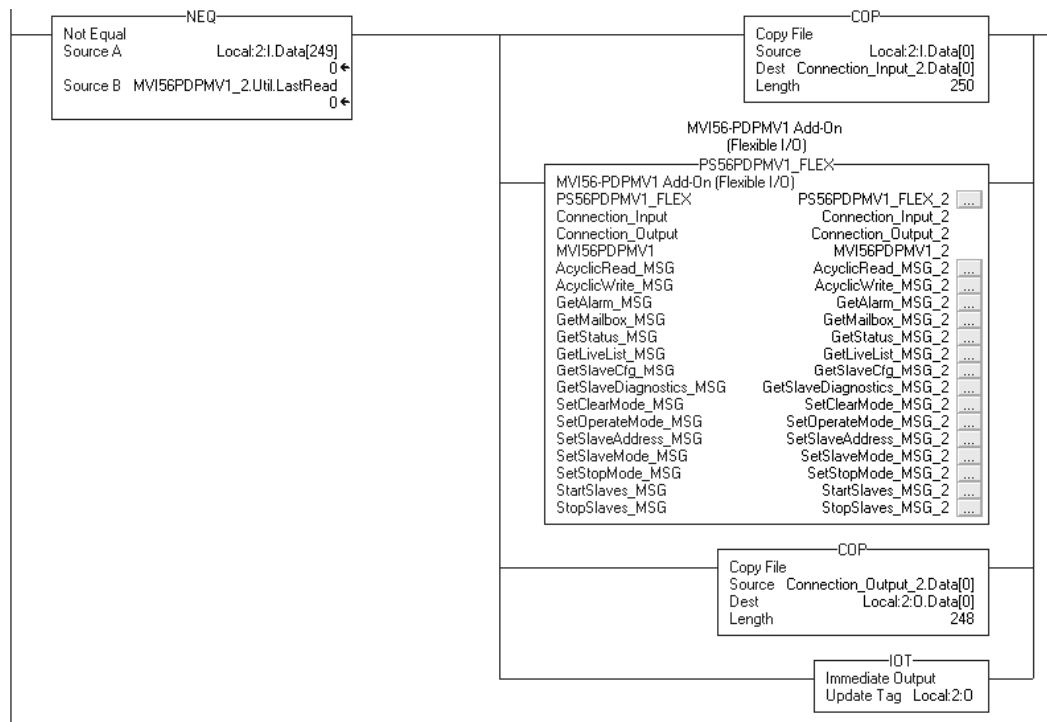
15 The module name associated with the project uses a different name than the standard "MVI56PDPMV1". You must re-define the path accordingly for each MESSAGE tag associated to the Add-On Instruction. Click the [...] symbol to set up each MESSAGE tag. Select the Communication path as follows:
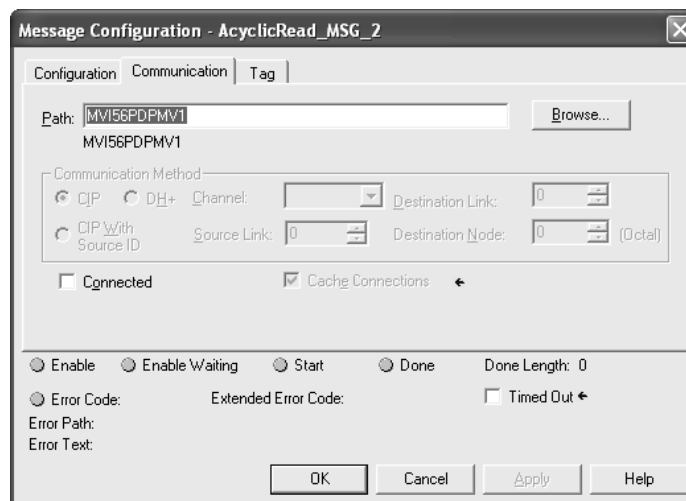
16 Select the **BROWSE** button and select the correct module to be associated to the MESSAGE instruction. Click **OK** to confirm.
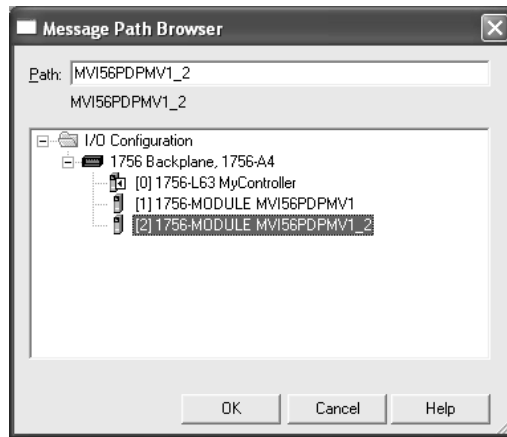
17 Click **OK** to confirm the MESSAGE configuration. Repeat this procedure for every MESSAGE tag associated to the Add-On Instruction.
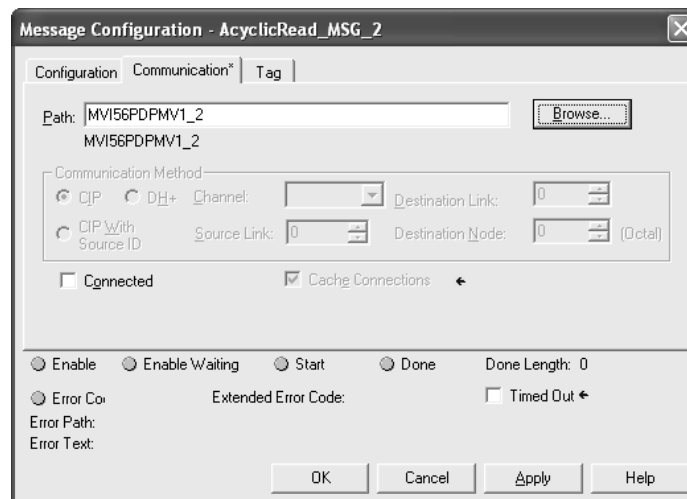
The setup procedure is now completed and the processor is ready to communicate with the MVI56-PDPMV1 module.

```
┌────────────NEQ────────────┐                                    ┌────────────COP────────────┐
│ Not Equal                 │                                    │ Copy File                 │
│ Source A     Local:2:I.Data[249]                               │ Source        Local:2:I.Data[0]
│                         0 ←                                     │ Dest   Connection_Input_2.Data[0]
│ Source B  MVI56PDPMV1_2.Util.LastRead                          │ Length                 250 │
│                         0 ←                                     └───────────────────────────┘
└───────────────────────────┘
```

MVI56-PDPMV1 Add-On
(Flexible I/O)
────PS56PDPMV1_FLEX────

MVI56-PDPMV1 Add-On (Flexible I/O)	
PS56PDPMV1_FLEX	PS56PDPMV1_FLEX_2 ...
Connection_Input	Connection_Input_2
Connection_Output	Connection_Output_2
MVI56PDPMV1	MVI56PDPMV1_2
AcyclicRead_MSG	AcyclicRead_MSG_2 ...
AcyclicWrite_MSG	AcyclicWrite_MSG_2 ...
GetAlarm_MSG	GetAlarm_MSG_2 ...
GetMailbox_MSG	GetMailbox_MSG_2 ...
GetStatus_MSG	GetStatus_MSG_2 ...
GetLiveList_MSG	GetLiveList_MSG_2 ...
GetSlaveCfg_MSG	GetSlaveCfg_MSG_2 ...
GetSlaveDiagnostics_MSG	GetSlaveDiagnostics_MSG_2 ...
SetClearMode_MSG	SetClearMode_MSG_2 ...
SetOperateMode_MSG	SetOperateMode_MSG_2 ...
SetSlaveAddress_MSG	SetSlaveAddress_MSG_2 ...
SetSlaveMode_MSG	SetSlaveMode_MSG_2 ...
SetStopMode_MSG	SetStopMode_MSG_2 ...
StartSlaves_MSG	StartSlaves_MSG_2 ...
StopSlaves_MSG	StopSlaves_MSG_2 ...

```
┌────────────COP────────────┐
│ Copy File                 │
│ Source   Connection_Output_2.Data[0]
│ Dest          Local:2:O.Data[0]
│ Length                 248 │
└───────────────────────────┘

┌────────────IOT────────────┐
│ Immediate Output          │
│ Update Tag  Local:2:O     │
└───────────────────────────┘
```

1.8 Common Settings for All Sample Programs

Certain parameter options are common to all the various versions of ladder logic and Add-On Instruction sample programs. Specific settings for these parameters vary based on which sample is used and the details of your specific custom application.

1.8.1 Choosing the Controller Type

This sample application shows a 1756-L63 ControlLogix 5563 Controller selected. If you are using a different model of the ControlLogix processor, you must configure the sample program to use the correct processor model.

To change the controller type

1   In RSLogix 5000, once you have loaded a sample program or started a new project, right-click the *Controller <Name>* folder at the top of the *Controller Organizer* pane to open a shortcut menu; or you may left-click the Processor icon [▯] in the *Online Toolbar* .
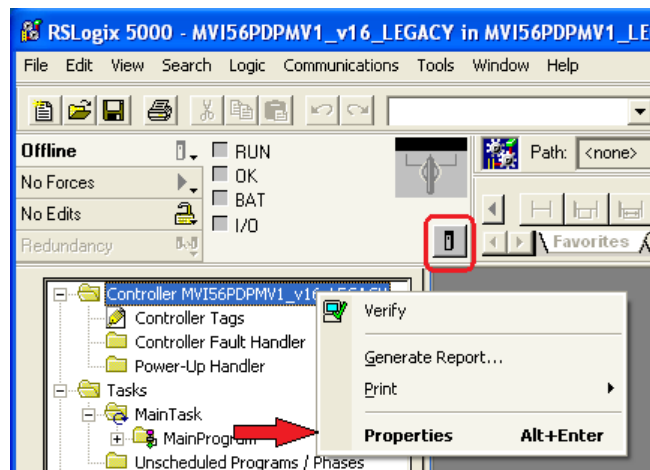
2 On the shortcut menu, choose **PROPERTIES**. This action opens the *Controller Properties* dialog box.
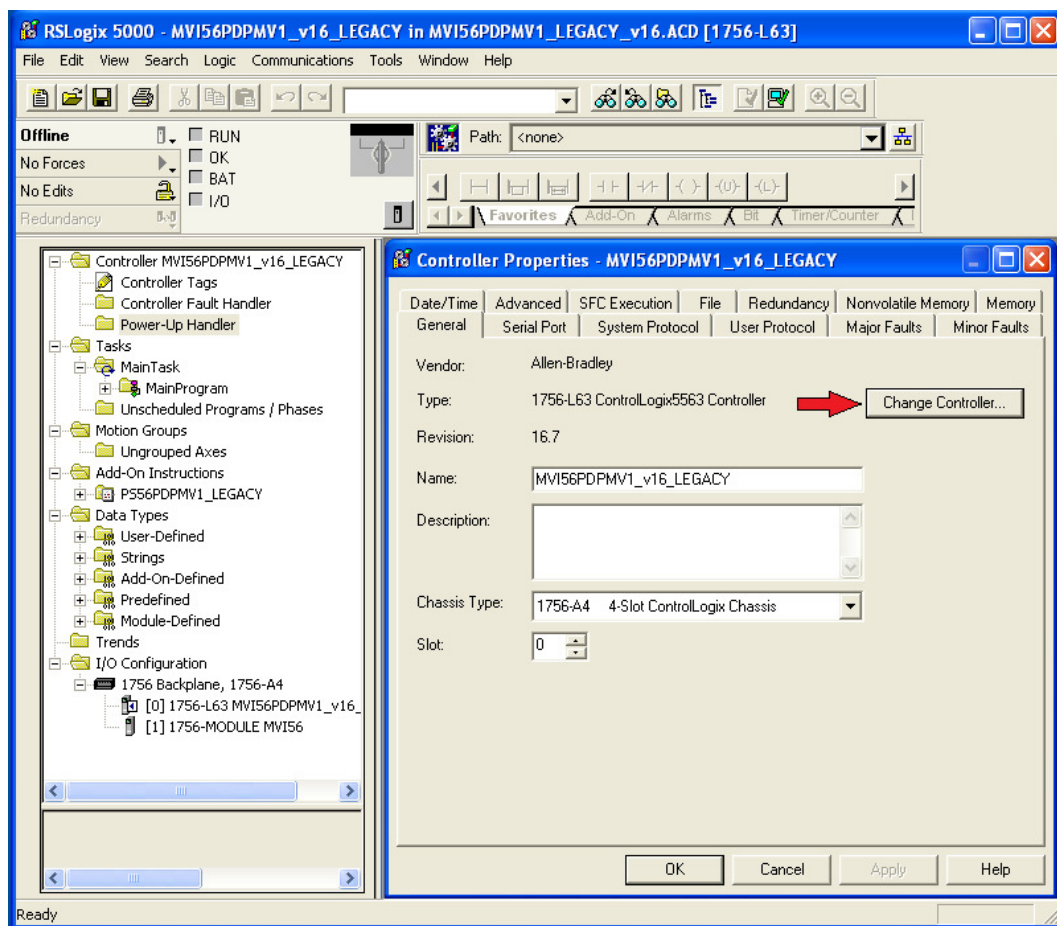
3 Click the **CHANGE CONTROLLER** button. This action opens the *Change Controller* dialog box.
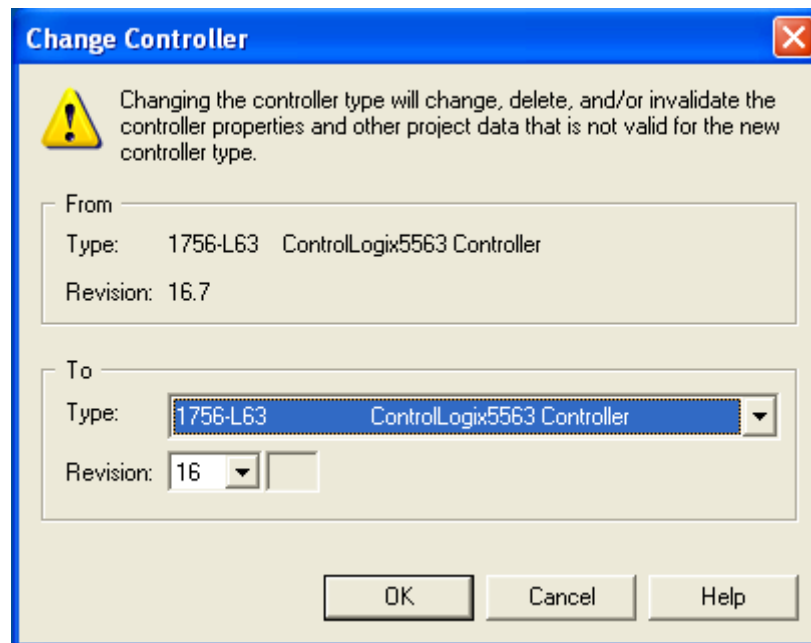
4 Open the **TYPE** drop-down list, and then select your ControlLogix controller.
5 Select the correct firmware revision for your controller.
6 Click **OK** to save your changes and return to the previous window.
7 Select the chassis type that matches your specific installation.

1.8.2 Selecting the Slot Number for the Module

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

To change the slot number

1   In the *Controller Organization* list, select the module and then click the right mouse button to open a shortcut menu.
2   On the shortcut menu, choose **PROPERTIES**. This action opens the *Module Properties* dialog box.
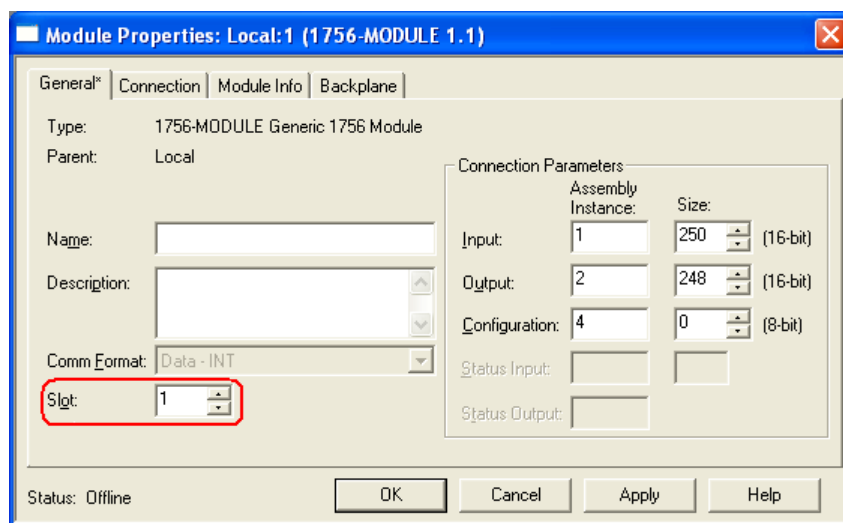
3   In the *Slot* field, use the spinners on the right side of the field to select the slot number where the module will reside in the rack, and then click **OK**.

RSLogix will automatically apply the slot number change to all tags, variables and ladder logic rungs that use the MVI56-PDPMV1 slot number for computation.

1.8.3 Selecting the Connection Parameters for the Module

The connection parameters will determine the number of words each backplane I/O block will contain for data transfer between the processor and the module. The input and output backplane block sizes will depend on the MVI56-PDPMV1 version and the selected mode (LEGACY or FLEX). Depending on these factors, the connection parameters may be set as Fixed or Flexible lengths.

While following the procedures in the appropriate sections, keep in mind to set the *Use Legacy Mode* parameter to match the sample program version (LEGACY or FLEX) that you selected and loaded. If you get a mismatch between this parameter setting and the sample ladder program loaded, the module will not function.

The following settings are configured in ProSoft Configuration Builder for modules with firmware version 1.21 or later only. Note that the parameters that are displayed on the *Edit* screen vary depending on whether **YES** or **NO** is selected for the *Use Legacy Mode* parameter. For more information, refer to Setting Module Parameters (page 82).

Select LEGACY mode (USE LEGACY MODE **= Y)**
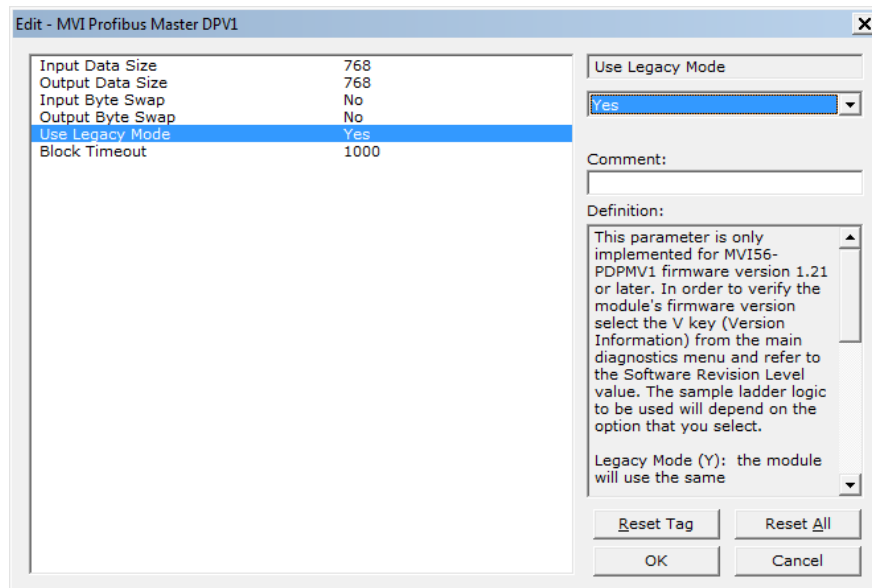
Select FLEX mode (USE LEGACY MODE **= N)**
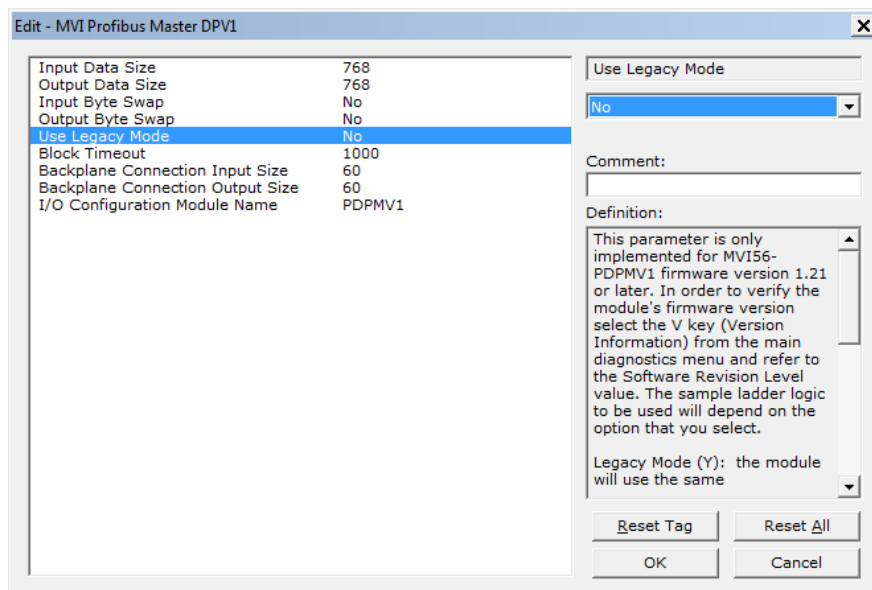
Set Module Properties in RSLogix 5000 According to Mode Selected
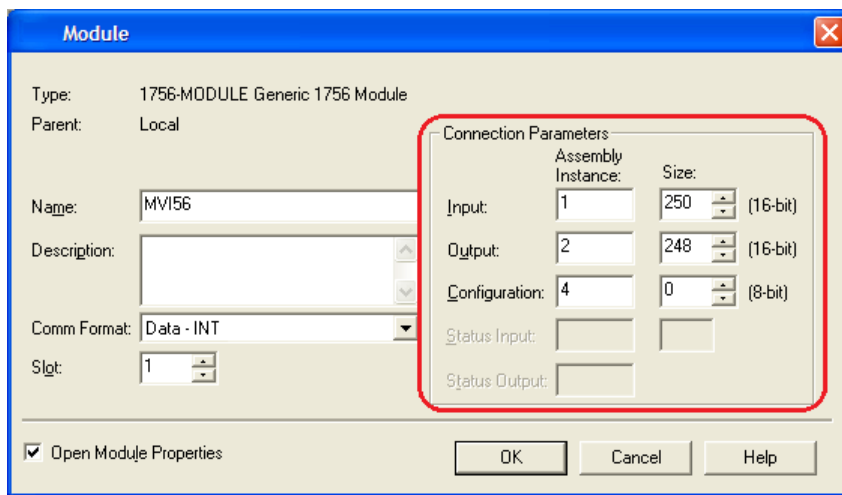
For additional details about setting connection parameters for FLEX mode, see
Adjust the Connection Sizes for Your Application (page 41).

1.8.4 Selecting the RPI Time for the Module

Refer to the Connection tab and enter the RPI time. The connection is scheduled to move data to or from the module at least this often.

The RPI time to enter depends on your application as follows:

RPI Time 5 to 750 milliseconds

For ANY of the following conditions

- MVI56-PDPMV1 module firmware version 0.30
- MVI56-PDPMV1 module firmware version 1.21 (or later) AND configured in LEGACY mode (**USE LEGACY MODE** = **YES**)
- MVI56-PDPMV1 module firmware version 1.21 (or later) AND configured in FLEX mode (**USE LEGACY MODE** = **NO**) AND MVI56-PDPMV1 is located in a remote rack through ControlNet

Enter the **RPI** time that will provide optimum backplane performance for your application, within the range from **5 MS** to **750 MS**.
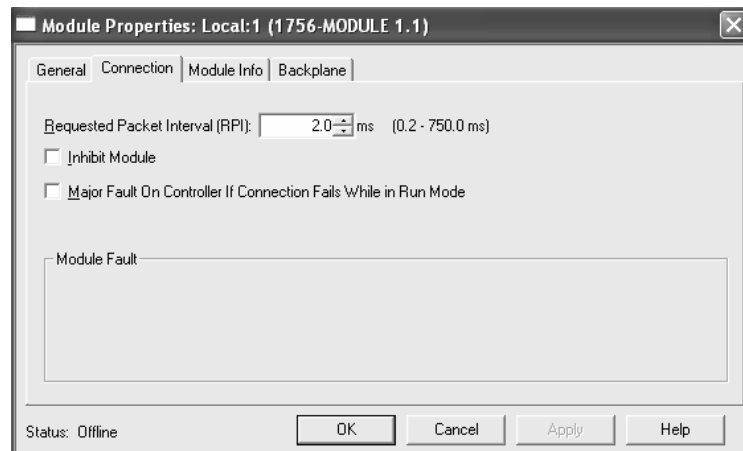
RPI Time 750 milliseconds

If ALL of the following conditions are true:

- MVI56-PDPMV1 module firmware version 1.21 (or later)
- MVI56-PDPMV1 module configured in FLEX mode (**USE LEGACY MODE** = **NO**)
- MVI56-PDPMV1 module is located in either the local rack or remote rack through EtherNet/IP adapters

For optimal performance set the *RPI* time as **750 MS**. For this scenario, the module will use the *IOT (Immediate Output)* instruction that will override the configured *RPI* time. Setting a lower *RPI* time will cause unnecessary interrupts to the module, delaying data updates and lowering data throughput.
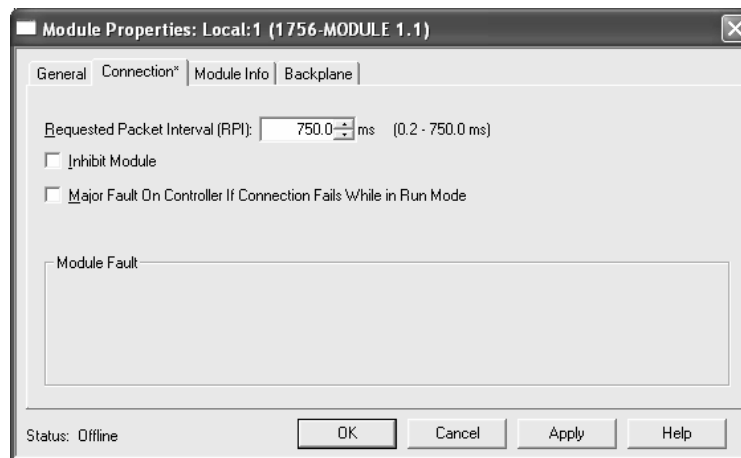
1.8.5 Changing and Importing Configuration Changes (for PCB v 2.2.0 and up only)

NOTE: This procedure must be used anytime you wish to change the module configuration after the first time an AOI rung import was performed. For changes to be effective, previous controller tags and ladder logic must be manually deleted before the new rung export file can be successfully imported. Failure to follow this procedure may result in the module becoming non-functional.

This procedure assumes you have already made your configuration changes and have re-exported the processor files according the procedure specified in Export the Processor Files for PCB v2.2.0 or later (page 107).

To successfully complete this update procedure, you must do the following steps in the order shown:

1   **DELETE** the existing ladder logic rung that contains the previously imported Add-On Instruction (AOI) ladder file.
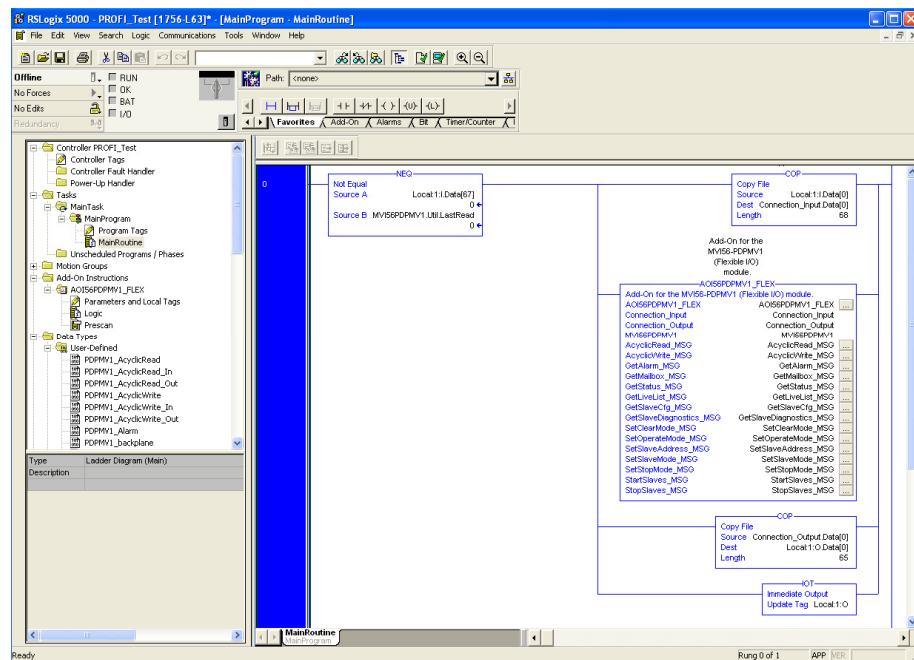
2   Go to the *Controller Tags* window and select the **EDIT TAGS** tab. **DELETE** all the controller tags associated with the AOI import.
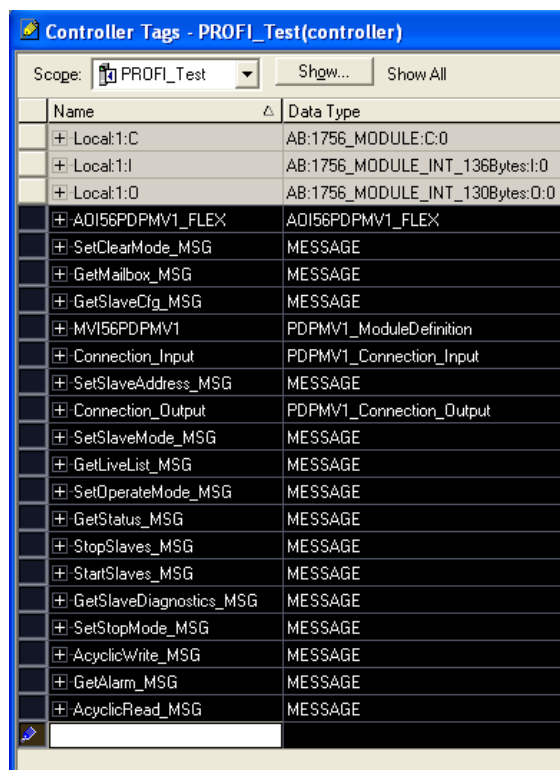
3 Go to the *Add-On* Instructions folder in the project tree. **DELETE** the "*AOI56PDPMV1_FLEX*" AOI that was created by the original import.
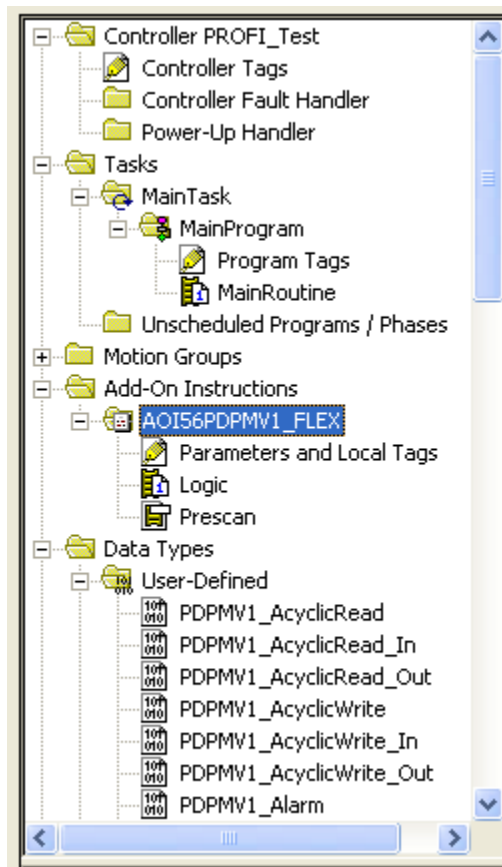
4 Go to the *Data Types* folder, to the *User-Defined* sub-folder. **DELETE** the following user-defined data types in the following order:

a) *PDPMV1_Connection_Input*
b) *PDPMV1_Connection_Output*
c) *PDPMV1_PBInF*
d) *PDPMV1_PBOutF*
e) *PDPMV1_ModuleDefinition*
f) *PDPMV1_UTIL*
g) *PDPMV1_Configuration*
h) *PDPMV1_Backplane_IOSize*
i) *PDPMV1_PROFIBUS_IOSize*
j) *PDPMV1_Status*
k) *PDPMV1_StatusControl*
l) *PDPMV1_Module*

5   Go to the *I/O Configuration* folder. Locate the MVI56-PDPMV1 and right-click on it to open a context menu, then left-click on **PROPERTIES** to open the *Module Properties* dialog box. Check and edit (if necessary) the **INPUT** and **OUTPUT** Sizes in the *Module Properties* dialog box so that they match those you configured for the module in the ProSoft Configuration Builder (PCB) *MVI Profibus Master DPV1* edit window.
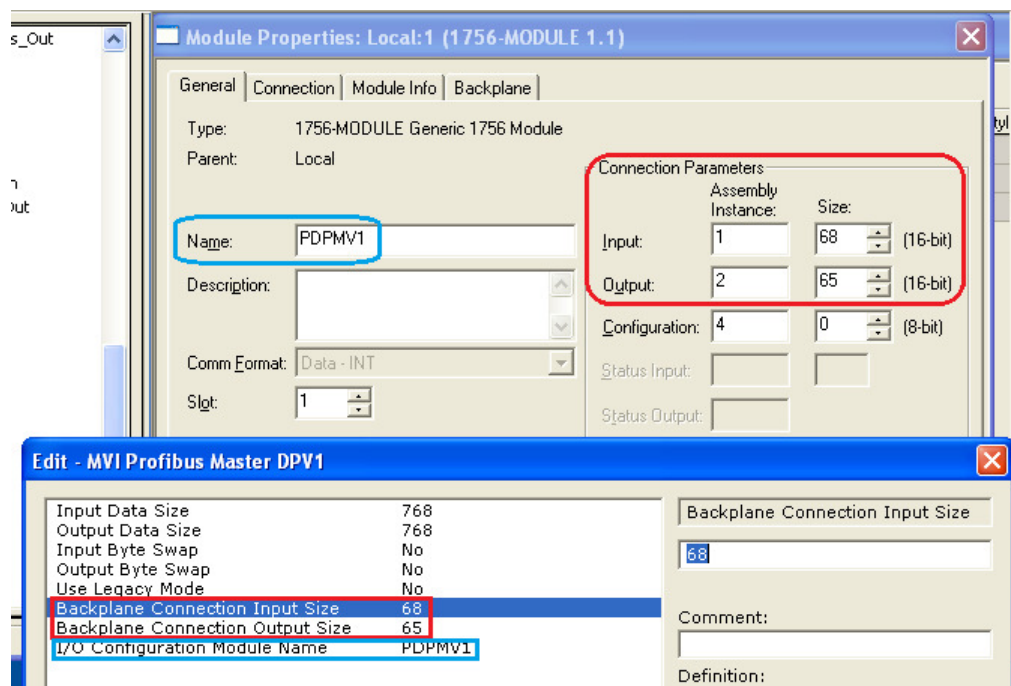
> **REMEMBER:** It is also critically important that the *I/O Configuration Module Name* on the PCB Edit box be set to match the *Name* parameter in the RSLogix5000 *Module Properties* box. The name you enter in PCB is used to create a communication path for the Status and Acyclic Data MSG instructions. If the two names are not identical, the import may appear to have completed successfully; but the module will not work correctly because the MSG instructions will not have the correct module name in the MSG instruction *Communication* tab *Path* parameter.
> If you receive any warning messages from RSLogix 5000, during or after the import, about duplicate data types being overwritten or not imported, you may safely ignore these messages. The module will function correctly. You can eliminate any such warning messages by deleting all UDTs before importing the new ladder rung file.

6   At this point, you are ready to import your new ladder rung file with your configurations changes. Please return to the previous section and follow the procedure found in the topic, Exporting the Processor Files for PCB v2.2.0 or later (page 107).

1.9   Downloading the Sample Program to the Processor

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

Note: The key switch on the front of the ControlLogix module must be in the **REM** position. Be sure you have saved a copy of the latest version of your RSLogix 5000 project file before you attempt a download.

1   Connect to the processor using the best available method.
2   If you are not already online with the processor, open the **COMMUNICATIONS** menu, and then choose **DOWNLOAD**. RSLogix will establish communication with the processor.
3   When communication is established, RSLogix will open a confirmation dialog box. Click the **DOWNLOAD** button to transfer your application project to the processor.
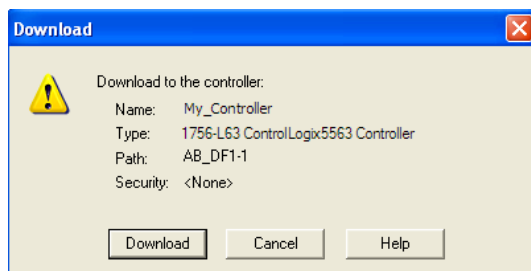
4   RSLogix will compile the program and transfer it to the processor. This process may take a few minutes.
5   When the download is complete, RSLogix will open another confirmation dialog box. Click **OK** to switch the processor from Program mode to **RUN** mode.
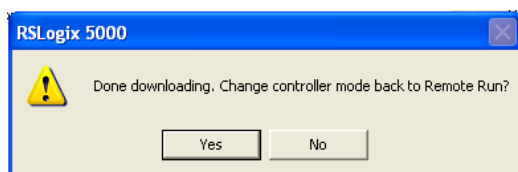
Note: If you receive an error message during these steps, refer to your RSLogix documentation to interpret and correct the error.

6   To verify that the processor is communicating with the module, open the *Main Program* folder in the *Controller Organization* pane in RSLogix, and double-click *MainRoutine*. You will be able to see the numbers change in the MVI56PDPMV1.Util.LastRead area in Rung 0 of the *MainRoutine* program.

1.10   Connecting Your PC to the Module

With the module securely mounted, connect your PC to the Configuration/Debug port using the RJ45-DB-9 Serial Adapter Cable and the Null Modem Cable included in the package with the MVI56-PDPMV1 module.

1   Connect the RJ45-DB-9 Serial Adapter Cable to the Null Modem Cable.
2   Insert the RJ45 cable connector from the RJ45-DB-9 cable into the Configuration/Debug port of the module.
3   Attach the other end to the serial port on your PC.

2   Configuring the MVI56-PDPMV1 Module

In This Chapter

Because the task of configuring the PROFIBUS network can be challenging, ProSoft Technology has provided a configuration tool called ProSoft Configuration Builder (PCB) that will help you with the following tasks:

- Creating a configuration project (page 80)
- Setting module parameters (page 82)
- Configuring the PROFIBUS network (page 84) (Master and slaves)
- Downloading the project to the module (page 111)

The following topics of this chapter explain each task step-by-step.

2.1 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. ProSoft Configuration Builder's window consists of a tree view on the left, and an information pane and 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.
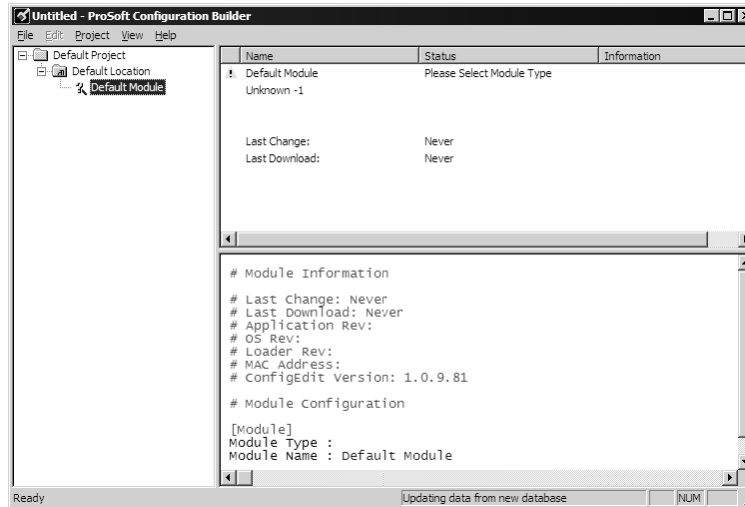
Start configuring the module by adding the MVI56-PDPMV1 module to the project.

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

2 On the shortcut menu, choose **CHOOSE MODULE TYPE**. This action opens the *Choose Module Type* dialog box.
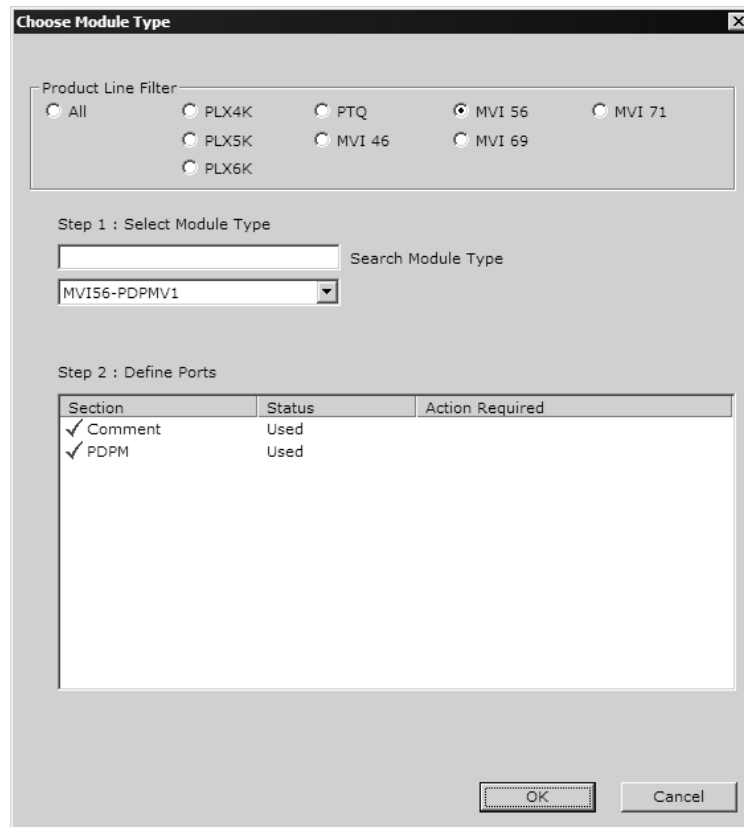
3 In the *Product Line Filter* area of the dialog box, select **MVI56**. In the *Select Module Type* dropdown list, select **MVI56-PDPMV1**, and then click **OK** to save your settings and return to the main PCB window.

2.2 Setting Module Parameters

Notice that the contents of the information pane and the configuration pane changed when you added the MVI56-PDPMV1 module to the project. The red "X" icon indicates that the module's configuration is incomplete.
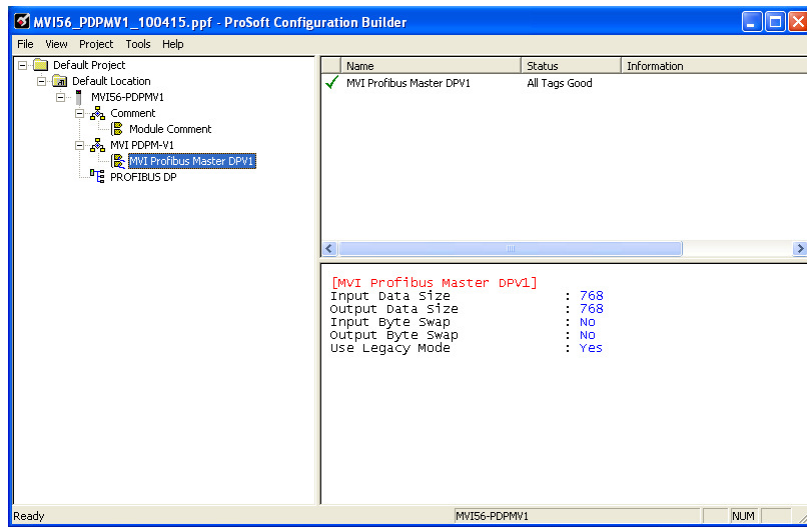
In the following steps, you will provide the missing information to make the red "X"s go away.

1   Click the plus sign **[+]** next to the module to expand the module tree, and then expand the *MVI PDPM-V1* tree.
2   Double-click the **MVI56 PROFIBUS MASTER** object. This action opens the *Edit* dialog box.
3   In the *Edit* dialog box, configure the values for **INPUT DATA SIZE** and **OUTPUT DATA SIZE** (PROFIBUS input and output words) to match the values required by your application. To change a value, select the parameter to modify in the left pane, and then type the new value in the edit field in the right pane.

For the sample application, the input and output data size values should both be **768**.
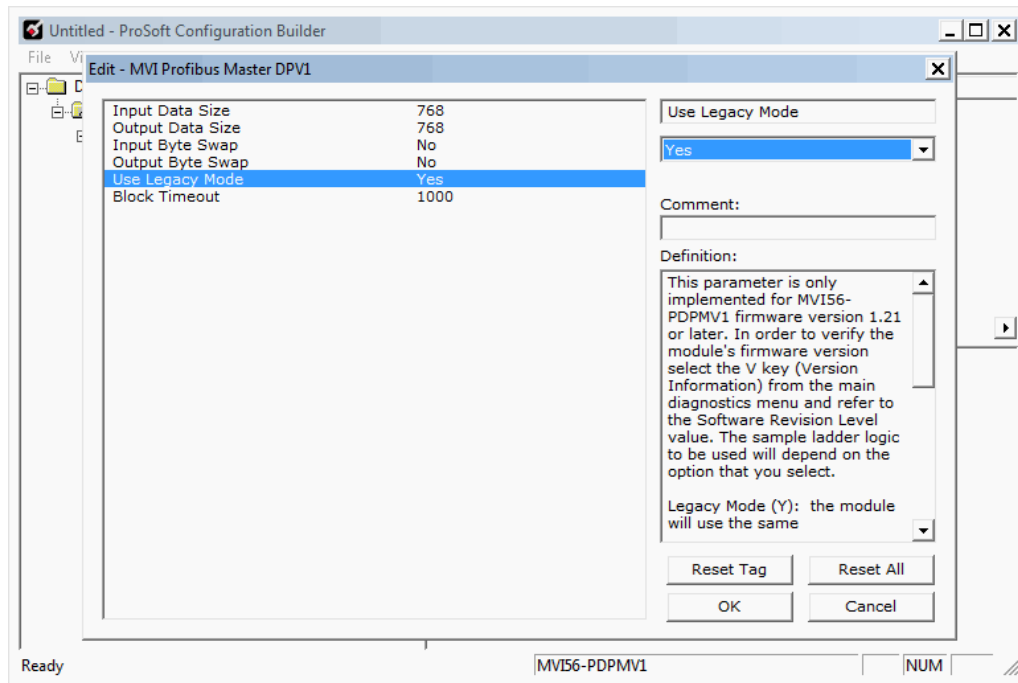
Note: Select the module mode (FLEX or LEGACY) to match the sample program previously selected as follows:
- If you chose a FLEX sample: Set USE LEGACY MODE = NO
- If you chose a LEGACY sample: Set USE LEGACY MODE = YES

4   Click **OK** to save your settings and return to the main PCB window.

At this time, you may wish to rename the **DEFAULT PROJECT** and **DEFAULT LOCATION** folders in the tree view.

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

2.3   Configuring the PROFIBUS Master

In this task, you will configure the PROFIBUS Master, and then add PROFIBUS slaves to the PROFIBUS network. When this step is complete, you will download the configuration information to the MVI56 module. You will also export the I/O maps for the processor.

1   In PCB tree view, click **[+]** to expand the *MVI56-PDPMV1* tree, and then double-click the **PROFIBUS DP** icon. This action opens the *PDPMV1 PROFIBUS Master Setup* dialog box.
2   In the *PDPMV1 PROFIBUS Master Setup* dialog box, click the **CONFIGURE PROFIBUS** button. This action opens the *ProSoft Configuration Builder for PROFIBUS* application.
3   Click **[+]** to expand the *PROFIBUS Master* tree.
4   Drag the PROFIBUS *Master* icon into the *Bus Configuration* window. This is automatically done by the software for new applications.
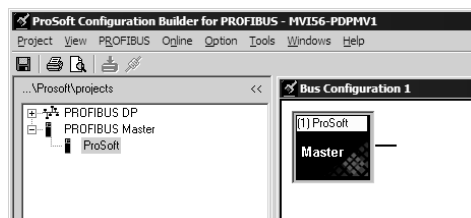
5   Double-click the **PROFIBUS MASTER** icon in the *Bus Configuration* window. This action opens the *Master Properties* dialog box.
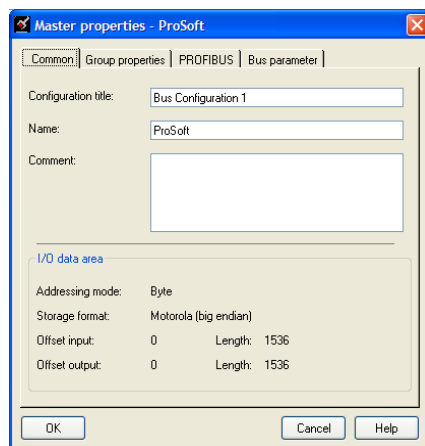
6   On the *Common* tab, name your PROFIBUS drop.

> **Note:** The *PROFIBUS* tab contains the address setting and advanced configuration settings for the Master. The default settings on this tab work best in most applications.

7   Click **OK** to save your changes and return to the *Bus Configuration* window.

2.3.1 Installing the GSD Files

ProSoft Configuration Builder (PCB) uses PROFIBUS slave device definition files (GSD files) to obtain basic configuration information about the PROFIBUS slaves you add to the network. The GSD configuration files identify the slave's capabilities so that the MVI56-PDPMV1 can communicate with it correctly. Slave device manufacturers provide the GSD files for the equipment they make. Slave device files sometimes come in various languages. When a manufacturer provides slave device files in several languages, it is a common practice to use the third letter of the file extension to indicate the language used in the file. For instance:

- o .GS**D** is the most commonly used file extension and will usually be in either English or German
- o .GS**E** will usually be in English
- o .GS**S** will usually be in Spanish
- o .GS**F** will usually be in French
- o other combinations may also be seen, as well as other languages using the letters indicated above

Follow these steps to install the GSD file or files for your slave device or devices.

Tip: GSD configuration files for popular PROFIBUS slaves and ProSoft Technology solutions are included with PCB. Before installing GSD files, browse the list of available slaves in the Tree View window to see if GSD files for your slave are already installed.
GSD files are often both model number specific as well as model revision specific. Just because you may have an older GSD file from a manufacturer for the particular make and model of your slave device does not guarantee it will work for a newer revision of that device. Be sure you obtain from the device manufacturer the correct GSD file or files for your PROFIBUS slave or slaves.

To install GSD files manually

1   In ProSoft Configuration Builder tree view, click **[+]** to expand the module tree, and then double-click the **PROFIBUS DP** icon. This action opens the *PDPMV1 PROFIBUS Master Setup* dialog box.
2   Click the **CONFIGURE PROFIBUS** button. This action opens the *ProSoft Configuration Builder for PROFIBUS* application.
3   Open the **TOOLS** menu, and then choose **INSTALL NEW GS* FILE**. This action opens a dialog box that allows you to browse for the location of the GSD configuration files to install. (Depending on the device and language used in the file, the actual extension may be ".GSD", ".GSE", ".GSS", or other combinations; hence the generic reference to ".GS*" files, where "*" is a wildcard that stands for any letter.)
4   Choose the file to install, and then click **OPEN**. If the file already exists in the configuration file path, you will be prompted to overwrite the file.
5   You will be prompted to associate the GSD configuration file with a bitmap image of the slave device. Use the **FILE / OPEN** dialog box to browse for the location of the image file to use. If you have no device-specific bitmap file, you may **CANCEL** the bitmap upload, and a generic device icon will be used in the *Bus Configuration* window for this slave device.

2.3.2 Configuring the PROFIBUS Slaves

There are two essential steps to configuring a slave:

1   Add the slave in ProSoft Configuration Builder (PCB) as a device connected to the PROFIBUS Master, specifying the slave address and any necessary input and output configuration. Download the PROFIBUS Master configuration to the MVI56-PDPMV1 module.
2   Configure the slave (using PCB or the configuration tool supplied by the manufacturer, for some PROFIBUS slaves). Verify that the slave address configured in the slave module matches the slave address configured in PCB. Download the PROFIBUS Slave configuration to the slave module.

Scanning for Slaves Manually

The following steps describe how to add and configure a Siemens EM 277 PLC as a slave module. The configuration information (.GSD file) for this module is already installed with ProSoft Configuration Builder.

1 In *ProSoft Configuration Builder for PROFIBUS*, click the plus sign **[+]** to expand the PROFIBUS DP tree.

2 Navigate to the *PLC/Siemens* folder containing the EM 277 slave device to add, and then click the plus sign **[+]** to expand the folder.
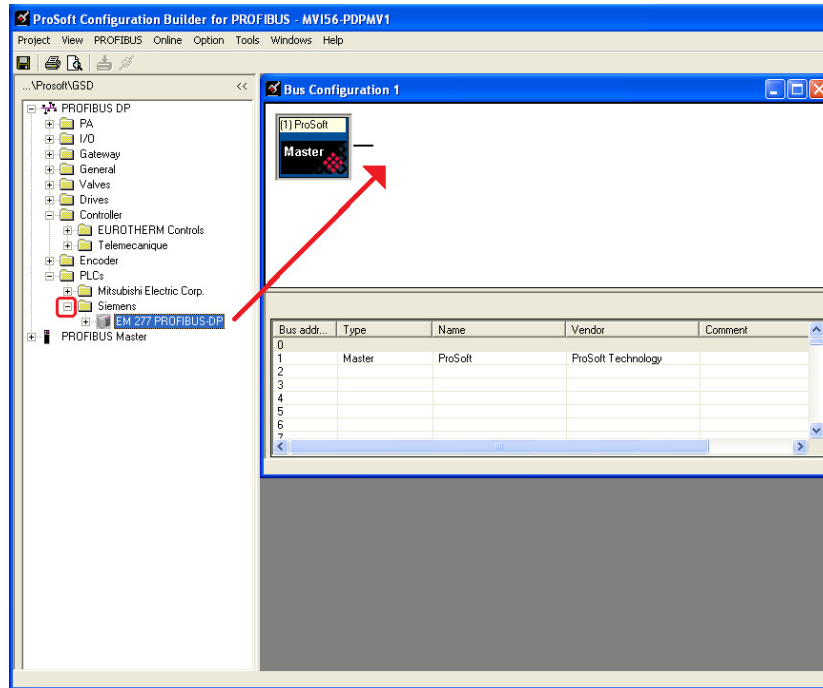
3 Drag the slave icon into the *Bus Configuration* window. This action adds the slave device and connects it to the Master in a network relationship.
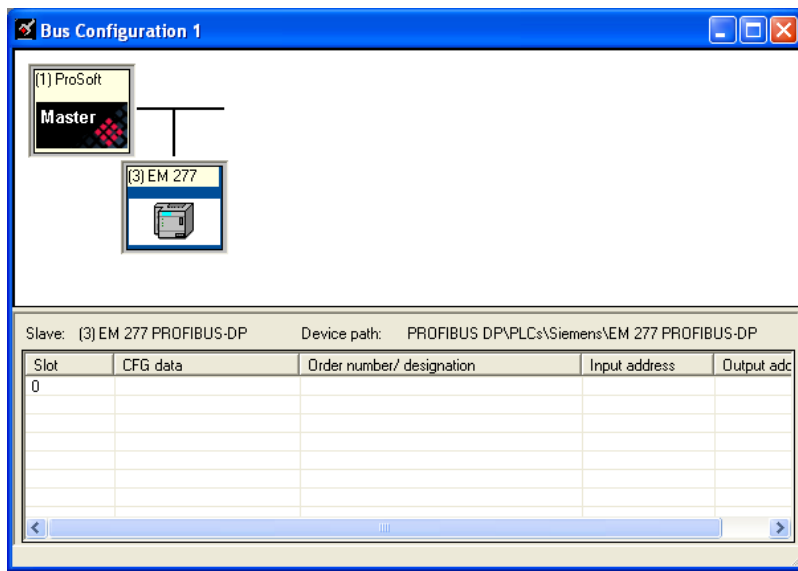
4 In the tree view, click the plus sign **[+]** to expand the slave device you added. This action opens a list of device configuration values. The following illustration shows the possible input/output configuration values for a Siemens EM 277 PLC Slave. The selections available for other devices will be different, so you should review the specifications for the product you are installing in order to determine the correct values to use.
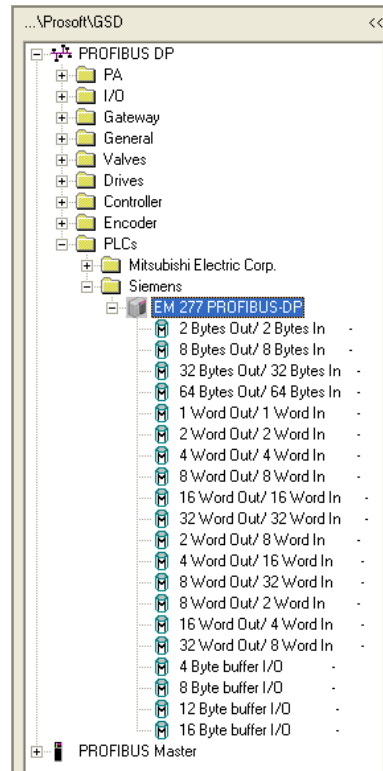
5 Drag the input and output parameters to the *Slot Location Grid (Subscriber List)* below the *Bus Configuration* window. This view displays the slot number, configuration data, and starting input and output addresses. The MVI56-PDPMV1 Master uses this information to identify and communicate with individual slaves on the network.

For this example, we will configure thirty-two (32) words of output and eight (8) words of input. These input and output words are assigned to addresses within the MVI56-PDPMV1 module's internal database.
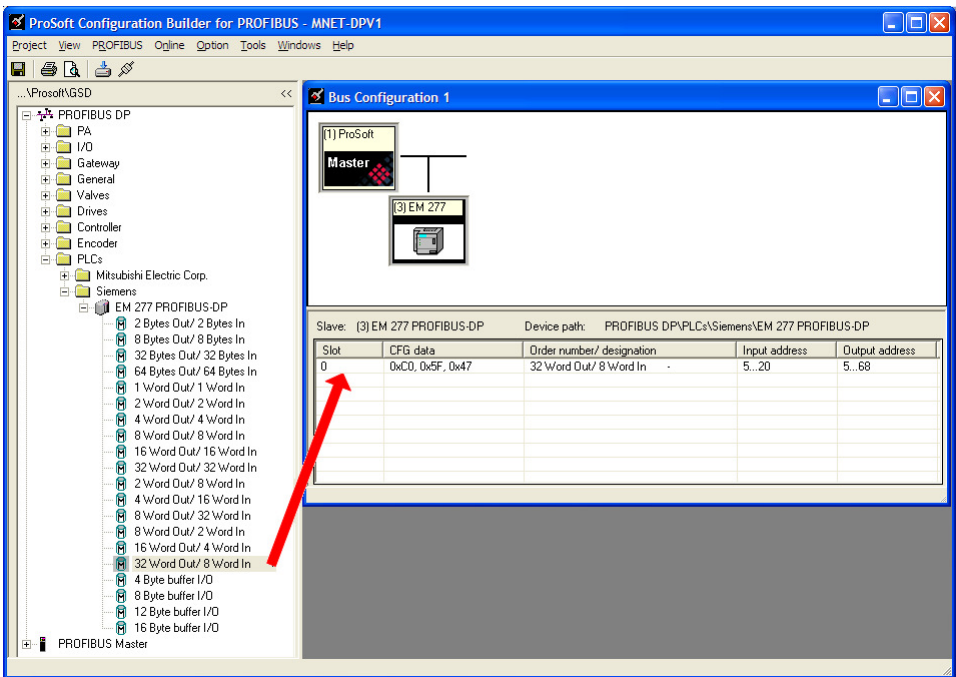
For each new slave added to the PROFIBUS network, PCB automatically converts the input/output byte addresses to word input/output addresses for the tag database in the processor.

Tip: To make it easier to view the data from individual slaves, give each slave a meaningful name, and place each slave's I/O areas in contiguous blocks. Use the Processor Network Memory Map (page 104) as an aid to cross-referencing the slave I/O data in RSLogix 5000.

6   Double-click the **SLAVE** icon to view the *Slave Properties*.
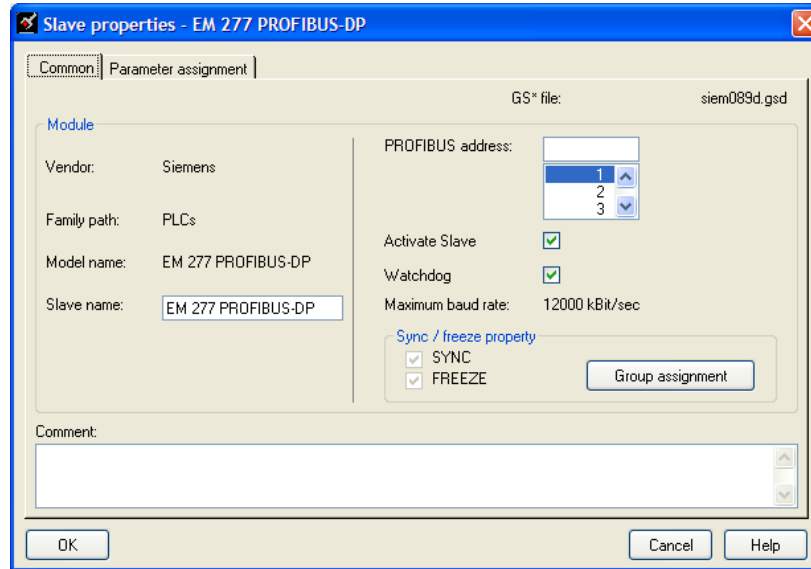
PCB automatically assigns a PROFIBUS address to each new slave. The address assignment begins at address **3**, and is incremented by 1 for each new slave added to the network. You can change the address in the *Common* tab of the *Slave Properties* dialog box. PCB will not allow you to assign a PROFIBUS address that is already in use by another device on this network.

Leave the remaining settings unchanged for now, and click **OK** to close the *Slave Properties* dialog box.

7   Repeat steps 2 through 6 for all slaves you intend to place on the network.

8   When you are finished adding slaves, open the **PROJECT** menu and choose **EXIT**. Click **YES** to save the project and return to the *PROFIBUS Master Setup* dialog box.

Using The Autoscan Feature

The concept of *Automatic network scanning* means that the user can instruct the *Bus Configuration* window to automatically gather information about slaves that are connected to the network. When the scan is completed the user can adopt the detected slaves to the *bus configuration* and download to the Master.

This is a quick way to get a network up and running. However, one should be aware that it is not guaranteed that any particular slave will enter data exchange since the user parameter data might not match. This is especially obvious if no associated GSD-file is found during the network scan, this means that no user parameter data would be sent to the slave.

NETWORK SCAN is selectable from the *Online* menu as well as from the drop-down menu for the **MASTER** icon.
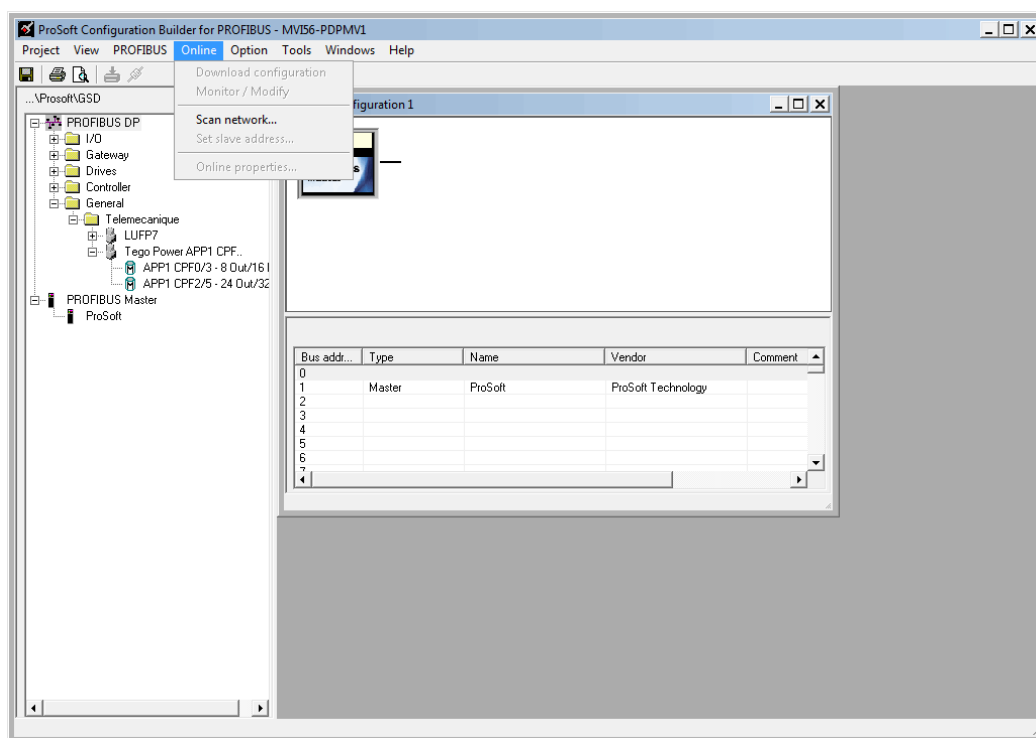
When the download is completed, the *PROFIBUS Master Configuration* window will initialize the Master to operate as a *Class 2 Master only*. In this mode it is possible to initialize the Master even if the database does not contain any slaves.

After successful initialization, the *PROFIBUS Master Configuration* window will issue the following mailboxes in order to gather information about the connected slaves:

1 1. Send FB_APPL_GET_LIVE_LIST in order to detect connected slaves,
2 2. Send FB_APPL_GET_SLAVE_DIAG (external request) to all devices identified as slaves according to the Live list.
3 3. Send FB_APPL_GET_SLAVE_CONFIG to all devices identified as slaves according to the Live list.

When the information is collected the *PROFIBUS Master Configuration* window will find a matching GSD-file and extract information from it. Refer to the flowchart below for this sequence:

```
                          ┌─────────────────────┐
                          │  Start network scan │
                          └──────────┬──────────┘
                                     │
                          ┌──────────▼──────────────────┐
                          │ Gather information about all │
                          │ slaves connected to the      │
                          │ network::                    │
                          │ -LiveList (find slaves)      │
                          │ -SlaveDiag (fetch Ident number)│
                          │ -GetCfg (fetch Identifier bytes)│
                          └──────────┬──────────────────┘
                                     │
                     NA = slave with lowest node address
                     ID = Ident number for slave NA
                     CFG = Identifier bytes for slave NA
                                     │
                                    ◇ Does GSD(s) exist that
                                      matches ID for slave NA?
                          No ────────  (refer to section "GSD selection
                                       algorithm")
                                     │ Yes
                          ┌──────────▼──────────────────┐
                          │ Find Module(s) in GSD that   │
                          │ matches CFG                  │
                          │ (refer to section "Module    │
                          │ selection algorithm")        │
                          └──────────┬──────────────────┘
```

GSD Selection Algorithm

If two or more matching GSD-files are found, the first one found should be selected. The other compatible files should be stored so that the user can select one of them instead. If the user selects another GSD-file, the *PROFIBUS Master Configuration* window will run through the *Module Selection Algorithm* (described below) again.

Module Selection Algorithm

The algorithm used to find modules in the GSD based on the Identifier byte(s) is as follows:

Select the module that matches the largest number of Identifier bytes. If the GSD contains two or more modules with the exact set of Identifier bytes, use the first module found.

Example:

If a slave responds with identifier bytes: 0x11, 0x21, 0x31 and that the associated GSD-file contains five modules: "A" = 0x11, "B" = 0x21, "C" = 0x31, "AB" = 0x11, 0x21 and "BC" = 0x21, 0x31. The *PROFIBUS Master Configuration* window will then select modules "AB" and "C".

> **Note**: If no matching module is found in the GSD, The *PROFIBUS Master Configuration* window will display the identifier byte(s) instead.

Network scan window

The information extracted from the GSD-file(s) will be displayed in the *Network scan* window.

Select

In this column all found slaves will be marked as selected by default, except for slaves with the special address 126 (refer to the next section that describes the Address column). Only selected slaves will be added to the *PROFIBUS Master Configuration* when the **ADOPT SELECTED SLAVES** button is clicked.

Address

In this column the node address of the slaves will be displayed. Found slaves should be listed in ascending order according to their node addresses.

Special address 126 -Set Slave address:

If a slave with node address 126 is detected during the network scan, the *PROFIBUS Master Configuration* window will display the address in red color. It will not be possible for the user to adopt the slave to the configuration since it is not allowed to exchange data with devices having this address. The check box in the *Select* column will be grayed out.
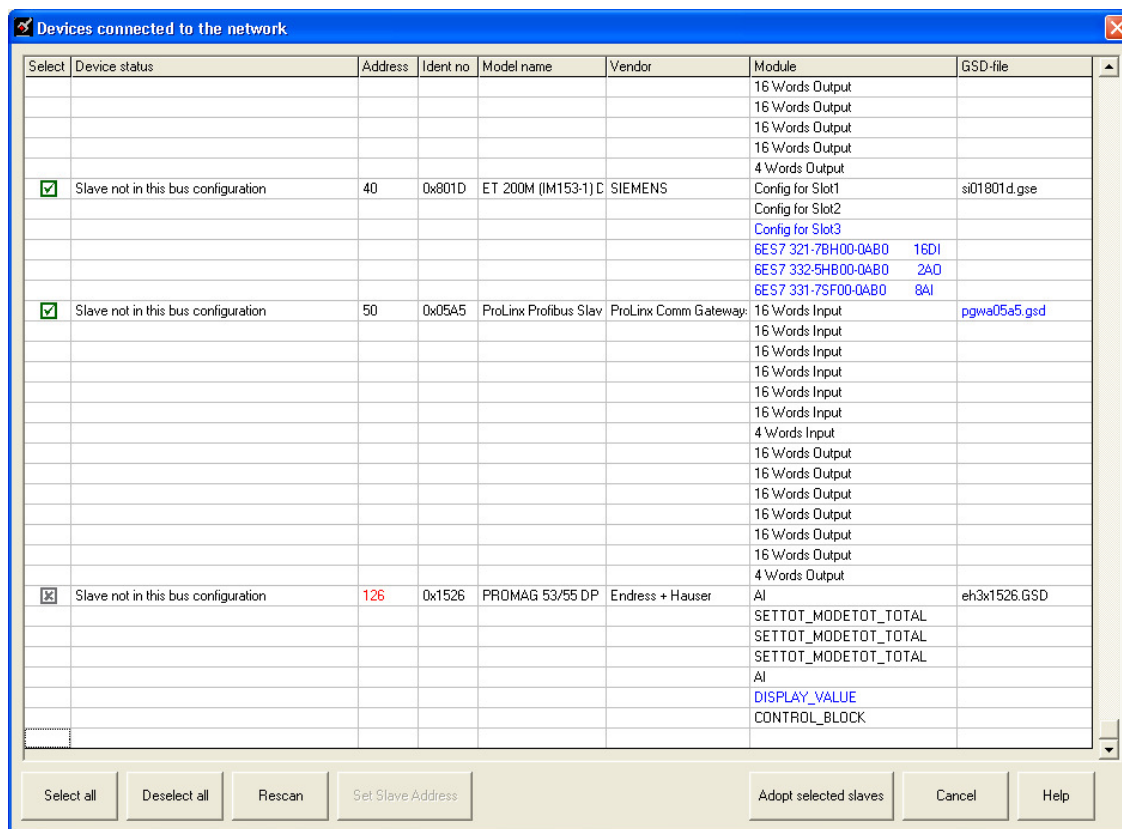
To be able to adopt a slave with address 126 the user must first assign a valid address by clicking the icon next to the node address. By doing so the *Set Slave Address* dialog box is started.

Note that the *Old slave address* is preset to a value of 126 that is not editable (grayed out).
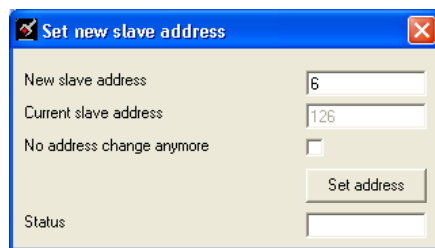
If the Slave is in the configuration already then it will not affect the addressing.

Example:
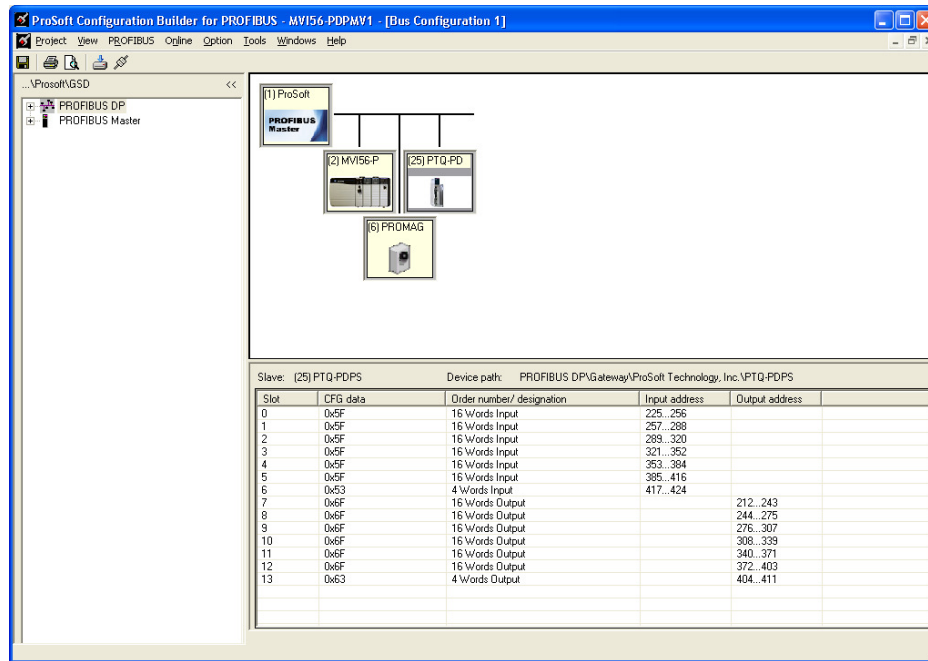
After scanning, the network finds these other slaves: 2, 6, 25, and 40
Slaves 2, 6, and 25 are found, but are marked as in the bus configuration (the mapping of the inputs and outputs will not be affected)
Slaves 40 is new and could be added and the input/output addressing will be appended to the end as shown on the last screen.
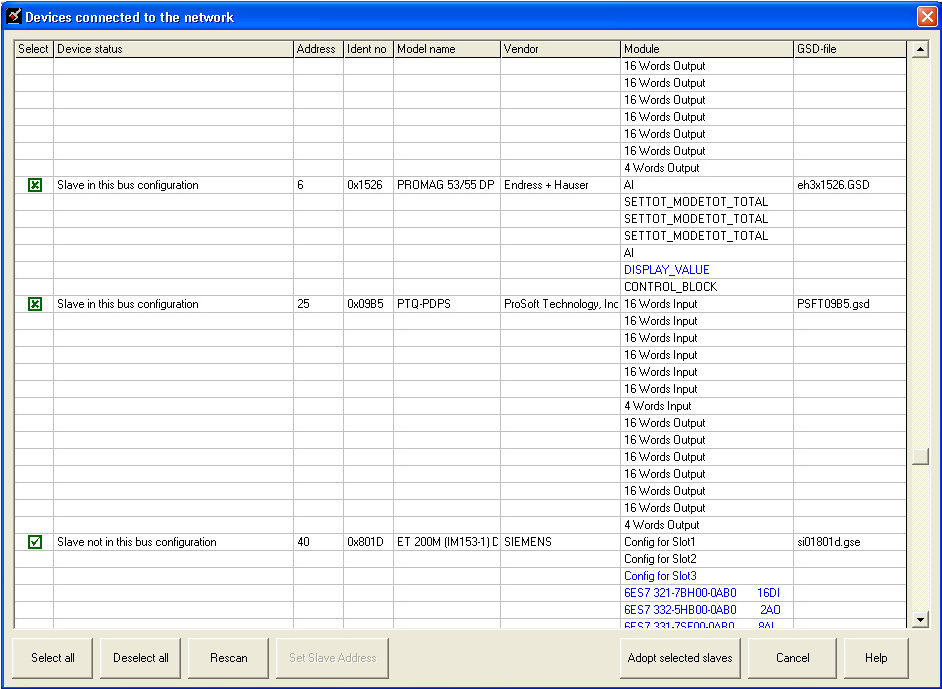
The *PROFIBUS Master Configuration* window will prevent the user from selecting a *New slave address* that is already occupied by another device; this includes detected Master stations as well. If the user selects an occupied address, a message similar to the one shown here will open.
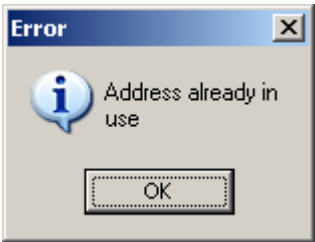
When an address has been successfully assigned, the *PROFIBUS Master Configuration* window will update the *Network scan* window as shown here. The node address will be updated to the one that the user selected in the *Set Slave* dialog box. The check box in the *Select* column will be marked allowing the user to adopt the slave to the configuration.

Slave

In this column the name of the slave as stated in the assigned GSD-file will be displayed. If no matching GSD-file is found the Ident number will be displayed in red color in the drop-down list.

Module

This column shows the name of the module(s) as stated in the assigned GSD-file, which matches the Identifier byte(s) derived from the *GetCfg* mailbox message. If no GSD-file or no matching module is found the Identifier byte(s) will be displayed in red color. If the configuration for a slave is constructed of several modules, the modules will be listed under each other.

If there is more than one module in the GSD-file that matches the Identifer bytes, the first matching module will be displayed in blue color in a drop-down list. The drop-down list will contain all other matching modules so that the user can select the desired one.

Note: Only modules that have the exact same Identifer bytes as the first matching module will be displayed in the drop-down list

GSD-file

This column shows the name of the GSD-file that matches the Ident number derived from the *SlaveDiag* mailbox message. If there are more files with the same Ident number in the device catalog, the first matching GSD-file will be displayed in blue color in a drop-down list.

This could be the case if the device catalog contains two or more brand labeled devices, or GSD-files for two or more languages (for example NICEDEV.GSD and NICEDEV.GSE) exist.

Note: If the user selects another GSD-file, The *PROFIBUS Master Configuration* window will update the modules for that slave accordingly.

If no GSD-file is found the user will be able to copy the expected GSD to the device catalog by clicking the icon next to the text *No GSD found*. This will start the Install *new GS*-file* dialog box. When the file is installed, the *PROFIBUS Master Configuration* window will verify that the installed file matches the slave and update the modules for the slave accordingly.

Rescan

Pressing the **YES** button will trigger a new network scan. Before proceeding with the scan a message similar to the one below will appear. If a new scan is accepted, detected slaves found during the previous scan will be lost.
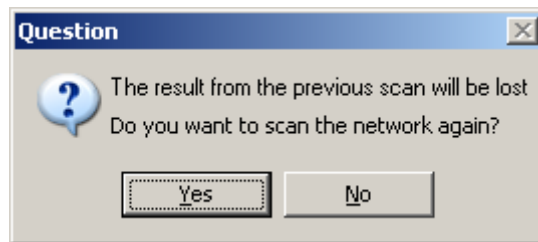
Note: The autoscan function cannot delete slaves in the current configuration, it can only change (if slaves differs) or add. The slave must be manually removed from the bus configuration in order to be able to add it again via the scan window.

Adopt selected slaves

Pressing this button will cause all selected slaves to be adopted to the *PROFIBUS Master Configuration* window. Before carrying on with this action a message similar to the one below will appear.
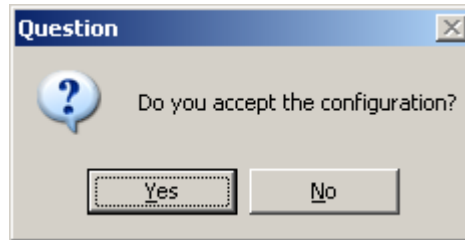
If accepted, the *network scan* window will close and the *PROFIBUS Master Configuration* window will be populated with the slaves that were found during the network scan.
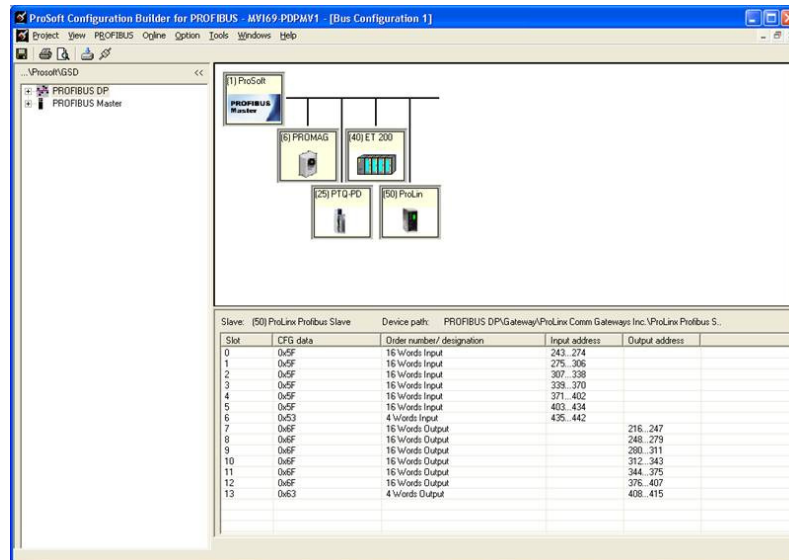
Note: *Slave:* is equal to the Ident number and that the *Device path:* and *Order number/designation* fields are left empty.

Cancel and Help

If the **CANCEL** button is pressed a message similar to the one below will appear.

If the **HELP** button is pressed the online help will start.

Set_Param (SAP61)

ProSoft PROFIBUS slave (PDPS) devices have a configurable parameter for SPC3 User Prm Byte. The following illustration shows the value of this parameter in *ProSoft Configuration Builder for PROFIBUS*, the configuration tool for ProSoft PROFIBUS Master devices.

Parameter Data Structure

SPC3 evaluates the first seven data bytes (without user prm data), or the first eight data bytes (with user prm data). The first seven bytes are specified according to the standard. The eighth byte is used for SPC3-specific communications. The additional bytes are available to the application.

Byte	Bit Position								Designation
	7	6	5	4	3	2	1	0	
0	Lock Reg	Unio Req	Sync Req	Free Req	WD on	Res	Res	Res	Station status
1									WD_Fact_1
2									WD_Fact_2
3									MinTSDR
4									Ident_Number_High
5									Ident_Number_Low
6									Group_Ident
7									Spec_User_Prm_Byte
8 to 243									User_Prm_Data

Byte 7	Spec_User_Prm_Byte		
Bit	Name	Significance	Default State
0	Dis_Startbit	The start bit monitoring in the receiver is switched off with this bit	Dis_Startbit = 1, Start bit monitoring is switched off.
1	Dis_Stopbit	Stop bit monitoring in the receiver is switched off with this bit	Dis_Stopbit = 0 Stop bit monitoring is not switched off.
2	WD_Base	This bit specifies the time base used to clock the watchdog. WD_Base = 0: time base 10 ms WD_Base = 1: time base 1 ms	WD_Base = 0 The time base is 10 ms.
3 to 4	Res	To be parameterized with 0	0
5	Publisher_Enable	DXB-publisher-functionality of the SPC3 is activated with this bit	Publisher_Enable = 0, DXB-request-telegrams are ignored; Publisher_Enable = 1, DXB-request-telegrams are processed
6 to 7	Res	To be parameterized with 0	0

2.3.3 Printing the Processor Network Memory Map

The *Processor Memory Map* dialog box uses the information about your
PROFIBUS Master and slaves to display the data types you configured. You will
need this information to determine the memory areas your application will use in
the ControlLogix processor.

To view or print the Processor Network Memory Map

1   On the *Master Setup* dialog box, click the **SHOW MEMORY MAP** button, near
the bottom of the window.
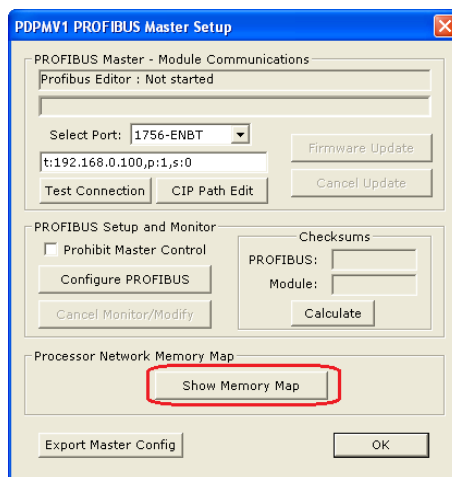
This action opens the *inRAx Data Type* dialog box.
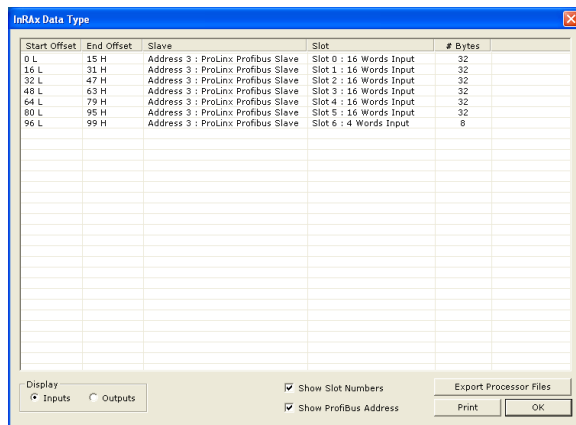
2   Notice that there are buttons in the *Display* area of the dialog box to show
INPUTS or **OUTPUTS**. These Input and Output maps correspond to the Input
and Output data you configured for the PROFIBUS Slaves (page 86). Notice
also that there are **CHECK BOXES** to optionally show (if checked) or hide (if not
checked) the slot numbers and PROFIBUS addresses.

3   Click **PRINT** to print the input and output maps for reference. Note that you
    must do this separately for each of input and output map by first clicking on
    the **INPUTS** radio button and then the **PRINT** button; then, click on the
    OUTPUTS radio button and then the **PRINT** button, again.
4   When you have finished printing the processor memory maps, you can click
    OK to close the dialog box and click **OK** again to close the *Master Setup*
    dialog box; or, leave them open for the next step, exporting the processor
    files.
5   Keep the printed memory maps available so you can refer to them when you
    configure the sample ladder logic in RSLogix. The sample ladder logic
    contains input and output arrays that must be cross-referenced with the
    variables.

2.3.4   Exporting the Processor Files Prior to PCB v2.2.0

NOTE: This is an optional step which is not required to set up your application, regardless of the
ProSoft Configuration Builder (PCB) version you are using. The release of PCB version 2.2.0 has
made this particular procedure obsolete. This topic has been left in the User Manual solely as
historical reference, in case you need to understand how older versions of PCB previously
functioned.

PCB versions prior to version 2.2.0 will create a text file containing the Processor
Network Memory Map that you printed in the previous step, which can be
imported into RSLogix 5000 as user-defined data types *MVI56PDPMV1_PBInF*
and *MVI56PDPMV1_PBOutF*. This simplifies the task of adding data types to
handle input and output from the PROFIBUS slaves in your ladder logic.

The sample program stores the PROFIBUS I/O data into two SINT arrays. If you
choose to use this export option, the data types exported will reflect the data
types that you have configured for your PROFIBUS network slave configuration.
If you decide to use this functionality, you must develop extra ladder logic to copy
the PROFIBUS data between controller tags defined with these data types and
the standard SINT arrays.

To export the processor files from PCB (versions prior to v2.2.0 only)

1 In the *inRAx Data Type* dialog box, click **EXPORT PROCESSOR FILES**. ProSoft Configuration Builder will suggest a file name with the extension .L5X (RSLogix 5000 export file).

InRAx Data Type

Start Offset	End Offset	Slave	Slot	# Bytes
0 L	15 H	Address 3 : ProLinx Profibus Slave	Slot 0 : 16 Words Input	32
16 L	31 H	Address 3 : ProLinx Profibus Slave	Slot 1 : 16 Words Input	32
32 L	47 H	Address 3 : ProLinx Profibus Slave	Slot 2 : 16 Words Input	32
48 L	63 H	Address 3 : ProLinx Profibus Slave	Slot 3 : 16 Words Input	32
64 L	79 H	Address 3 : ProLinx Profibus Slave	Slot 4 : 16 Words Input	32
80 L	95 H	Address 3 : ProLinx Profibus Slave	Slot 5 : 16 Words Input	32
96 L	99 H	Address 3 : ProLinx Profibus Slave	Slot 6 : 4 Words Input	8

Display
⦿ Inputs ○ Outputs

☑ Show Slot Numbers
☑ Show ProfiBus Address

Export Processor Files
Print | OK

2 Choose a location for the file that you can access easily from RSLogix 5000 (*My Documents* or *Desktop*, for example), and then click **SAVE**.

3 Click **OK** to close the *inRAx Data Type* dialog box.

2.3.5 Exporting the Processor Files for PCB v2.2.0 or later

> **NOTE:** The files created in this step are for use with RSLogix 5000, version 16 or higher. Earlier versions of RSLogix did not support the Add-On Instructions (AOIs) created by this procedures; so the files created here will not work with RSLogix 5000, versions 15 or earlier. For those earlier versions of RSLogix, use the standard sample ladder logic files without AOIs. For details, see Using the MVI56-PDPMV1 Sample Ladder Logic (page 23).

For your convenience, beginning with version 2.2.0, and later, ProSoft Configuration Builder (PCB) will create a complete, customized Add-On Instruction (AOI) Rung Export .L5X file that has been tailored to the specific configuration you created for your PROFIBUS network and module configuration. This makes integrating the MVI56-PDPMV1 module into a new or existing application easier than it has ever been.

For additional details on how to integrate the rung import file into your RSLogix 5000 project, see Importing the Ladder Rung - LEGACY (page 27) or Importing the Ladder Rung - FLEX (page 53).

The following procedure will work for either mode and will export appropriate but different rung import files, depending on which option is selected in the **USE LEGACY MODE** parameter.

To export the AOI Rung Import file from PCB (for version 2.2.0 and later only)

1 In the *inRAx Data Type* dialog box, click **EXPORT PROCESSOR FILES**.
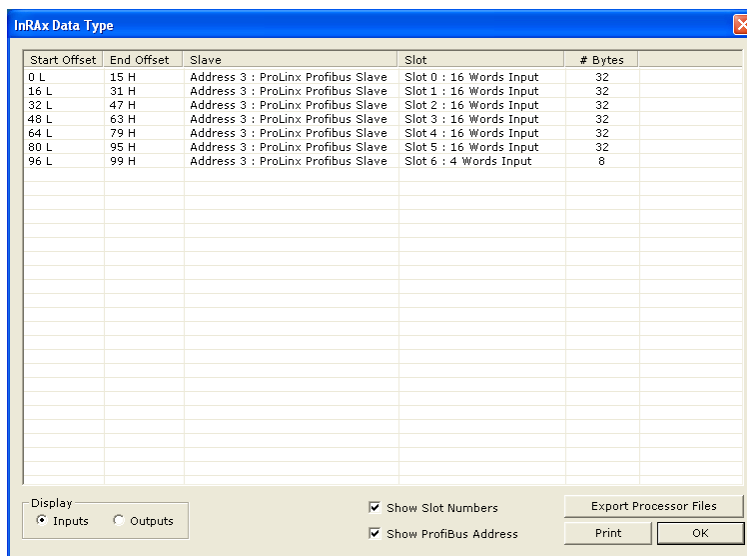
ProSoft Configuration Builder will suggest a file name with the extension L5X (RSLogix AOI rung import file). It is highly recommended that you use the default filename.

For LEGACY Mode:
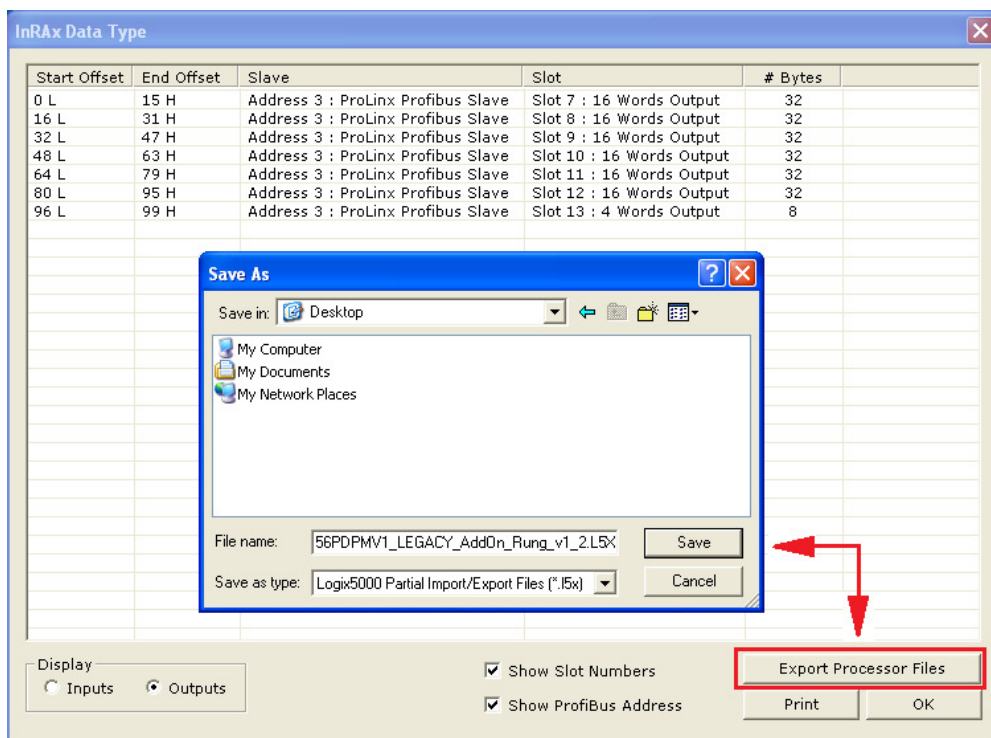
For FLEX Mode:
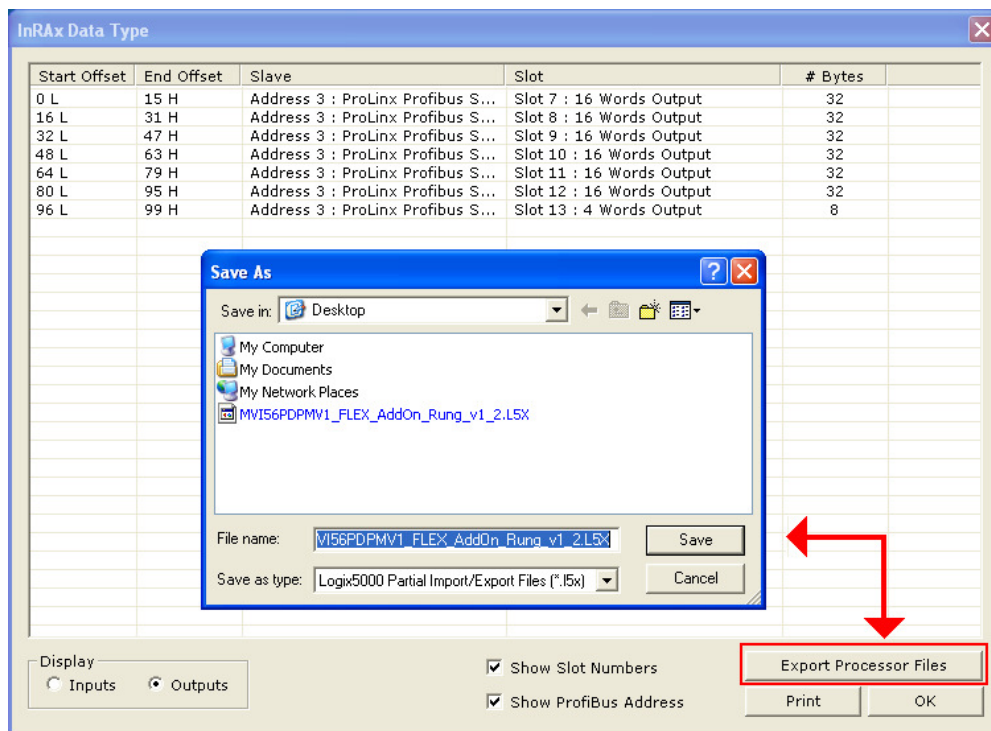
2 Choose a location for the file that you can access easily from RSLogix 5000 (*My Documents* or *Desktop*, for example), and then click **SAVE**.
3 Click **OK** to close the *inRAx Data Type* dialog box.

NOTE: If you make any changes to your module or PROFIBUS configurations, you will probably want to create new AOI Rung Import files using this procedure. If you create new files, before you can import them into your existing application, some of the previously imported controller tags and ladder logic will need to be manually deleted. For additional information, please see the topic *Changing an Existing Configuration and Importing Changes (page 72).*

2.3.6 Backing Up the Project

In this step, you will create a backup copy of your project and configuration files. The backup procedure saves your data for reuse on another machine, or allows you to restore your data in the event of a system failure.

To save your project and configuration files

1 In *ProSoft Configuration Builder* tree view, click **[+]** to expand the *MVI56-PDPMV1* tree, and then double-click the **PROFIBUS DP** icon. This action opens the *PDPMV1 PROFIBUS Master Setup* dialog box.
2 In the *PDPMV1 PROFIBUS Master Setup* dialog box, click the **EXPORT MASTER CONFIG** button. This action saves the PROFIBUS network configuration for your module in an XML file. The recommended location for this file is your *My Documents* folder.

Tip: You can use the XML file created by ProSoft Configuration Builder in this step to simplify the task of configuring additional PROFIBUS network modules. Because it saves the entire network configuration, you can add modules quickly by modifying only the items that are unique for each device, typically the slot number and I/O addresses. To use this saved configuration, open Windows Explorer, navigate to the folder where you saved the Master Configuration XML file, and then drag the file onto the new PROFIBUS DP icon in the ProSoft Configuration Builder tree view.

---

3 In *ProSoft Configuration Builder*, open the **FILE** menu, and then choose **SAVE AS**.
4 Name the project file, and click **SAVE**. The recommended location for this file is your *My Documents* folder.

Note: All PCB project files and module-related files are automatically saved to C:\PCBExportfFiles.

A complete backup consists of the Project and Master Configuration files, plus the GSD configuration files. The default location for the GSD files is *C:\Documents and Settings\All Users\Application Data\ProSoft\GSD (Windows XP / 2000)* or *C:\My Documents*.

PCB does this complete backup for you automatically. The default location for these backup files is *C:\PCBExportFiles.* All the files associated with your PCB configuration will be stored in a folder with the same name as the name you used to save your PCB configuration (.ppf) file. When you exit PCB, you will be prompted to overwrite your *Export* folder files.
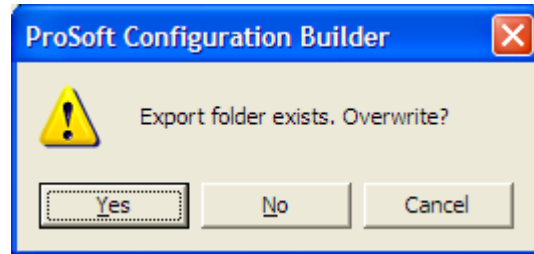
If you have made changes to your configuration settings, you should click the **YES** button every time you see this dialog box to have the backup files updated. Having all the files for your PCB configuration stored in one folder makes it easier to transfer the application from one system to the other or to send your files to ProSoft Technical Support when you need assistance.

2.4   Downloading the Project to the Module

1   Right-click the MVI56-PDPMV1 module and choose **DOWNLOAD FROM PC TO DEVICE**.
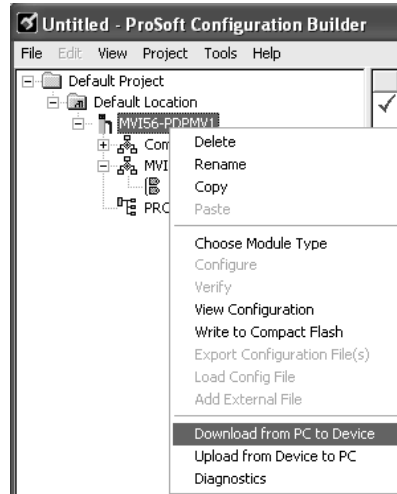
2   In the *Select Connection Type* dropdown list, choose **1756-ENBT**. The default path appears in the text box, as shown in the following illustration.
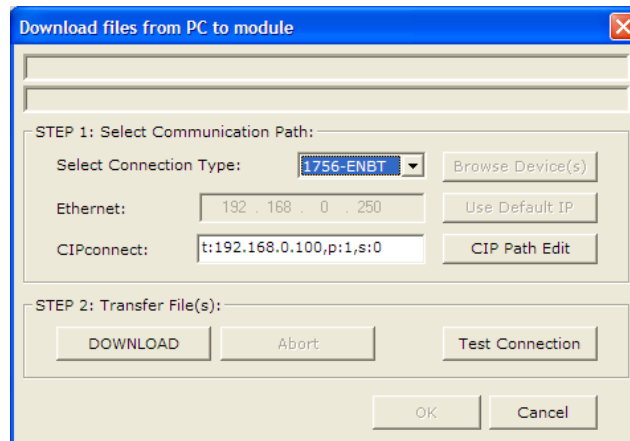
3 Click **CIP PATH EDIT** to open the *CIPconnect Path Editor* dialog box.
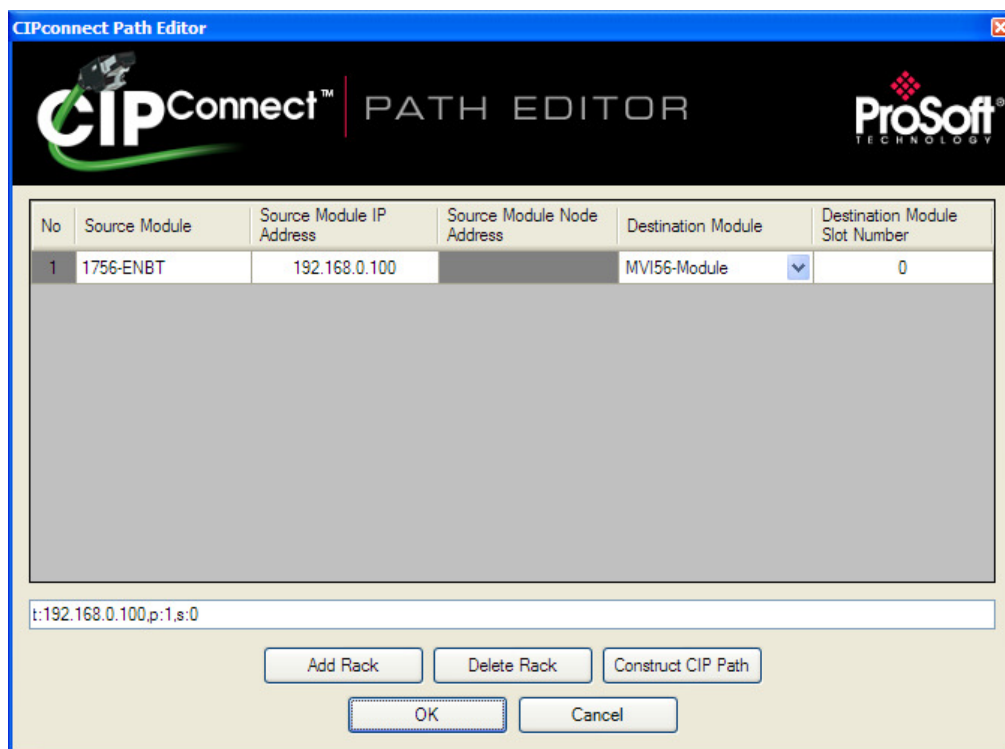
The *CIPconnect Path Editor* allows you to define the path between the PC and the MVI56-PDPMV1 module. The first connection from the PC is always the 1756-ENBT module (Ethernet/IP).

Each row corresponds to a physical rack in the CIP path.

▪ If the MVI56-PDPMV1 module is located in the same rack as the first 1756-ENBT module, you should only set rack 1 and configure the associated parameters.
▪ If the MVI56-PDPMV1 is available in a remote rack (accessible through ControlNet or EtherNet/IP) then the user should include all racks (through the **ADD RACK** button).

Parameter	Description
Source Module	Source module type. This field is automatically selected depending on the destination module of the last rack (1756-CNB or 1756-ENBT).
Source Module IP Address	IP address of the source module (only applicable for 1756-ENBT)
Source Module Node Address	Node address of the source module (only applicable for 1756-CNB)
Destination Module	Select the destination module associated to the source module in the rack. The connection between the source and destination modules is performed through the backplane.
Destination Module Slot Number	The slot number where the destination module is located.

To use the CIPconnect Path Editor, follow these steps.

1 Configure the path between the 1756-ENBT connected to your PC and the MVI56-PDPMV1 module. If the module is located in a remote rack, add more racks to configure the full path. The path can only contain ControlNet or EtherNet/IP networks. The maximum number of supported racks is six.
2 Click **CONSTRUCT CIP PATH** to build the path in text format
3 Click **OK** to confirm the configured path.

The following examples should provide a better understanding on how to set up the path for your network.

2.4.1 Example 1: Local Rack Application

For this example, the MVI56-PDPMV1 module is located in the same rack as the 1756-ENBT that is connected to the PC.
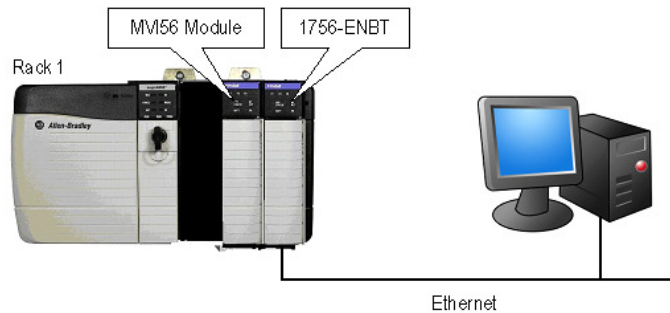
Rack 1

Slot	Module	Network Address
0	ControlLogix Processor	-
1	Any	-
2	MVI56-PDPMV1	-
3	1756-ENBT	IP=105.102.0.12

1 In the *Download* window, click **CIP PATH EDIT.**

2 Configure the path as shown in the following illustration, and click
   CONSTRUCT CIP PATH to build the path in text format.

3 Check the new path in the *CIPconnect* text box.

4 Click **TEST CONNECTION** to verify that the physical path is available. The following message should be displayed upon success.

5 Click **DOWNLOAD** to download the configuration files to the module through the path.

You can also monitor the PROFIBUS network through the CIPconnect path. Refer to the PROFIBUS DP section in ProSoft Configuration Builder, and verify the same path you have just created through the Download button. You can also re-edit the path through the CIP Path Edit button.

1 Click **CONFIGURE PROFIBUS.**

2 Select **ONLINE-MONITOR/MODIFY** to monitor the PROFIBUS network through the CIPconnect path .

The configured slave(s) should have a green border indicating successful
PROFIBUS communication.

2.4.2 Example 2: Remote Rack Application

For this example, the MVI56-PDPMV1 module is located in a remote rack
accessible through ControlNet, as shown in the following illustration.
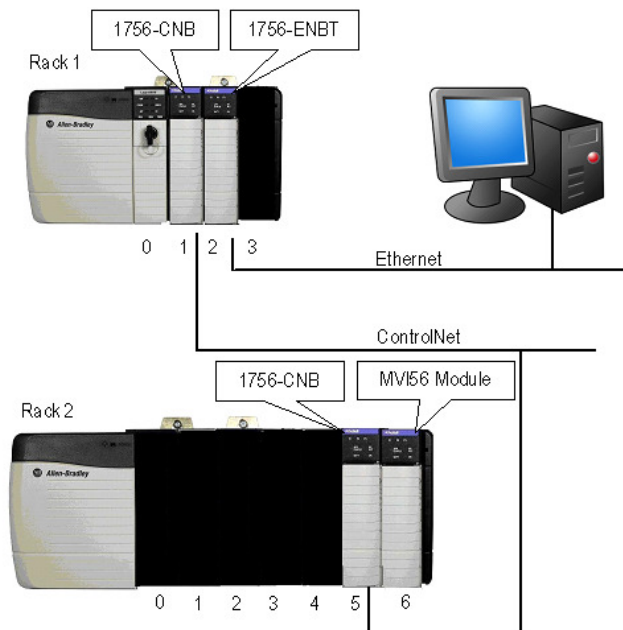
Rack 1

Slot	Module	Network Address
0	ControlLogix Processor	-
1	1756-CNB	Node = 1
2	Any	-
3	1756-ENBT	IP=105.102.0.12

Rack 2

Slot	Module	Network Address
0	Any	-
1	Any	-
2	Any	-
3	Any	-
4	Any	-
5	1756-CNB	Node = 2
6	MVI56-PDPMV1	-

1 In the *Download* window, click **CIP PATH EDIT**.
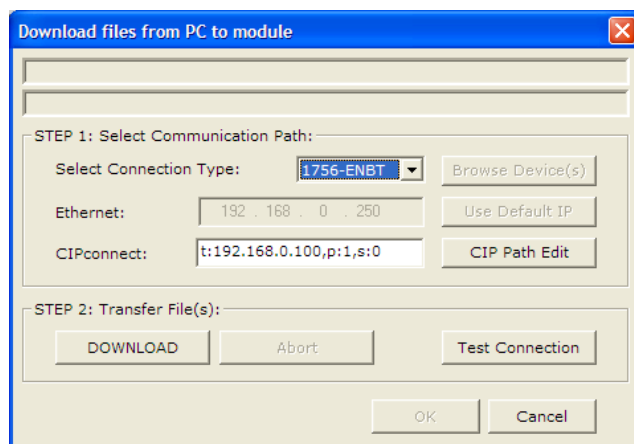
2 Configure the path as shown in the following illustration for this example and click **CONSTRUCT CIP PATH** to build the path in text format.

3 Check the new path in the *CIPconnect* text box.

4   Click **TEST CONNECTION** to verify that the physical path is available. The following message should be displayed upon success.

5   Click **DOWNLOAD** to download the configuration files to the module through the path.

You can also monitor the PROFIBUS network through the CIPconnect path. Refer to the PROFIBUS DP section in ProSoft Configuration Builder, and verify the same path you have just created through the Download button. You can also re-edit the path through the CIP Path Edit button.

1 Click **CONFIGURE PROFIBUS.**

2 Select **ONLINE-MONITOR/MODIFY** to monitor the PROFIBUS network through the CIPconnect path .

The configured slave(s) should have a green border indicating successful
PROFIBUS communication.

2.5 Verifying Correct Operation

The final step is to verify that the MVI56-PDPMV1 module is communicating with the PROFIBUS slave. Choose the method that best suits your connection and proximity to the modules and the processor.

- If you are able to view the module directly: Look at the PROFIBUS LEDs on the MVI56-PDPMV1 (page 123)
- If you are able to connect to the module using ProSoft Configuration Builder and a serial cable: View the Fieldbus Data from the MVI56-PDPMV1's Configuration/Debug menu (page 126)
- If you are able to connect to the processor using RSLogix 5000: View the Controller Tags in RSLogix 5000 (page 129)
- If you are able to connect to the processor using RSLogix 5000: Send a Mailbox Message in RSLogix 5000 (page 130)

2.5.1 Checking the PROFIBUS LEDs on the MVI56-PDPMV1

- If all four PROFIBUS LEDs are illuminated green, then the PROFIBUS Master is communicating and exchanging data with all configured PROFIBUS slaves.
- If the COM STAT LED is green and flashing, the PROFIBUS Master is communicating and exchanging data with at least one of the configured slaves.
- If the COM STAT LED is red, there is a communication or configuration error.
- If the COM STAT LED is off, the PROFIBUS Master is not communicating with any slaves. The most likely reason is that the slaves are not correctly configured.
- After restarting the MVI56-PDPMV1 Master and the PROFIBUS slave, look at the PROFIBUS cable connection. If all the PROFIBUS LEDs on the MVI56-PDPMV1 are illuminated green, then the Master is communicating successfully with all slaves.

2.5.2 Viewing the Online Status of the PROFIBUS Network

Note: For this procedure, you must connect a serial cable from the serial port on your PC to the RJ45 to DB9M adaptor cable on the MVI56-PDPMV1 module.

1 In *ProSoft Configuration Builder for PROFIBUS*, open the **ONLINE** menu, and then choose **MONITOR/MODIFY**. ProSoft Configuration Builder will establish communication with the MVI56-PDPMV1 module, and will indicate communication status.
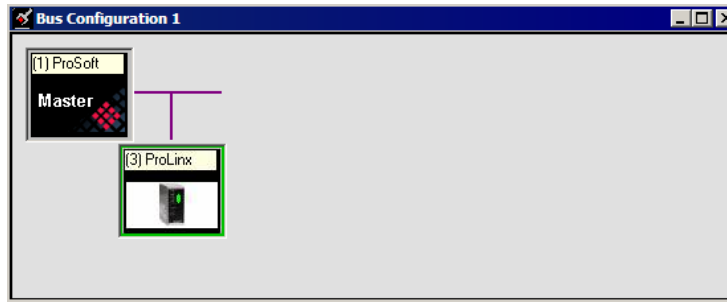
   o If the slave icon in the *Bus Configuration* window has a green border, then the MVI56-PDPMV1 module is correctly communicating with the PROFIBUS slave.
   o If the slave icon in the *Bus Configuration* window has a red border, then the module is not communicating with the slave.
   o If the slave icon in the *Bus Configuration* window has a blue border, the slave is communicating with the Master, but is generating diagnostic data. To view diagnostic data for the slave, select the slave, and click the right mouse button to open a shortcut menu. On the shortcut menu, choose **ONLINE PROPERTIES**.

2 In the *Online Slave Properties* dialog box, click the **DIAGNOSTIC** tab, and select **(CHECK) DETAILS FOR SLAVE DIAGNOSTIC**. Slave diagnostic information will appear in the *Diagnostic* window. Refer to the documentation for your PROFIBUS slave to determine the meaning of the diagnostic data.
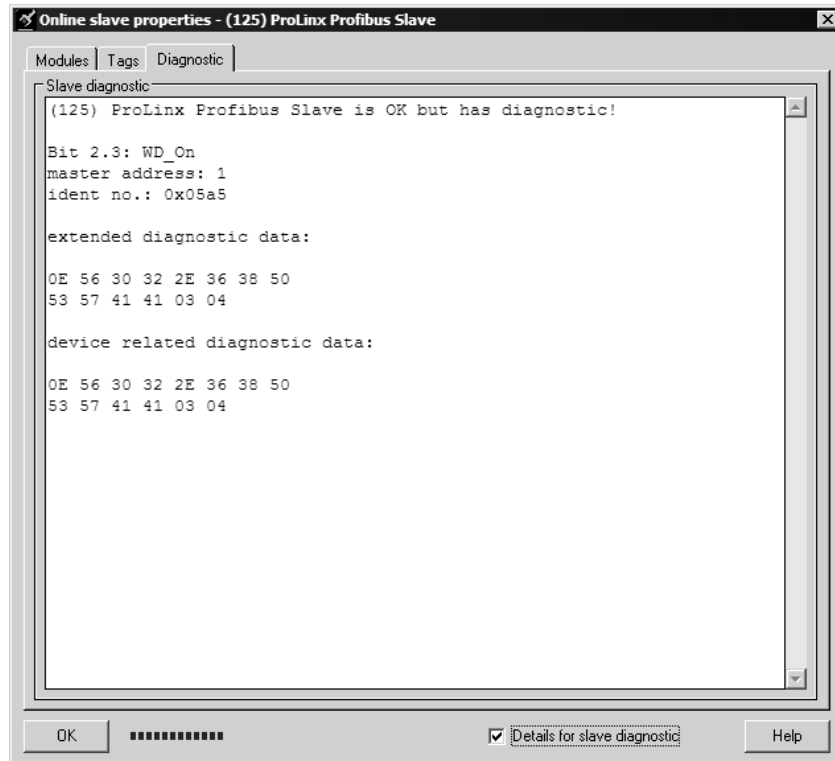
2.5.3 Viewing the Fieldbus Data from the MVI56-PDPMV1's Configuration/Debug Menu

Note: For this procedure, you must connect a serial cable from the serial port on your PC to the RJ45 to DB9M adaptor cable on the MVI56-PDPMV1 module.

1 In ProSoft Configuration Builder, select the **MVI56-PDPMV1 MODULE**, then click the right mouse button to open a shortcut menu.
2 On the shortcut menu, choose **DIAGNOSTICS**. This action opens the *Diagnostics* dialog box.
3 Press **[?]** to display the main menu, and then press **[2]** to view *Fieldbus data*.
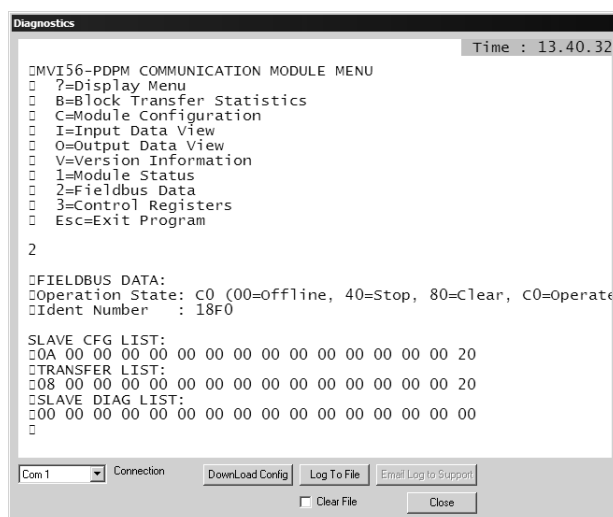
4 Note the value in Operation State. If the Master and the slave are communicating correctly, the operation state will be C0 (hex).
5 Note the values in SLAVE CFG LIST, TRANSFER LIST and SLAVE DIAG LIST. If the Master and slave are communicating successfully, the values in SLAVE CFG LIST will match the values in TRANSFER LIST.

Note: Each list is a hexadecimal representation of a bitmap of slave addresses on the PROFIBUS network. In the illustration above, a value of 20h in the rightmost columns of both of the first two lists means one slave at address 125 is configured **and** communicating with the Master. Notice that the values in the leftmost columns of the first two lists do not match. This means that the slave at address 1 is configured, but is not communicating with the Master. The slave at address 3, however, is both configured **and** communicating with the Master. Below is an explanation on how to read these bitmaps.

Each (XX XX) grouping represents one word containing 16 slave addresses.

In SLAVE CFG LIST in the example above
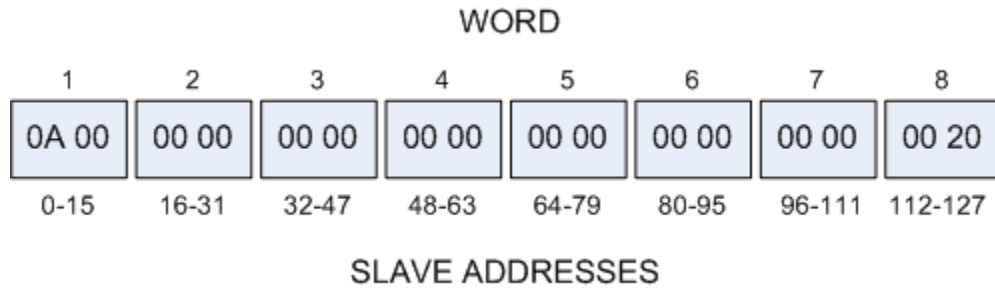
To find out which slave addresses in Word 1 are occupied by configured slaves, expand Word 1's hexadecimal numbers into a bitmap.
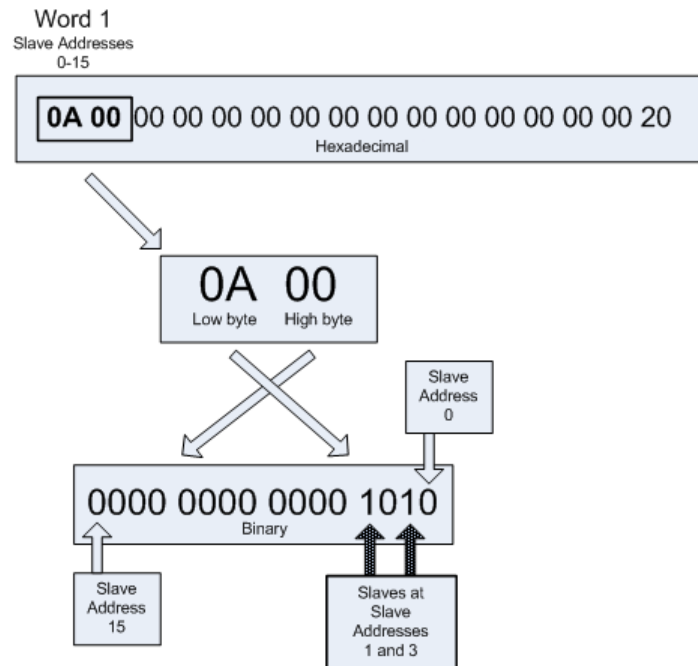
Note: A zero (**0**) in the bitmap indicates an unoccupied slave address; one (**1**) indicates that the slave address is occupied by a configured slave.

Here is the bitmap for Word 8:
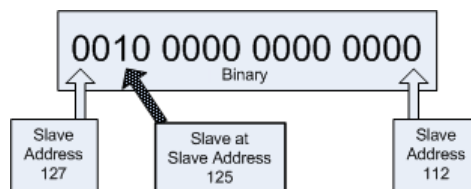
So from the SLAVE CFG LIST, we can tell that there are configured slaves at slave addresses 1, 3 and 125.

The TRANSFER LIST indicates which configured slaves are actually communicating with the Master. Notice that Word 1 is different in the TRANSFER LIST.

Here is the bitmap for Word 1 in the TRANSFER LIST:
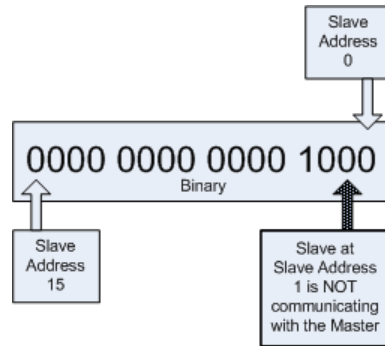
In conclusion, there are configured slaves at addresses 1,3 and 125, but only the slaves at addresses 3 and 125 are communicating with the Master.

Tip: This slave status data can also be found in the *MVI56PDPMV1.Status.Fieldbus* controller tag structure in the Add-On Instruction ladder. The status data in these controller tags can be used as part of your custom application alarm and control logic.

2.5.4 Viewing the Controller Tags in RSLogix 5000

Note: For this procedure, you must connect a serial cable from the serial port on your PC to the DB9M connector on the ControlLogix processor.

1   Open the sample ladder logic in RSLogix 5000.
2   Open the **COMMUNICATIONS** menu, and choose **GO ONLINE**.
3   In the *Controller Organization* window, double-click **CONTROLLER TAGS**.
4   Expand the MVI56PDPMV1, MVI56PDPMV1.Status and MVI56PDPMV1.Status.Fieldbus tags.

⊟ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List		{...}
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[0]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[1]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[2]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[3]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[4]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[5]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[6]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[7]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[8]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[9]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[10]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[11]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[12]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[13]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[14]		2#0000_0000
⊞ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[15]	▼	2#0010_0000

Note: For the FLEX sample program, you must toggle the command bit **MVI56PDPMV1.Command.GetStatus** in order to update the status data. For the LEGACY sample program the status data is updated automatically.

In the illustration above, the binary value in MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[15] indicates that a slave is configured at address 125 (the PROFIBUS Slave Address configured in the Slave Configuration (page 86) procedure).

5 Next, expand the MVI56PDPMV1.Status.Fieldbus.FB_Slave_xfer tag. The value for MVI56PDPMV1.Status.Fieldbus.FB_Slave_xfer should match the corresponding value in MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List, indicating that the slave is configured and communicating.

─MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer	{...}
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[0]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[1]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[2]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[3]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[4]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[5]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[6]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[7]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[8]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[9]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[10]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[11]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[12]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[13]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[14]	2#0000_0000
⊞-MVI56PDPMV1.Status.Fieldbus.FB_Slave_Data_xfer[15]	2#0010_0000

Tip: You can add ladder logic to perform an action based on the values of these tags, for example to determine if a slave is misconfigured.

If the values of these tags show that the slave is configured, but is not transferring data, the slave is not configured correctly. Refer to the documentation provided with your PROFIBUS Slave for troubleshooting procedures.

2.5.5 Sending a Mailbox Message in RSLogix 5000

Note: For this procedure, you must connect a serial cable from the serial port on your PC to the DB9M connector on the ControlLogix processor.

1 Open the sample ladder logic in RSLogix 5000.
2 Open the **COMMUNICATIONS** menu, and choose **GO ONLINE**.
3 In the *Controller Organization* window, double-click **CONTROLLER TAGS.**
4 Expand the MVI56PDPMV1 and MVI56PDPMV1.[Mailbox]Command tags.
5 Enter "1" (number one) in the tag MVI56PDPMV1.[Mailbox]Command.GetLiveList, and then press [Enter] to send the mailbox message from the processor to the MVI56-PDPMV1.

Note: The [Mailbox] string will only be available for the Legacy sample program. For the FLEX sample program the controller tag is MVI56PDPMV1.Command.GetLiveList.

	⊟ MVI56PDPMV1.MailboxCommand	{...}
	── MVI56PDPMV1.MailboxCommand.StopMode	0
	── MVI56PDPMV1.MailboxCommand.OperateMode	0
	── MVI56PDPMV1.MailboxCommand.ClearMode	0
	── MVI56PDPMV1.MailboxCommand.SynchCRC	0
✎	── MVI56PDPMV1.MailboxCommand.GetLiveList	1
	── MVI56PDPMV1.MailboxCommand.GetSlaveConfig	0
	── MVI56PDPMV1.MailboxCommand.SetSlaveMode	0
	── MVI56PDPMV1.MailboxCommand.GetSlaveDiagnostics	0
	── MVI56PDPMV1.MailboxCommand.SetSlaveAddress	0
	── MVI56PDPMV1.MailboxCommand.AcyclicRead	0
	── MVI56PDPMV1.MailboxCommand.AcyclicWrite	0

6 Next, expand the tag MVI56PDPMV1.Mailbox.GetLiveList.Station, to verify that the MVI56-PDPMV1 responded to the GetLiveList message.

For MVI56PDPMV1.Mailbox.GetLiveList.Station[1] (the PROFIBUS Master Address), notice that the value has changed from 4 (not communicating) to 3 (Master).

	⊟ MVI56PDPMV1.Mailbox	{...}
	⊟ MVI56PDPMV1.Mailbox.GetLiveList	{...}
	⊞ MVI56PDPMV1.Mailbox.GetLiveList.ReturnCode	0
	⊞ MVI56PDPMV1.Mailbox.GetLiveList.FaultInfo	0
	⊟ MVI56PDPMV1.Mailbox.GetLiveList.Station	{...}
	⊞ MVI56PDPMV1.Mailbox.GetLiveList.Station[0]	4
▶	⊞ MVI56PDPMV1.Mailbox.GetLiveList.Station[1]	3
	⊞ MVI56PDPMV1.Mailbox.GetLiveList.Station[2]	4
	⊞ MVI56PDPMV1.Mailbox.GetLiveList.Station[3]	4

For MVI56PDPMV1.Mailbox.GetLiveList.Station[125] (the PROFIBUS Slave Address configured in the Slave Configuration (page 86) procedure), notice that the value has changed from 4 (not communicating) to 0 (Slave OK)

		⊞ MVI56PDPMV1.Mailbox.GetLiveList.Station[122]	4
		⊞ MVI56PDPMV1.Mailbox.GetLiveList.Station[123]	4
		⊞ MVI56PDPMV1.Mailbox.GetLiveList.Station[124]	4
▶		⊞ MVI56PDPMV1.Mailbox.GetLiveList.Station[125]	0
		⊞ MVI56PDPMV1.Mailbox.GetLiveList.Station[126]	4

3 Mailbox Messaging

In This Chapter

The MVI56-PDPMV1 PROFIBUS DP Master uses a process called Mailbox Messaging to exchange parameter data between the processor, Master, and slave devices. This process provides a way to encapsulate and prioritize commands and data sent between the PROFIBUS Master and slaves.

The PROFIBUS DP-V1 protocol specifies two types of data transmission messages (telegrams): Cyclic Data Telegrams and Acyclic Data Telegrams. Cyclic data communication is the exchange of normal slave input and output (I/O) data and is handled automatically by the Master in a defined, recurring, deterministic sequence based on the configuration you create in ProSoft Configuration Builder (PCB).

Acyclic communication extends data communication beyond normal I/O data to allow moving field device parameterization and calibration data over the bus during runtime and to allow for extended diagnostics and alarm messages. Acyclic data telegrams are transmitted in the gaps between cyclic data telegrams and, therefore, have a lower priority and get less bandwidth than cyclic data.

Mailbox Messaging commands are incorporated into the sample ladder logic. Mailbox messages and responses to mailbox messages are stored in mailbox data types.

The following chapter discusses these features in more detail.

3.1 Mailbox Message Queuing

The MVI56-PDPMV1 module operates asynchronously on the Mailbox Messages and as such provides for the queuing of the messages as they are received. The queue sizes used in the module are as follows:

Queue Type	Queue Size Max	Description
Output message from processor	126	Number of messages that the MVI56 module will queue by type of message. Note that status of the queues can be monitored via the Queue Message Count values.
Input messages for processor	126	
Alarm messages from slaves for processor	100	

3.1.1 Queue Timeouts

The MVI56-PDPMV1 module will only allow a message to stay in a queue for up to 10 seconds. If the PROFIBUS Master (for output messages) or the processor (for input and alarm messages) has not successfully received a message within 10 seconds, the module will clear the message out of the queue.

3.2   Flex Mode Mailbox Communication

The processor sends a MSG instruction to the module with a *Set Attribute Single* type to request the module to process a mailbox function.
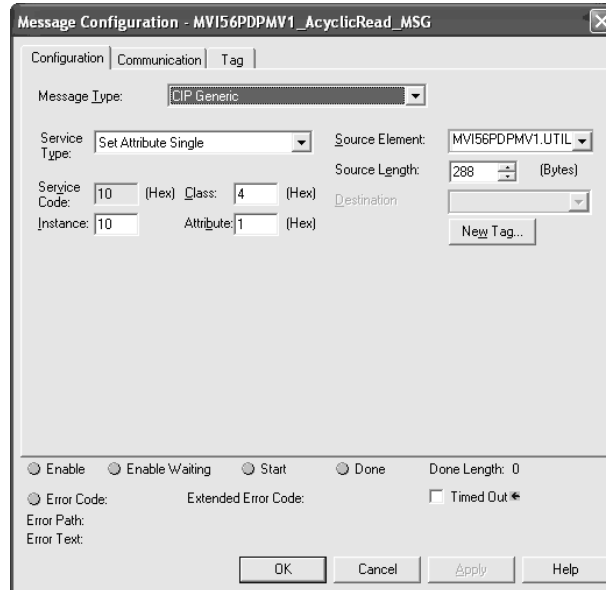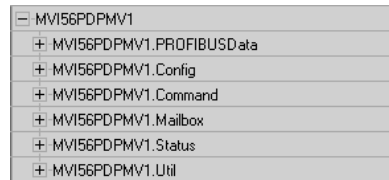
When the module has a response ready to be processed, it will latch bit 15 of the CIP Mailbox MSG Index word of the input block. The ladder logic recognizes this condition and sends another MSG instruction (now *Get Attribute Single*) to retrieve the mailbox response from the module:
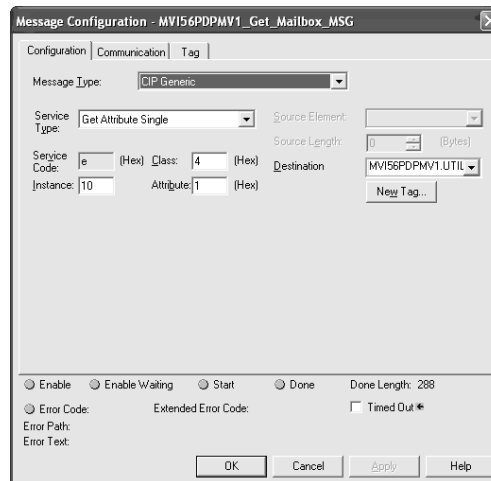
3.2.1 Using FLEX Mode MSG Instructions

This section will provide further information on how to use the controller tags in the FLEX sample program (if Use Legacy Mode=No).

The main controller tag structure, *MVI56PDPMV1*, is organized as follows.

Controller Tag	Description
MVI56PDPMV1.PROFIBUSData	PROFIBUS input and output data transferred between the processor and the module
MVI56PDPMV1.Config	Module configuration data copied from the module (or calculated in ladder)
MVI56PDPMV1.Command	Command bits to trigger mailbox commands, reboot the module and retrieve status information
MVI56PDPMV1.Mailbox	Mailbox output and input data
MVI56PDPMV1.Status	Status information
MVI56PDPMV1.Util	Generic tags used for internal ladder processing (DO NOT MODIFY)

3.2.2 Retrieving Status Information

In order to retrieve status data from the module, toggle the
MVI56PDPMV1.Command.GetStatus bit.

− MVI56PDPMV1	{...}
+ MVI56PDPMV1.PROFIBUSData	{...}
+ MVI56PDPMV1.Config	{...}
− MVI56PDPMV1.Command	{...}
MVI56PDPMV1.Command.AcyclicRead	0
MVI56PDPMV1.Command.AcyclicWrite	0
MVI56PDPMV1.Command.GetLiveList	0
MVI56PDPMV1.Command.GetSlaveConfig	0
MVI56PDPMV1.Command.GetSlaveDiagnostics	0
MVI56PDPMV1.Command.GetStatus	1

The bit is reset when the MSG instruction is enabled to request status data from
the module.

− MVI56PDPMV1	{...}
+ MVI56PDPMV1.PROFIBUSData	{...}
+ MVI56PDPMV1.Config	{...}
− MVI56PDPMV1.Command	{...}
MVI56PDPMV1.Command.AcyclicRead	0
MVI56PDPMV1.Command.AcyclicWrite	0
MVI56PDPMV1.Command.GetLiveList	0
MVI56PDPMV1.Command.GetSlaveConfig	0
MVI56PDPMV1.Command.GetSlaveDiagnostics	0
MVI56PDPMV1.Command.GetStatus	0

When the MSG execution is successfully completed, the controller tag
MVI56PDPMV1.Util.MSG.GetStatus.DNCounter.ACC will be incremented. If an
error occurs during message execution, then the controller tag
MVI56PDPMV1.Util.MSG.GetStatus.ERCounter.ACC will be incremented. The
error code will be stored in the controller tag
MVI56PDPMV1.Util.MSG.GetStatus.LastError.

− MVI56PDPMV1.Util.MSG	{...}
+ MVI56PDPMV1.Util.MSG.MailboxRequest	{...}
+ MVI56PDPMV1.Util.MSG.MailboxResponse	{...}
− MVI56PDPMV1.Util.MSG.GetStatus	{...}
− MVI56PDPMV1.Util.MSG.GetStatus.DNCounter	{...}
+ MVI56PDPMV1.Util.MSG.GetStatus.DNCounter.PRE	100000
+ MVI56PDPMV1.Util.MSG.GetStatus.DNCounter.ACC	1
MVI56PDPMV1.Util.MSG.GetStatus.DNCounter.CU	0
MVI56PDPMV1.Util.MSG.GetStatus.DNCounter.CD	0
MVI56PDPMV1.Util.MSG.GetStatus.DNCounter.DN	0
MVI56PDPMV1.Util.MSG.GetStatus.DNCounter.OV	0
MVI56PDPMV1.Util.MSG.GetStatus.DNCounter.UN	0
− MVI56PDPMV1.Util.MSG.GetStatus.ERCounter	{...}
+ MVI56PDPMV1.Util.MSG.GetStatus.ERCounter.PRE	100000
+ MVI56PDPMV1.Util.MSG.GetStatus.ERCounter.ACC	0
MVI56PDPMV1.Util.MSG.GetStatus.ERCounter.CU	0
MVI56PDPMV1.Util.MSG.GetStatus.ERCounter.CD	0
MVI56PDPMV1.Util.MSG.GetStatus.ERCounter.DN	0
MVI56PDPMV1.Util.MSG.GetStatus.ERCounter.OV	0
MVI56PDPMV1.Util.MSG.GetStatus.ERCounter.UN	0
+ MVI56PDPMV1.Util.MSG.GetStatus.LastError	16#0000
MVI56PDPMV1.Util.MSG.GetStatus.Pending	0

The status data retrieved from the module is stored in the *MVI56PDPMV1.Status* controller tag structure. It includes:

- General module information
- Backplane communication status
- PROFIBUS slaves status (configured, exchanging data, reporting diagnostics)

− MVI56PDPMV1.Status	{...}
− MVI56PDPMV1.Status.Pattern	{...}
+ MVI56PDPMV1.Status.Pattern[0]	'M'
+ MVI56PDPMV1.Status.Pattern[1]	'V'
+ MVI56PDPMV1.Status.Pattern[2]	'I'
+ MVI56PDPMV1.Status.Pattern[3]	'5'
+ MVI56PDPMV1.Status.Pattern[4]	'P'
+ MVI56PDPMV1.Status.Pattern[5]	'D'
+ MVI56PDPMV1.Status.Pattern[6]	'P'
+ MVI56PDPMV1.Status.Pattern[7]	'M'
+ MVI56PDPMV1.Status.Pattern[8]	'V'
+ MVI56PDPMV1.Status.Pattern[9]	'1'
+ MVI56PDPMV1.Status.Pattern[10]	'$00'
+ MVI56PDPMV1.Status.Pattern[11]	'$00'
+ MVI56PDPMV1.Status.Input_Size	200
+ MVI56PDPMV1.Status.Output_Size	200
+ MVI56PDPMV1.Status.Reserved1	0
+ MVI56PDPMV1.Status.Reserved2	0
+ MVI56PDPMV1.Status.Reserved3	0
+ MVI56PDPMV1.Status.Input_Swap	0
+ MVI56PDPMV1.Status.Output_Swap	0
+ MVI56PDPMV1.Status.Module_Major_Ver	1
+ MVI56PDPMV1.Status.Module_Minor_Ver	21
− MVI56PDPMV1.Status.Fieldbus	{...}
− MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List	{...}
+ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[0]	2#0000_1000
+ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[1]	2#0000_0000
+ MVI56PDPMV1.Status.Fieldbus.FB_Slave_Cfg_List[2]	2#0000_0000

3.2.3 Sending Mailbox Commands

The mailbox commands allow the processor to request specific tasks from the module. The following table describes the available mailbox commands:

Mailbox	Controller Tag		Overview
	Trigger	**Data**	
Acyclic Read	MVI56PDPMV1.Command. AcyclicRead	MVI56PDPMV1.Mailbox. AcyclicRead	Reads acyclic data information from DPV1 slaves
Acyclic Write	MVI56PDPMV1.Command. AcyclicWrite	MVI56PDPMV1.Mailbox. AcyclicWrite	Writes acyclic data information from DPV1 slaves
Get Live List	MVI56PDPMV1.Command. GetLiveList	MVI56PDPMV1.Mailbox. GetLiveList	Reads the live list from the module. The live list shows which slave addresses are currently communicating with the module
Get Slave Configuration	MVI56PDPMV1.Command. GetSlaveConfig	MVI56PDPMV1.Mailbox. GetSlaveConfig	Reads the configuration (identifier bytes) from a specific connected PROFIBUS slave
Get Slave Diagnostics	MVI56PDPMV1.Command. GetSlaveDiagnostics	MVI56PDPMV1.Mailbox. GetSlaveDiagnostics	Reads slave diagnostics from a specified slave
Set Clear Mode	MVI56PDPMV1.Command. SetClearMode	-	Sets the module in CLEAR mode. This is an option for the Set Operating Mode mailbox.
Set Operate Mode	MVI56PDPMV1.Command. SetOperateMode	-	Sets the module in OPERATE mode. This is an option for the Set Operating Mode mailbox.
Set Stop Mode	MVI56PDPMV1.Command. SetStopMode	-	Sets the module in STOP mode. This is an option for the Set Operating Mode mailbox.
Set Slave Address	MVI56PDPMV1.Command. SetSlaveAddress	MVI56PDPMV1.Mailbox. SetSlaveAddress	Sets the node address of a specified slave (if supported at slave)
Set Slave Mode	MVI56PDPMV1.Command. SetSlaveMode	MVI56PDPMV1.Mailbox. SetSlaveMode	Sets control commands to slaves or group of slaves (sync, freeze, unsynch, unfreeze)
Start Slaves	MVI56PDPMV1.Command. StartSlaves	MVI56PDPMV1.Mailbox. StartSlaves	Reinitializes communication with a slave after interrupted through Stop Slaves mailbox
Stop Slaves	MVI56PDPMV1.Command. StopSlaves	MVI56PDPMV1.Mailbox. StopSlaves	Stops communication with specified slaves

To send a mailbox request

1 Enter the mailbox configuration parameters (if applicable) in the following controller tag:
MVI56PDPMV1.Mailbox.<MailboxName>.Out.<ParameterName>

Note: This step is not applicable to the following mailboxes (no parameters required):
- SetClearMode
- SetStopMode
- SetOperatingMode
- GetLiveList

2 Toggle the mailbox trigger bit *MVI56PDPMV1.Command.<MailboxName>*
3 Verify that the mailbox trigger bit is automatically reset
4 Verify that the message to request the mailbox was successfully executed through the increment of the following controller tag:
MVI56PDPMV1.Util.MSG.MailboxRequest.DNCounter.ACC
5 Verify that the message to retrieve the mailbox response was successfully executed through the increment of the following controller tag:
MVI56PDPMV1.Util.MSG.MailboxResponse.DNCounter.ACC
6 Refer to the mailbox response data (if applicable) in the following controller tag: *MVI56PDPMV1.Mailbox.<MailboxName>.In.<ResponseDataName>*

Note: this step is not applicable to the following mailboxes (no response data):
- SetClearMode
- SetStopMode
- SetOperatingMode

Example - GetSlaveConfiguration Mailbox

The following example shows how to use a mailbox command to retrieve the slave configuration (identifier bytes) from a PROFIBUS slave.

The first step is to configure the parameters that will be required for the mailbox. For the GetSlaveConfig mailbox, the only parameter required is the PROFIBUS slave address (*MVI56PDPMV1.Mailbox.GetSlaveConfig.Out.SlaveAddress*).

− MVI56PDPMV1.Mailbox.GetSlaveConfig	{...}
− MVI56PDPMV1.Mailbox.GetSlaveConfig.Out	{...}
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.Out.SlaveAddress	3
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In	{...}

Toggle the trigger bit to initiate the mailbox message request:

− MVI56PDPMV1.Command	{...}
MVI56PDPMV1.Command.AcyclicRead	0
MVI56PDPMV1.Command.AcyclicWrite	0
MVI56PDPMV1.Command.GetLiveList	0
MVI56PDPMV1.Command.GetSlaveConfig	1

The trigger bit is automatically reset when the message is enabled for execution:

− MVI56PDPMV1.Command		{...}
	MVI56PDPMV1.Command.AcyclicRead	0
	MVI56PDPMV1.Command.AcyclicWrite	0
	MVI56PDPMV1.Command.GetLiveList	0
	MVI56PDPMV1.Command.GetSlaveConfig	0

When the mailbox is successfully executed, the counter ACC values will be incremented:

- *MVI56PDPMV1.Util.MSG.MailboxRequest.DNCounter.ACC* (Request)
- *MVI56PDPMV1.Util.MSG.MailboxResponse.DNCounter.ACC* (Response)

− MVI56PDPMV1.Util.MSG	{...}
− MVI56PDPMV1.Util.MSG.MailboxRequest	{...}
− MVI56PDPMV1.Util.MSG.MailboxRequest.DNCounter	{...}
+ MVI56PDPMV1.Util.MSG.MailboxRequest.DNCounter.PRE	100000
+ MVI56PDPMV1.Util.MSG.MailboxRequest.DNCounter.ACC	1
MVI56PDPMV1.Util.MSG.MailboxRequest.DNCounter.CU	0
MVI56PDPMV1.Util.MSG.MailboxRequest.DNCounter.CD	0
MVI56PDPMV1.Util.MSG.MailboxRequest.DNCounter.DN	0
MVI56PDPMV1.Util.MSG.MailboxRequest.DNCounter.OV	0
MVI56PDPMV1.Util.MSG.MailboxRequest.DNCounter.UN	0
+ MVI56PDPMV1.Util.MSG.MailboxRequest.ERCounter	{...}
+ MVI56PDPMV1.Util.MSG.MailboxRequest.LastError	16#0000
MVI56PDPMV1.Util.MSG.MailboxRequest.Pending	0
− MVI56PDPMV1.Util.MSG.MailboxResponse	{...}
− MVI56PDPMV1.Util.MSG.MailboxResponse.DNCounter	{...}
+ MVI56PDPMV1.Util.MSG.MailboxResponse.DNCounter.PRE	100000
+ MVI56PDPMV1.Util.MSG.MailboxResponse.DNCounter.ACC	1
MVI56PDPMV1.Util.MSG.MailboxResponse.DNCounter.CU	0
MVI56PDPMV1.Util.MSG.MailboxResponse.DNCounter.CD	0
MVI56PDPMV1.Util.MSG.MailboxResponse.DNCounter.DN	0
MVI56PDPMV1.Util.MSG.MailboxResponse.DNCounter.OV	0
MVI56PDPMV1.Util.MSG.MailboxResponse.DNCounter.UN	0
+ MVI56PDPMV1.Util.MSG.MailboxResponse.ERCounter	{...}
+ MVI56PDPMV1.Util.MSG.MailboxResponse.LastError	16#0000
MVI56PDPMV1.Util.MSG.MailboxResponse.Pending	0

When the mailbox execution was successfully performed, the mailbox response will be stored in the controller tag structure MVI56PDPMV1.Mailbox.GetSlaveConfig.In, which includes:

- *MVI56PDPMV1.Mailbox.GetSlaveConfig.In.ByteCount*: number of identifier bytes returned
- *MVI56PDPMV1.Mailbox.GetSlaveConfig.In.SlaveAddress*: slave address
- *MVI56PDPMV1.Mailbox.GetSlaveConfig.In.ReturnCode*: 0 if successful
- *MVI56PDPMV1.Mailbox.GetSlaveConfig.In.FaultInformation*: 0 if successful

- *MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[ ]*: configuration identifier bytes

− MVI56PDPMV1.Mailbox.GetSlaveConfig		{...}
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.Out		{...}
− MVI56PDPMV1.Mailbox.GetSlaveConfig.In		{...}
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.ByteCount		10
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.SlaveAddress		3
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.ErrorCode		{...}
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.ReturnCode		16#0000
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.FaultInformation		16#0000
− MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier		{...}
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[0]		16#59
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[1]		16#59
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[2]		16#59
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[3]		16#59
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[4]		16#59
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[5]		16#69
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[6]		16#69
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[7]		16#69
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[8]		16#69
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[9]		16#69
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[10]		16#00
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[11]		16#00
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[12]		16#00
+ MVI56PDPMV1.Mailbox.GetSlaveConfig.In.Identifier[13]		16#00

The identifier bytes will match the byte sequence displayed through PCB for PROFIBUS (*CFG data* column):
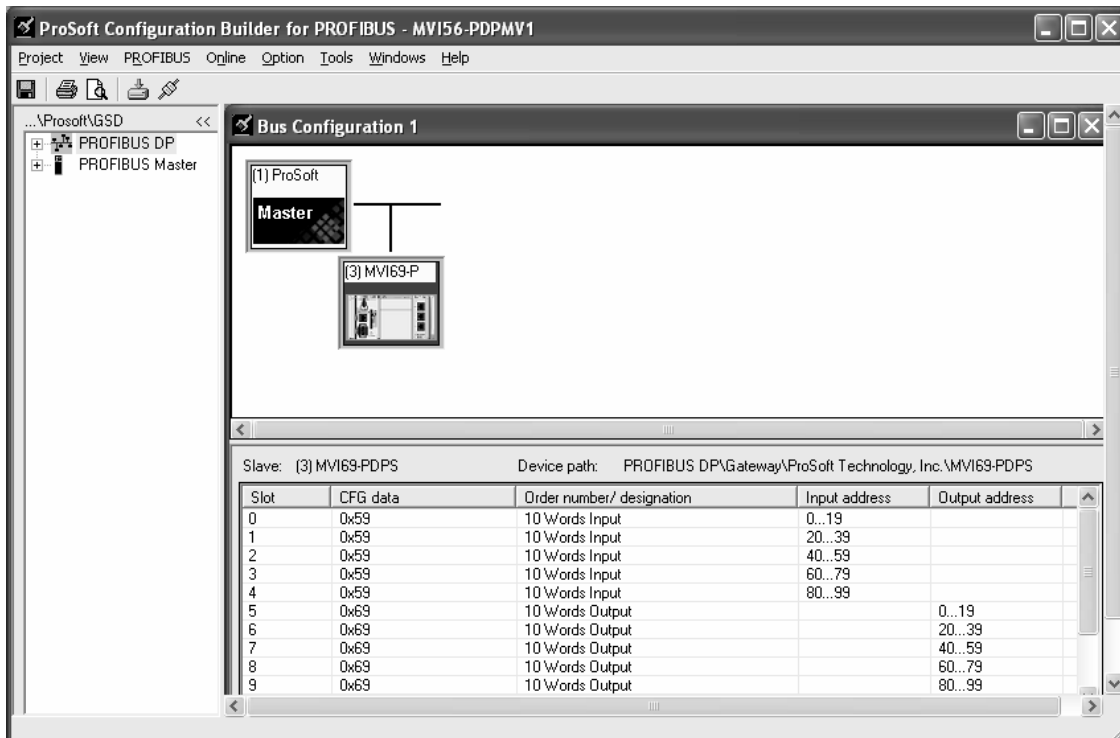
3.2.4 Reading Alarms

The module will automatically read alarms received from DPV1 slaves into specific controller tags.

When the ladder recognizes that the module has received a new alarm, the module will latch the *MVI56PDPMV1.Util.CurrentAlarm.15* bit. The sample ladder logic will automatically recognize that the module has an alarm pending and execute a message instruction to poll for the alarm.

When the message is successfully executed the ladder logic will increment the following controller tag:

MVI56PDPMV1.Util.MSG.GetAlarm.DNCounter.ACC

If the message is not successfully executed the ladder logic will increment the following controller tag:

MVI56PDPMV1.Util.MSG.GetAlarm.ERCounter.ACC

The error code will be copied to the following controller tag:

MVI56PDPMV1.Util.MSG.GetAlarm.LastError

− MVI56PDPMV1.Util.MSG.GetAlarm	{...}
− MVI56PDPMV1.Util.MSG.GetAlarm.DNCounter	{...}
+ MVI56PDPMV1.Util.MSG.GetAlarm.DNCounter.PRE	100000
+ MVI56PDPMV1.Util.MSG.GetAlarm.DNCounter.ACC	1
MVI56PDPMV1.Util.MSG.GetAlarm.DNCounter.CU	0
MVI56PDPMV1.Util.MSG.GetAlarm.DNCounter.CD	0
MVI56PDPMV1.Util.MSG.GetAlarm.DNCounter.DN	0
MVI56PDPMV1.Util.MSG.GetAlarm.DNCounter.OV	0
MVI56PDPMV1.Util.MSG.GetAlarm.DNCounter.UN	0
− MVI56PDPMV1.Util.MSG.GetAlarm.ERCounter	{...}
+ MVI56PDPMV1.Util.MSG.GetAlarm.ERCounter.PRE	100000
+ MVI56PDPMV1.Util.MSG.GetAlarm.ERCounter.ACC	0
MVI56PDPMV1.Util.MSG.GetAlarm.ERCounter.CU	0
MVI56PDPMV1.Util.MSG.GetAlarm.ERCounter.CD	0
MVI56PDPMV1.Util.MSG.GetAlarm.ERCounter.DN	0
MVI56PDPMV1.Util.MSG.GetAlarm.ERCounter.OV	0
MVI56PDPMV1.Util.MSG.GetAlarm.ERCounter.UN	0
+ MVI56PDPMV1.Util.MSG.GetAlarm.LastError	16#0000
MVI56PDPMV1.Util.MSG.GetAlarm.Pending	0

The alarm data will be stored in the *MVI56PDPMV1.Mailbox.Alarm* controller tag structure:

− MVI56PDPMV1.Mailbox.Alarm	{...}
+ MVI56PDPMV1.Mailbox.Alarm.ByteCount	5
+ MVI56PDPMV1.Mailbox.Alarm.SlotNumber	1
+ MVI56PDPMV1.Mailbox.Alarm.SlaveAddress	3
+ MVI56PDPMV1.Mailbox.Alarm.AlarmSpecAck	1
+ MVI56PDPMV1.Mailbox.Alarm.SeqNumber	1
+ MVI56PDPMV1.Mailbox.Alarm.ExtDiag	0
+ MVI56PDPMV1.Mailbox.Alarm.AlarmType	0
+ MVI56PDPMV1.Mailbox.Alarm.FaultInformation	0
− MVI56PDPMV1.Mailbox.Alarm.Data	{...}
+ MVI56PDPMV1.Mailbox.Alarm.Data[0]	5
+ MVI56PDPMV1.Mailbox.Alarm.Data[1]	32
+ MVI56PDPMV1.Mailbox.Alarm.Data[2]	11
+ MVI56PDPMV1.Mailbox.Alarm.Data[3]	0
+ MVI56PDPMV1.Mailbox.Alarm.Data[4]	0

3.2.5 Rebooting the Module

In order to reboot the module toggle the *MVI56PDPMV1.Command.Reboot* controller tag.

MVI56PDPMV1.Command	{...}
MVI56PDPMV1.Command.AcyclicRead	0
MVI56PDPMV1.Command.AcyclicWrite	0
MVI56PDPMV1.Command.GetLiveList	0
MVI56PDPMV1.Command.GetSlaveConfig	0
MVI56PDPMV1.Command.GetSlaveDiagnostics	0
MVI56PDPMV1.Command.GetStatus	0
MVI56PDPMV1.Command.Reboot	1

3.3 Legacy Mode Mailbox Communication

The processor requests a mailbox block by sending a block with a block number between 100 and 109. The request block contains the mailbox request according to the data listed in Output Mailbox: Legacy Mode. (page 146)

The module immediately responds with a confirmation message consisting of the *Mailbox Out Message Response* block to the processor, containing the same block ID as the request block.

After the mailbox request is processed by the module, it sends the *Mailbox In Message Response* block to the processor (block ID between 100 and 109).

NOTE: The block ID in the *Mailbox In Message Response* block may not necessarily be the same as the one sent in the *Mailbox Out Message* during the mailbox request. This is because using Mailbox Messaging is a multi-stage process:

1 Send a special Output block to tell the module to send the Acyclic (Mailbox) Message to a specific slave
2 Wait for the module to receive the receive the slave's response and notify the ladder logic that a response in available
3 Send a special Output block to tell the module to send the slave's response to the Acyclic Message.

It takes two separate actions, using special block IDs, to complete a Mailbox Message cycle. First, a Mailbox Message is sent and then, once the reply has been received, a second action is performed to retrieve the slave's response. Therefore, the ladder should refer to the command data, rather than block ID, to copy the mailbox response to the correct controller tag.
For example:
command = 02 => Set Operating Mode Response 1

3.3.1 Input Mailbox: Legacy Mode

This block contains the actual mailbox response sent by the module.

Block Format for Mailbox In Message (Input Image)

Word Offset Start	Word Offset End	Description
0	143	Mailbox input message from module
144	247	Reserved for future use
248	248	Index of Output image to receive (1 to 4 for data, 0 or -1 if no data)
249	249	Block number 100 to 109

3.3.2 Output Mailbox: Legacy Mode

The following block requests a mailbox message from the module.

Block Format for Mailbox Out Message Request (Output Image)

Word Offset Start	Word Offset End	Description
0	0	Block number 100 to 109
1	144	Mailbox output message for module
145	247	Reserved for future use

Block Format for Mailbox Out Message Response (Input Image)

After the module receives the mailbox request, it immediately sends a data block as confirmation that the block was received.

Word Offset Start	Word Offset End	Description
0	0	Block number 100 to 109
1	1	Status of out mailbox request: 0 = request successfully added to mailbox queue. 1 = queue allocation error 2 = queue is full
2	247	Reserved for future use
248	248	Index of output image to receive (1 to 4 for data, 0 or -1 if no data)
249	249	Block number 100 to 109

3.3.3 Receiving Mailbox Message Responses from the Module: Legacy Mode

After a mailbox message has been sent, a response from the command, usually containing the requested data or the status of the command, is returned from the MVI56 module to the processor. The response is returned from the MVI56-PDPMV1 via the PROFIBUS Input data block.

Remembering the PROFIBUS Input Data Memory Map:

Word Offset	Name	Description
0 to 72		Configuration, status and control data
73		Number of messages in the In Mailbox queue
74		Number of messages in the Out Mailbox queue
75		Number of messages in the alarm queue
76		Last Out Mailbox Message ID processed from output image
77		Current In Mailbox Control Index
78		Current Alarm Control Index
79 to 222		Incoming Mailbox Message data
223 to *n*		PROFIBUS input data

The important section relevant to the Mailbox Messaging discussion is the Incoming Mailbox Data section (Byte Offsets 79 to 222). Within this section of data, the following structure exists:

Mailbox Message Structure: From MVI56 module

Word Offset	Type	Description
79	Message ID	Message ID value will match value used to generate the outgoing mailbox message
80	Message Info	See individual commands for data values to be entered in each of these register locations
81	Command	
82	Data Size	
83	Frame Count	
84	Frame Number	
85	Offset high	
86	Offset Low	
87	Extended Word 1	
88	Extended Word 2	
89	Extended Word 3	
90	Extended Word 4	
91	Extended Word 5	
92	Extended Word 6	
93	Extended Word 7	
94	Extended Word 8	
-	See individual commands	
222		

3.4   Special Function Mailbox Messaging Commands

The MVI56-PDPMV1 module supports some extended PROFIBUS functions, using a mailbox data exchange mechanism implemented in the module.

The MVI56-PDPMV1 module supports the following special functions through this mailbox messaging scheme:

Initiated from Processor

Message	Description
Set Operation Mode	Controls the operating state of the PROFIBUS Master
Set Slave Mode	Sends special control command to one or several slaves (Sync/Freeze)
Get Slave Diag	Gets diagnostic information from a slave
Get Slave Config	Gets slave configuration
Set Slave Address	Sets node address of a slave (if supported by the slave)
Get Live List	Gets information from all nodes on the network
MSAC1 Read	DPV1 acyclic read (Class 1)
MSAC1 Write	DPV1 acyclic write (Class 1)
Start Slave	Starts slaves dynamically
Stop Slave	Stops slaves dynamically

DPV1 Alarm Handling: Generated by Slave Devices

Message	Description
Alarm Indications	Spontaneous alarm indication from DPV1 slave. Structure of data is slave-dependent
Alarm Confirmation	This message is sent by the MVI56 module automatically as a confirmation to the alarm indications.

The provided Function Blocks simplify the processor logic required to implement these messaging mailbox exchanges.

Sending a mailbox message to the MVI56-PDPMV1 module is a relatively simple process; however, it is important to follow a certain sequence.

Mailbox Message Structure: To MVI56 Module

Word Offset	Type	Description
0	Message ID	Processor logic or user-set. The Message ID field is used by the MVI56 module to detect a new message in the PROFIBUS output data image.
		When a non-zero value is detected, the message is processed immediately. The sample ladder logic assigns specific message IDs to different mailbox messages.
1	Message Info	See individual commands for data values to be entered in each of these register locations
2	Command	
3	Data Size	
4	Frame Count	
5	Frame Number	
6	Offset high	
7	Offset Low	
8	Extended Word 1	
9	Extended Word 2	
10	Extended Word 3	
11	Extended Word 4	
12	Extended Word 5	
13	Extended Word 6	
14	Extended Word 7	
15	Extended Word 8	
-	See individual commands	
149		

3.5 Mailbox Message Commands

3.5.1 Mailbox Message: Set Slave Mode

In addition to station-related user data transfer, which is executed automatically, the Master can send control commands to a single slave, a group of slaves, or all slaves simultaneously. These control commands are transmitted as multicast commands. This permits use of sync and freeze modes for event-controlled synchronization of the slaves.

The slaves begin sync mode when they receive a sync command from their assigned Master. The outputs of all addressed slaves are then frozen in their current state. During subsequent user data transmissions, the output data are stored on the slaves, but the output states remain unchanged. The stored output data are not sent to the outputs until the next sync command is received. Sync mode is concluded with the unsync command.

Similarly, a freeze control command causes the addressed slaves to assume freeze mode. In this operating mode, the states of the inputs are frozen until the Master sends the next freeze command. Freeze mode is concluded with the unfreeze command.

Note 1: It is only possible to send control commands when operating mode is either "CLEAR" or "OPERATE".
Note 2: Not all slaves support this feature. Refer to the documentation for the actual slave for more information.

Parameter	Description
Command Initiator	Application
Command Name	SET SLAVE MODE
Command Number	0300h
Fragmented	No
Extended Header Data	Fault information may be returned in the header of the response.

Command and Response Layout: Set Slave Mode

	Command		Response		
Message ID	(ID)		(ID)		
Message information	4002h		0002h		
Command	0003h		0003h		Set Slave Mode
Data size	0000h		0000h		
Frame count	0001h		0001h		
Frame number	0001h		0001h		
Offset high	0000h		0000h		
Offset low	0000h		0000h		
Extended word 1	Slave Address	Group Select	Slave Address	Group Select	
Extended word 2	Control Command		Control Command		
Extended word 3	-		-		
Extended word 4	-		-		
Extended word 5	-		-		
Extended word 6	-		-		
Extended word 7	-		Extended Fault Information		
Extended word 8			Fault Information		

Message Information

Refer to Message Information (page 173).

Slave Address

Range 1 to 125; 127

If the request applies for only one slave, that slave address must be entered in the range 1 to 125. If a slave group is to be addressed, slave address should be 127 (multicast address).

Group Select

Range 01h to FFh (Bit Coded)

This parameter determines which group to address. Refer to the following example:

b7	b6	b5	b4	b3	b2	b1	b0
Group 8	Group 7	Group 6	Group 5	Group 4	Group 3	Group 2	Group 1

Example:

To address Group 1, 2, and 4, the Group Select value should be D0h. If an individual slave should be addressed, the correct group selection must also be made, because the slave will ignore the message if it does not belong to the requested group(s).

What group(s) a slave belongs to is determined during network configuration, and is downloaded during initialization to each slave via the PROFIBUS telegram *Set_Prm.*

Control Command

This parameter specifies the command to send:

Bit	Explanation
0 (LSB)	Reserved, set to zero
1	Reserved, set to zero
2	Unfreeze input data
3	Freeze input data
4	Unsynchronize output data
5	Synchronize output data
6	Reserved, set to zero
7 (MSB)	Reserved, set to zero

Combinations of the bits (Unsync/Sync and Unfreeze/Freeze)

Bits 0 or 6	Bits 1 or 7	Explanation
0	0	No function
0	1	Function will be activated
1	0	Function will be inactive
1	1	Function will be inactive

Fault Information and Extended Fault Information

"Fault Information" Contents		"Extended Fault Information" Contents	
0100h	Address out of range	-	
0200h	Group number 0 not permitted	-	
0A00h	Failed to send Global Control request	0A00h	Incorrect operation mode (Clear/Operate Only)
		0150h	Invalid Freeze Group (Group is not initiated to be Freeze Group)
		0250h	Invalid Sync Group (Group is not initiated to be Sync Group)
		0350h	Incorrect Control Command
		0450h	No Sync -/ or Freeze groups enabled in Master configuration
FE00h	Command not possible in Class 2 only mode	-	
FF00h	Module not initialized	-	

3.5.2 Mailbox Message: Get Slave Diagnostics

This command reads diagnostic data from a specified slave.

> **Note:** The response data size depends on the actual slave implementation. Range 6 to 244.

Parameter	Description
Command Initiator	Application
Command Name	GET SLAVE DIAGNOSTICS
Command Number	0400h
Fragmented	No
Extended Header Data	Fault information may be returned in the header of the response.

Command and Response Layout: Get Slave Diagnostics

	Command		Response		
Message ID	(ID)		(ID)		
Message information	4002h		0002h		
Command	0004h		0004h		Get Slave Diagnostics
Data size	0000h		(Size of data)		
Frame count	0001h		0001h		
Frame number	0001h		0001h		
Offset high	0000h		0000h		
Offset low	0000h		0000h		
Extended word 1	Slave Address	Type of Request	Slave Address	Type of Request	
Extended word 2	-		-		
Extended word 3	-		-		
Extended word 4	-		-		
Extended word 5	-				
Extended word 6	-				
Extended word 7	-		Extended Fault Info		
Extended word 8	-		Fault Information		
			Station Status 1	Station Status 2	Response data word 1
			Station Status 3	Station Status 4	Response data word 2
			Ident Number		Response data word 3
			Extended Diagnostic Data		Response data word 4 ... ... Response data word n

Message Information

Refer to Message Information (page 173).

Slave Address

Range 1 to 125; specifies the slave from which to read diagnostics.

Type of request

00h: Internal slave diagnostic request. Returns the diagnostic information stored in the Master. Can only be requested for slaves configured by the Master.

Note: Not allowed when operating in "Class 2-Only" mode.

01h: External slave diagnostic request. Sends a diagnostic request on the network to the specified slave. Can be requested for all slaves on the network.

Error code [1 ...4]

If "Return Code" equals 8030h ("Negative indication from lower layer"), status values according to the DP-specification may be available in "Error Code 1". Error Codes 2 to 4 are reserved.

Refer to Mailbox Messaging Error Codes (page 173).

Return Code

Refer to Mailbox Messaging Error Codes (page 173)

Fault Information

If "Invalid Other" is returned in the Message Information word in the header of the response, information about the fault can be found here.

0100h: Address out of range.

0200h: Incorrect "Type of request"

0A00h: Failed to read diagnostic data from slave. Refer to Return Codes (page 174) for additional fault information.

0B00h: Remote station failure. Refer to Return Codes (page 174) for additional fault information.

FE00h: Command not possible; module operates as a Class 2 Master only.

FF00h: Module offline (not initialized or no valid database).

Station Status [1 ... 3]

Refer to EN50170 Vol. 2 for more information.

Master Address

Address of the Master that parameterized the slave.

Ident Number

Unique ID assigned by the PROFIBUS User Organization.

Extended Diagnostic Data

Slave user-specific data. Refer to the documentation for the actual slave for more information.

3.5.3   Mailbox Message: Get Slave Configuration

This command reads the actual configuration (identifier bytes) of a specified slave.

Note: The response data size depends on the actual slave implementation. Range 6 to 244.

Parameter	Description
Command Initiator	Application
Command Name	GET SLAVE CONFIGURATION
Command Number	0500h
Fragmented	No
Extended Header Data	Fault information may be returned in the header of the response.

Command and Response Layout: Get Slave Configuration

	Command		Response		
Message ID	(ID)		(ID)		
Acyclic Message Status Word	4002h		0002h		
Command	0005h		0005h		Get Slave Configuration
Data size	0000h		(Size of data)		Number of identifier bytes (n)
Frame count	0001h		0001h		
Frame number	0001h		0001h		
Offset high	0000h		0000h		
Offset low	0000h		0000h		
Extended word 1	Slave Address		Slave Address		
Extended word 2	-		-		
Extended word 3	-		-		
Extended word 4	-		-		
Extended word 5	-		Error Code 1	Error Code 2	
Extended word 6	-		Error Code 3	Error Code 4	
Extended word 7	-		Return Code		
Extended word 8	-		Fault Information		
			Identifier byte 1		Response data word 1
			Identifier byte 2		Response data word 2
			Identifier byte 3		Response data word 3
			...		...
			Identifier byte n		Response data word n

Acyclic Message Status Word

Refer to Acyclic Message Status Word (page 173).

Slave Address

Range 1 to 125; specifies the slave from which to read the configuration.

Error Code [1 ... 4]

If "Return Code" equals 3080h ("Negative indication from lower layer"), status values according to the DP-specification may be available in "Error Code 1", Error Codes 2 through 3 are reserved. Refer to Mailbox Messaging Error Codes (page 173).

Return Code

Refer to Mailbox Messaging Error Codes (page 173).

Fault Information

If "Invalid other" is returned in the Acyclic Message Status Word in the header of the response, information about the fault can be found here. Refer to Acyclic Message Status Word (page 173).

0100h: Address out of range.

0A00h: Failed to execute request. Refer to Return Codes (page 174) for additional information.

0B00h: Remote station failure. Refer to Return Codes (page 174) for additional information.

FF00h: Module not initialized.

Identifier Bytes [1 ... n]

Refer to EN50170 Vol. 2 for information on the structure of these bytes. In addition, refer to the documentation provided with the slave device for more information.

3.5.4 Mailbox Message: Set Slave Address

This command makes it possible to set the node address of a specified slave, if the slave supports this feature.

Note: The message data size depends on the actual slave implementation. Range 0 to 240 bytes.

Parameter	Description
Command Initiator	Application
Command Name	SET SLAVE ADDRESS
Command Number	0600h
Fragmented	No
Extended Header Data	Fault information may be returned in the header of the response.

Command and Response Layout: Set Slave Address

	Command		Response		
Message ID	(ID)		(ID)		
Message information	4002h		0002h		
Command	0006h		0006h		Set Slave Address
Data size	(Size of data)		(Size of data)		No. of Slave Data bytes (n)
Frame count	0001h		0001h		
Frame number	0001h		0001h		
Offset high	0000h		0000h		
Offset low	0000h		0000h		
Extended word 1	Current Slave Addr.	New Slave Address	Current Slave Addr.	New Slave Address	
Extended word 2	Slave Ident Number		Slave Ident Number		
Extended word 3	No_add_ Chg		No_add_ Chg		
Extended word 4	-		-		
Extended word 5	-		Error Code 1	Error Code 2	
Extended word 6	-		Error Code 3	Error Code 4	
Extended word 7	-		Return Code		
Extended word 8			Fault Information		
Message Data byte 1	Slave Data 1		Slave Data 1		
Message Data byte 2	Slave Data 2		Slave Data 2		
Message Data byte 3	Slave Data 3		Slave Data 3		
...	...		...		
Message Data byte "n"	Slave Data n		Slave Data n		

Message Information

Refer to Message Information (page 173).

Current Slave Address

Range 1 to 125; specifies the current address of the slave.

New Slave Address

Range 1 to 125; specifies the new address of the slave.

Slave Ident Number

Ident number for the slave; which address should be altered.

No_add_Chg

This parameter specifies whether it is allowed to change the slave address again later. If this is not allowed, it is only possible to change the address with this function after initial reset. After the initial reset, the slave takes the default address of 126.

00h: Change of address is still possible at a later stage.

01h-FFh: Change of address is only possible after the initial address (the default address = 126).

Error Code [1 ...4]

If "Return Code" equals 3080h ("Negative indication from lower layer"), status values according to the DP-specification in available in "Error Code 1". Error Codes 2 and 3 are reserved. Refer to Return Codes (page 174).

Return Code

Refer to Return Codes (page 174).

Fault Information

If "Invalid Other" is returned in the Message Information word in the header of the response, information about the fault can be found here:

0100h: Current slave address out of range.

0200h: New slave address out of range.

0A00h: Failed to execute request.

0B00h: Remote station failure.

FF00h: Module not initialized.

Refer to Mailbox Messaging Error Codes (page 173).

Slave Data

With this parameter, it is possible to deliver user-specific data. The data is stored in the slave if possible (EEPROM, FLASH, or other storage media).

3.5.5 Mailbox Message: Get Live List

This command returns 127 bytes of information about the nodes on the network. Every byte stands for one bus subscriber, and the position of the byte in the response data assigns the address (0 to 126), the content assigns the Station Type.

This command can be sent in all operation modes (STOP, CLEAR, and OPERATE), however the module must be initialized properly.

Parameter	Description
Command Initiator	Application
Command Name	GET LIVE LIST
Command Number	1800h
Fragmented	No
Extended Header Data	Fault information may be returned in the header of the response.

Command and Response Layout: Get Live List

	Command	Response	
Message ID	(ID)	(ID)	
Message information	4002h	0002h	
Command	0018h	0018h	Get Live List
Data size	0000h	0007h	127 Bytes of Data
Frame count	0001h	0001h	
Frame number	0001h	0001h	
Offset high	0000h	0000h	
Offset low	0000h	0000h	
Extended word 1	-	-	
Extended word 2	-	-	
Extended word 3	-	-	
Extended word 4	-	-	
Extended word 5	-	-	
Extended word 6	-	-	
Extended word 7	-	Return Code	
Extended word 8	-	Fault Information	
Message Data byte 1		Station Type 0	Response Data Byte 1
Message Data byte 2		Station Type 1	Response Data Byte 1
Message Data byte 3		Station Type 2	Response Data Byte 1
...		...	Response Data Byte 1
Message Data byte "n"		Station Type 126	Response Data Byte 1

Message Information

Refer to Message Information (page 173).

Station Type [0 ... 126]

00h: Slave Station

01h: Master Station not yet ready for Token ring (station only physically at the bus)

02h: Master Station ready to enter Token ring (there is not yet any Token transmission)

03h: Master Station in Token Ring (Token transmission through the station)

04h: Station does not exist

Fault Information

If "Invalid Other" is returned in the Message Information word in the header of the response, information about the fault can be found here. Refer to Message Information (page 173).

0AH00: Failed to build Live List.

FF00h: Module offline (not initialized or no valid database)

3.5.6 Mailbox Message: Acyclic Data Read: Class 1

This command initiates a DPV1 Class 1 acyclic read request. Refer to EN50170 (DPV1) for more information.

Parameter	Description
Command Initiator	Application
Command Name	MSAC1 READ
Command Number	2000h
Fragmented	No
Extended Header Data	Fault information may be returned in the header of the response.

Command and Response Layout: Acyclic Read

	Command		Response		
Message ID	(ID)		(ID)		
Message information	4002h		0002h		
Command	0020h		0020h		Acyclic Read
Data size	0000h		(Size of data)		Number of data bytes (n)
Frame count	0001h		0001h		
Frame number	0001h		0001h		
Offset high	0000h		0000h		
Offset low	0000h		0000h		
Extended word 1	Slave Addr.	Slot Number	Slave Addr.	Slot Number	
Extended word 2	Index	Length	Index	Length	
Extended word 3	-		-		
Extended word 4	-		-		
Extended word 5	-			Error Decode	
Extended word 6	-		Error Code 1	Error Code 2	
Extended word 7	-		Extended Fault information		
Extended word 8	-		Fault Information		
			Data 1		Response Data byte 1
			Data 2		Response Data byte 1
			Data 3		Response Data byte 1
			...		...
			Data n		Response Data byte 1

Message Information
Refer to Message Information (page 173).

Slave Address
Station address of the slave responder.

Slot Number and Slot Index
Used in the slave to address the desired data block.

Length

This parameter specifies the number of bytes of the data block to read. If the server data block length is less than requested, the length of the response will be the actual length of the data block. If the server data block is greater or equal, the response will contain the same amount of data.

The slave may answer with an error response if data access is not allowed.

Data [1 ... n]

Returned data.

Fault Information and Extended Fault Information

If "Invalid Other" is returned in the Message Information word in the header of the response, information about the fault can be found here.

"Fault Information"		"Extended Fault Information" Contents
0100h	Address out of range	-
0A00h	Failed to execute request	Refer to Return Codes (page 174).
0B00h	Remote station failure	
1000h	Remote station DPV1 failure	Function_Number
1100h	Length out of range (>240 bytes)	-
1200h	Slave does not support DPV1	-
1300h	Slave not active or not present in configuration	-
FE00h	Command not possible in "Class 2-Only" mode	-
FF00h	Module offline (not initialized or no valid database)	-

Error Decode, Error Code 1 and Error Code 2

If "Fault Information" contains error code 1000h, more information according to the DPV1 specification can be found here.

3.5.7 Mailbox Message: Acyclic Data Write: Class 1

This command initiates a DPV1 Class 1 acyclic write request. Refer to EN50170 (DPV1) for more information.

Parameter	Description
Command Initiator	Application
Command Name	MSAC1 WRITE
Command Number	2100h
Fragmented	No
Extended Header Data	Fault information may be returned in the header of the response.

Command and Response Layout: Acyclic Write

	Command		Response		
Message ID	(ID)		(ID)		
Acyclic Message Status Word	4002h		0002h		
Command	0021h		0021h		Acyclic Write
Data size	(Size of data)		(Size of data)		Number of data bytes (n)
Frame count	0001h		0001h		
Frame number	0001h		0001h		
Offset high	0000h		0000h		
Offset low	0000h		0000h		
Extended word 1	Slave Addr.	Slot Number	Slave Addr.	Slot Number	
Extended word 2	Index	Length	Index	Length	
Extended word 3	-		-		
Extended word 4	-		-		
Extended word 5	-			Error Decode-	
Extended word 6	-		Error Code 1	Error Code 2	
Extended word 7	-		Extended Fault information		
Extended word 8	-		Fault Information		
Message Data byte 1	Data 1		Data 1		
Message Data byte 2	Data 2		Data 2		
Message Data byte 3	Data 3		Data 3		
...	...		...		
Message Data byte n	Data n		Data n		

Acyclic Message Status Word

Refer to Acyclic Message Status Word (page 173).

Slave Address

Station address of the slave responder.

Slot Number and Slot Index

Used in the slave to address the desired data block.

Length

This parameter specifies the number of bytes to write. If the destination data block size is less than requested, the response will contain an error message. If the data block length is greater than or equal to the requested length, the response contains the number of bytes that have been written. The slave may answer with an error response if data access is not allowed.

Data [1 ... n]

Data that should be written.

Fault Information and Extended Fault Information

If "Invalid Other" is returned in the Acyclic Message Status Word in the header of the response, information about the fault can be found here:

"Fault Information"		"Extended Fault Information" Contents
0100h	Address out of range	-
0A00h	Failed to execute request	Refer to Return Codes (page 174).
0B00h	Remote station failure	
1000h	Remote station DPV1 failure	Function_Number
1100h	Length out of range (>240 bytes)	-
1200h	Slave does not support DPV1	-
1300h	Slave not active or not present in configuration	-
FE00h	Command not possible in "Class 2-Only" mode	-
FF00h	Module offline (not initialized or no valid database)	-

Error Decode, Error Code 1, and Error Code 2

If "Fault Information" contains error code 1000h, more information according to the DPV1 specification can be found here.

3.5.8 Mailbox Message: Alarm Indication

This message indicates to the application that a DPV1 slave has transferred an alarm message to the Master. This mailbox message is sent spontaneously by the module; the module itself initiates the mailbox communications.

Detailed information about the cause of the alarm is presented in extended words 1 to 3 and in the message data field (see below).

Command and Response Layout: Alarm Indication

	Command		Response	
Message ID	(ID)		(ID)	
Message information	4002h		0002h	
Command	0022h		0022h	Alarm Indication
Data size	(request length)		0000h	
Frame count	0001h		0001h	
Frame number	0001h		0001h	
Offset high	0000h		0000h	
Offset low	0000h		0000h	
Extended word 1	Slave Address	Slot Number	-	
Extended word 2	Seq Number	Alarm Spec Ack	-	
Extended word 3	Alarm Type	Ext Diag	-	
Extended word 4	-		-	
Extended word 5	-		-	
Extended word 6	-		-	
Extended word 7	-		-	
Extended word 8	Fault Information		-	
Message Data byte 1	Data 1			
Message Data byte 2	Data 2			
Message Data byte 3	Data 3			
...	...			
Message Data byte n	Data n			

Slave Address

Station address of the slave that issued the alarm.

Slot Number

Used by the slave to indicate the source of the alarm. Range 0 to 254.

Seq Number

Unique identification number of the alarm. Range 0 to 31.

Alarm Spec Ack

Provides additional information about the alarm, such as if an error appears or disappears. Also indicates whether the slave needs additional knowledge from the Master. For example: writing to a certain memory area with an Acyclic Write request.

Alarm Type

Identifies the alarm type such as Process Alarm, Plug Alarm, and so on. Range 1 to 6, 32 to 126.

Extended Diagnostic Flag

FFh: Slave sends an alarm message with "Extended Diag flag" set

00h: Slave sends an alarm message with "Extended Diag flag" cleared

Data [1 ... n]

Additional manufacturer-specific alarm information (Alarm - PDU)

Fault Information

If the Message Information word in the header of the message indicates "Invalid Other", addition information is available in this register.

3E00h: Module has received an invalid alarm indication data structure from a DPV1 slave ("Slave Address" contains the node address of the slave that issued the erroneous indication).

Refer to the PNO document "Extensions to EN50170 (DPV)" for more information on how to interpret these parameters.

3.5.9 Mailbox Message: Set Operating Mode

This command allows setting the operating mode of the module (STOP, CLEAR, or OPERATE).

Parameter	Description
Command Initiator	Application
Command Name	SET OPERATING MODE
Command Number	0200h
Fragmented	No
Extended Header Data	Fault information may be returned in the header of the response.

Command and Response Layout: Set Operating Mode

	Command		Response		
Message ID	(ID)		(ID)		
Message information	4002h		0002h		
Command	0002h		0002h		Set Operation Mode
Data size	0000h		0000h		
Frame count	0001h		0001h		
Frame number	0001h		0001h		
Offset high	0000h		0000h		
Offset low	0000h		0000h		
Extended word 1	Mode	Conf. Req	Mode	Conf. Req	
Extended word 2	-		-		
Extended word 3	-		-		
Extended word 4	-		-		
Extended word 5	-		-		
Extended word 6	-		-		
Extended word 7	-		-		
Extended word 8	-		Fault Information		

Mode

40h: STOP

80h: CLEAR

C0h: OPERATE

Conf. Req.

00h: Confirmation is not required

01h: Confirmation required. All confirmations are automatically sent by the
Master; the user is not required to send a confirmation message.

Fault Information

If "Invalid Other" is returned in the Message Information word in the header of the
response, information about the fault can be found here. Refer to Return Codes
(page 174) for more information.

0100h: Invalid operating mode

FF00h: Module not initialized

3.5.10 Mailbox Message: Start Slave

This mailbox message starts a selection of slaves that was previously removed from the processing cycle by means of the mailbox message FB_APPL_STOP_SLAVE.

This message is allowed in all Operation modes (STOP, CLEAR and OPERATE).

Note: The message will be accepted even if one or several slaves are not part of the configuration and can therefore obviously not be started. However, the application can find out about this situation by evaluating the "Fault information" and "Message data words" of the response.

Command and Response Layout: Start Slave

	Command		Response		
Message ID	(ID)		(ID)		
Message information	4002h		0002h		
Command	000Bh		000Bh		Start Slave
Data size	007Eh		007Eh		
Frame count	0001h		0001h		
Frame number	0001h		0001h		
Offset high	0000h		0000h		
Offset low	0000h		0000h		
Extended word 1					
Extended word 2	-		-		
Extended word 3	-		-		
Extended word 4	-		-		
Extended word 5	-		-		
Extended word 6	-		-		
Extended word 7	-		Additional Fault Information		
Extended word 8	-		Fault Information		
Message data word 1	Slave 0	Slave 1	Slave 0	Slave 1	
Message data word 2	Slave 2	Slave 3	Slave 2	Slave 3	
Message data word 3 to 62	...	...	...	..	
Message data word 63	Slave 124	Slave 125	Slave 124	Slave 125	

Command:

▪ Message data word 1-63

Byte-array stating which slave/slaves to start. Array index is equal to slave address.
0: Do not affect slave
1: Start slave
2-255: Reserved

Response:

▪ Message information (in response header)

"Invalid Data Size" is returned if Data size in the command header does not equal 126.
If "Invalid Other" is returned, further information is to be found in Extended word 8.

▪ Additional Fault information (Extended word 7)

If Extended word 8 equals 0x000A -"Failed to execute request" additional info can be found here.

▪ Fault information (Extended word 8)

0x0001: Invalid setting in Message data word 1-63 of the command.
0x0002: At least one slave reports a warning. Refer to Message data word 1-63.
0x000A: Failed to execute request. Additional fault information is to be found in Extended word 7.
0x00FE: Command not possible, module operates as class 2 Master only.
0x00FF: Module not initialized (this command is only possible after END_INIT).

▪ Message data word 1-63

Byte-array stating the status of the slaves. Array index is equal to slave address.
0: Slave unaffected
1: Slave started
2: Warning - Slave could not be started because it is not part of the configuration

3.5.11 Mailbox Message: Stop Slave

This mailbox message stops a selection of slaves from the processing cycle.

This message is allowed in all Operation modes (STOP, CLEAR and OPERATE).

Note: The message will be accepted even if one or several slaves are not part of the configuration and can therefore obviously not be stopped. However, the application can find out about this situation by evaluating the "Fault information" and "Message data words" of the response.

Command and Response Layout: Stop Slave

	Command		Response		
Message ID	(ID)		(ID)		
Message information	4002h		0002h		
Command	000Ch		000Ch		Stop Slave
Data size	007Eh		007Eh		
Frame count	0001h		0001h		
Frame number	0001h		0001h		
Offset high	0000h		0000h		
Offset low	0000h		0000h		
Extended word 1	-		-		
Extended word 2	-		-		
Extended word 3	-		-		
Extended word 4	-		-		
Extended word 5	-		-		
Extended word 6	-		-		
Extended word 7	-		Additional Fault Information		
Extended word 8	-		Fault Information		
Message data word 1	Slave 0	Slave 1	Slave 0	Slave 1	
Message data word 2	Slave 2	Slave 3	Slave 2	Slave 3	
Message data word 3 to 62	...	...	...	...	
Message data word 63	Slave 124	Slave 125	Slave 124	Slave 125	

Command:

▪ Message data word 1-63

Byte-array stating which slave/slaves to stop. Array index is equal to slave address.
0: Do not affect slave
1: Stop slave
2-255: Reserved

Response:

▪ Message information (in response header)

"Invalid Data Size" is returned if Data size in the command header does not equal 126.
If "Invalid Other" is returned, further information is to be found in Extended word 8.

▪ Additional Fault information (Extended word 7)

If Extended word 8 equals 0x000A -"Failed to execute request" additional info can be found here.

▪ Fault information (Extended word 8)

0x0001: Invalid setting in Message data word 1-63 of the command.
0x0002: At least one slave reports a warning. Refer to Message data word 1-63.
0x000A: Failed to execute request. Additional fault information is to be found in Extended word 7.
0x00FE: Command not possible; module operates as class 2 Master only.
0x00FF: Module not initialized (this command is only possible after END_INIT).

▪ Message data word 1-63

Byte-array stating the status of the slaves. Array index is equal to slave address.
0: Slave unaffected
1: Slave stopped
2: Warning - Slave could not be stopped because it is not part of the configuration
3: Warning - Slave already stopped

3.6 Mailbox Messaging Error Codes

3.6.1 Acyclic Message Status Word

This register contains bit and code information about the mailbox message. The register is divided into five areas according to the following illustration:

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Message Type								ERR	C/R	(reserved)		Error Code			

Bit / Field	Description	Contents	
ERR	This bit indicates if the received command contained any errors.	0: 1:	Message OK Error (See also "Error Code" below)
C/R	This bit specifies whether the message is a command or a response.	0: 1:	Response Message Command Message
Error Code	If the ERR bit is set, this field contains additional information about the error.	0h: 1h: 2h: 3h: 4h: 5h: 6h: 8h: 9h: Fh:	Invalid Message ID Invalid Message Type Invalid Command Invalid Data Size Message header malformed (offset 008h) Message header malformed (offset 00Ah) Message header malformed (offset 00Ch to 00Dh) Invalid Response Flash Config Error **Invalid Other**
		(All other values are reserved)	
Message Type	This field specifies the type of the message.	1h: Application Message 2h: PROFIBUS Specific Message 3h: Memory Message 5h: Reset Message	
		(All other values are reserved)	

3.6.2 Return Codes

Possible error codes in Message Data word "Return Code" *(The Return Codes can be byte swapped)*

Return Code	Name	Meaning
8010h	DPMC_ERR_V1C_CLOSED	Internal DPMC instance no longer exists
8011h	DPMC_ERR_V1C_STOPPED	Internal DPMC instance has already been stopped
8012h	DPMC_ERR_V1C_STARTED	Internal DPMC instance has already been started
8013h	DPMC_ERR_V1C_STATE_UNKNOWN	Internal DPMC instance has entered an undefined state
8021h	DPMC_ERR_V1C_REQ_ACTIVE	A request is already active
8022h	DPMC_ERR_V1C_NOT_ALLOWED	Internal DPMC module not initialized correctly
8023h	DPMC_ERR_V1C_INVALID_PAR	Invalid parameter in user request
8024h	DPMC_ERR_V1C_MEM_ALLOC	Internal memory allocation error
8025h	DPMC_ERR_V1C_L2_REQ	Unknown opcode in the confirmation
8026h	DPMC_ERR_V1C_TIMEOUT	Active request terminated with timeout
8028h	DPMC_ERR_V1C_INVALID_LEN	Invalid length in user request
8030h	DPMC_ERR_V1C_REQ_NEG[1]	Negative indication from lower layer
8031h	DPMC_ERR_V1C_REQ_RE	Message frame format error in response
8042h	DPMC_ERR_V1C_REQ_WITHDRAW	Request was recalled
8043h	DPMC_ERR_V1C_REQ_NOT_FOUND	Associated request block not found
80C1h	DPMC_ERR_V1C_MM_FE	Format error in request frame
80C2h	DPMC_ERR_V1C_MM_NI	Function not implemented
80C3h	DPMC_ERR_V1C_MM_AD	Access denied
80C4h	DPMC_ERR_V1C_MM_EA	Area too large
80C5h	DPMC_ERR_V1C_MM_LE	Data block length too large
80C6h	DPMC_ERR_V1C_MM_RE	Format error in response frame
80C7h	DPMC_ERR_V1C_MM_IP	Invalid parameter
80C8h	DPMC_ERR_V1C_MM_SC	Sequence conflict
80C9h	DPMC_ERR_V1C_MM_SE	Sequence error
80CAh	DPMC_ERR_V1C_MM_NE	Area non-existent
80CBh	DPMC_ERR_V1C_MM_DI	Data incomplete or incorrect
80CCh	DPMC_ERR_V1C_MM_NC	Master parameter set not compatible

3.6.3 Error Codes

If the return code indicates DPMC_ERR_V1C_REQ_NEG, the status values according to the DP-standard may be available in Error Code 1. Refer to the PROFIBUS DP specification for information on how to interpret these status values.

Error Code	Name	Meaning
01h	L2_STATUS_UE	
02h	L2_STATUS_RR	
03h	L2_STATUS_RS	
0Ch	L2_STATUS_RDL	Refer to PROFIBUS DP specification
0Dh	L2_STATUS_RDH	
0Fh	L2_STATUS_NA	

3.6.4 DP-V1 Error Codes

Possible error codes in Message Data word "Return Code".

Return Code	Name	Meaning
0003h	DPMC_ERR_M_MEM_ALLOC	Internal memory allocation error
0004h	DPMC_ERR_M_L2_REQ	Unknown opcode in the configuration
0005h	DPMC_ERR_M_INVALID_PAR	Invalid parameter in user request
0007h	DPMC_ERR_M_NOT_IN_DATA	Slave is not in DataExchange (thus no DP-V1 request can exist)
0012h	DPMC_ERR_M_REQ_ACTIVE	A request is already active
0018h	DPMC_ERR_M_NOT_ALLOWED	Internal DPMC module not initialized correctly
0021h	DPMC_ERR_M_CLOSED	Internal DPMC instance no longer exists
0022h	DPMC_ERR_M_STOPPED	Internal DPMC instance has already been stopped
0023h	DPMC_ERR_M_STARTED	Internal DPMC instance has already been started
0024h	DPMC_ERR_M_STATE_UNKNOWN	Internal DPMC instance has entered an undefined state
002Fh	DPMC_ERR_M_SLAVE_NOT_FOUND	Slave does not respond
0031h	DPMC_ERR_M_TIMEOUT	Active request terminated with timeout
0034h	DPMC_ERR_M_INVALID_LEN	Invalid length in user request
0035h	DPMC_ERR_M_REQ_NEG	Negative indication from lower layer
0036h	DPMC_ERR_M_REQ_RE	Message frame format error in response
0037h	DPMC_ERR_M_REQ_WITHDRAW	Request was recalled
0038h	DPMC_ERR_M_REQ_NOT_FOUND	Associated request block not found
0040h	DPMC_ERR_M_MM_FE	Format error in request frame
0041h	DPMC_ERR_M_MM_NI	Function not implemented
0042h	DPMC_ERR_M_MM_AD	Access denied
0043h	DPMC_ERR_M_MM_EA	Area too large
0044h	DPMC_ERR_M_MM_LE	Data block length too large
0045h	DPMC_ERR_M_MM_RE	Format error in response frame
0046h	DPMC_ERR_M_MM_IP	Invalid parameter
0047h	DPMC_ERR_M_MM_SC	Sequence conflict
0048h	DPMC_ERR_M_MM_SE	Sequence error
0049h	DPMC_ERR_M_MM_NE	Area non-existent
004Ah	DPMC_ERR_M_MM_DI	Data incomplete or incorrect
004Bh	DPMC_ERR_M_MM_NC	Master parameter set not compatible
004Ch	DPMC_ERR_M_S7_XA	
004Dh	DPMC_ERR_M_S7_XR	PROFIBUS error for DP-V1 (NRS-PDU received)
004Eh	DPMC_ERR_M_S7_XW	

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

4.1 Basic Troubleshooting Steps

1   Verify that the module is installed correctly, and is communicating with the processor.
2   Install the most current version of *ProSoft Configuration Builder*.
3   Note the color and behavior of the LED Status Indicators (lights) on the front panel. Refer to the chart in the following section for examples.

4.2 LED Status Indicators: Front of MVI56 Module

The LEDs indicate the module's operating status. The module has two sets of LED status indicators:

- MVI56 Module Status LEDs on the front of the module near the top
- PROFIBUS Master Status LEDs behind the door on the front of the module.
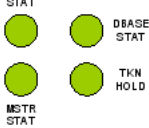
4.2.1 Module Faceplate Status Indicators

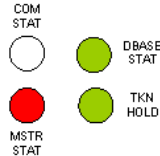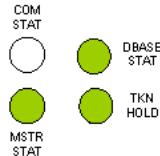| Indicator	Color	Status	Indication
CFG	Green	ON	Configuration/Debug Port is active
		OFF	Configuration/Debug Port is inactive
P1	Not Used		
P2	Not Used		
APP STATUS	Amber	ON	Configuration Error: This LED is illuminated when the PROFIBUS and module CRC values do not match between input/output blocks. Verify that the values match the values generated with the Calculate Checksums button in *ProSoft Configuration Builder*. If they do not match, you must manually change the values at the appropriate address locations listed in the Input and Output block definitions in this manual.
		OFF	Normal operation.
BP ACT	Amber	ON	The LED is ON when the module is able to communicate over the backplane.
		OFF	The LED is OFF when the module is unable to communicate with the processor. The processor is either absent or not running.
BATT	Red	OFF	The battery voltage is OK and running.
		ON	The battery voltage is low or the battery is not present. The battery LED will illuminate briefly upon the first installation of the module or if the unit has not had power for an extended period of time. This behavior is normal; however, should the LED come ON in a working installation, please contact ProSoft Technology.
OK	Red/ Green	OF	The card is not receiving any power and is not securely plugged into the rack.
		GREEN	The module is operating normally.
		RED	The program has detected an error or is being configured. If the LED remains RED for over 10 seconds, the processor program may be incompatible with the module. Check the module version (page 18) and mode (page 20) (Legacy or Flex) to determine the correct sample ladder logic. Remove the card from the rack and re-insert the card to restart the module's program.

4.2.2 PROFIBUS Master Indicators

LED	State	Description
TKN HLD	GREEN	The module has the token
	OFF	The module does not have the token
DBASE STAT	GREEN	Database OK
	GREEN-Flashing	Database download in progress
	RED	Invalid database
	OFF	No databases have been downloaded
MSTR STAT	GREEN	Operating mode
	GREEN-Flashing	Clear mode
	RED	Stop mode
	OFF	Offline
COM STAT	GREEN	Data exchange with all configured slaves
	GREEN-Flashing	Data exchange with at least one configured slave but not all configured slaves
	RED	Bus control error (possible bus short circuit or configuration error)
	OFF	No data exchange with any configured slave
ALL LEDs	RED	Fatal error

4.2.3 Examples

The following table shows some of the possible status indicators.

	Processor Status	Module Status	PROFIBUS Master Status	PROFIBUS Master Status LED Description
Normal Operation	RUN (ON)	ACTIVE (ON)		**COM STAT (GREEN):** Master is communicating with all configured slaves. **DBASE STAT (GREEN):** PROFIBUS has been configured. **TKN HOLD (GREEN):** Module is holding the PROFIBUS token. **MSTR STAT (GREEN):** Master is in operating mode.

	Processor Status	Module Status	PROFIBUS Master Status	PROFIBUS Master Status LED Description
PROFIBUS Master is stopped	RUN	ACTIVE	COM STAT (white/off), DBASE STAT (green), MSTR STAT (red), TKN HOLD (green)	**COM STAT (OFF):** Master is not communicating with configured slaves. **DBASE STAT (GREEN):** PROFIBUS has been configured. **TKN HOLD (GREEN):** Module is holding the PROFIBUS token. **MSTR STAT (RED):** Master is in STOP mode.
CPU is in Program mode	PROG	ACTIVE	COM STAT (white/off), DBASE STAT (green), MSTR STAT (red), TKN HOLD (green)	**COM STAT (OFF):** Master is not communicating with configured slaves. **DBASE STAT (GREEN):** PROFIBUS has been configured. **TKN HOLD (GREEN):** Module is holding the PROFIBUS token. **MSTR STAT (RED):** Master is in STOP mode.
CPU is in Program mode	PROG	ACTIVE	COM STAT (white/off), DBASE STAT (green), MSTR STAT (green), TKN HOLD (green)	**COM STAT (OFF):** Master is operating but there are no slaves responding. **DBASE STAT (GREEN):** PROFIBUS has been configured. **TKN HOLD (GREEN):** Module is holding the PROFIBUS token. **MSTR STAT (GREEN):** Master is in operating mode.

As previously indicated in this User Manual, the module requires the correct sample program loaded to the processor for correct initialization and operation. If the sample program is not set up correctly, then the module might not fully initialize or allow access to the debug menu through PCB.

As indicated below, it is required to have a sample program loaded in the processor in order to allow debug menu access through PCB and to be able to reconfigure the MVI56-PDPMV1 module.

Also, if you observe the MSTR STAT LED as RED or blinking GREEN/RED it means that an incorrect sample ladder logic was loaded to the processor (not matching the actual module mode (LEGACY or FLEX).

The following tables will help you to troubleshoot different scenarios and the LED behavior.

4.2.4 Legacy Mode (Use Legacy Mode = Yes)

Scenario	Debug Access After Power Up	LED Behavior After Power Up
- Module located in slot that is not configured OR - No processor OR - Neither sample logics loaded to processor (LEGACY or FLEX)	No (Waiting for connection to processor)	MSTR STAT= OFF COM STAT=OFF DBASE STAT=GREEN TKN HLD= OFF OK = RED BP ACT = AMBER APP STATUS= AMBER
LEGACY Ladder loaded to processor (module is inserted in configured slot)	Yes	MSTR STAT= GREEN COM STAT= GREEN or OFF (depends if the module is communicating with PROFIBUS slaves) DBASE STAT= GREEN TKN HLD= GREEN OK = GREEN BP ACT = AMBER APP STATUS= AMBER
FLEX Ladder	Yes	MSTR STAT= GREEN (1sec) /RED (2 secs) COM STAT= OFF DBASE STAT= GREEN TKN HLD= GREEN OK = GREEN BP ACT = OFF APP STATUS= AMBER

4.2.5 Flex Mode (Use Legacy Mode=No)

Scenario	Debug Access After Power Up	LED Behavior After Power Up
- Module located in slot that is not configured OR - No processor OR - Neither sample logics loaded to processor (LEGACY or FLEX)	No (Waiting for connection to processor)	MSTR STAT= OFF COM STAT=OFF DBASE STAT=GREEN TKN HLD= OFF OK = RED BP ACT = AMBER APP STATUS= AMBER
LEGACY Ladder	Yes	MSTR STAT= RED COM STAT= OFF DBASE STAT= GREEN TKN HLD= GREEN OK = GREEN BP ACT = OFF APP STATUS= AMBER
FLEX Ladder	Yes	MSTR STAT= GREEN COM STAT= GREEN or OFF (depends if the module is communicating with PROFIBUS slaves) DBASE STAT= GREEN TKN HLD= GREEN OK = GREEN BP ACT = AMBER APP STATUS= OFF

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

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

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

4.3.1   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.
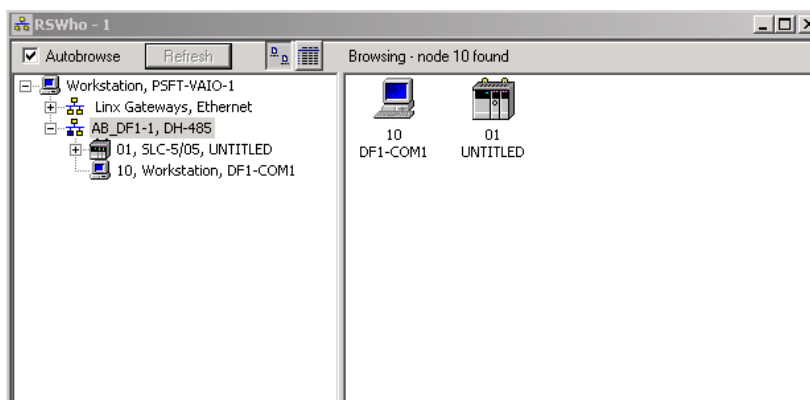
3   Notice how the DF1 driver is opened, and the driver is looking for a processor on node 1. If the network is being browsed, then you will not be able to stop this driver. To stop the driver your *RSWho* screen should look like this:
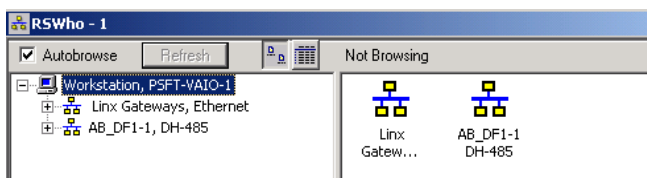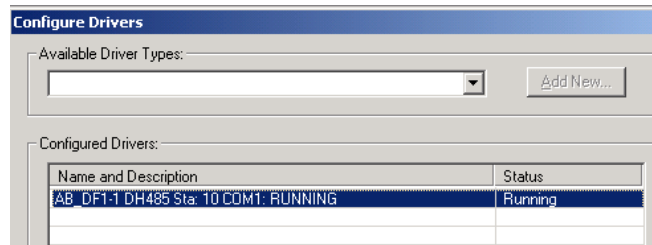
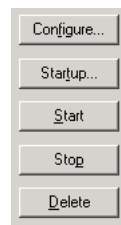Branches are displayed or hidden by clicking on the ⊞ or the ⊟ icons.

白 品 AB_DF1-1, DH-485
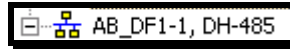
4 When you have verified that the driver is not being browsed, go to
COMMUNICATIONS > **CONFIGURE DRIVERS**.

You may see something like this:

Configure Drivers
Available Driver Types:
[▼] Add New...

Configured Drivers:
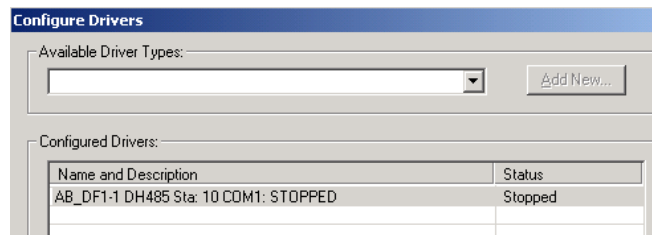| Name and Description	Status
AB_DF1-1 DH485 Sta: 10 COM1: RUNNING	Running

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:

Configure...
Startup...
Start
Stop
Delete

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

Configure Drivers
Available Driver Types:
[▼] Add New...

Configured Drivers:
| Name and Description	Status
AB_DF1-1 DH485 Sta: 10 COM1: STOPPED	Stopped

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.

4.3.2 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.
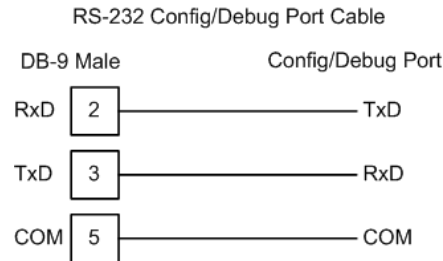

4.3.3 DB9 to RJ45 Adaptor (Cable 14)

Wiring Diagram
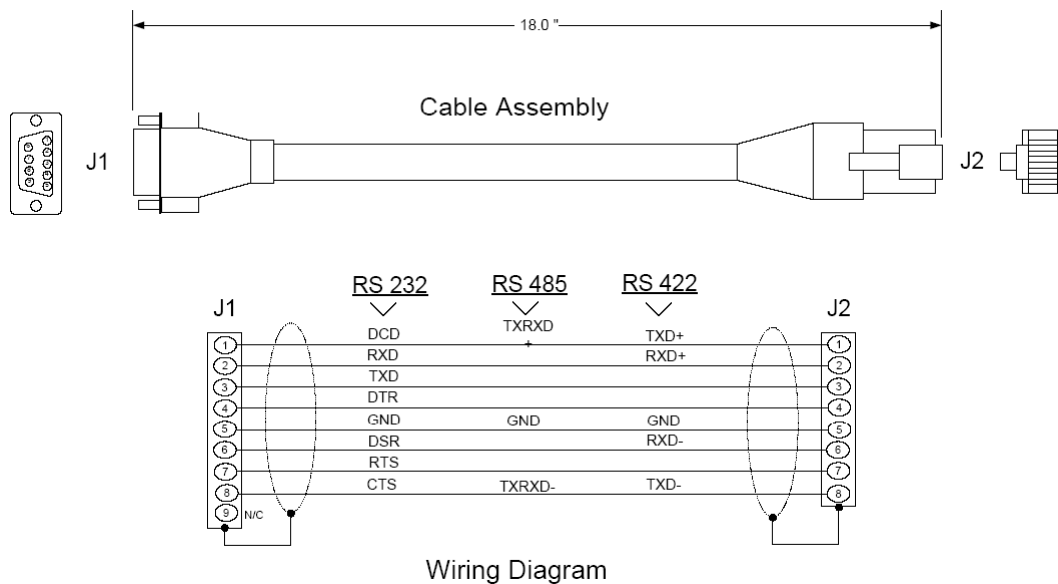
4.3.4 Using the Diagnostic Window in ProSoft Configuration Builder
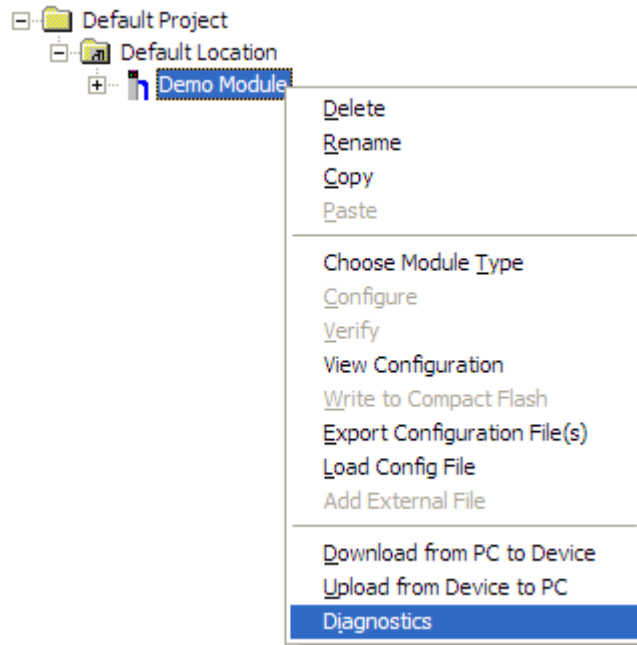
Tip: You can have a ProSoft Configuration Builder Diagnostics window open for more than one module at a time.

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

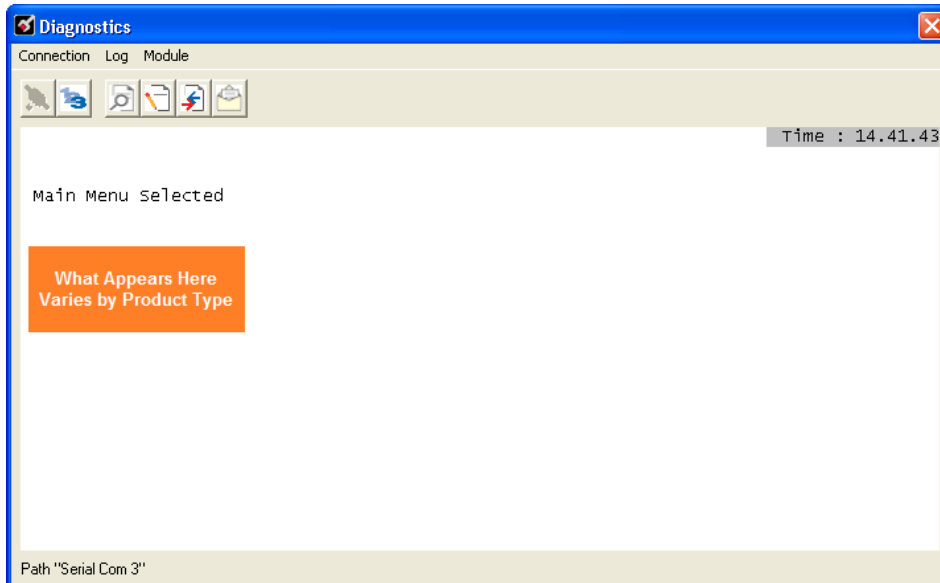1   Start *PCB*, and then select the module to test. Click the right mouse button to open a shortcut menu.
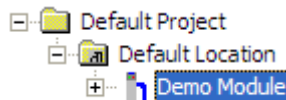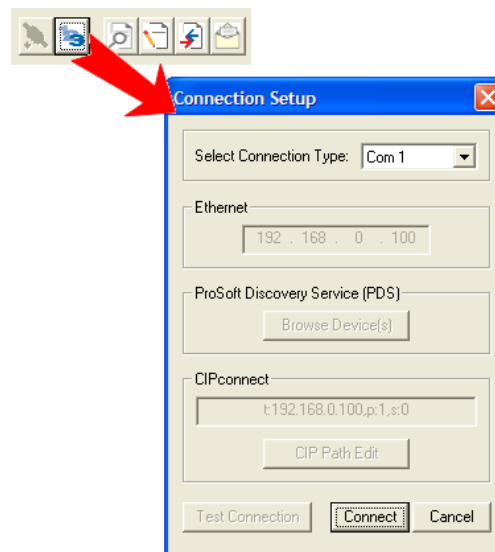

2   On the shortcut menu, choose **DIAGNOSTICS**.

This action opens the *Diagnostics* dialog box.

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

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

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

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

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

Navigation

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

The organization of the menu structure is represented in simplified form in the following illustration:
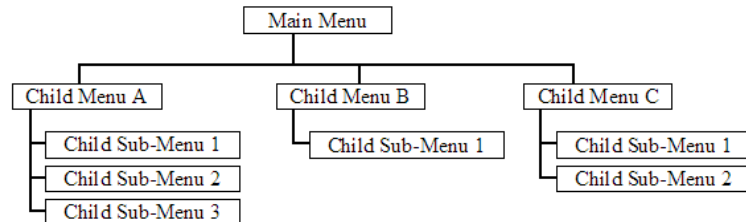
The remainder of this section shows the menus available for this module, and briefly discusses the commands available to you.

Keystrokes

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

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

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

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.

```
MVI56-PDPM COMMUNICATION MODULE MENU
 ?=Display Menu
 B=Block Transfer Statistics
 C=Module Configuration
 I=Input Data View
 O=Output Data View
 V=Version Information
 1=Module Status
 2=Fieldbus Data
 3=Control Registers
 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 communication 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 Engineers.
There may be some special command keys that are not listed on the menu but that may activate additional diagnostic or debugging features. If you need these functions, you will be advised how to use them by Technical Support. Please be careful when pressing keys so that you do not accidentally execute an unwanted command.

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 Block Transfer Statistics

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

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

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

Viewing Module Configuration

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

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

Viewing Input Data

Press **[I]** to open the *Input Database View* menu. Use this command to see the contents of data placed in the Input database.

Viewing Output Data

Press **[O]** to open the *Output Database* menu. Use this command to see the contents of data placed in the Output database.

Viewing Version Information

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

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

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

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

Viewing Module Status

Press **[1]** to view status information about the module. This screen also contains useful information for mailbox troubleshooting:

- Scan count
- Mailbox counters
- Alarm counters
- Number of acyclic read and write operations performed by the module

You can also view the number of mailbox messages in the input and output queues, and the number of alarms in the alarm queue.

Viewing PROFIBUS Data

Press **[2]** to view PROFIBUS data. Use this command to view information related to the status of each slave in the PROFIBUS network, and to verify that each slave is configured (SLAVE CFG LIST), exchanging data with the Master (TRANSFER LIST) and in diagnostic mode (SLAVE DIAG LIST).

You can also check the module's operation state, where:

- 00 = Offline
- 40 = Stop
- 80 = Clear
- C0 = Operate

Viewing Control Registers

Press **[3]** to view information about the PROFIBUS Master's Control Registers. Use this command to view general information about the module, such as the firmware version and its serial number. The module status contains two possible codes:

- 0400 = module is running but not communicating with slaves
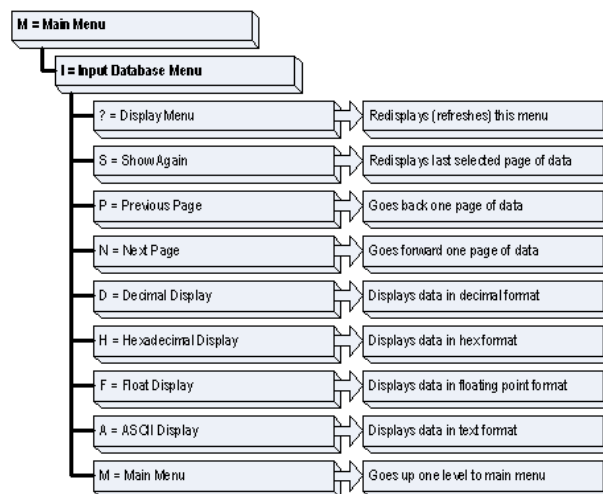- 0401 = module is running and communicating with slaves

If the module is in STOP mode, the status code is displayed as 0400.

Exiting the Program

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

Input Database View Menu

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

M = Main Menu		
I = Input Database Menu		
? = Display Menu	Redisplays (refreshes) this menu
S = Show Again	Redisplays last selected page of data
P = Previous Page	Goes back one page of data
N = Next Page	Goes forward one page of data
D = Decimal Display	Displays data in decimal format
H = Hexadecimal Display	Displays data in hex format
F = Float Display	Displays data in floating point format
A = ASCII Display	Displays data in text format
M = Main Menu	Goes up one level to main menu

Displaying the Current Page of Registers Again

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

```
DATABASE DISPLAY 0 TO 99 (DECIMAL)
    100     101     102       4       5       6       7       8       9      10
     11      12      13      14      15      16       0       0       0       0
      0       0       0       0       0       0       0       0       0       0
      0       0       0       0       0       0       0       0       0       0
      0       0       0       0       0       0       0       0       0       0
      0       0       0       0       0       0       0       0       0       0
      0       0       0       0       0       0       0       0       0       0
      0       0       0       0       0       0       0       0       0       0
      0       0       0       0       0       0       0       0       0       0
      0       0       0       0       0       0       0       0       0       0
```

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

Viewing the Previous Page of Registers

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

Viewing the Next Page of Registers

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

Viewing Data in Decimal Format

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

Viewing Data in Hexadecimal Format

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

Viewing Data in Floating-Point Format

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

Viewing Data in ASCII (Text) Format

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

Returning to the Main Menu

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

Output Database View Menu

Press **[O]** from the *Main* menu to open the *Database View* menu. Use this menu command to view the current contents of the module's database. Press **[?]** to view a list of commands available on this menu.
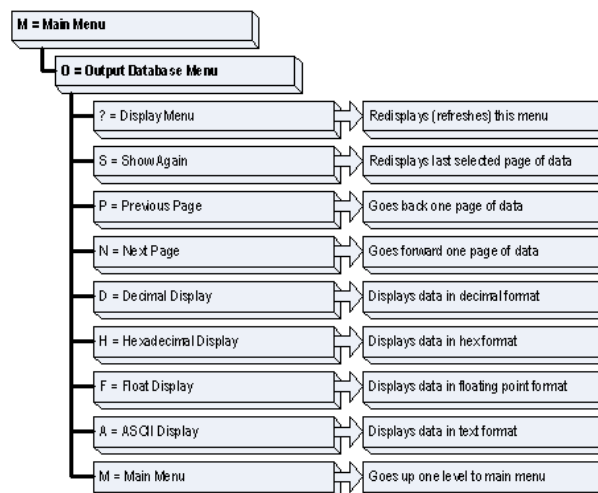
Displaying the Current Page of Registers Again

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

```
DATABASE DISPLAY 0 TO 99 <DECIMAL>
   100    101    102      4      5      6      7      8      9     10
    11     12     13     14     15     16      0      0      0      0
     0      0      0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0      0      0
```

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

Viewing the Previous Page of Registers

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

Viewing the Next Page of Registers

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

Viewing Data in Decimal Format

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

Viewing Data in Hexadecimal Format

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

Viewing Data in Floating-Point Format

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

Viewing Data in ASCII (Text) Format

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

Returning to the Main Menu

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

Viewing PROFIBUS Slave Diagnostic Bytes

Use the Get Slave Diagnostics mailbox message (page 153) to retrieve PROFIBUS slave diagnostic bytes. Refer to Mailbox Messaging (page 133) to see how to view the bytes returned by this mailbox message.

Standard PROFIBUS Slave Diagnostic Bytes

The diagnostic information consists of 6 bytes of standard diagnostic information plus any user-related diagnostic information. The standard information is shown in the tables below.

Byte	Description
0	Station status 1
1	Station status 2
2	Station status 3
3	Master address
4	Ident number high
5	Ident number low

Byte 0 - Station Status 1 Bits

Bit	Description
0	Station not existent
1	Station not ready
2	Configuration fault
3	Extended diagnostic data present
4	Not supported
5	Invalid slave response
6	Parameter fault
7	Master lock

Byte 1 - Station Status 2 Bits

Bit	Description
0	Parameter request
1	Static diagnostic
2	Slave device
3	Watchdog on
4	Freeze mode
5	Sync mode
6	Reserved
7	Slave deactivated

Byte 2 - Station Status 3 Bits

Bit	Description
0	Reserved
1	Reserved
2	Reserved
3	Reserved
4	Reserved
5	Reserved
6	Reserved
7	Extended diagnostic overflow

Byte 3 - Master Address

This byte shows the address of the assigned PROFIBUS Master after parameterization. If there is an error during the parameterization process, this byte will display the value FF (hexadecimal).

Byte 4 - Ident Number High

This byte shows the high byte of the specific Ident Number assigned to the module by the PROFIBUS User Organization.

Byte 5 - Ident Number Low

This byte shows the low byte of the specific Ident Number assigned to the module by the PROFIBUS User Organization.

5   Reference

In This Chapter

5.1 Product Specifications

The PROFIBUS DPV1 Master module is a powerful communication interface for Rockwell Automation® ControlLogix® processors. Developed under license from Rockwell Automation, the module incorporates proprietary backplane technology that enables powerful data exchange with ControlLogix processors.

The module supports complete Master specifications according to IEC 61158. Acyclic parameter data can be transferred with Class 1 or Class 2 DPV1 services, allowing processors to easily communicate with slave devices supporting PROFIBUS DPV0/V1 protocol.

The module acts as an input/output module between the PROFIBUS network and the ControlLogix processor. Data transfer from the processor is asynchronous with the I/O data transfer on the PROFIBUS network.

5.1.1 General Specifications

- Single slot - 1756 backplane compatible
- The module is recognized as an Input/Output module and has access to processor memory for data transfer between processor and module
- Ladder Logic is used for data transfer between module and processor. Sample ladder file (add-on instruction) included.
- Local or remote rack
- Configurable I/O backplane sizes allow optimal performance for remote rack applications

- CIPconnect® enabled. Module supports communications with ProSoft Configuration Builder and FDT comDTM via Ethernet routing using Rockwell Automation Ethernet interface.

5.1.2 Hardware Specifications

Specification	Description
Backplane Current Load	800 mA @ 5 Vdc 3 mA @ 24 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)
LED Indicators	Module Status Backplane Transfer Status Application Status Serial Activity and Error LED Status
Debug/Configuration port (CFG)	
CFG Port (CFG)	RJ45 (DB-9M with supplied cable) RS-232 only CIPConnect® Ethernet routing using 1756-ENBT, 1756-EN2T or similar
Shipped with Unit	RJ45 to DB-9M cables for each port 6-foot RS-232 configuration cable

5.1.3 Functional Specifications

The inRAx PDPMV1 PROFIBUS Master module acts as a PROFIBUS network scanner, transferring input and output data between PROFIBUS devices and processor data memory over the backplane.

- Master Busview configuration interface via included ProSoft Configuration Builder software (Part Number PSW-PCB)
- Project-unique GSD file import library
- Monitoring and modification of process data and DPV1 acyclic data
- Multi-drop on a PROFIBUS DPV1 network with other compatible devices
- Automatic project documentation
- Automatic Bus Parameter calculation
- Online slave diagnostics
- Supports all standardized baud rates, up to 12 Mbps
- Supports extended diagnostic data (DPV1)
- Multiple Masters can be placed in a single rack
- CRC checksum determination of slave configuration consistency to processor
- Master Status LED indicators for Operations, Network Communication, Master Token-Hold and Network Configuration
- FDT/DTM PROFIBUS Master transport communication DTM software included (Part Number PSW-CDTM-PDPM)

PROFIBUS Master Specifications

- Up to 125 slaves can be connected with a repeater
- Up to 1536 cyclic bytes input and 1536 bytes output data
- Supports Extended Diagnostic Data
- RS-485 optically isolated PROFIBUS interface with on board DC-DC converter
- Acyclic communications (DPV1), Read and Write
- Alarm handling (DPV1)
- Supports Sync and Freeze commands
- Supports PROFIdrive 3.1 compliant parameter read and write operations
- Supports Multicast and Broadcast telegrams (DPV1)

PROFIBUS Master Port

- DB-9F optically isolated RS-485
- Ready, Run, Error and Token LED Indicators
- PROFIBUS DPV1 RS-485 interface with a 9-pin D shell female connector and isolated opto-couplers

5.2   About the PROFIBUS Protocol

PROFIBUS (Process Field Bus) is a widely-used, open-standards protocol created by a consortium of European factory automation suppliers in 1989.

PROFIBUS is a Master/slave protocol. The Master establishes a connection to one or more remote slaves. When the connection is established, the Master sends the PROFIBUS poll messages (called telegrams in PROFIBUS) to the slave or slaves. The MVI56-PDPMV1 module works as a Master only. It cannot be a slave to some other Master.

The MVI56-PDPMV1 module also acts as an input/output module between devices on a PROFIBUS network and the Rockwell Automation ControlLogix processor. The module uses an internal database to pass data and mailbox requests and responses between the processor and the slave devices on the PROFIBUS network.

PROFIBUS specifications include a variety of network types. The network type supported by the MVI56-PDPMV1 module is PROFIBUS DP version 1.0, which is designed for remote I/O systems, motor control centers, and variable speed drives.

5.2.1   PROFIBUS DP Architecture

The PROFIBUS DPV1 network supports multiple Master systems with several slaves.

The following table shows the most important features of PROFIBUS DPV1:

Standard	EIN 501 70 DIN 19245
Transmission equipment (Physical)	EIA RS-485 IEC 1158-2 (through link or coupler) Fiber Optic Cable (not available)
Transfer procedure	Half-duplex
Bus topology	Linear bus with active bus termination
Bus cable type	Shielded twisted pair conductors
Connector	9-pin D-Sub
Number of nodes on the bus	Max: 32 with no repeaters Max: 125 with 3 repeaters in 4 segments

5.2.2   Bus Access

Two different bus access procedures handle the various communication requirements for the PROFIBUS DPV1 topology:

- Token Passing
- Polling

Token Passing

Token passing ring is the basis for communication between the more complex, active stations. All stations have the same rights in that a token is passed from station to station in a logical ring. The token is passed to each station with a maximum, definable token cycle time. A station is given transmission rights for the duration of time that it has the token.

Master/Slave Polling

Master/slave polling guarantees a cyclic, real-time based data exchange between the station with transmission rights, the active station, and its subordinates, the passive stations. In this case, the Master is able to pass data to the slave and/or receive data. The services in layer 2 (field-bus data link in ISO-OSI reference model) organize this communication.

5.2.3 Master/Slave Communication Phases

The communication between the Master and the slaves takes place in the following phases:

▪ Parameterization and configuration phase
▪ Usable data transfer phase

Before a DP slave can be integrated into the usable data transfer phase, the parameterization and configuration phase runs a device identification test that verifies that the planned configuration matches the actual device configuration for each slave in the PROFIBUS network. The test verifies that:

▪ The device is actually there
▪ It is the right type of device
▪ The station address set on the device matches the station address in the bus configuration
▪ The formats, telegram length information, and bus parameters are correct and
▪ The number of configured inputs and outputs is correct

5.2.4 How Cable Length Affects Communication Rate

Max Bus Cable Length Per Segment	Baud Rates (for 12 Mbps cable)
1.2 km	9.6 kbit/sec
1.2 km	19.2 kbit/sec
1.2 km	93.75 kbit/sec
1.0 km	187.5 kbit/sec
0.5 km	500 kbit/sec
0.2 km	1.5 Mbit/sec
0.1 km	3 Mbit/sec
0.1 km	8 Mbit/sec
0.1 km	12 Mbit/sec

5.2.5 PROFIBUS Master Port

The following diagram has been imported from the PROFIBUS Master documentation. Note that the signals to reference are the D-Sub signals in the table.

D-Sub (male)	Board to Board	Screw Terminal	Signal
Housing	1	5	Cable shield
1	4	-	-
2	7	-	-
3	6	4	B-Line
4	3	6	RTS
5	2	2	GND_BUS
6	8	1	+5V BUS (output)
7	9	-	-
8	5	3	A-Line
9	10	-	-
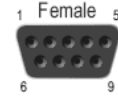
5.2.6 Constructing a Bus Cable for PROFIBUS DP

The bus cable for connecting PROFIBUS DP devices must be constructed by the user. A special PROFIBUS cable (twisted pair) is required here. This standard cable is available from various manufacturers and is a Belden part number 3079A.

If you plan to construct your own bus cable, the following part numbers are provided for your convenience.

- PROFIBUS connector: Siemens part number 6ES7972-0BA40-0XA0
- PROFIBUS cable: Belden part number 3079A.

To construct the cable, proceed as follows:

1   Cut the cable to the required length.
2   Prepare the cable ends as shown in the illustration (dimensions in mm):
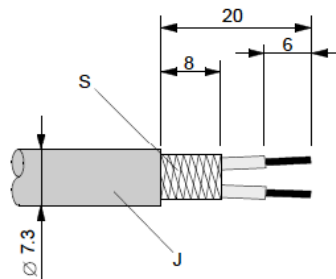
   J   PVC Jacket
   S   Braided shielding
3   Remove the PVC jacket J to the indicated length.
4   Wrap the provided copper shielding F around the shield braiding S:
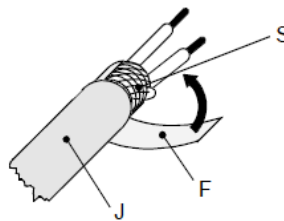
   J   PVC jacket
   S   Braided shielding
   F   Copper foil shielding
   Additional foil can be obtained from 3M.
5   Plug the leads of the corresponding cable(s) into the terminals as shown:
   o   Green leads in terminal A
   o   Red lead in terminal B

- **Note:** Do **not** tighten the corresponding screws yet.

Connection terminal assignment on the PROFIBUS DP:
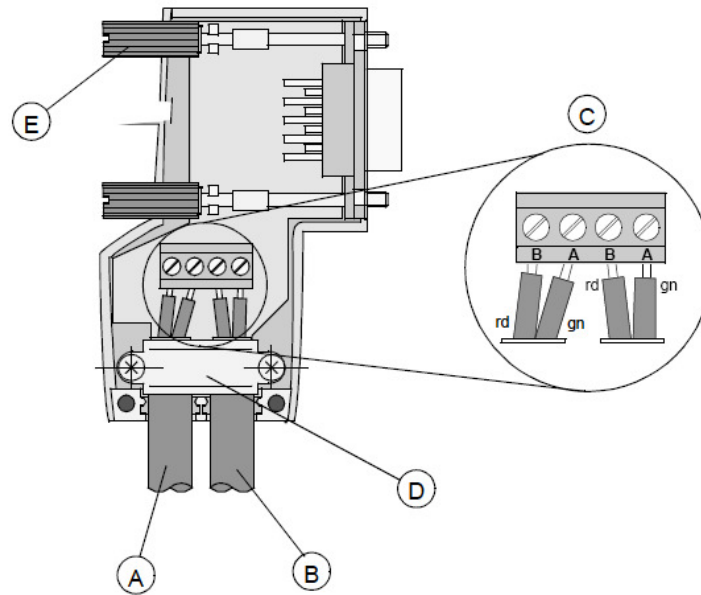
A Incoming cable
B Outgoing cable
C Connection terminals (only once (B,A))
D Cable cleat for reliving tension
E Bus connector screws

6   Attach the cables with the provided cable cleat to create a robust shielded connection and to relieve any tension as shown:
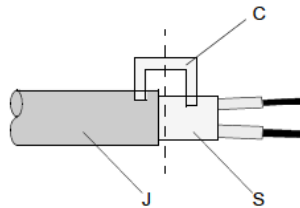
J PVC Jacket
S Braided shielding with foil shielding
C Cable cleat

▪   **Note**: Half of the cable jacket must lie under the cable cleat!

Pay attention to the cable cleat installation instructions.

7   Fasten the individual wires of the PROFIBUS cable to the terminals
8   Close the connector housing.

▪ **Note:** The shielding of both cables is connected internally with the metal housing of the connector.

9   Complete the Central Shielding Measures (below) and grounding operations for the shielding before you connect the cable connector to the module.
10 Plug the PROFIBUS DP connector into the module and secure it with the screws.

Bus Begin and Bus End

The PROFIBUS connector with termination is required at the beginning and the end of the bus. These connectors emulate the line impedance.

It is recommended that at least one connector with diagnostics interface is used.

Wiring diagram for a PROFIBUS DP cable
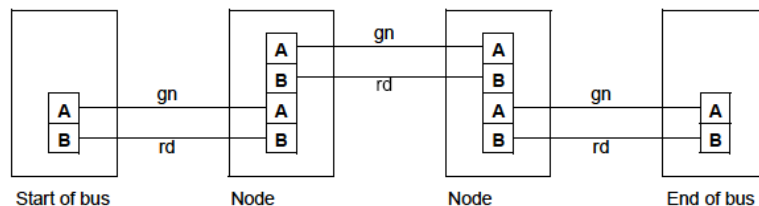
Grounding and Shielding for Systems with Equipotential Bonding

Each cable shield should be galvanically grounded with the earth using FE/PE grounding clamps immediately after the cable has been connected to the cabinet.

This example indicates the shielding connection from the PROFIBUS cable to the FE/PE rail.
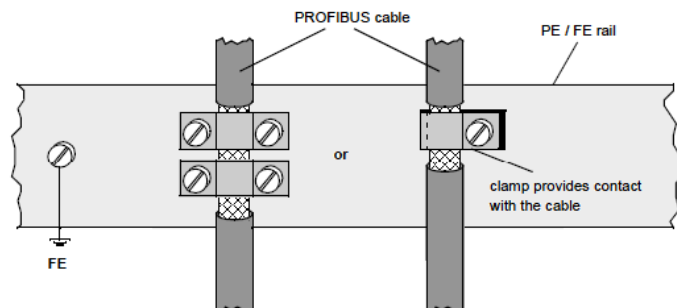
Note: An equalization current can flow across a shield connected at both ends because of fluctuations in ground potential. To prevent this, it is imperative that there is potential equalization between all the attached installation components and devices.

This example indicates the system components and devices in a system with equipotential bonding.
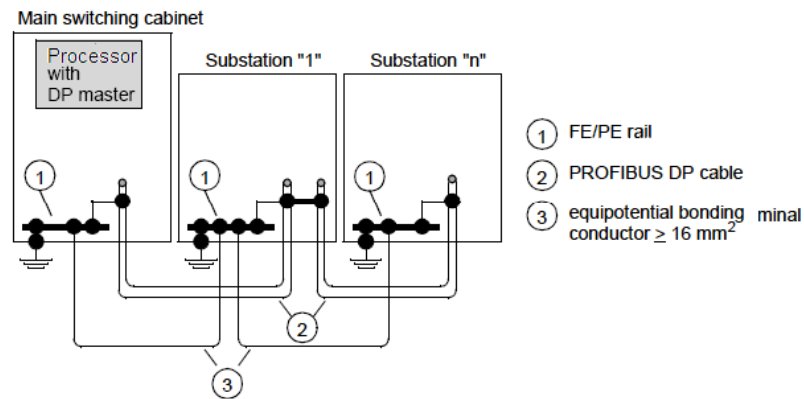
Grounding and Shielding for Systems without Equipotential Bonding

Note: Grounding and shielding is to be carried out the same as for systems **with** equipotential bonding.

If this is not possible because of system or construction specific reasons however, use distributed ground with a capacitive coupling of high frequency interference signals.

This representation shows distributed grounding with capacitive coupling.
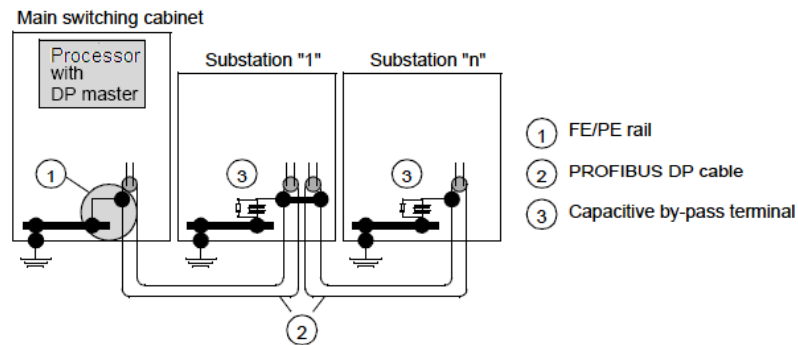
5.2.7 Supported PROFIBUS Services

The following table lists all available services according to the PROFIBUS specification.

Service	PROFIBUS Version	Master Class 1		Master Class 2	
		Request	Response	Request	Response
DDLM_Data-Exchange	DP-V0	Yes		No	
DDLM_Set_Prm	DP-V0	Yes		No	
DDLM_Chk_cfg	DP-V0	Yes		No	
DDLM Slave Diag	DP-V0	Yes		No	
DDLM_Global_Control	DP-V0	Yes		No	
DDLM_Get_Cfg	DP-V0			Yes	
DDLM_Set_Slave_Add	DP-V0			Yes	
DDLM_Read_Input	DP-V0			No	
DDLM_Read_Output	DP-V0			No	
DDLM_Get_Master_Diag	DP-V0		Yes		
DDLM_Start_Seq	DP-V0		No	No	
DDLM_Download	DP-V0		No	No	
DDLM_Upload	DP-V0		No	No	
DDLM_End_Seq	DP-V0		No	No	
DDLM_Act_Param_Brct	DP-V0		No	No	
DDLM_Act_Param	DP-V0		No	No	
MSAC1_Read	DP-V1	Yes			
MSAC1_Write	DP-V1	Yes			
MSAL1_Alarm	DP-V1		Yes		
MSAL1_Alarm_Ack	DP-V1		Yes		
MSAC2_Initiate	DP-V1			No	
MSAC2_Read	DP-V1			No	
MSAC2_Write	DP-V1			No	
MSAC2_DataTransport	DP-V1			No	
MSAC2_Abort	DP-V1			No	
Data_eXchange_Broadcast	DP-V2	No			
Isochrone_mode (Takt sync)	DP-V2	No			
Extended_Set_Prm (Subscriber)	DP-V2	No			

5.3   PROFIBUS comDTM

DTM (Device Type Manager) is a standard way to provide all necessary data and functionality for a communication device, for example a PROFIBUS DP card. This technology is similar to the way Microsoft Windows supports printer drivers supplied by manufacturers and available to any Windows application, rather than requiring a custom printer driver for each specific application.

PROFIBUS comDTM, distributed by ProSoft Technology, is a DTM for PTQ and MVI series PDPMV1 modules and ProLinx PDPMV1 gateways. Configuration is available through Ethernet for the PTQ and ProLinx series PDPMV1, and through RS-232 serial for the MVI series PDPMV1 modules. Ethernet configuration is also available through the CIPConnect® feature for MVI56-PDPMV1 and MVI69-PDPMV1 modules (see note).

PROFIBUS comDTM allows configuration tools, instruments, and communication devices on a field network to recognize and use the module's capabilities.

Communication Channels Supported

	Serial	Ethernet	
Product	**Local RS-232 Port**	**Local Ethernet Port**	**CIPconnect**
MVI46-PDPMV1	Supported		
MVI56-PDPMV1	Supported		Supported (see note)
MVI69-PDPMV1	Supported		Supported (see note)
PTQ-PDPMV1		Supported	
ProLinx PDPMV1		Supported	

Note: MVI56-PDPMV1 requires firmware version 1.28.000 or later to support CIPconnect-enabled PC programs (1756-ENBT support). The 1756-ENBT card must be located in the same rack as the MVI56-PDPMV1. Bridging through multiple racks is not supported by the comDTM software. MVI69-PDPMV1 requires firmware version 1.37.002 or later to support CIPconnect through the Ethernet port to L32E and L35E CompactLogix processors.
Note: This functionality requires comDTM version 1.0.1.5 with install version 1.01.0003. For information on how to check the comDTM version and install version, refer to Verifying the comDTM Version and comDTM Install Version (page 223).

5.3.1 ProSoft Technology Product Availability

Part Number	Description
PSW-cDTM-PDPM	PROFIBUS DPV1 Master comDTM software gateway

5.3.2 Introduction to PROFIBUS comDTM

Why Use PROFIBUS comDTM?

Customers from around the world have different machines, fieldbusses, and other end-user equipment. Each is equipped with the field bus requested by their end-user. Since there are so many variations, the automation solution in their standard machine ends up being different from case to case.

This means that service engineers need to have different configuration tools for every fieldbus. Or maybe even one for every device. You want one, but the reality is you have many. This is where PROFIBUS comDTM can help with configuring and communicating with different networks, products and suppliers.

What is PROFIBUS?

PROFIBUS (Process Field Bus) is a widely-used, open-standards protocol created by a consortium of European factory automation suppliers in 1989.

PROFIBUS is a Master/slave protocol. The Master establishes a connection to the remote slave. When the connection is established, the Master sends the PROFIBUS commands to the slave.

PROFIBUS supports a variety of network types. The network type supported by the PTQ and MVI series PROFIBUS modules is PROFIBUS DP (Device Bus) version 1.0, which is designed for remote I/O systems, motor control centers, and variable speed drives.

What is FDT?

FDT (Field Device Tool) is the specification for software interfaces for the integration of field devices, drives, and controls in engineering and configuration tools. FDT is manufacturer-independent and allows for trouble-free parameterization and configuration of the user's specific processing system.

FDT technology standardizes the communication interface between field devices and systems. The key feature is its independence from the communication protocol and the software environment of either the device or the host system. FDT allows any device to be accessed from any host through any protocol.

The FDT container implements the FDT specification. It serves as an interface between FDT and a superior application. It uses the DTMs to gain access to the devices. FDT frame application is an engineering or configuration tool which has an FDT container.

FDT technology comprises three key components: the Frame Application, Device DTMs, and Communication DTMs.

- The DTM (Device Type Manager) is used for the configuration and maintenance of field devices, drives and so on. It is only functional with an FDT container.
- The FDT container implements the FDT specification. It serves as interface between FDT and a superior application. It uses the DTMs to gain access to devices.
- FDT frame application is an engineering or configuration tool that has an FDT container. The user interface of the DTMs is displayed here.

To better understand the functionality of these components, consider the analogy of the Internet - a standard web browser allows users to view countless web pages created by many content providers. The host system supplier typically creates the Frame Application, just as Microsoft supplies the Internet Explorer web browser. Just as a web browser opens a web page that contains code from the company that makes the web page, an FDT frame opens the Device DTM, which is the plug-in software from the device vendor.

Similar to a web browser, the Frame Application has menu bars, toolbars, and a navigation tree. Visually, the frame application surrounds the device vendor's DTM. Like opening a web page from a 'favorites' navigation tree, a user can navigate down a tree that lists the field device tags, click on one, and open the device vendor's DTM inside the frame. And, like web pages that let users interact with a reservation system or a shopping service, the Device DTMs let the user interact with the field device in a wide variety of ways. The Device DTM vendor can create a graphically rich user interface that does virtually anything possible in an advanced Windows PC-type interface. The third part of the technology, the Communication DTM, provides a standardized communication Application Process Interface (API) inside the PC, interfacing between the Device Vendor's DTM and the host system's specific driver that handles pass-through communications from the PC down to the fieldbus interface card.

The host system vendor supplies a Communication DTM (comDTM) for each supported fieldbus protocol. This ensures that the details of the PC, network, interface cards, and pass-through protocols of the host system, are transparent to the device vendor's DTM. This correlates back to the internet analogy where: the web page is transparent to the PC it's running in, the brand of the network interface card in the PC, or whether communication is DSL or broadband cable.

FDT technology complements and expands existing device description languages. It does not replace but rather builds upon existing DDs.

In particular, FDT expands the capabilities of DD for complex devices. Device Description languages have limitations in the graphical representation of the device at the user interface and allow only a limited integration of special features. FDT/DTM removes these limitations.

Typical frame applications are

- Pactware from The PACTware Consortium e.V (freeware)
- FieldCare from Endress & Hauser
- Field Control from ABB

What is DTM?

DTM (Device Type Manager) is a standard way to provide all necessary data and functionality for a communication device, for example a PROFIBUS DP card. This technology is similar to the way Microsoft Windows supports printer drivers supplied by manufacturers and available to any Windows application, rather than requiring a custom printer driver for each specific application.

PROFIBUS comDTM, distributed by ProSoft Technology, is a DTM for PTQ and MVI series PDPMV1 modules and ProLinx PDPMV1 gateways. It allows configuration tools, instruments and communication devices on a field network to recognize and use the module's capabilities.
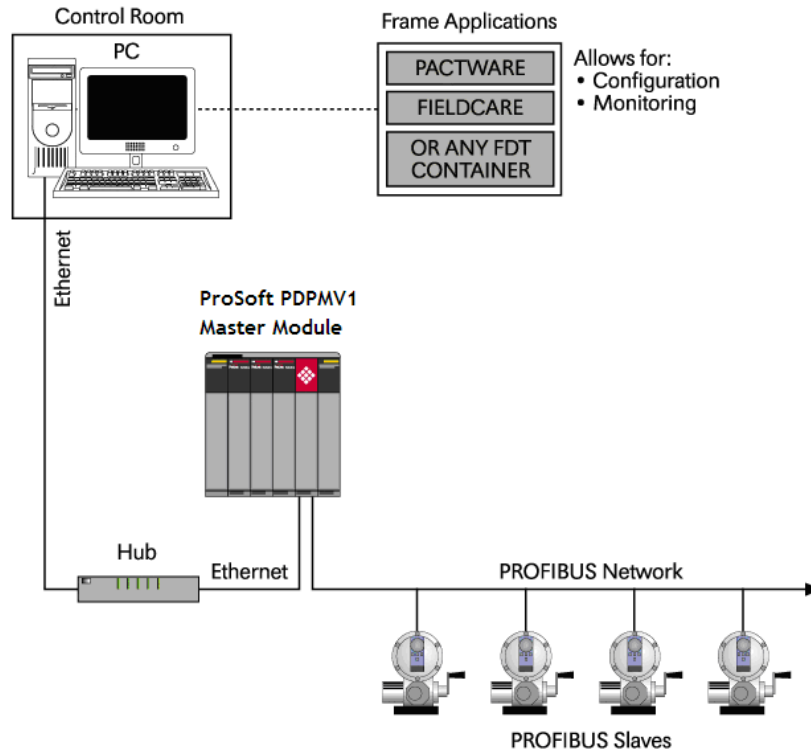
5.3.3 System Requirements

Confirm that your system meets the following hardware and software requirements before you start the installation.

Hardware Requirements (Recommended)

- Pentium 4 Processor rated for at least 2 GHz
- 450 MB hard drive space for DTM Libraries
- Video card capable of 1024 X 768 resolution at 64k colors
- Ethernet Network Interface Card (NIC)
- One of the following ProSoft Technology PROFIBUS DPV1 Master communication modules:
  - ProLinx PDPMV1 Gateway, Ethernet only, serial port not supported
  - PTQ-PDPMV1 Ethernet only, serial port not supported
  - MVI series PDPMV1 RS232 serial
  - MVI56-PDPMV1 RS232 serial, or with CIPconnect and 1756-ENBT module for Ethernet support
  - MVI69-PDPMV1 RS232 serial, or with CIPconnect for Ethernet support

Note for MVI series and PTQ users: The Ethernet connection implements a version of UDP protocol that dynamically allocates a random UDP port for every connection. This implementation limits the possibility of using most serial-to-Ethernet converters to access the MVI series PDPMV1 serial port, since many Ethernet-to-serial converters require the configuration of a fixed UDP port, which is not available for the current implementation.

Note for MVI56-PDPMV1 users: The Ethernet connection uses CIPconnect to communicate with the MVI56-PDPMV1 (firmware version 1.28.000 is CIPconnect enabled and is required for 1756-ENBT support). The 1756-ENBT card must be located in the same rack as the MVI56-PDPMV1. Bridging through multiple racks is not supported by the comDTM software.

Note for MVI69-PDPMV1 users: The Ethernet connection uses CIPconnect to communicate with the MVI69-PDPMV1 (requires firmware version 1.37.002 or later to support CIPconnect through the Ethernet port to L32E and L35E CompactLogix processors).

Note: This functionality requires comDTM version 1.0.1.5 with install version 1.01.0003. For information on how to check the comDTM version and install version, refer to Verifying the comDTM Version and comDTM Install Version (page 223).

Software Requirements (Minimum)
- Windows NT 4.0 Service Pack 6A, Windows 2000 SP3 or Windows XP Professional SP2, or better
- Microsoft Internet Explorer Version 6.0, or better
- FDT 1.2.1 compliant FDT frame application. Compatible applications include:
  - PACTware
  - FieldCare
  - M&M fdtCONTAINER

Some FDT Containers require the following components:
- Microsoft Management Console
- Adobe Acrobat Reader 5.0, or better

5.3.4 Installation

Important: You must have Administrator rights on your computer to install this application.
Important: Please open and read the *Readme* file before starting the installation. The *Readme* file is located in Utilities > comDTM > Readme on the *ProSoft Solutions Product CD-ROM*.

To install comDTM

1 Insert the *ProSoft Solutions Product CD-ROM* in an available CD-ROM drive in your computer. Wait for the startup screen to appear.
2 On the startup screen, click **PRODUCT DOCUMENTATION**. This action opens a *Windows Explorer* file tree window.
3 Double-click to open the **UTILITIES** folder, then navigate to **COMDTM** > **COMDTM INSTALL**.
4 Double-click the **SETUP.EXE** file. This action starts the installation wizard.
5 Follow the instructions on the installation wizard to install the program.
6 Click **FINISH** to complete the installation. If you are prompted to restart your computer, save your work in any applications that are running, close the applications, and allow the computer to restart.

Note: The CIPconnect® components are installed with comDTM version 1.0.1.5 with install version 1.01.0003 and later. For information on how to check the comDTM version and install version, refer to Verifying the comDTM Version and comDTM Install Version (page 223).
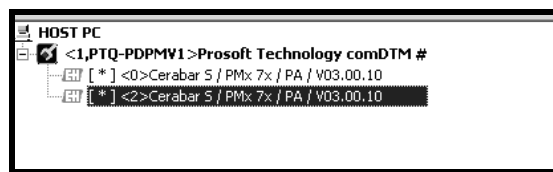
5.3.5 Quick Start

The following steps demonstrate how to start the FDT (Field Device Tool) program and configure the PROFIBUS comDTM.

Starting FDT

1 Start the FDT program and login as administrator. The following procedures use PACTware 3.0 software.
2 Click the **UPDATE DEVICE CATALOG** button. If PROFIBUS comDTM was installed successfully, it will appear in the *Device Catalog* window.

3 Select the **PROSOFT TECHNOLOGY COMDTM** entry in the device catalog, and then click **ADD**.
4 Repeat steps 1 through 3 to add any other manufacturer's device DTMs installed on your computer. Select the correct address for each device, and then click **OK** to proceed.

Note: You must use the same PROFIBUS device address that you used when setting up the device.
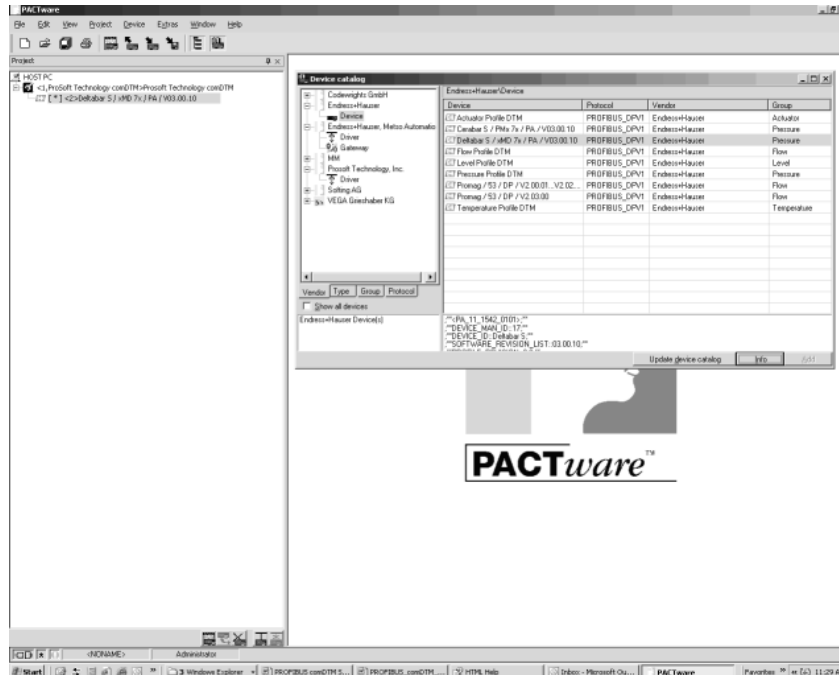
Connecting the comDTM to the Master to Establish Communication

Note: The features described in this section require the current version of PROFIBUS comDTM. You can always download the newest version from www.prosoft-technology.com.

1 From the Windows **START** button, navigate to **PROGRAMS** > **PROSOFT TECHNOLOGY**, and then choose **PROSOFT TRANSPORT PATH EDITOR**.

2 If you have not created a communication path, click the **ADD** button. If you have already created a path, skip to step 5.
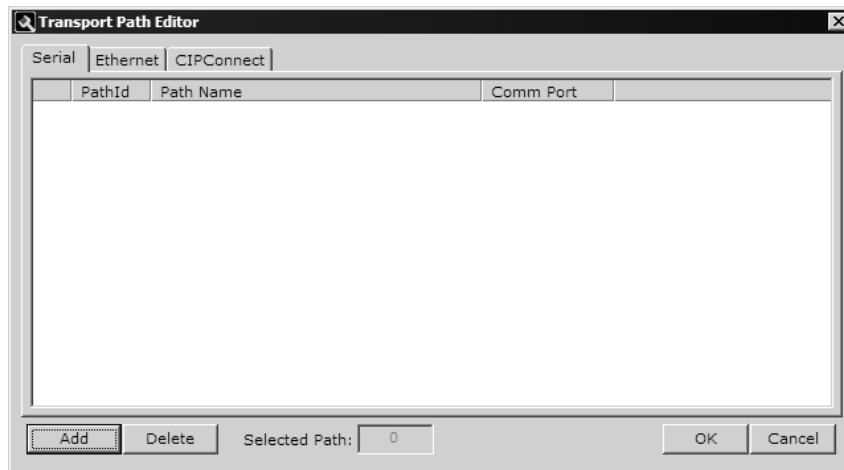
3 For PTQ-PDPMV1 and ProLinx-PDPMV1 users: Select the **ETHERNET** tab, and enter the IP Address of the module or gateway. Enter a descriptive path name.

Note: Do not include the underscore (_) for the path name.
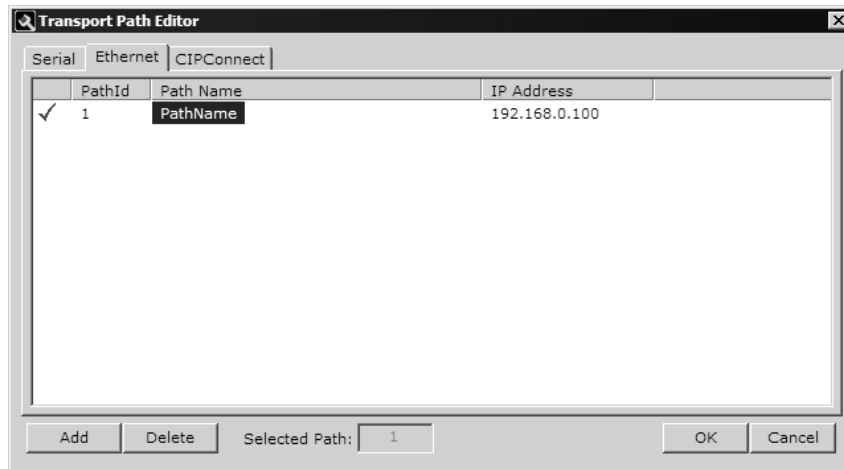
For MVI-PDPMV1 users: Select the **SERIAL** tab and enter the COM port number on your PC connected to the module. Enter a descriptive path name.
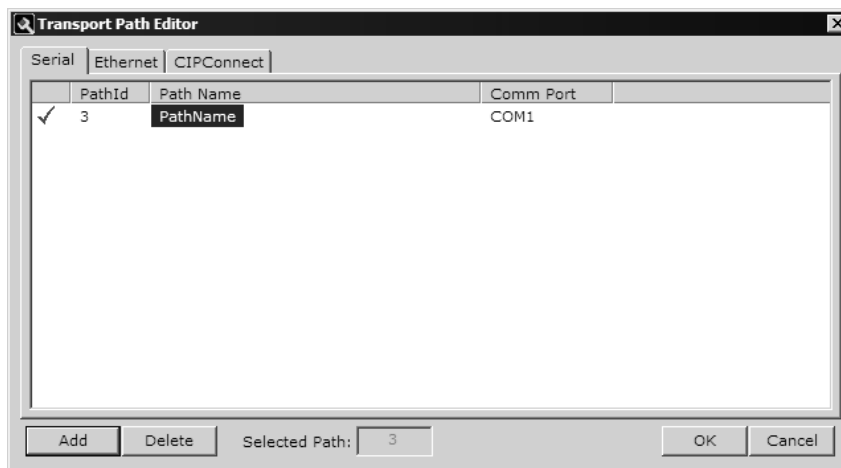
For MVI56-PDPMV1 and MVI69-PDPMV1 users with CIPconnect: Select the **CIPCONNECT** tab, and then click the **ADD** button.
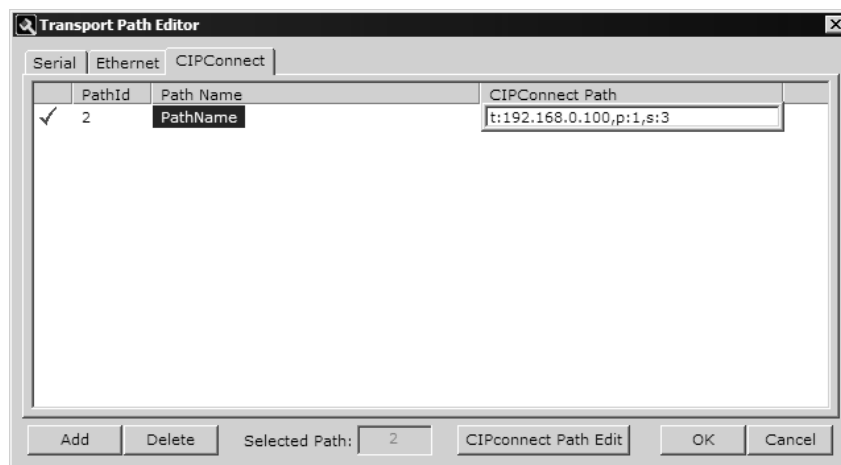
Click the **CIPCONNECT PATH EDIT** button to define the path for this application. The CIPconnect Path Editor allows the configuration of the path between your PC and the MVI56-PDPMV1 or MVI69-PDPMV1 module.

For the following example, the PC will be connected through Ethernet to a 1756-ENBT communication card (IP=192.168.0.100) and the MVI56-PDPMV1 card is located in slot 3 of the same rack.
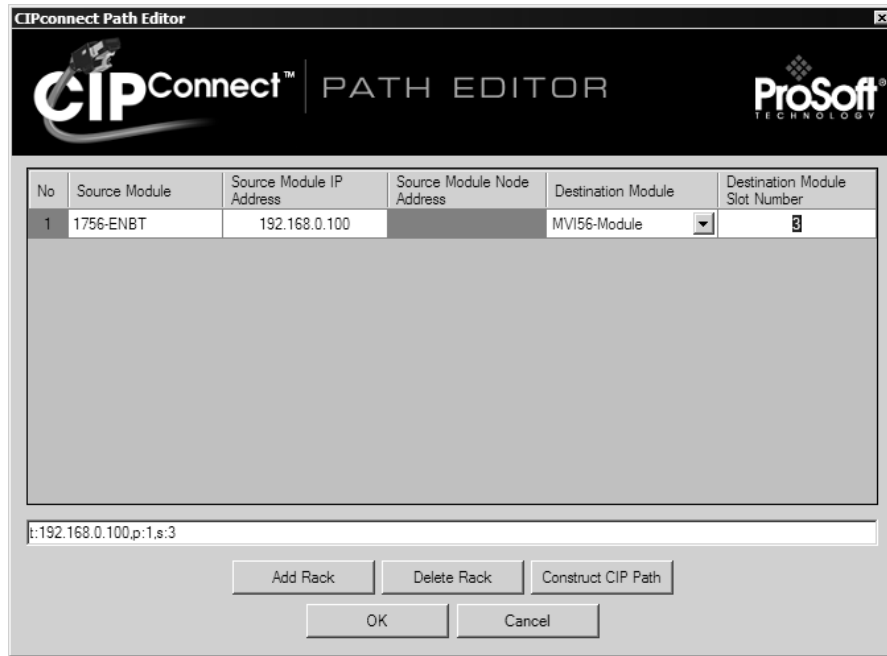
For more information, please refer to Using the CIPconnect Path Editor (page 222).

Note: CIPconnect is available for MVI56-PDPMV1 firmware version 1.28.000 (or later) and for MVI69-PDPMV1 firmware version 1.37.002 (or later). This functionality requires comDTM version 1.0.1.5 with install version 1.01.0003 (or later). For information on how to check the comDTM version and install version, refer to Verifying the comDTM Version and comDTM Install Version (page 223).

4 When you have configured the communication path, click the **OK** button to confirm. The communication path will be displayed at the top grid panel as shown in the following illustration.
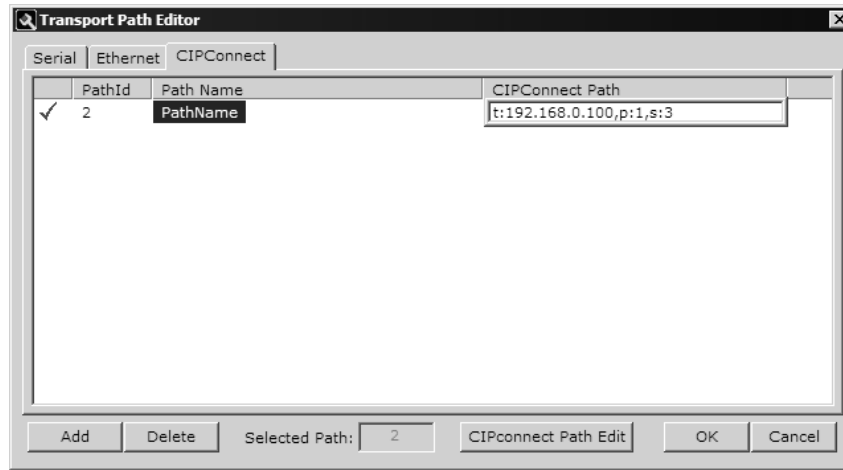
5 Select the path and click the **OK** button to exit the *Transport Path Editor* window.
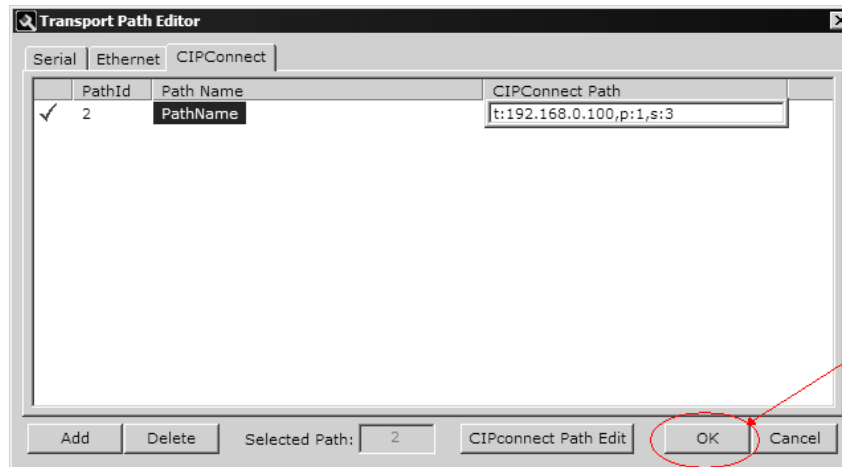
6 Select the **COMDTM** icon and click the right mouse button to open a shortcut menu. On the shortcut menu, choose **CONNECT**. If the connection is successful, the icon will be highlighted, as shown in the following illustration.

When the comDTM is connected with the Master, PACTware indicates the
connection Master by displaying a green plug in the status bar.
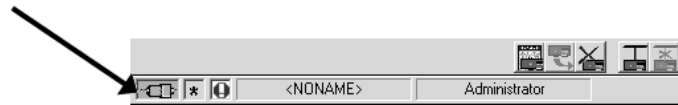
This completes the installation and Quick Start Guide for the ProSoft Technology
PROFIBUS comDTM. Refer to the online help and documentation additional
information on each DTM component you have installed and configured.

The comDTM provides a *Guided Tour* section in the online help that explains the
basic features and operation of the program. To open the online help, click the
right mouse button on **PROSOFT TECHNOLOGY COMDTM**, and choose **ADDITIONAL
FUNCTIONS** > **ONLINE HELP** from the shortcut menu.

Click the **GUIDED TOUR** icon. Use the navigation buttons on each help page to
view the help topics.
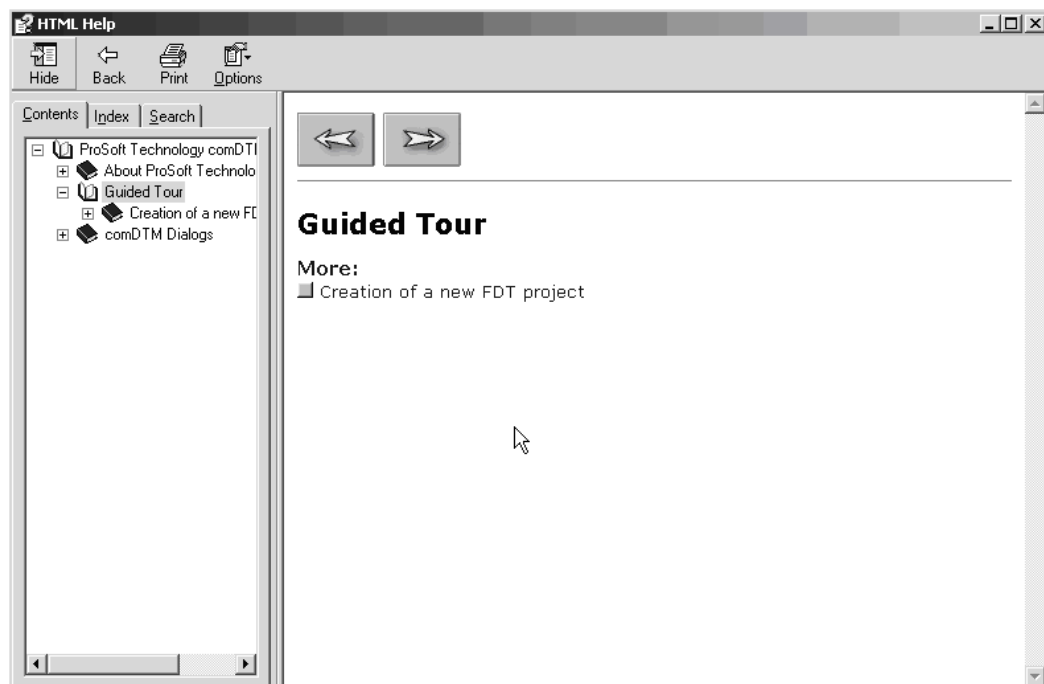
Refer to the documentation and online help for your FDT frame program for
specific FDT frame instructions.

Using the CIPconnect Path Editor

Example
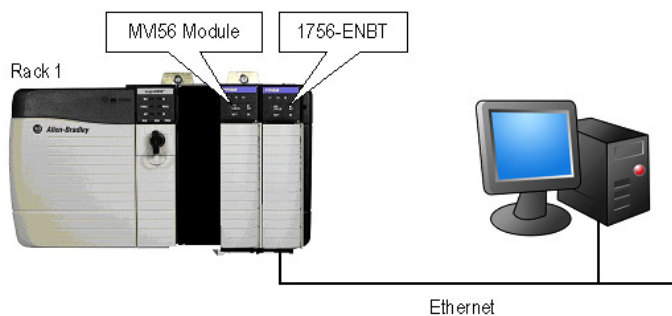
Rack 1

Slot	Module	Network Address
0	ControlLogix Processor	-
1	Any	-
2	MVI56-PDPMV1	-
3	1756-ENBT	IP=105.102.0.12

1 In the *Transport Path Editor*, click **ADD**, then click **CIPCONNECT PATH EDIT.**
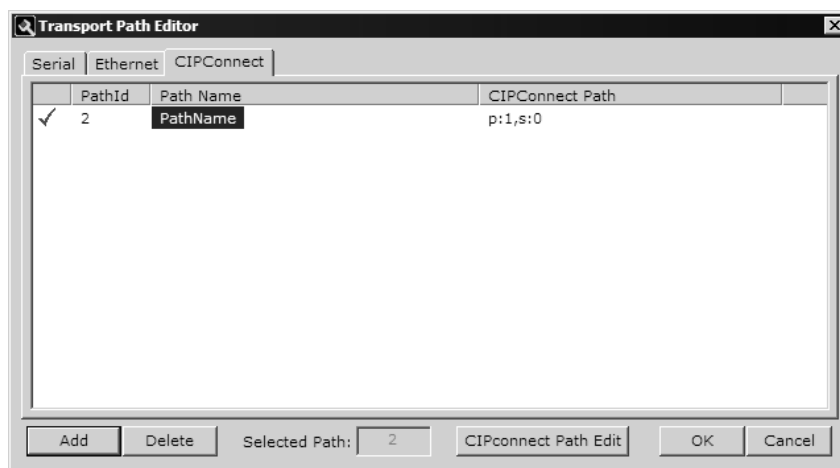
2 Configure the path as shown in the following illustration, and click **CONSTRUCT CIP PATH** to build the path in text format.
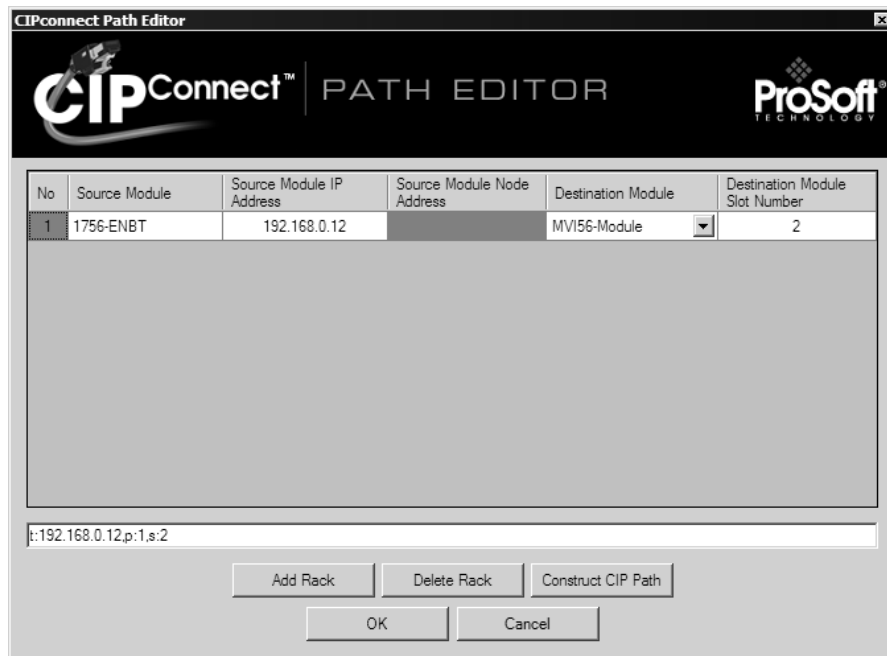
3 Check the new path in the *Transport Path Editor*.
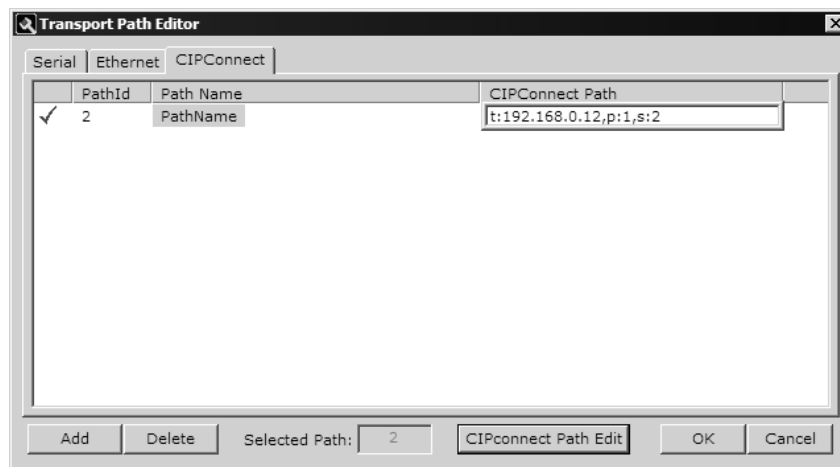
5.3.6 Verifying the comDTM Version and comDTM Install Version

Introduction

There are two versions associated to the comDTM – the comDTM version and the comDTM install version. Starting with comDTM version 1.0.1.5, each upgrade will indicate the same comDTM version but a different comDTM install version.

This section describes how to check the comDTM version and comDTM install version.

Checking the comDTM Version

Refer to the *Version* column indicated when you add the comDTM to the DTM Container project.
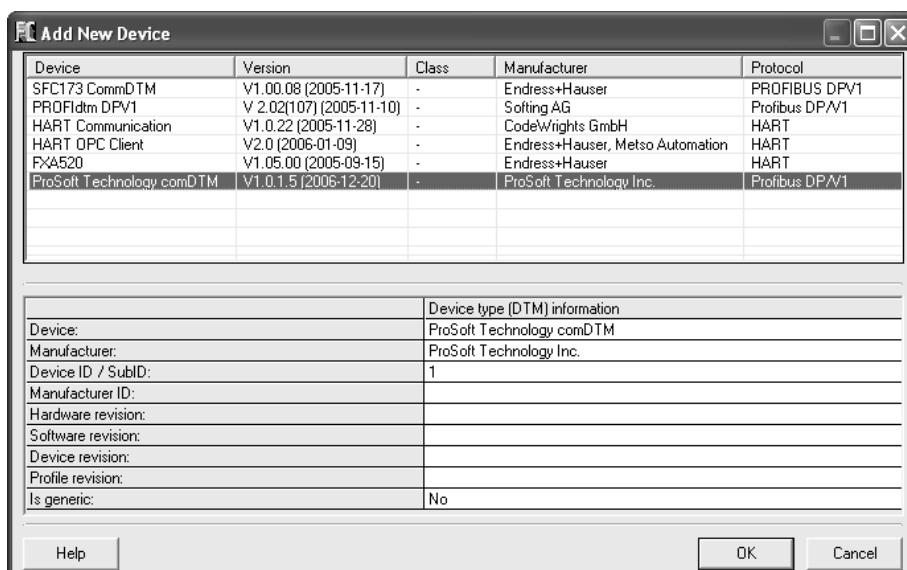
Checking the comDTM Install Version

1 Click the **START** menu and then choose **CONTROL PANEL**.
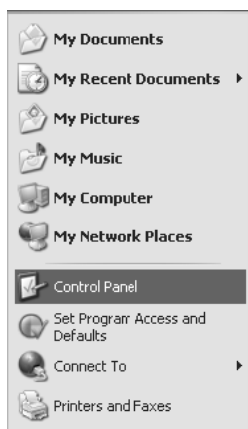
      2 In the list of **CONTROL PANEL** applets, select **ADD OR REMOVE PROGRAMS**.
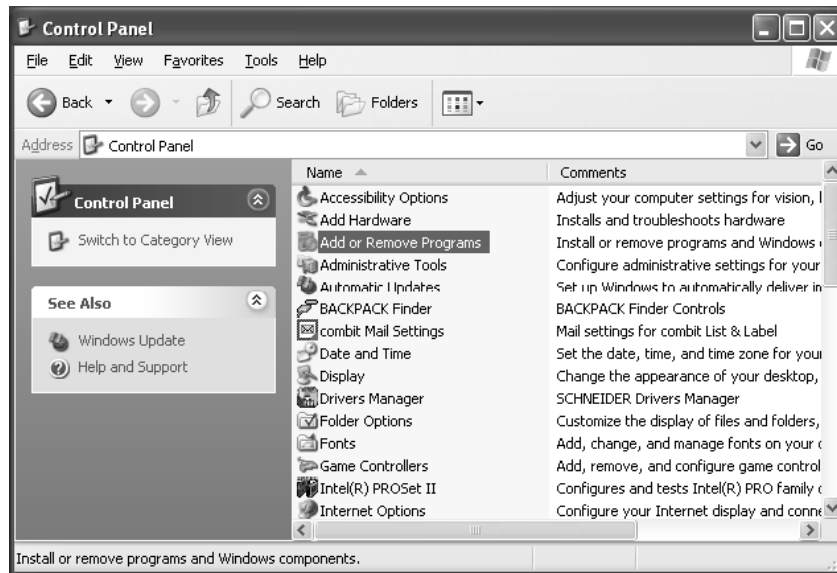
      3 Select **PROSOFT TECHNOLOGY COM-DTM**, and then click on the link **CLICK HERE FOR SUPPORT INFORMATION**.
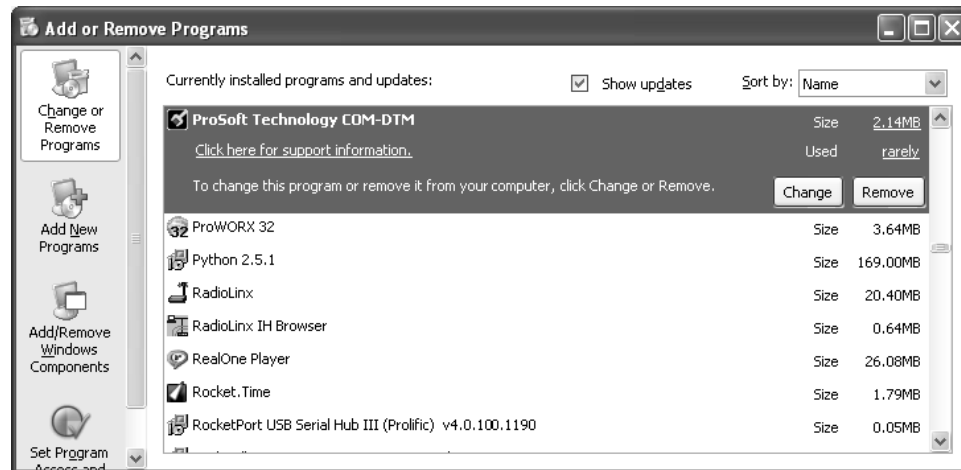
4 You will see the comDTM Install Version in the *Version* field, as shown in the following illustration (1.01.0003 for this example).
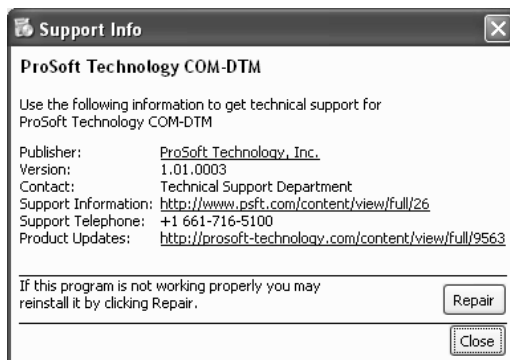
Checking the Install Version for Vista

1 Select **CONTROL PANEL**.
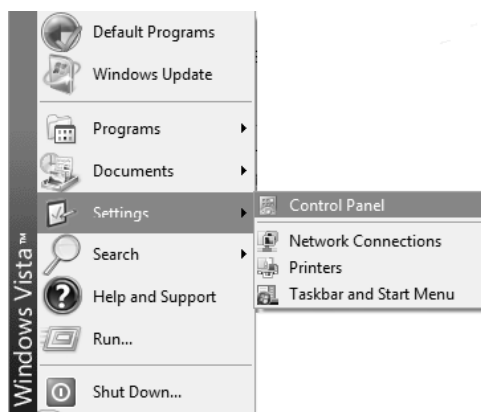
2 Select **UNINSTALL PROGRAMS**.
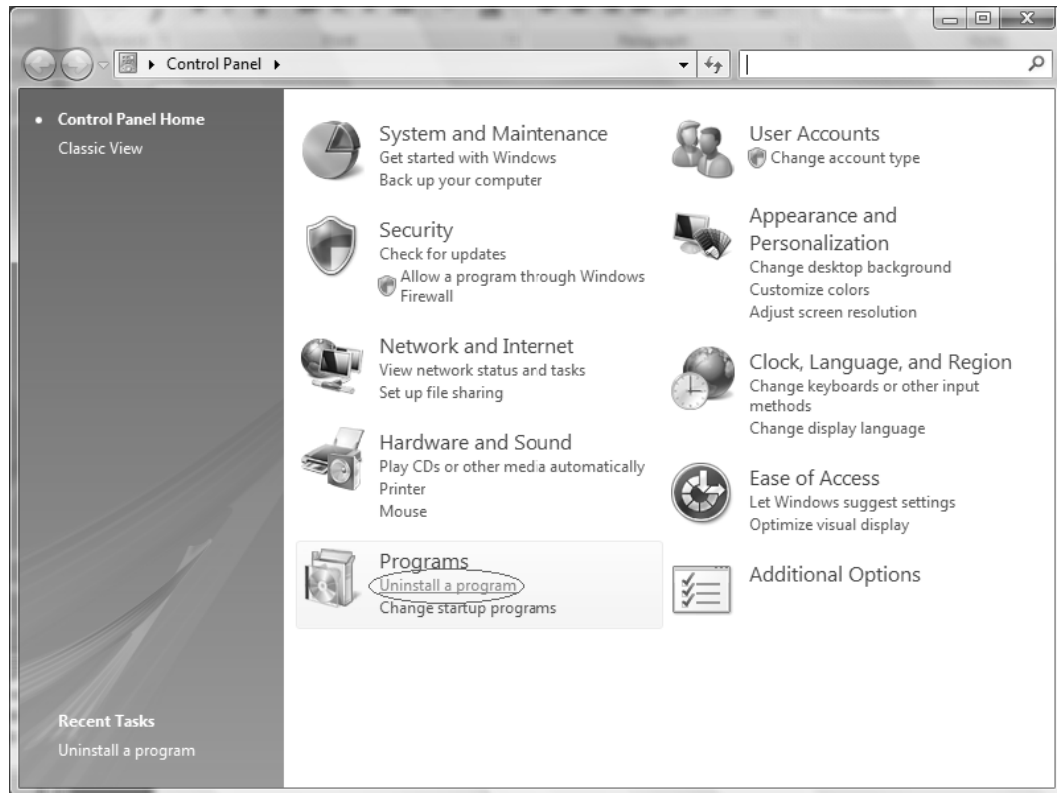
3 Select Prosoft Technology COM-DTM (click once)
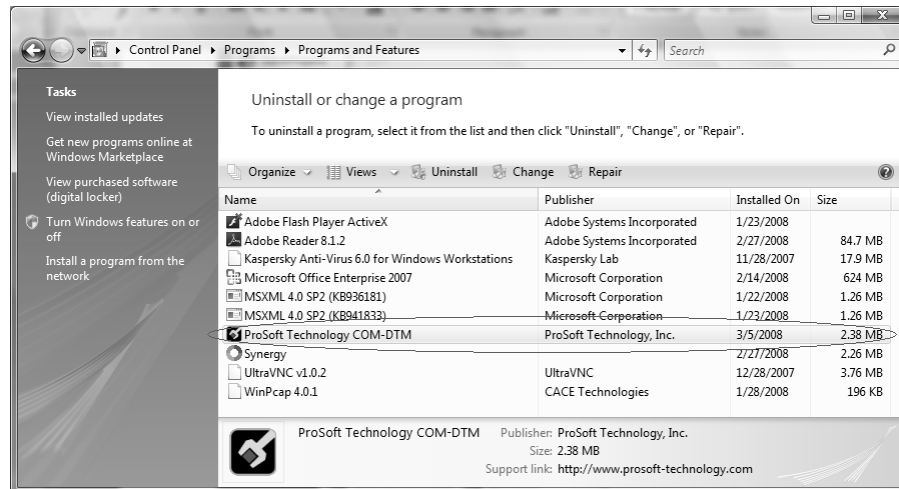
4 Click the **ORGANIZE** tab and select **LAYOUT** > **DETAILS PANE**.
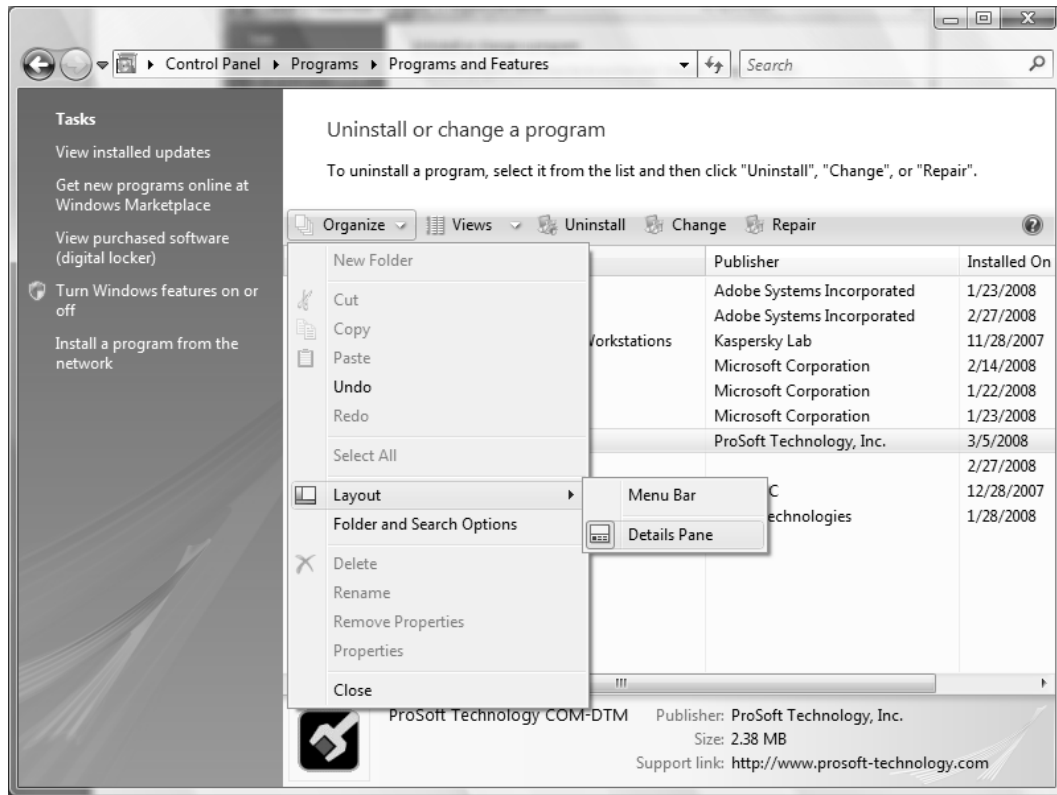
5 Check the Install Version at the bottom right portion of the window.
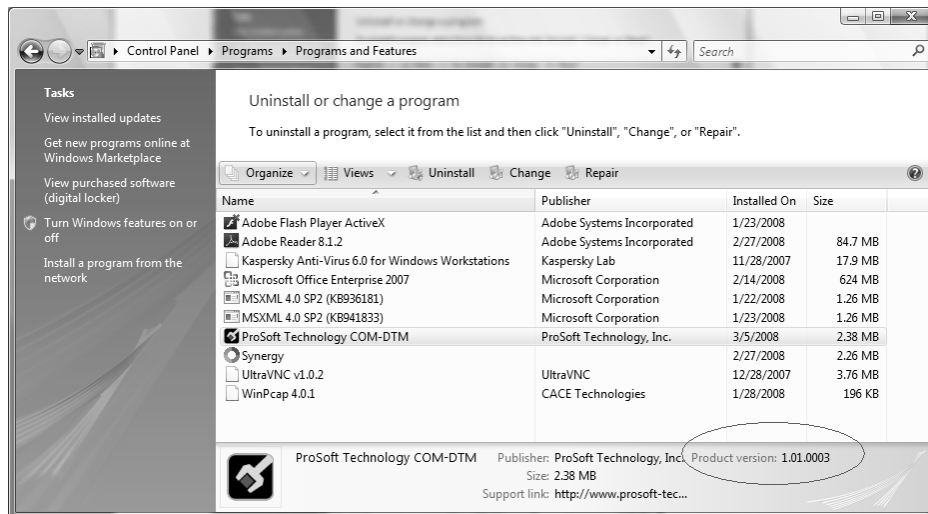
5.4   Module Functional Overview

The MVI56 module communicates with the processor over the backplane using only the following two blocks of data:

- MVI56 Input Data block
- MVI56 Output Data block

This section of the Application Reference Guide describes the data structures and transfer mechanisms used to transfer data between the MVI56-PDPMV1 module and the ControlLogix processor.

The following illustration shows the Input/Output Data block flow between the ControlLogix processor and the MVI56-PDPMV1 module.
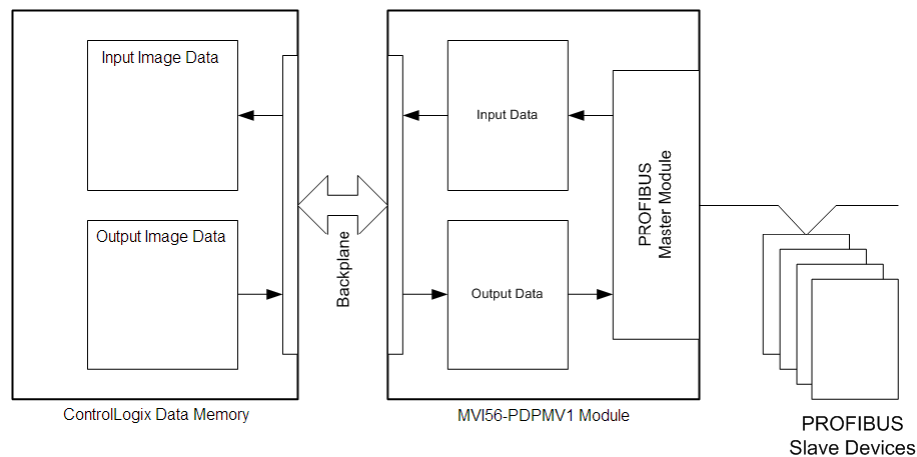
These two data blocks (Input Data and Output Data) consist of a data structure that provides for the movement of:

- Input Data image from PROFIBUS slave devices
- Output Data image for writing to PROFIBUS slave devices
- Module configuration and status (from module to processor)
- PROFIBUS Messaging Mailbox commands (from processor to module)
- PROFIBUS Messaging Mailbox responses (from module to processor)

In LEGACY mode, the module reports this information through regular I/O blocks. In FLEX mode, this data is exchanged through MSG instructions.

The transfer method depends on the configured mode:

- LEGACY (page 230) (firmware version 0.30 or version 1.21 or later and Use Legacy Mode = Yes)
- FLEX (page 236) (firmware version 1.21 or later and Use Legacy Mode = No)

5.4.1 Legacy Mode Input and Output Data Blocks

The following illustration shows how the transfer of user data, mailbox block, and status data is performed between the MVI56-PDPMV1 and the ControlLogix processor.
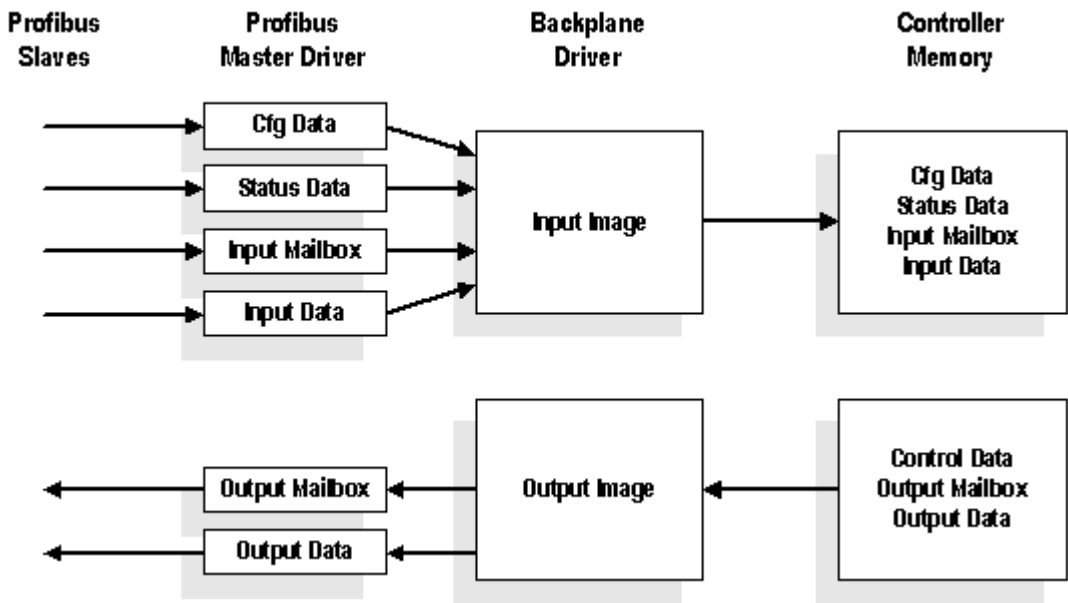
The blocks that are transferred between the module and the processor can perform different tasks, according to the Block ID numbers, as listed in the following table.

Block ID Numbers for Transfer

Block #s	Type	Description
-1 or 0	Status	Block containing status data (refer to Status Objects)
1 to 4	Input Data	Input data from PROFIBUS network with each block containing up to 200 words of data (refer to Input Data Block)
1 to 4	Output Data	Output data for PROFIBUS network with each block containing up to 200 words of data (refer to Output Data Block)
100 to 109	Mailbox	These block numbers send or receive mailbox messages (refer to PDPMV1_Mailbox data type)
200 to 209	Alarms	These block numbers send alarm messages from the PROFIBUS network to the processor (refer to PDPMV1_Alarm).

The size of the input and output areas should be set to the smallest possible values to maximize data throughput.

Handshaking is required between the in and out messages used by the module and the controller.

Normal Backplane Operation

The MVI56-PDPMV1 module's application code initiates the data transfers at the end of every ControlLogix PLC ladder scan. As such, the MVI56-PDPMV1 module is able to actively read and write the PROFIBUS Cyclic Input/Output data blocks in the appropriate locations.

Input Data Block (Block ID = 1 to 4)

This block contains the previous input data received from the slaves.

Input Data Block Format (Input Image) (Local:1:I.Data Controller Tag)

You can find the contents of the Input Data Block (Input Image) in the *Local:1:I.Data* user controller tags in RSLogix.

Word Offset			
Start	End	Name	Description
0	199	PROFIBUS Input Data	Data received from the PROFIBUS slave devices on the network.
			Total size of the PROFIBUS Input Data block is a function of the PROFIBUS network configuration.
			Block ID 1 = Slave bytes 0 to 399
			Block ID 2 = Slave bytes 400 to 799
			Block ID 3 = Slave bytes 800 to 1199
			Block ID 4 = Slave bytes 1200 to 1535
200	200	Last out mailbox message ID processed from output image	The module confirms the receipt of a mailbox by copying its ID code (Message ID) to this register.
201	201	Current in mailbox control index	Incremented after the module has transferred a new mailbox response to the processor.
202	202	Current alarm control index	Incremented after the module has transferred a new alarm to the processor.
203	203		Reserved for future use
204	205	PROFIBUS CRC32 value	CRC32 checksum for PROFIBUS Master configuration downloaded from configuration utility
206	207	Module CRC32 value	MVI56-PDPMV1 module configuration checksum for module configuration downloaded from configuration utility
208	247		Reserved for future use
248	248	Write Block ID	Write block ID of Output image to receive (1 to 4 for data, 0 or -1 if no data)
249	249	Read Block ID	Read Block ID number -1 to 4

Input Block Status Data Details (Block ID = 0 to -1)

The MVI56-PDPMV1 module periodically transfers a status block to the ControlLogix containing relevant information about the module operation (Block IDs 0 to -1).

The types of data returned in the PROFIBUS Input Status area include:

- Module configuration data values
- MVI56 software revision level
- Key PROFIBUS configuration values
- PROFIBUS Master hardware/software revision level
- MVI56 module statistics
- Mailbox messaging control and status

You can find the contents of the Status Data block in the *MVI56PDPMV1.Status* user controller tag structure in RSLogix. These tags reference the *PDPMV1_Status* user-defined data type.

The following table is received periodically from the MVI56 module.

Byte Offset			
Start	End	Name	Description
0	9	Module ID string	Unique module 10-byte pattern as text "MVI5PDPMV1"
10	11	Reserved	Reserved for future use
12	13	PROFIBUS Input Data size	The number of words of PROFIBUS input data to transfer from the PROFIBUS Master to the processor within the Input Data blocks. Value is selected during user configuration
14	15	PROFIBUS Output Data size	The number of words to transfer from the processor in the PROFIBUS output space during the output data transfer. Value is selected during user configuration
16	17	Reserved	Reserved for future use
18	19	Reserved	Reserved for future use
20	21	Reserved	Reserved for future use
22	22	Input Data Byte Swap	User configured flag to indicate if input data is swapped before being placed in the input image for the controller. 0=No swapping Not 0=Swap bytes
23	23	Output Data Byte Swap	User configured flag to indicate if output data is swapped after being received from the output image of the controller. 0=No swapping Not 0=Swap bytes
24	24	Module software major version number.	MVI56 module application software version number
25	25	Module software minor version number	
26	41	PROFIBUS Slave Configured List	This is a 16-byte array with bit fields where one bit is assigned to each slave station address. The associate bit is set if the slave is present in the database. For the bit/slave relationship, refer to the Slave List Structure below.
42	57	PROFIBUS Data Transfer Status	This is a 16-byte array with bit fields where one bit is assigned to each slave station address. The associated bit is set if the slave has reached or retained the "Data Exchange" state at least once during the last three data cycles. For the bit/slave relationship, refer to the Slave List Structure below.

For rows 24/25, the Description column contains a diagram:

Bits

16	8	7	0
High Byte		Low Byte	

Byte Offset

Start	End	Name	Description
58	73	PROFIBUS Slave Diagnostic Status	This is a 16-byte array with bit fields where one bit is assigned to each slave station address. When a slave leaves the "Deactive" state for the first time, the associated bit is set. This bit is then cleared when the slave enters "Data Exchange" state. If a slave indicates "Extended Diagnostics" when it is in "Data Exchange" state, the associated bit is set. For the bit/slave relationship, refer to the Slave List Structure below.

Slave List Structure

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	Slave 7	Slave 6	Slave 5	Slave 4	Slave 3	Slave 2	Slave 1	Slave 0
Byte 1	Slave 15	Slave 14	Slave 13	Slave 12	Slave 11	Slave 10	Slave 9	Slave 8
Byte 2	Slave 23	Slave 22	Slave 21	Slave 20	Slave 19	Slave 18	Slave 17	Slave 16
Byte 3	Slave 31	Slave 30	Slave 29	Slave 28	Slave 27	Slave 26	Slave 25	Slave 24
Byte 4	Slave 39	Slave 38	Slave 37	Slave 36	Slave 35	Slave 34	Slave 33	Slave 32
...	...	...	...	...	...	...	...	...
Byte 15	Slave 127	Slave 126	Slave 125	Slave 124	Slave 123	Slave 122	Slave 121	Slave 120

Byte Offset

Start	End	Name	Description
74	74		PROFIBUS pad byte to make data word aligned (not used)
75	75	PROFIBUS Master Operating State	PROFIBUS Master operating state 0x0000=Offline 0x4000=Stop 0x8000=Clear 0xC000=Operate
76	77	PROFIBUS Ident Number	PROFIBUS Master PNO Ident number. Bytes will be swapped
78	81	PROFIBUS Master Serial Number	Unique 32-bit serial number for the PROFIBUS Master
82	83	PROFIBUS Software Version	Software version number for the PROFIBUS Master software Example for Version 1.40: High Byte - 0x40 Low Byte - 0x01

Byte Offset			
Start	End	Name	Description
84	85	PROFIBUS Master Module Status	Represents the PROFIBUS Master module's operating status
			Bit 2=Application Status
			0 - Application Stopped
			1 - Application Running
			Bit 8=Data exchange (FBRS)
			0 - There is no data exchange with any of the assigned slaves
			1 - There is Data Exchange with at least one of the assigned slaves
			Bit 9=Slave input frozen/cleared (FBFC)
			0 - A slave's inputs in the IN area are cleared if a slave is not in Data Exchange
			1 - A slave's inputs in the IN area are frozen if a slave is not in Data Exchange
			Bit 12=Reset (RDR)
			0 - No action
			1 - A reset is requested by the PROFIBUS Master module because a new database has been downloaded
86	89	PROFIBUS configuration Checksum	CRC32 checksum for PROFIBUS Master configuration downloaded from configuration utility
90	93	MVI56 Module configuration checksum	MVI56-PDPMV1 for module configuration downloaded from configuration utility
94	95	Application program scan counter	MVI56-PDPMV1 module program scan counter. Can be used to gauge application code scan time performance
96	97	Module PROFIBUS output image data update counter	Counter representing the number of times the output data image is transferred to the module's internal Master
98	99	Module PROFIBUS input image data update counter	Counter representing the number of times the input data image is transferred from the module's internal Master
100	101	Module out mailbox counter	Module out mailbox counter
102	103	Module in mailbox counter	Module in mailbox counter
104	105	Module alarm IND receive counter	Number of spontaneous alarm messages received from slave
106	107	Module alarm CON receive counter	Number of confirmation messages received from slaves indicating that the slave received the confirmation message from the MVI56-PDPMV1 module
108	109	Reserved	
110	111	Reserved	
112	113	Module backplane read count	Rollover counter of the number of MVI56-PDPMV1 to processor backplane read data transfers
114	115	Module backplane write count	Rollover counter of the number of MVI56-PDPMV1 to processor backplane write data transfers
116	117	Module backplane error count	Rollover counter of the number of MVI56-PDPMV1 to processor backplane data transfers that have failed

Byte Offset

Start	End	Name	Description
118	119	File Error Word	This is bit-mapped value that indicates which files are not present. Bit 0: Problem with PDPMV1.CFG file Bit 1: Not used Bit 2: Problem with PDPMV1.DDB file Bit 3: Problem with PDPMV1.XML file Bits 4 to 15 not used. If this word has a value other than 0, the CFG ERR LED on the module will be illuminated.
120	145	Reserved	Reserved for future use
146	147	In mailbox queue count	Number of message in the in mailbox queue
148	149	Out mailbox queue count	Number of message in the out mailbox queue
150	151	Alarm queue count	Number of message in the alarm queue
152	153	Last out mailbox message ID processed from output image	Last out mailbox message ID processed from output image
154	155	Current in mailbox control index	Current in mailbox control index
156	157	Current alarm control index	Current alarm control index
158	496		Reserved for future use
496	497		Index of Output image to receive (1 to 4 for data, 0 or -1 if no data)
498	499		Block index of -1 or 0 for status block

Output Data Block (Block ID = 1 to 4)

This block contains the PROFIBUS output data to be sent from the processor to the module.

Output Data Block Format (Output Image)

You can find the contents of the Output Data Block (Output Image) in the *MVI56PDPMV1.Output* user controller tag structure in RSLogix.

Word Offset

Start	End	Name	Description
0	0	Block ID	Block number 1 to 4
1	200	PROFIBUS Output Data	PROFIBUS Output Data: Data going to the PROFIBUS Network.
201	201	Reserved	Reserved for future use
202	202	Reserved	Reserved for future use
203	203	Reserved	Reserved for future use

Word Offset			
Start	**End**	**Name**	**Description**
204	205	PROFIBUS CRC32 value	Computed for PROFIBUS Config
206	207	Module CRC32 value	Computed for module data
			When the module first starts up or recognizes an initialization of the processor, it will compare the values of the two CRC's in the input and output images. If either one of the CRCs do not match, the module will be placed in Stop mode. If each set matches, the module will be placed in Operate mode.
208	247	Reserved	Reserved for future use

5.4.2 Flex Mode Input and Output Data Blocks

The following illustration shows how the transfer of data, mailbox block, and status data is performed between the MVI56-PDPMV1 and the ControlLogix processor.
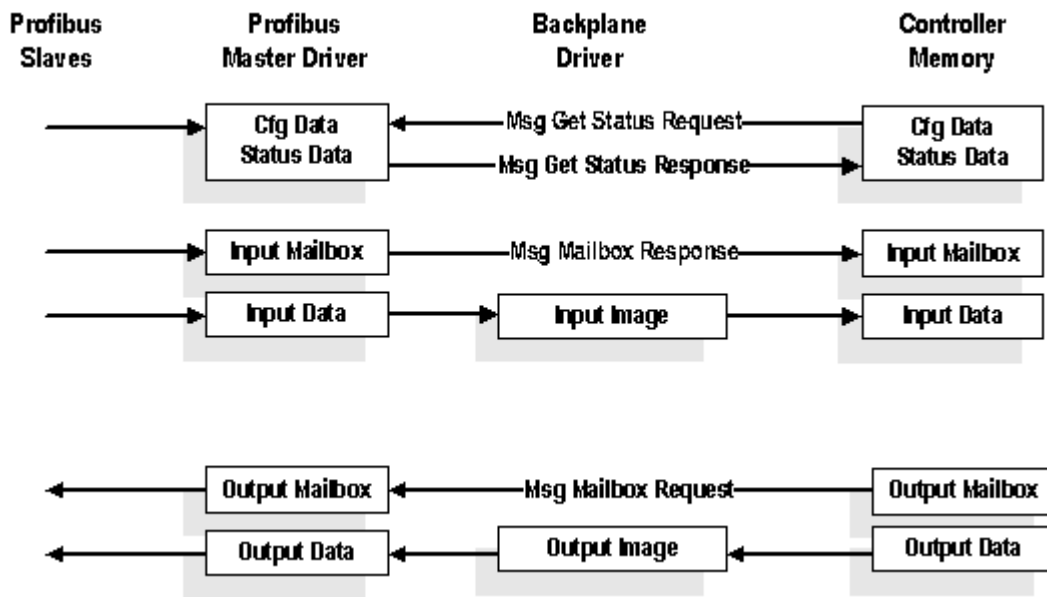
The PROFIBUS I/O data is transferred through the backplane I/O images. The status and mailbox data is transferred through MSG instructions.

Block ID Numbers for Transfer

Block	Type	Description
0	Input Data	This block is generated only if the module has a single input block (the configured PROFIBUS input data fits into one input backplane block). It allows the switching between two blocks with different block IDs but with same content. So in fact block 0 and block 1 will transfer the same PROFIBUS input data.
1 to *n*	Input Data	Input data from PROFIBUS network with each block containing up to 242 words of data (refer to Input Data Block (page 237))
1 to *n*	Output Data	Output data for PROFIBUS network with each block containing up to 243 words of data (refer to Output Data Block (page 242))

Note: The maximum block count *n* will depend on the configured PROFIBUS I/O size and the backplane block size for each application. For example, if the PROFIBUS input = 100 and each backplane block will transfer 20 words of PROFIBUS data, then *n*=5.

Normal Operation

The MVI56-PDPMV1 module's application code initiates the data transfers at the end of every ControlLogix PLC ladder scan. As such, the MVI56-PDPMV1 module is able to actively read and write the PROFIBUS Cyclic Input/Output data blocks in the appropriate locations.

Input Initialization Block (Block ID = 9990)

Block 9990 is the first block transferred by the module to the processor after one of the following events:

- Module power up
- Module configuration through ProSoft Configuration Builder
- Module reboot
- ControlLogix mode switch from PROG to RUN

It contains general module configuration data, which allows the ladder logic to correctly handle the handshaking between module and processor for proper backplane communication.

Input Data Block Format (Input Image) (Local:1:I.Data Controller Tag)

You can find the contents of the Input Data Block (Input Image) in the *Local:1:I.Data* user controller tags in RSLogix.

Word Offset			
Start	**End**	**Name**	**Description**
0	0	Block Write ID	Block ID of the next write block. It will contain a value of 9990 because the module expects the next block from the processor to have the same block ID.

Word Offset			
Start	End	Name	Description
1	1	Read Block Size (MVI56PDPMV1.Util.ReadBlockSize)	Read block size (backplane connection input length) as configured by the user through RSLogix in the I/O Configuration section. The size indicates the number of words to transfer from the module to the processor at each time.
2	2	Write Block Size (MVI56PDPMV1.Util.WriteBlockSize)	Write block size (backplane connection output length) as configured by the user through RSLogix in the I/O Configuration section. The size indicates the total number of words to transfer from the processor to the module at each time.
3	3	PROFIBUS Total Input Data Size (MVI56PDPMV1.Config.PROFIBUS.InputDataSize)	PROFIBUS Input size configured in PCB through the Input Data Size parameter.
4	4	PROFIBUS Total Output Data Size (MVI56PDPMV1.Config.PROFIBUS.OutputDataSize)	PROFIBUS Output size configured in PCB through the Output Data Size parameter.
5	5	Read Block Count (MVI56PDPMV1.Util.ReadBlockCount)	Number of read blocks to transfer from the module to the processor in order to transfer the configured PROFIBUS input data.
6	6	Write Block Count (MVI56PDPMV1.Util.WriteBlockCount)	Number of write blocks to transfer from the processor to the module in order to transfer the configured PROFIBUS output data.
7	7	PROFIBUS Input Data Word Count Per Read Block (MVI56PDPMV1.Util.InputWordsPerReadBlock)	Maximum number of PROFIBUS input data words that fit into one read block.
8	8	PROFIBUS Input Data Word Count Per Write Block (MVI56PDPMV1.Util.OutputWordsPerWriteBlock)	Maximum number of PROFIBUS output data words that fits into one write block.
9	9	PROFIBUS Input Data Word Count In Last Read Block (MVI56PDPMV1.Util.InputWordsInLastReadBlock)	Number of PROFIBUS input data words in the last read block. The last read block is the one with highest block ID.
10	10	PROFIBUS Output Data Words In Last Write Block (MVI56PDPMV1.Util.OutputWordsInLastWriteBlock)	Number of PROFIBUS output data words in the last write block. The last write block is the one with highest block ID.
11	11	Read Block ID	9990

Input Data Block (Block ID = 1 to n)

This block contains the previous input data received from the slaves.

Input Data Block Format (Input Image) (Local:1:I.Data Controller Tag)

You can find the contents of the Input Data Block (Input Image) in the *Local:1:I.Data* user controller tags in RSLogix.

Word Offset			
Start	**End**	**Name**	**Description**
0	0	Block Write ID	Block ID of the next write block to transfer from the processor to the module
1	2	PROFIBUS CRC32 value	CRC32 checksum for PROFIBUS Master configuration downloaded from configuration utility
3	4	Module CRC32 value	MVI56-PDPMV1 module configuration checksum for module configuration downloaded from configuration utility
5	5	CIP Mailbox MSG Index	Latches bit 15 if new mailbox response is available
6	6	CIP Alarm MSG Index	Latches bit 15 if new alarm response is available
7	*n*-1	PROFIBUS Input Data	PROFIBUS input data transferred from the PROFIBUS slaves
n	*n*	Block Read ID	Block ID of this read block

MSG Status Data Details

The ControlLogix processor may periodically send a MSG request to retrieve the status data from the MVI56-PDPMV1 module.

The types of data returned in the PROFIBUS Input Status area include:

- Module configuration data values
- MVI56 software revision level
- Key PROFIBUS configuration values
- PROFIBUS Master hardware/software revision level
- MVI56 module statistics
- Mailbox messaging control and status.

You can find the contents of the Status Data block in the *MVI56PDPMV1.Status* user controller tag structure in RSLogix. These tags reference the PDPMV1_Status user-defined data type.

The following table shows the data structure of the GetStatus mailbox response.

Byte Offset			
Start	**End**	**Name**	**Description**
0	9	Module ID string	Unique module 10-byte pattern as text "MVI5PDPMV1"
10	11	Reserved	Reserved for future use
12	13	PROFIBUS Input Data size	The number of words of PROFIBUS input data to transfer from the PROFIBUS Master to the processor within the Input Data blocks. Value is selected during user configuration

Byte Offset

Start	End	Name	Description
14	15	PROFIBUS Output Data size	The number of words to transfer from the processor in the PROFIBUS output space during the output data transfer. Value is selected during user configuration
16	17	Reserved	Reserved for future use
18	19	Reserved	Reserved for future use
20	21	Reserved	Reserved for future use
22	22	Input Data Byte Swap	User-configured flag to indicate if input data is swapped before being placed in the input image for the controller. 0=No swapping Not 0=Swap bytes
23	23	Output Data Byte Swap	User-configured flag to indicate if output data is swapped after being received from the output image of the controller. 0=No swapping Not 0=Swap bytes
24	24	Module software major version number.	MVI56 module application software version number
25	25	Module software minor version number	
26	41	PROFIBUS Slave Configured List	This is a 16-byte array with bit fields where one bit is assigned to each slave station address. The associate bit is set if the slave is present in the database. For the bit/slave relationship, refer to the Slave List Structure below.
42	57	PROFIBUS Data Transfer Status	This is a 16-byte array with bit fields where one bit is assigned to each slave station address. The associated bit is set if the slave has reached or retained the "Data Exchange" state at least once during the last three data cycles. For the bit/slave relationship, refer to the Slave List Structure below.
58	73	PROFIBUS Slave Diagnostic Status	This is a 16-byte array with bit fields where one bit is assigned to each slave station address. When a slave leaves the "Deactive" state for the first time, the associated bit is set. This bit is then cleared when the slave enters "Data Exchange" state. If a slave indicates "Extended Diagnostics" when it is in "Data Exchange" state, the associated bit is set. For the bit/slave relationship, refer to the Slave List Structure below.

For rows 24–25, the Description column contains a diagram labeled "Bits":

Bits			
16	8	7	0
High Byte		Low Byte	

Slave List Structure

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	Slave 7	Slave 6	Slave 5	Slave 4	Slave 3	Slave 2	Slave 1	Slave 0
Byte 1	Slave 15	Slave 14	Slave 13	Slave 12	Slave 11	Slave 10	Slave 9	Slave 8
Byte 2	Slave 23	Slave 22	Slave 21	Slave 20	Slave 19	Slave 18	Slave 17	Slave 16
Byte 3	Slave 31	Slave 30	Slave 29	Slave 28	Slave 27	Slave 26	Slave 25	Slave 24
Byte 4	Slave 39	Slave 38	Slave 37	Slave 36	Slave 35	Slave 34	Slave 33	Slave 32
...	...	...	...	...	...	...	...	...
Byte 15	Slave 127	Slave 126	Slave 125	Slave 124	Slave 123	Slave 122	Slave 121	Slave 120

Byte Offset			
Start	**End**	**Name**	**Description**
74	74		PROFIBUS pad byte to make data word aligned (not used)
75	75	PROFIBUS Master Operating State	PROFIBUS Master operating state 0x0000=Offline 0x4000=Stop 0x8000=Clear 0xC000=Operate
76	77	PROFIBUS Ident Number	PROFIBUS Master PNO Ident number. Bytes will be swapped
78	81	PROFIBUS Master Serial Number	Unique 32-bit serial number for the PROFIBUS Master
82	83	PROFIBUS Software Version	Software version number for the PROFIBUS Master software Example for Version 1.40: High Byte - 0x40 Low Byte - 0x01
84	85	PROFIBUS Master Module Status	Represents the PROFIBUS Master module's operating status Bit 2=Application Status 0 - Application Stopped 1 - Application Running Bit 8=Data exchange (FBRS) 0 - There is no data exchange with any of the assigned slaves 1 - There is data exchange with at least one of the assigned slaves Bit 9=Slave input frozen/cleared (FBFC) 0 - A slave's inputs in the IN area are cleared if a slave is not in Data Exchange 1 - A slave's inputs in the IN area are frozen if a slave is not in Data Exchange Bit 12=Reset (RDR) 0 - No action 1 - A reset is requested by the PROFIBUS Master module because a new database has been downloaded
86	89	PROFIBUS configuration Checksum	CRC32 checksum for PROFIBUS Master configuration downloaded from configuration utility
90	93	MVI56 Module configuration checksum	MVI56-PDPMV1 for module configuration downloaded from configuration utility
94	95	Application program scan counter	MVI56-PDPMV1 module program scan counter. Can be used to gauge application code scan time performance
96	97	Module PROFIBUS output image data update counter	Counter representing the number of times the output data image is transferred to the module's internal Master
98	99	Module PROFIBUS input image data update counter	Counter representing the number of times the input data image is transferred from the module's internal Master
100	101	Module out mailbox counter	Module out mailbox counter
102	103	Module in mailbox counter	Module in mailbox counter
104	105	Module alarm IND receive counter	Number of spontaneous alarm messages received from slave
106	107	Module alarm CON receive counter	Number of confirmation messages received from slaves indicating that the slave received the confirmation message from the MVI56-PDPMV1 module
108	109	Reserved	Reserved for future use

Byte Offset

Start	End	Name	Description
110	111	Reserved	Reserved for future use
112	113	Module backplane read count	Rollover counter of the number of MVI56-PDPMV1 to processor backplane read data transfers
114	115	Module backplane write count	Rollover counter of the number of MVI56-PDPMV1 to processor backplane write data transfers
116	117	Module backplane error count	Rollover counter of the number of MVI56-PDPMV1 to processor backplane data transfers that have failed
118	119	File Error Word	Bitmapped value that indicates which files are not present
			Bit 0: Problem with PDPMV1.CFG file
			Bit 1: Not used
			Bit 2: Problem with PDPMV1.DDB file
			Bit 3: Problem with PDPMV1.XML file
			Bits 4 to 15 not used.
			If this word has a value other than 0, the CFG ERR LED on the module will be illuminated.
120	145	Reserved	Reserved for future use
146	147	In mailbox queue count	Number of message in the in mailbox queue
148	149	Out mailbox queue count	Number of message in the out mailbox queue
150	151	Alarm queue count	Number of message in the alarm queue
152	153	Last out mailbox message ID processed from output image	Last out mailbox message ID processed from output image
154	155	Current in mailbox control index	Current in mailbox control index
156	157	Current alarm control index	Current alarm control index
158	496		Reserved for future use
496	497		Index of Output image to receive (1 to 4 for data, 0 or -1 if no data)
498	499		Block index of -1 or 0 for status block

Output Data Block (Block ID = 1 to n)

This block contains the PROFIBUS output data to be sent from the processor to the module.

Output Data Block Format (Output Image)

You can find the contents of the Output Data Block (Output Image) in the *MVI56PDPMV1.Output* user controller tags in RSLogix.

Byte Offset

Start	End	Name	Description
0	0	Write Block ID	Block number of this write block 1 to *n*
1	2	PROFIBUS CRC32 value	Computed for PROFIBUS Config

Byte Offset			
Start	**End**	**Name**	**Description**
3	4	Module CRC32 value	Computed for module data
			When the module first starts up or recognizes an initialization of the processor, it will compare the values of the two CRC's in the input and output images. If either one of the CRCs do not match, the module will be placed in Stop mode. If each set matches, the module will be placed in Operate mode.
5	*n*	PROFIBUS Output Data	PROFIBUS output data transferred to the PROFIBUS slaves

CIP Message Support

CIP Messaging is supported for mailbox communication and status retrieval.

CIP messaging is executed using a Generic CIP Message instruction using the Class value of 4 (assembly object). The instance value identifies the data used by the MSG instruction. The only two service types supported by the module are Get Attribute Single (0x0E) and Set Attribute Single (0x10). The Class parameter must always be set to a value of 4. The instance value selects the data set to interface with for the instruction. The Attribute field is not used by the module and should be set to 0.

For more information on the CIP message support, refer to the sample ladder logic.

The following topics describe how CIP messages are handled.

CIP Generic Messages

Get Attribute Single (0x0E)	Set Attribute Single (0x10)	Instance
Status	-	9
Mailbox	-	10
-	Mailbox	10
Alarm	-	11

6 Support, Service & Warranty

In This Chapter

Contacting Technical Support

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

1 Product Version Number
2 System architecture
3 Network details

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

1 Module configuration and 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, an after-hours answering system allows 24-hour/7-days-a-week pager access to one of our qualified Technical and/or Application Support Engineers. Detailed contact information for all our worldwide locations is available on the following page.*

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

6.1 Return Material Authorization (RMA) Policies and Conditions

The following Return Material Authorization (RMA) Policies and Conditions (collectively, "RMA Policies") apply to any returned product. These RMA Policies are subject to change by ProSoft Technology, Inc., without notice. For warranty information, see Limited Warranty (page 248). In the event of any inconsistency between the RMA Policies and the Warranty, the Warranty shall govern.

6.1.1 Returning Any Product

a) In order to return a Product for repair, exchange, or otherwise, the Customer must obtain a Return Material Authorization (RMA) number from ProSoft Technology and comply with ProSoft Technology shipping instructions.

b) In the event that the Customer experiences a problem with the Product for any reason, Customer should contact ProSoft Technical Support at one of the telephone numbers listed above (page 245). A Technical Support Engineer will request that you perform several tests in an attempt to isolate the problem. If after completing these tests, the Product is found to be the source of the problem, we will issue an RMA.

c) All returned Products must be shipped freight prepaid, in the original shipping container or equivalent, to the location specified by ProSoft Technology, and be accompanied by proof of purchase and receipt date. The RMA number is to be prominently marked on the outside of the shipping box. Customer agrees to insure the Product or assume the risk of loss or damage in transit. Products shipped to ProSoft Technology using a shipment method other than that specified by ProSoft Technology, or shipped without an RMA number will be returned to the Customer, freight collect. Contact ProSoft Technical Support for further information.

d) A 10% restocking fee applies to all warranty credit returns, whereby a Customer has an application change, ordered too many, does not need, etc. Returns for credit require that all accessory parts included in the original box (i.e.; antennas, cables) be returned. Failure to return these items will result in a deduction from the total credit due for each missing item.

6.1.2 Returning Units Under Warranty

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

a) A replacement module will be shipped and invoiced. A purchase order will be required.

b) Credit for a product under warranty will be issued upon receipt of authorized product by ProSoft Technology at designated location referenced on the Return Material Authorization

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

ii. If defect is customer generated and is repairable, customer can authorize ProSoft Technology to repair the unit by providing a purchase order for 30% of the current list price plus freight charges, duties and taxes as applicable.

6.1.3 Returning Units Out of Warranty

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

The following is a list of non-repairable units:

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

6.2 LIMITED WARRANTY

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

6.2.1 What Is Covered By This Warranty

a) *Warranty On New Products*: ProSoft warrants, to the original purchaser, that the Product that is the subject of the sale will (1) conform to and perform in accordance with published specifications prepared, approved and issued by ProSoft, and (2) will be free from defects in material or workmanship; provided these warranties only cover Product that is sold as new. This Warranty expires three (3) years from the date of shipment for Product purchased **on or after** January 1st, 2008, or one (1) year from the date of shipment for Product purchased **before** January 1st, 2008 (the "Warranty Period"). If the Customer discovers within the Warranty Period a failure of the Product to conform to specifications, or a defect in material or workmanship of the Product, the Customer must promptly notify ProSoft by fax, email or telephone. In no event may that notification be received by ProSoft later than 39 months from date of original shipment. Within a reasonable time after notification, ProSoft will correct any failure of the Product to conform to specifications or any defect in material or workmanship of the Product, with either new or remanufactured replacement parts. ProSoft reserves the right, and at its sole discretion, may replace unrepairable units with new or remanufactured equipment. All replacement units will be covered under warranty for the 3 year period commencing from the date of original equipment purchase, not the date of shipment of the replacement unit. Such repair, including both parts and labor, will be performed at ProSoft's expense. All warranty service will be performed at service centers designated by ProSoft.

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

6.2.2 What Is Not Covered By This Warranty

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

b) This Warranty does not cover the failure of the Product to perform specified functions, or any other non-conformance, defects, losses or damages caused by or attributable to any of the following: (i) shipping; (ii) improper installation or other failure of Customer to adhere to ProSoft's specifications or instructions; (iii) unauthorized repair or maintenance; (iv) attachments, equipment, options, parts, software, or user-created programming (including, but not limited to, programs developed with any IEC 61131-3, "C" or any variant of "C" programming languages) not furnished by ProSoft; (v) use of the Product for purposes other than those for which it was designed; (vi) any other abuse, misapplication, neglect or misuse by the Customer; (vii) accident, improper testing or causes external to the Product such as, but not limited to, exposure to extremes of temperature or humidity, power failure or power surges; or (viii) disasters such as fire, flood, earthquake, wind and lightning.

c) The information in this Agreement is subject to change without notice. ProSoft shall not be liable for technical or editorial errors or omissions made herein; nor for incidental or consequential damages resulting from the furnishing, performance or use of this material. The user guide included with your original product purchase from ProSoft contains information protected by copyright. No part of the guide may be duplicated or reproduced in any form without prior written consent from ProSoft.

6.2.3 Disclaimer Regarding High Risk Activities

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

6.2.4 Intellectual Property Indemnity

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

a) Any documentation included with Product purchased from ProSoft is protected by copyright and may not be duplicated or reproduced in any form without prior written consent from ProSoft.

b) ProSoft's technical specifications and documentation that are included with the Product are subject to editing and modification without notice.

c) Transfer of title shall not operate to convey to Customer any right to make, or have made, any Product supplied by ProSoft.

d) Customer is granted no right or license to use any software or other intellectual property in any manner or for any purpose not expressly permitted by any license agreement accompanying such software or other intellectual property.

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

f) **Additional Restrictions Relating To Software And Other Intellectual Property**

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

6.2.5 Disclaimer of all Other Warranties

The Warranty set forth in What Is Covered By This Warranty (page 249) are in lieu of all other warranties, express or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose.

6.2.6 Limitation of Remedies **

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

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

6.2.7 Time Limit for Bringing Suit

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

6.2.8 No Other Warranties

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

6.2.9 Allocation of Risks

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

6.2.10 Controlling Law and Severability

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

Index