ProSoft® TECHNOLOGY

Where Automation Connects.



MVI56E-GSC/GSCXT

ControlLogix® Platform

Enhanced Generic ASCII Serial Communication Module

October 13, 2025

USER MANUAL

Your Feedback Please

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

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

MVI56E-GSC/GSCXT User Manual

For Public Use.

October 13, 2025

ProSoft Technology®, is a registered copyright of ProSoft Technology, Inc. All other brand or product names are or may be trademarks of, and are used to identify products and services of, their respective owners.

Content Disclaimer

This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither ProSoft Technology nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. Information in this document including illustrations, specifications and dimensions may contain technical inaccuracies or typographical errors. ProSoft Technology makes no warranty or representation as to its accuracy and assumes no liability for and reserves the right to correct such inaccuracies or errors at any time without notice. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

No part of this document may be reproduced in any form or by any means, electronic or mechanical, including photocopying, without express written permission of ProSoft Technology. All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components. When devices are used for applications with technical safety requirements, the relevant instructions must be followed. Failure to use ProSoft Technology software or approved software with our hardware products may result in injury, harm, or improper operating results. Failure to observe this information can result in injury or equipment damage.

Copyright © 2025 ProSoft Technology, Inc. All Rights Reserved.



For professional users in the European Union

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



Prop 65 Warning – Cancer and Reproductive Harm – www.P65Warnings.ca.gov

Agency Approvals and Certifications

Please visit our website: www.prosoft-technology.com

Open-Source Information

Open-Source Software used in the product

The product contains, among other things, Open-Source Software files, as defined below, developed by third parties and licensed under an Open-Source Software license. These Open-Source Software files are protected by copyright. Your right to use the Open-Source Software is governed by the relevant applicable Open-Source Software license conditions. Your compliance with those license conditions will entitle you to use the Open-Source Software as foreseen in the relevant license. In the event of conflicts between other ProSoft Technology, Inc. license conditions applicable to the product and the Open-Source Software license conditions, the Open-Source Software conditions shall prevail. The Open-Source Software is provided royalty-free (i.e. no fees are charged for exercising the licensed rights). Open-Source Software contained in this product and the respective Open-Source Software licenses are stated in the module webpage, in the link Open-Source.
If Open-Source Software contained in this product is licensed under GNU General Public License (GPL), GNU Lesser General Public License (LGPL), Mozilla Public License (MPL) or any other Open-Source Software license, which requires that source code is to be made available and such source code is not already delivered together with the product, you can order the corresponding source code of the Open-Source Software from ProSoft Technology, Inc. - against payment of the shipping and handling charges - for a period of at least 3 years since purchase of the product. Please send your specific request, within 3 years of the purchase date of this product, together with the name and serial number of the product found on the product label to:

ProSoft Technology, Inc.
Director of Engineering
9201 Camino Media, Suite 200
Bakersfield, CA 93311
USA

Warranty regarding further use of the Open-Source Software

ProSoft Technology, Inc. provides no warranty for the Open-Source Software contained in this product, if such Open-Source Software is used in any manner other than intended by ProSoft Technology, Inc. The licenses listed define the warranty, if any, from the authors or licensors of the Open-Source Software. ProSoft Technology, Inc. specifically disclaims any warranty for defects caused by altering any Open-Source Software or the product's configuration. Any warranty claims against ProSoft Technology, Inc. in the event that the Open-Source Software contained in this product infringes the intellectual property rights of a third party are excluded. The following disclaimer applies to the GPL and LGPL components in relation to the rights holders:
"This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License and the GNU Lesser General Public License for more details."
For the remaining Open-Source components, the liability exclusions of the rights holders in the respective license texts apply. Technical support, if any, will only be provided for unmodified software.

Contents

Your Feedback Please 2
Content Disclaimer 2

1 Start Here 7

1.1 What's New? 7
1.2 What's Different? 7
System Requirements 8
1.3 Deployment Checklist 9
1.4 Package Contents 10
1.5 Setting Jumpers 11
1.6 Installing the Module in the Rack 12
1.7 Using ProSoft Configuration Builder Software 13
1.7.1 Setting Up the Project 13
1.7.2 Connecting Your PC to the Module's Ethernet Port 15
1.7.3 Setting Up a Permanent IP Address 19
1.8 Before You Begin 32
1.8.1 About the Optional Add-On Instruction 32
1.9 Creating a New RSLogix 5000 Project 33
1.9.1 Creating the Module 34
1.9.2 Importing the Ladder Rung 36
1.10 Connecting Your PC to the ControlLogix Processor 44
1.11 Downloading the Sample Program to the Processor 45

2 MVI56E-GSC Configuration 46

2.1 GSC.UTIL.BackplaneFail 47
2.2 GSC.CONFIG.PortX 48
2.2.1 Port[x].Enabled 48
2.2.2 Port[x].Type 49
2.2.3 Port[x].Baudrate 50
2.2.4 Port[x].Parity 50
2.2.5 Port[x].DataBits 51
2.2.6 Port[x].StopBits 51
2.2.7 Port[x].RTSOn 51
2.2.8 Port[x].RTSOff 51
2.2.9 Port[x].Handshaking 51
2.2.10 Port[x].RTermCnt 51
2.2.11 Port[x].RTermChar 52
2.2.12 Port[x].RPacketLen 52
2.2.13 Port[x].RTimeout 52
2.2.14 Port[x].RDelay 52
2.2.15 Port[x].WTermCnt 52
2.2.16 Port[x].WTermChar 52
2.2.17 Port[x].WPacketLen 52
2.2.18 Port[x].WTimeout 52
2.2.19 Port[x].Spare 53
2.2.20 Port[x].WMinDelay 53
2.3 Changing Parameters During Operation 53

3 Diagnostics and Troubleshooting 54

3.1 Reading Status Data from the Module 54
3.2 The Diagnostics Menu 55
3.2.1 Using the Diagnostics Menu in ProSoft Configuration Builder 56
3.3 Monitoring Module Information 59
3.3.1 Version Menu 59
3.3.2 Config 59
3.3.3 NIC Status 59
3.4 Monitoring Backplane Information 60
3.4.1 Backplane Status Menu 60
3.5 Data Analyzer 61
3.5.1 Starting the Data Analyzer 61
3.5.2 Stopping the Data Analyzer 63
3.5.3 Data Analyzer Tips 64
3.6 Scrolling LED Status Indicators 66
3.6.1 Initialization Messages 66
3.6.2 Initialization Error Messages 67
3.6.3 Operation Message 67
3.7 Ethernet LED Indicators 67
3.8 Non-Scrolling LED Status Indicators 68
3.9 ControlLogix Processor Not in RUN or REM RUN 68
3.10 Clearing a Fault Condition 69
3.11 Troubleshooting 70
3.11.1 Processor Errors 70
3.11.2 Module Errors 70

4 Reference 71

4.1 Product Specifications 71
4.1.1 General Specifications 71
4.1.2 Functional Specifications 72
4.1.3 Hardware Specifications 73
4.2 General Concepts 74
4.2.1 Backplane Data Transfer 74
4.2.2 Data Flow between MVI56E-GSC Module and ControlLogix Processor 75
4.2.3 Termination of Received Data 79
4.3 Normal Data Transfer 83
4.3.1 Block Request from the Processor to the Module 83
4.3.2 Read Block 84
4.4 Special Function Blocks 88
4.4.1 Block 9998: Warm Boot 88
4.4.2 Block 9999: Cold Boot 88
4.4.3 Configuration Data Transfer Block 88
4.5 Using the Sample Add-On Instruction 90
4.5.1 Input/Output (I/O) Configuration and Module Properties 90
4.5.2 User-Defined Data Types 91
4.5.3 Controller Tags 99
4.5.4 Add-On-Defined Data Types 105
4.6 Using the Optional Add-On Instruction 110
4.6.1 Before You Begin 110
4.6.2 Overview 110
4.6.3 Importing the Optional Add-On Instruction Rung 111

4.6.4 Reading Ethernet Settings from the Module 115
4.6.5 Writing the Ethernet Settings to the Module 117
4.6.6 Reading the Clock Value from the Module 119
4.6.7 Writing the Clock Value to the Module 120
4.7 Using the Sample Program - RSLogix 5000 Version 15 and earlier 121
4.7.1 Opening the Sample Program in RSLogix 121
4.7.2 Choosing the Controller Type 124
4.7.3 Select the Slot Number for the Module 125
4.7.4 Downloading the Sample Program to the Processor 126
4.8 Adding the Sample Ladder to an Existing Application 127
4.9 Error/Configuration Word 129
4.10 Cable Connections 130
4.10.1 Ethernet Cable Specifications 130
4.10.2 Ethernet Performance 130
4.10.3 Ethernet Cable Configuration 131
4.10.4 RS-232 Application Port(s) 131
4.10.5 RS-422 134
4.10.6 RS-485 Application Port(s) 134
4.10.7 DB9 to RJ45 Adaptor (Cable 14) 135

5 Support, Service & Warranty 136

5.1 Contacting Technical Support 136
5.2 Warranty Information 136

1 Start Here

To get the most benefit from this User Manual, you should be familiar with:

- **Rockwell Automation® RSLogix™ software:** launch the program, configure ladder logic, and transfer the ladder logic to the processor
- **Microsoft Windows®:** install and launch programs, execute menu commands, navigate dialog boxes, and enter data
- **Hardware installation and wiring:** install the module, and safely connect generic ASCII serial and ControlLogix devices to a power source and to the MVI56E-GSC module's application port(s)

1.1 What's New?

MVI56E products are **backward compatible** with existing MVI56 products, ladder logic, and module configuration files already in use. Easily swap and upgrade to benefit from an array of new features designed to improve interoperability and enhance ease of use.

- **ProSoft Configuration Builder (PCB):** Microsoft Windows®-based utility software for diagnostics. Connect through the module's Ethernet port or use CIPconnect® to access troubleshooting features and functions.
- **ProSoft Discovery Service (PDS):** New Windows-based utility software to find and display a list of MVI56E modules on the network and to temporarily change a module's IP address to be able to connect with a module's web page.
- **CIPconnect-enabled:** Allows PC-to-module diagnostics from the Ethernet network through a ControlLogix® 1756-ENxT EtherNet/IP™ module.
- **Personality Card:** An industrial-grade compact flash memory card storing the module's Ethernet settings, allowing quick and easy replacement.
- **LED Scrolling Diagnostic Display:** 4-character, alphanumeric display, providing English messages for status and alarm data, and for processor and network communication status.

1.2 What's Different?

The MVI56E-GSC Generic ASCII Serial Communication module is configured in RSLogix™ 5000 software using the sample ladder or Add-On Instruction (AOI). It also uses ProSoft Discovery Service (PDS), ProSoft Configuration Builder (PCB), as well as all required product documentation.

- PDS is the software utility used to allow your PC to connect to the module to set a temporary Ethernet IP address. Then you can connect to the module's web page to retrieve or change the module's firmware though an Ethernet link.
- PCB is the software used to provide access to the module's diagnostic menus and application serial port communication data analyzer features.

System Requirements

The MVI56E-GSC module requires the following minimum hardware and software components:

- Rockwell Automation ControlLogix® processor (firmware version 10 or higher) with compatible limited voltage power supply and one free slot in the rack for the MVI56E-GSC module. The module requires 800mA of available 5 VDC and 3 mA of available 24 VDC power.



- Rockwell Automation RSLogix 5000 programming software
  - Version 16 or higher required for Add-On Instruction
  - Version 15 or lower must use Sample Ladder, available from www.prosoft-technology.com
- Rockwell Automation RSLinx® communication software version 2.51 or higher
- ProSoft Configuration Builder (PCB) (included)
- Pentium® II 450 MHz minimum. Pentium III 733 MHz (or better) recommended
- Supported operating systems:
  - Microsoft Windows 10
  - Microsoft Windows 7 Professional (32-or 64-bit)
  - Microsoft Windows 2000 Professional with Service Pack 1, 2, or 3
  - Microsoft Windows Server 2003
- 128 Mbytes of RAM minimum, 256 Mbytes of RAM recommended
- 100 Mbytes of free hard disk space (or more based on application requirements)

Note: The Hardware and Operating System requirements in this list are the minimum recommended to install and run software provided by ProSoft Technology®. Other third-party applications may have different minimum requirements. Refer to the documentation for any third-party applications for system requirements.

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

1.3 Deployment Checklist

Before you begin configuring the module, consider the following questions. Your answers will help you determine the scope of your project, and the configuration requirements for a successful deployment.

1 _____ Are you creating a new application or integrating the module into an existing application?

Most applications can use the Sample Add-On Instruction or Sample Ladder Logic without any edits to the Sample Program.

2 _____ Which slot number in the chassis will the MVI56E-GSC module occupy?

For communication to occur, you must enter the correct slot number in the sample program.

3 _____ Are RSLogix 5000 and RSLinx installed?

RSLogix and RSLinx are required to communicate to the ControlLogix processor (1756-L1, L5x, L6x). Sample Ladder programs are available for different versions of RSLogix 5000.

4 _____ How many words of data do you need to transfer in your application (from ControlLogix to Module / to ControlLogix from Module)?

The MVI56E-GSC module can transfer a maximum of 5000 (16-bit) registers to and from the ControlLogix processor. The Sample Ladder transfers 600 words to the ControlLogix processor (into the Read Data array), and obtains 600 words from the ControlLogix processor (from the Write Data array)

5 Serial Communication Parameters for the network:

_____ Baud rate?
_____ Data bits?
_____ Parity?
_____ Stop bits?

6 _____ Wiring type to use (RS232, 422 or 485). Configured by jumper settings.

Required for proper implementation of the module.

Note: If you are installing your module into a new system, and plan to use our Sample Ladder Logic, refer to the printed Quick Start Guide in the module package for simple installation procedures.

- For version 16 or newer of RSLogix 5000, refer to Upload the Add-On Instruction from the Module.
- For EXISTING system installations, refer to Using the Sample Program - RSLogix 5000 Version 15 and earlier (page 120).

Note: Most applications can use the Sample Ladder Logic *without modifying the sample program.*

1.4 Package Contents

The following components are included with your MVI56E-GSC module, and are all required for installation and configuration.

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

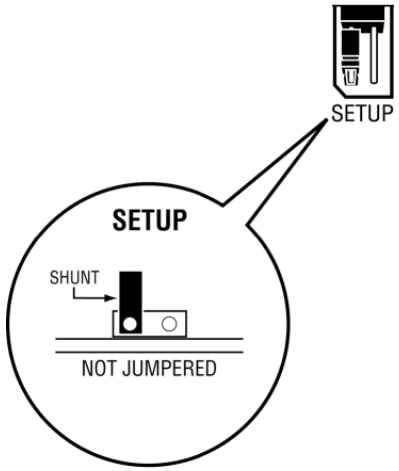
Qty.	Part Name	Part Number	Part Description
1	MVI56E-GSC Module	MVI56E-GSC	Enhanced Generic ASCII Serial Communication Module
2	Cable	Cable #14, RJ45 to DB9 Male Adapter cable	For DB9 Connection to Module's Application Serial Port
2	Adapter	1454-9F	Two Adapters, DB9 Female to Screw Terminal. For RS422 or RS485 Connections to Port 1 and 2 of the Module

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

1.5 Setting Jumpers

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

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



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

Security considerations:

Leaving the Setup pin jumpered leaves the module open to unexpected firmware updates.

You should consider segmenting the data flow for security reasons. Per IEC 62443-1-1, you should align with IEC 62443 and implement segmentation of the control system. Relevant capabilities are firewalls, unidirectional communication, DMZ. Oil and Gas customers should also see DNVGL-RP-G108 for guidance on partitioning.

You should practice security by design, per IEC 62443-4-1, including layers of security and detection. The module relies on overall network security design, as it is only one component of what should be a defined zone or subnet.

1.6 Installing the Module in the Rack

Make sure your ControlLogix processor and power supply are installed and configured, before installing the MVI56E-GSC module. Refer to your Rockwell Automation product documentation for installation instructions.

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

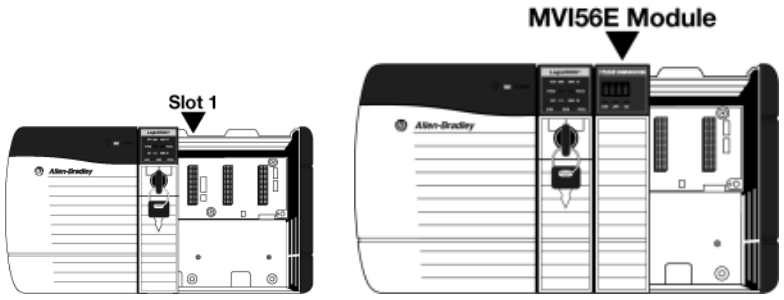
After you have checked the placement of the jumpers, insert the MVI56E-GSC into the ControlLogix chassis. Use the same technique recommended by Rockwell Automation to remove and install ControlLogix modules.

You can install or remove ControlLogix system components while chassis power is applied and the system is operating. However, please note the following warning.

Warning: When you insert or remove the module while backplane power is on, an electrical arc can occur. An electrical arc can cause personal injury or property damage by sending an erroneous signal to the system's actuators. This can cause unintended machine motion or loss of process control. Electrical arcs may also cause an explosion when they happen in a hazardous environment. 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 Align the module with the top and bottom guides, and then slide it into the rack until the module is firmly against the backplane connector.



2 With a firm, steady push, snap the module into place.
3 Check that the holding clips on the top and bottom of the module are securely in the locking holes of the rack.
4 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.
5 Turn power ON.

Note: If the module is improperly inserted, the system may stop working or may behave unpredictably.

Note: When using the MVI56E-GSCXT, you must use the 1756-A5XT or 1756-A7LXT chassis to uphold the XT specifications. In these chassis, modules are spaced further apart than in standard ControlLogix chassis. Blank spaces are inserted between active modules.

1.7 Using ProSoft Configuration Builder Software

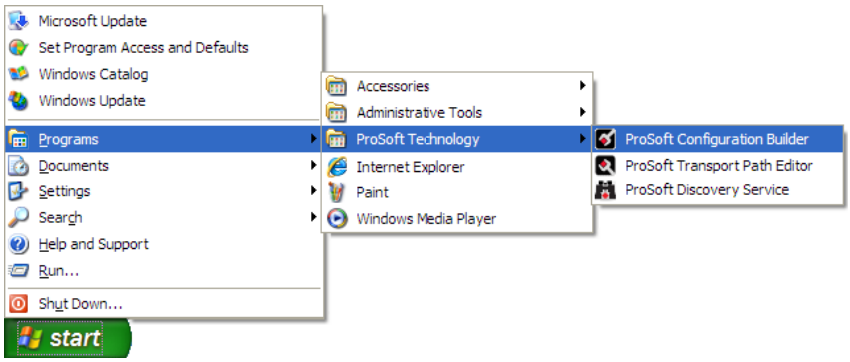
ProSoft Configuration Builder (PCB) provides a quick and easy way to manage module configuration files customized to meet your application needs. *PCB* is not only a powerful solution for new configuration files, but also allows you to import information from previously installed (known working) configurations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

The ProSoft Discovery Service (PDS) is available as a stand-alone application, or as part of ProSoft Configuration Builder. ProSoft Discovery Service shows you all the MVI56E modules available on your local area network.

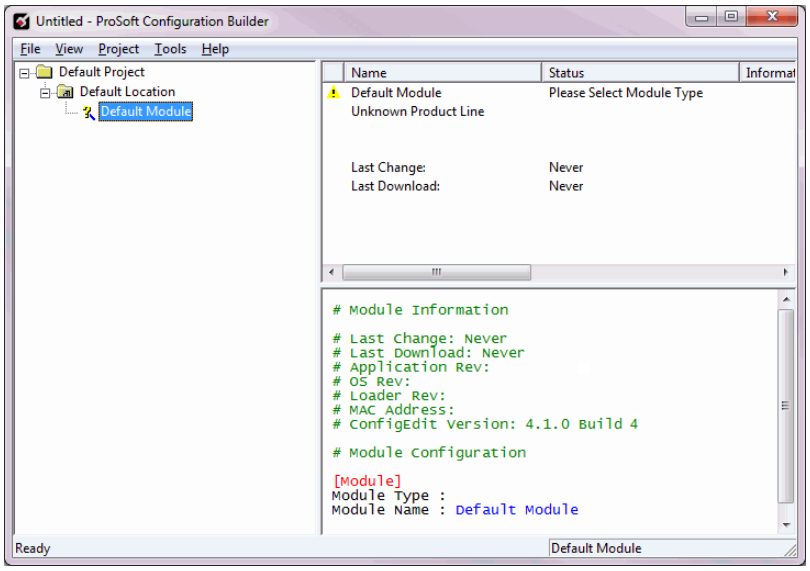
Note: The MVI56E-GSC module receives its protocol and backplane configuration information from the Ladder Logic. Use ProSoft Configuration Builder to configure the module's Ethernet settings.

1.7.1 Setting Up the Project

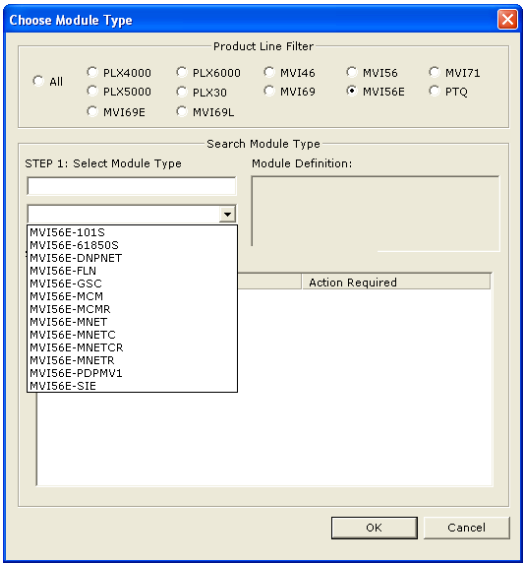
To begin, start **PROSOFT CONFIGURATION BUILDER** (PCB).



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



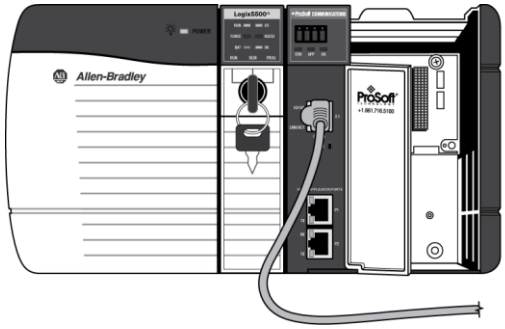
1. Use the mouse to select **DEFAULT MODULE** in the tree view, and then click the right mouse button to open a shortcut menu.
2. On the shortcut menu, select **CHOOSE MODULE TYPE**. This action opens the *Choose Module Type* dialog box.



3 In the *Product Line Filter* area of the dialog box, select **MVI56E**. In the *Select Module Type* dropdown list, select **MVI56E-GSC**, and then click **OK** to save your settings and return to the *ProSoft Configuration Builder* window.

1.7.2 Connecting Your PC to the Module's Ethernet Port

With the module securely mounted, connect one end of the Ethernet cable to the **CONFIG (E1)** Port, and the other end to an Ethernet hub or switch accessible from the same network as your PC. Or, you can connect directly from the Ethernet Port on your PC to the **CONFIG (E1)** Port on the module.

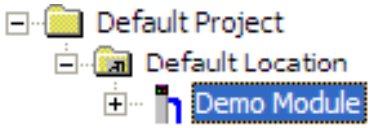


Setting Up a Temporary IP Address

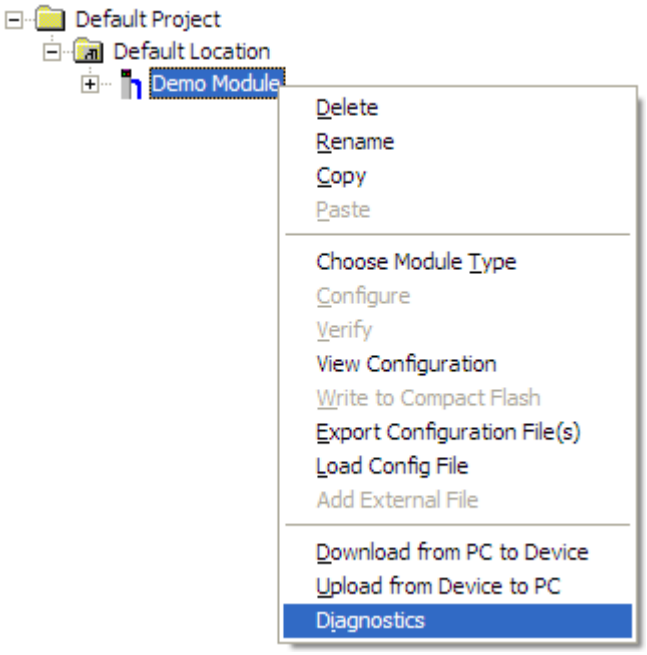
Important: *ProSoft Configuration Builder* locates MVI56E-GSC modules through UDP broadcast messages. These messages may be blocked by routers or layer 3 switches. In that case, *ProSoft Discovery Service* will be unable to locate the modules.

To use *ProSoft Configuration Builder*, arrange the Ethernet connection so that there is no router/ layer 3 switch between the computer and the module OR reconfigure the router/ layer 3 switch to allow routing of the UDP broadcast messages.

1 In the tree view in *ProSoft Configuration Builder*, select the **MVI56E-GSC** module.



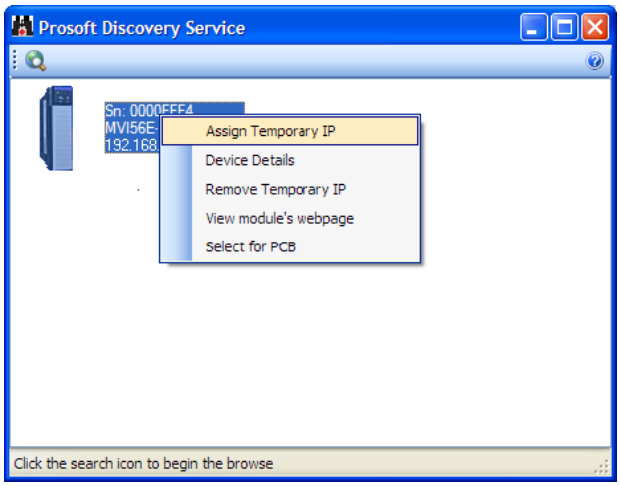
2 Click the right mouse button to open a shortcut menu. On the shortcut menu, choose **DIAGNOSTICS.**



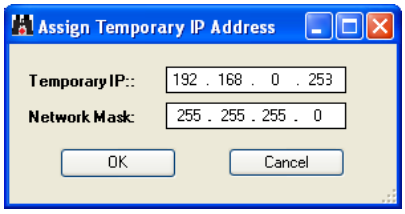
3 In the *Diagnostics* window, click the **SET UP CONNECTION** button.



4 In the *Connection Setup* dialog box, click the **BROWSE DEVICE(S)** button to open the *ProSoft Discovery Service*. Select the module, then right-click and choose **ASSIGN TEMPORARY IP**.

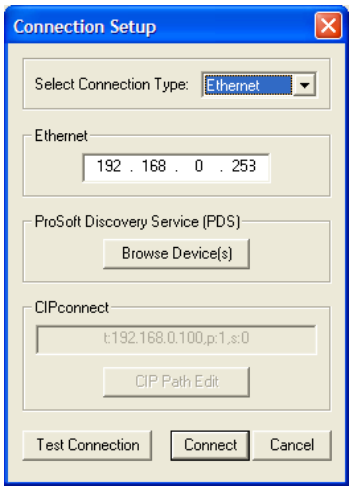


5 The module's default IP address is usually 192.168.0.250. Choose an unused IP within your subnet, and then click **OK**.

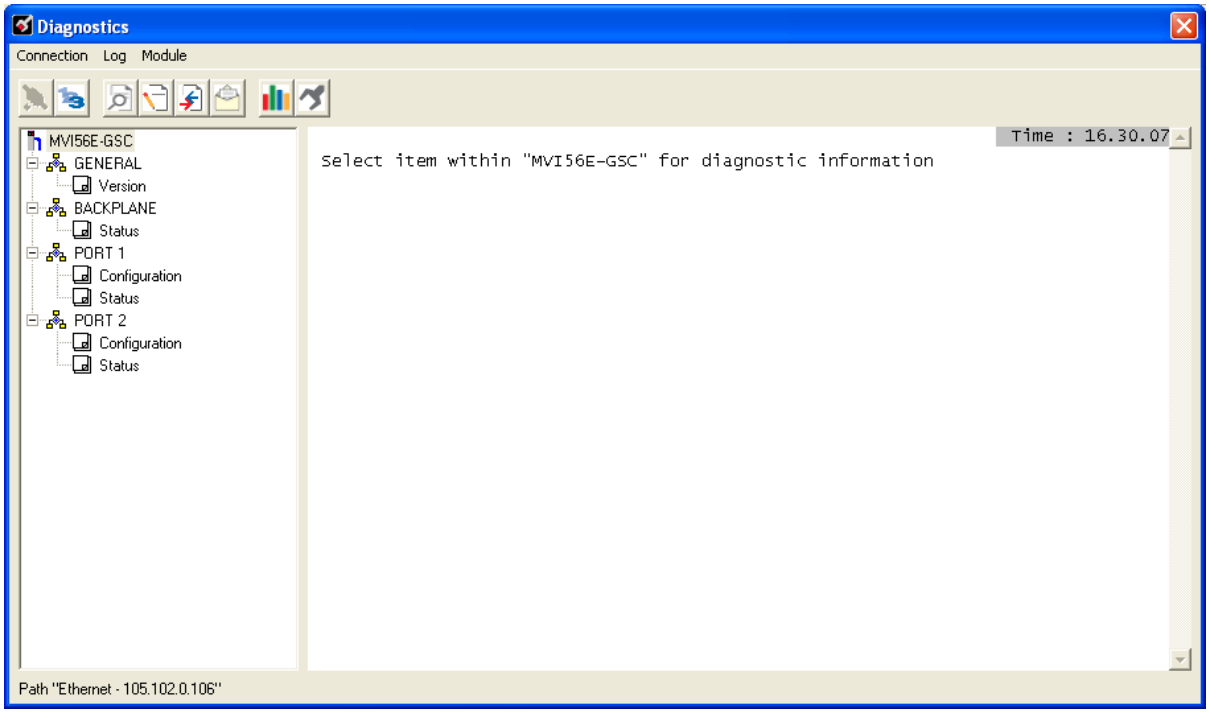


Important: The temporary IP address is only valid until the next time the module is initialized. For information on how to set the module's permanent IP address, see Setting Up a Permanent IP Address (page 18).

6 Close the *ProSoft Discovery Service* window. Enter the temporary IP in the Ethernet address field of the *Connection Setup* dialog box, then click the **TEST CONNECTION** button to verify that the module is accessible with the current settings.



7 If the *Test Connection* is successful, click **CONNECT**. The *Diagnostics* menu displays in the *Diagnostics* window.

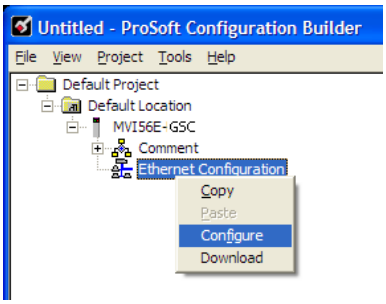


1.7.3 Setting Up a Permanent IP Address

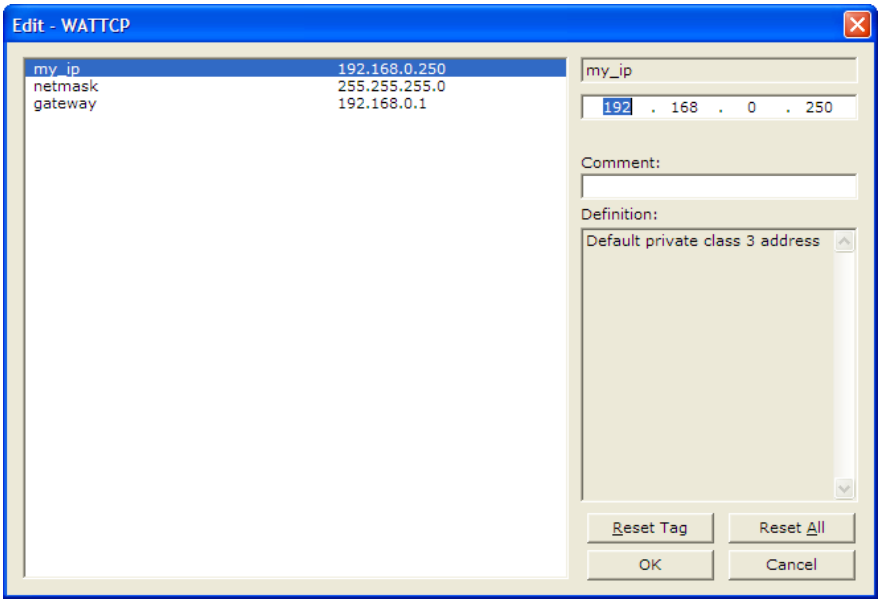
Note: For alternative methods of connecting to the module with your PC, refer to Using CIPconnect® to Connect to the Module (page 22) or Using RSWho to Connect to the Module (page 31).

These steps show you how to set a permanent IP address on the module. This example assumes module's default IP address is 192.168.0.250.

1. Start ProSoft Configuration Builder.
2. Select the **MVI56E-GSC** icon, and then click the [+] symbol to expand the **MVI56E-GSC** tree.
3. Select **ETHERNET CONFIGURATION**, and then click the right mouse button to open a shortcut menu.
4. On the shortcut menu, select **CONFIGURE.**

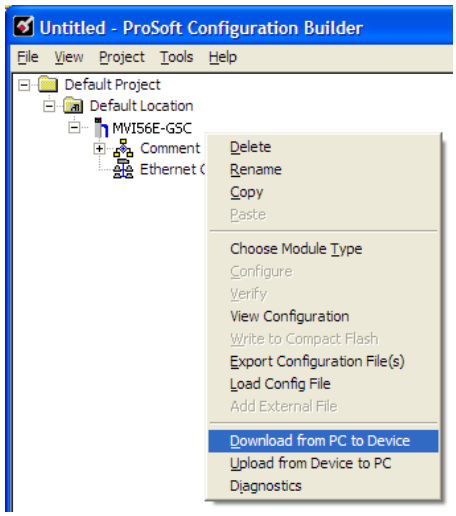


This action opens the *Edit-WATTCP* dialog box. Use this dialog box to enter the MVI56E-GSC module's permanent IP Address **(MY_IP), SUBNET MASK (NETMASK)** and **DEFAULT GATEWAY (GATEWAY)**.

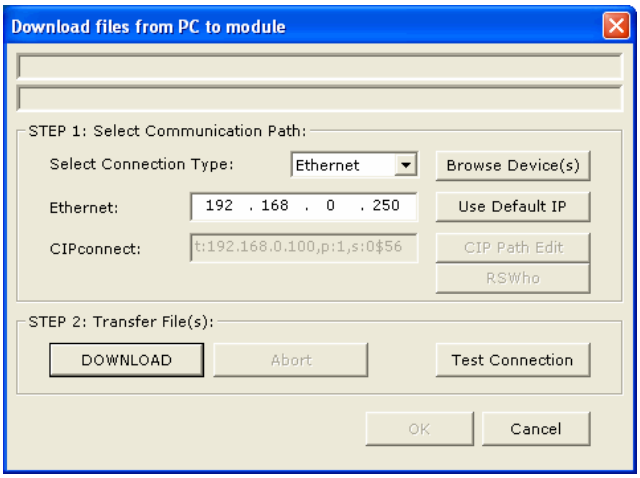


5 Click **OK** to save the updated Ethernet configuration.

6 Next, select the MVI56E-GSC icon, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose **DOWNLOAD FROM PC TO DEVICE**.



This action opens the **DOWNLOAD** dialog box. Notice that the Ethernet address field contains the temporary IP address you assigned previously. ProSoft Configuration Builder will use this temporary IP address to connect to the module.

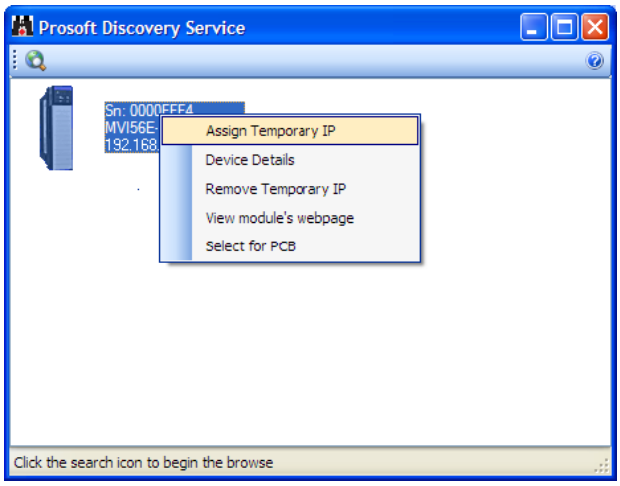


Click **TEST CONNECTION** to verify that the temporary IP address is correct.

7 If the connection succeeds, click **DOWNLOAD** to transfer the Ethernet configuration to the module.

If the Test Connection procedure fails, you will see an error message. To correct the error, follow these steps.

1 Click **OK** to dismiss the error message.
2 On the **DOWNLOAD** dialog box, click **BROWSE DEVICES** to open **PROSOFT DISCOVERY SERVICE**.



3 Select the module, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose **SELECT FOR PCB**.
4 Close Prosoft Discovery Service.
5 Click **DOWNLOAD** to transfer the Ethernet configuration to the module.

Using CIPconnect® to Connect to the Module

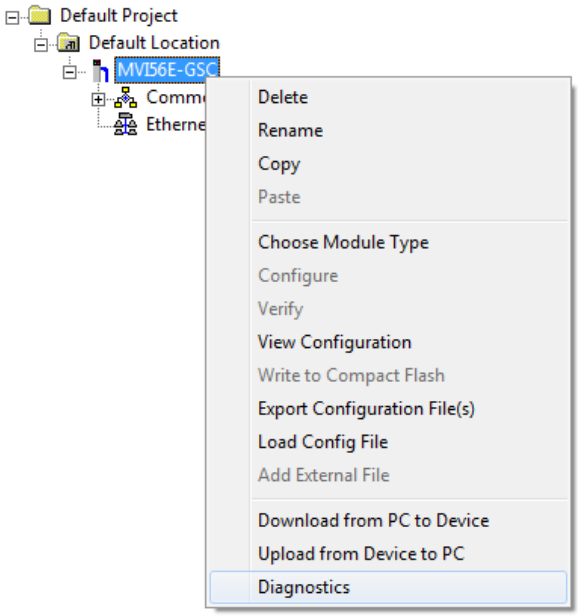
You can use CIPconnect® to connect a PC to the ProSoft Technology MVI56E-GSC module over Ethernet using Rockwell Automation's 1756-ENBT EtherNet/IP® module. This allows you to configure the MVI56E-GSC network settings and view module diagnostics from a PC. RSLinx is not required when you use CIPconnect. All you need are:

- The IP addresses and slot numbers of any 1756-ENBT modules in the path
- The slot number of the MVI56E-GSC in the destination ControlLogix chassis (the last ENBTx and chassis in the path).

To use CIPconnect, follow these steps.

1 In the tree view in *ProSoft Configuration Builder*, right-click the **MVI56E-GSC** icon to open a shortcut menu.

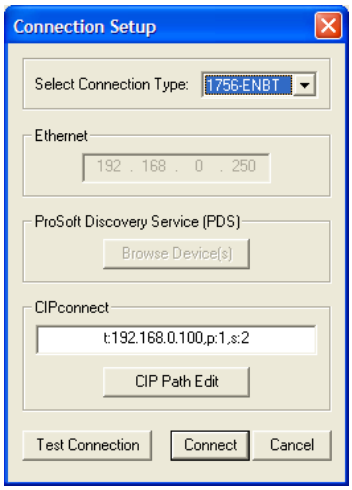
2 On the shortcut menu, choose **DIAGNOSTICS.**



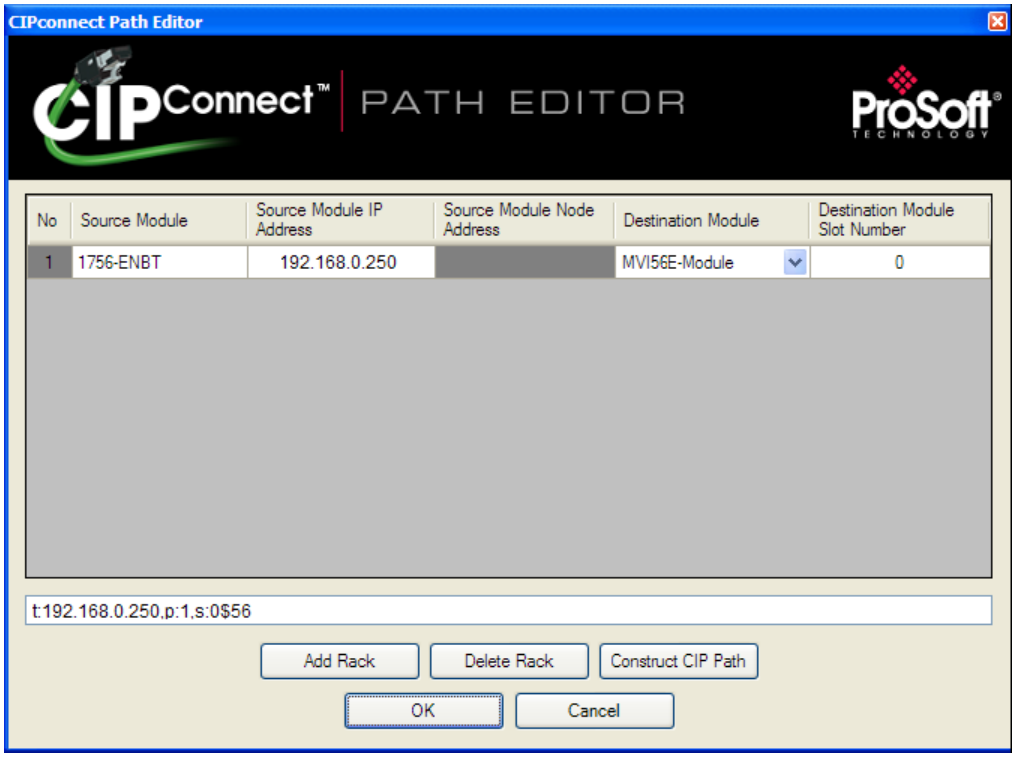
3 In the *Diagnostics* window, click the **SET UP CONNECTION** button.



4 In the *Select Connection Type* dropdown list, choose **1756-ENBT**. The default path appears in the text box, as shown in the following illustration.



5 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 MVI56E-GSC module. The first connection from the PC is always a 1756-ENBT (Ethernet/IP) module.

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

- If the MVI56E-GSC module is located in the same rack as the first 1756-ENBT module, select **RACK NO. 1** and configure the associated parameters.
- If the MVI56E-GSC is available in a remote rack (accessible through ControlNet or Ethernet/IP), include all racks (by using 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 MVI56E 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 MVI56E-GSC 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.

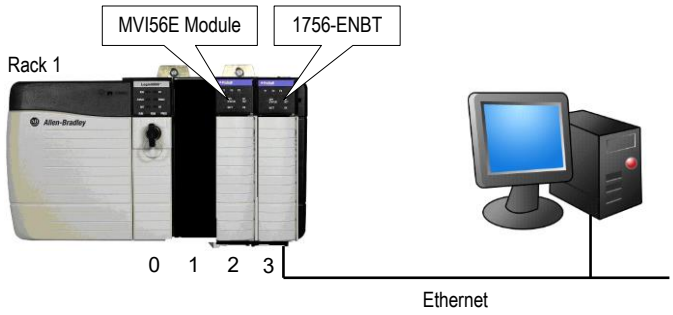
2 Click **CONSTRUCT CIP PATH** to build the path in text format

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.

Example 1: Local Rack Application

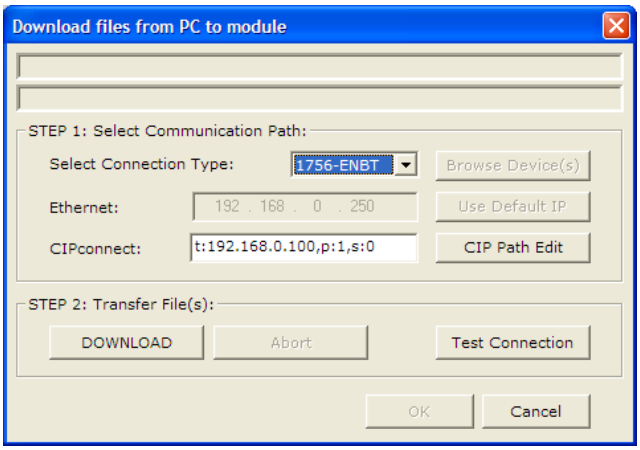
For this example the MVI56E-GSC module is located in the same rack as the 1756-ENBT that is connected to the PC.



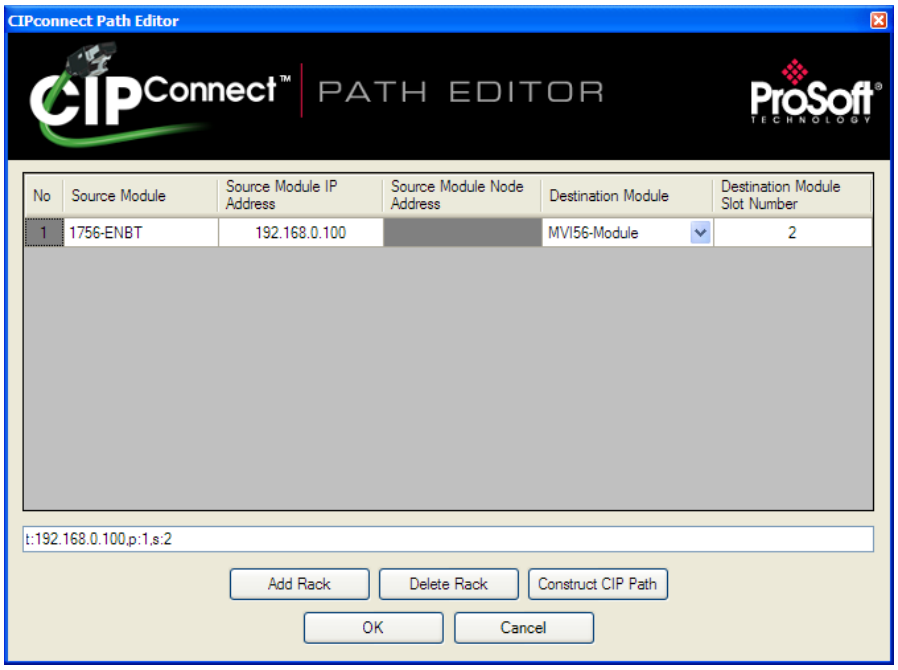
Rack 1

Slot	Module	Network Address
0	ControlLogix Processor	-
1	Any	-
2	MVI56E-GSC	-
3	1756-ENBT	IP=192.168.0.100

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

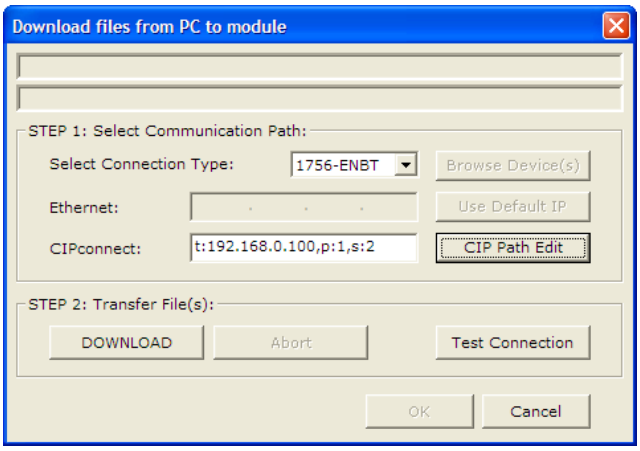


2 Configure the path as shown in the following illustration, and click **CONSTRUCT CIP PATH** to build the path in text format.

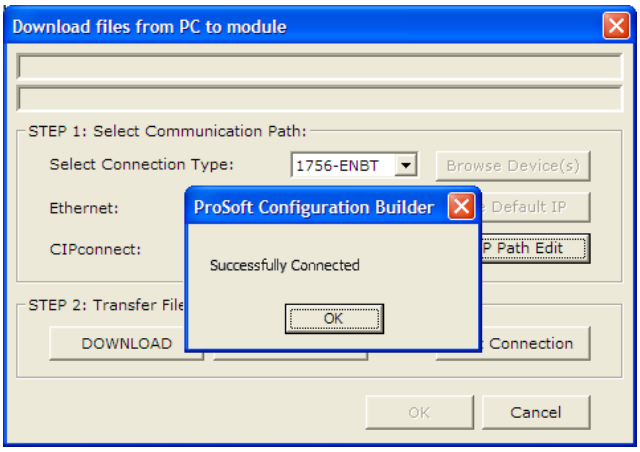


3 Click **OK** to close the **CIP PATH EDITOR** and return to the *Download* dialog box.

4 Check the new path in the download text box.



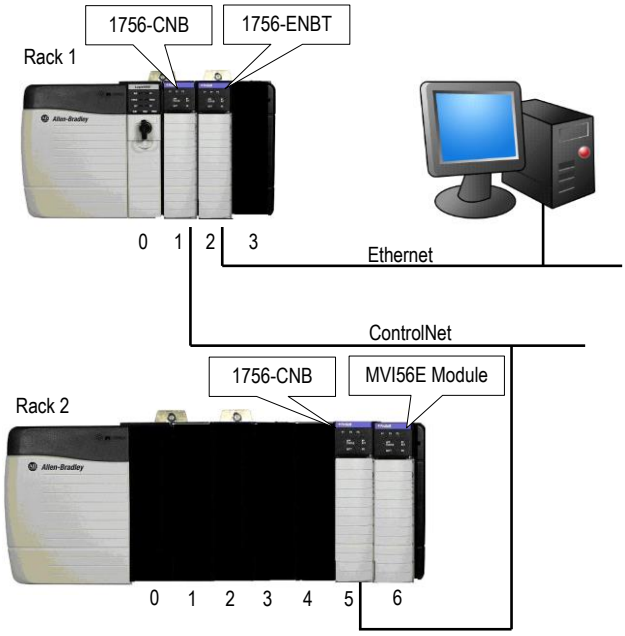
5 Click **TEST CONNECTION** to verify that the physical path is available. The following message should be displayed upon success.



6 Click **OK** to close the *Test Connection* pop-up. You are now ready to use CIPconnect for downloading changes to the module's Ethernet configuration settings. You may also use this connection to access module diagnostics screens and menus. For more information, see the chapter on Diagnostics and Troubleshooting. For more information, see the chapter on Diagnostics and Troubleshooting (page 54).

Example 2: Remote Rack Application - CIPconnect No Download

For this example, the MVI56E-GSC module is located in a remote rack accessible through ControlNet, as shown in the following illustration.



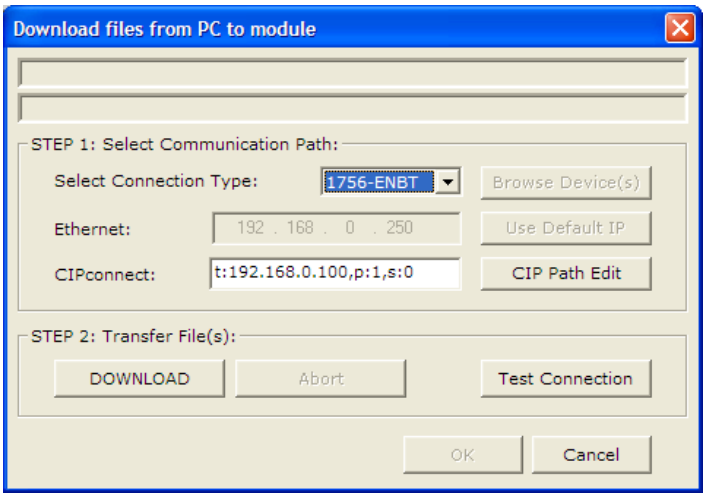
Rack 1

Slot	Module	Network Address
0	ControlLogix Processor	-
1	1756-CNB	Node = 1
2	1756-ENBT	IP=192.168.0.100
3	Any	-

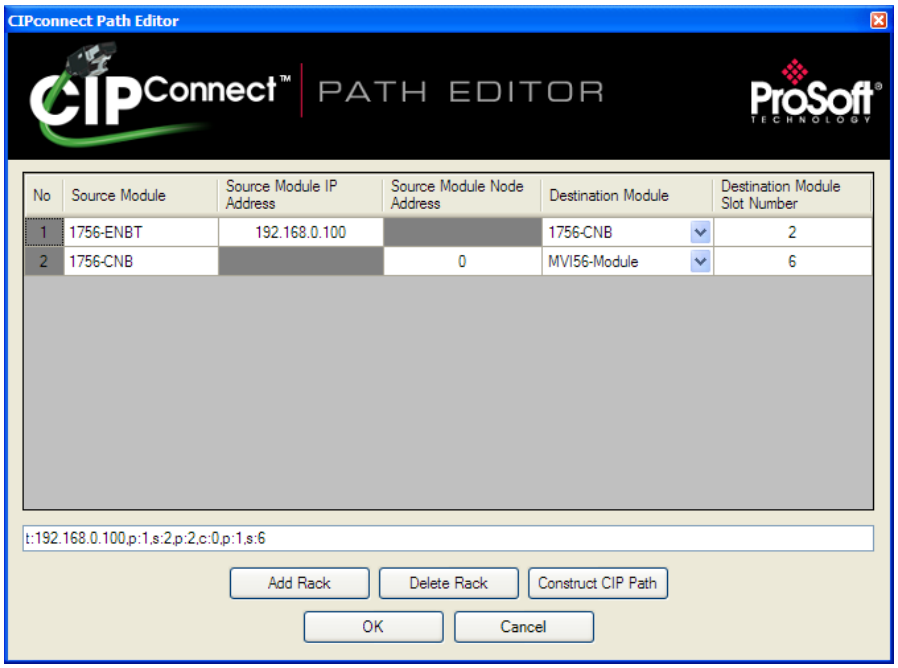
Rack 2

Slot	Module	Network Address
0	Any	-
1	Any	-
2	Any	-
3	Any	-
4	Any	-
5	1756-CNB	Node = 2
6	MVI56E-GSC	-

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

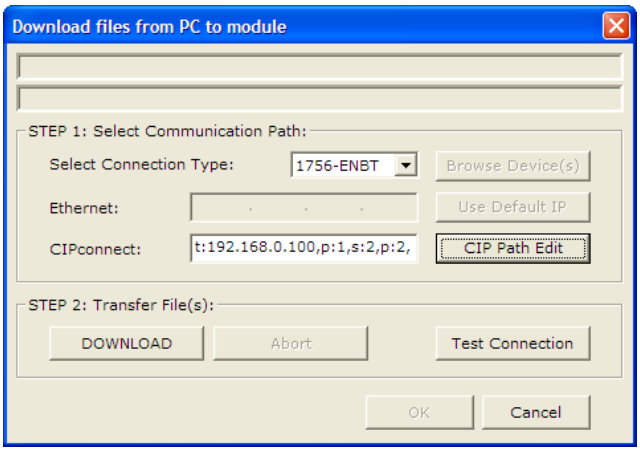


2 Configure the path as shown in the following illustration for this example and click **CONSTRUCT CIP PATH** to build the path in text format.

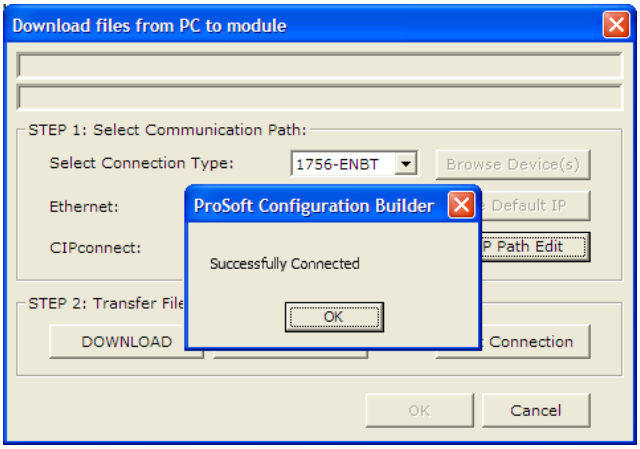


3 Click **OK** to close the **CIP PATH EDITOR** and return to the **DOWNLOAD** dialog box.

4 Check the new path in the download text box.



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

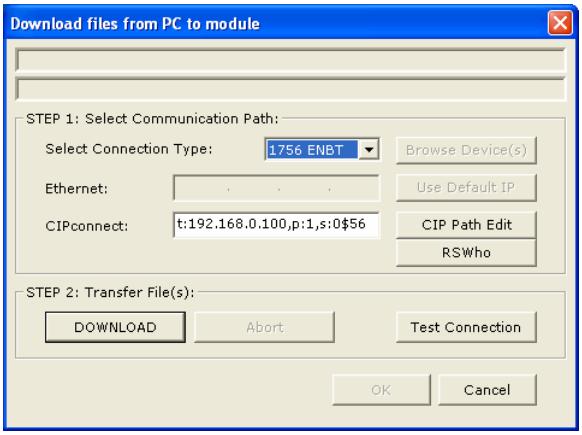


6 Click **OK** to close the *Test Connection* pop-up. You are now ready to use CIPconnect for downloading changes to the module's Ethernet configuration settings. You may also use this connection to access module diagnostics screens and menus. For more information, see the chapter on Diagnostics and Troubleshooting. For more information, see the chapter on Diagnostics and Troubleshooting (page 54).

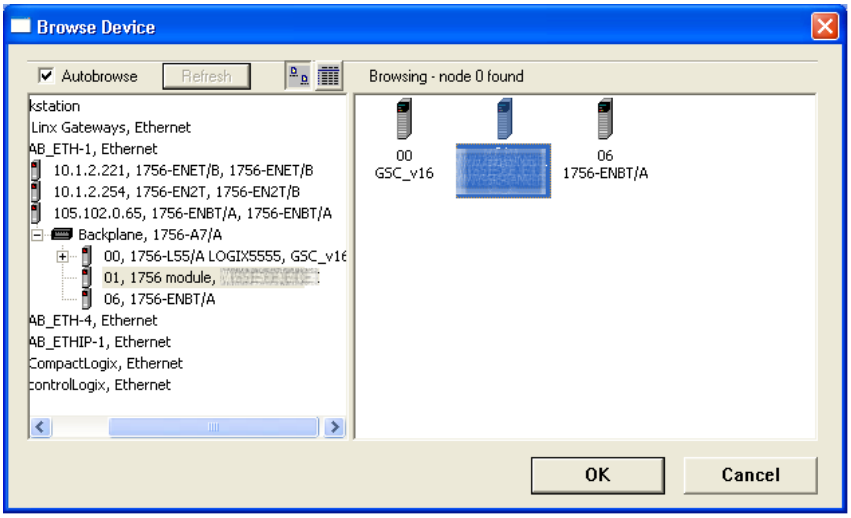
Using RSWho to Connect to the Module

You need to have RSLinx installed on your PC to use this feature. You also need an ENBT module set up in the rack. For information on setting up the ENBT module, see Using CIPconnect® to Connect to the Module (page 22).

1. In the tree view in *ProSoft Configuration Builder*, right-click the **MVI56E-GSC** module.
2. From the shortcut menu, choose **DOWNLOAD FROM PC TO DEVICE**.
3. In the *Download* dialog box, choose **1756 ENBT** from the *Select Connection Type* dropdown box.



4. Click **RSWHO** to display modules on the network. The MVI56E-GSC module will automatically be identified on the network.



5. Select the module, and then click **OK**.

1.8 Before You Begin

Note: This section only applies if your processor is using RSLogix 5000 version 16 or higher. If you have an earlier version, please see Using the Sample Program - RSLogix 5000 Version 15 and earlier (page 120).

Two Add-On Instructions are provided for the MVI56E-GSC module. The first is required for setting up the module; the second is optional.

Download them from www.prosoft-technology.com. Save them to a convenient location in your PC, such as *Desktop* or *My Documents*.

File Name	Description
MVI56EGSC_AddOn_Rung_xxx.L5X	L5X file containing Add-On Instruction, user defined data types, controller tags and ladder logic required to configure the MVI56E-GSC module
MVI56EGSC_Optional_Rung_xxx.L5X	Optional L5X file containing additional Add-On Instruction with logic for changing Ethernet configuration and clock settings.

1.8.1 About the Optional Add-On Instruction

The Optional Add-On Instruction performs the following tasks:

- Read/Write Ethernet Configuration

  Allows the processor to read or write the module IP address, subnet mask, and network gateway IP address.
- Read/Write Module Clock Value

  Allows the processor to read and write the module clock settings. The module's free-running clock also stores the last time that the Ethernet configuration was changed or the last time the module was restarted or rebooted. The date and time of the last change or restart is displayed on the scrolling LED during module power-up/start-up sequence.

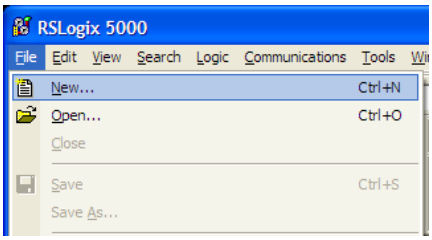
For more information, see Using the Optional Add-On Instruction (page 110).

Note: You can also set the date and time from the module's home page.

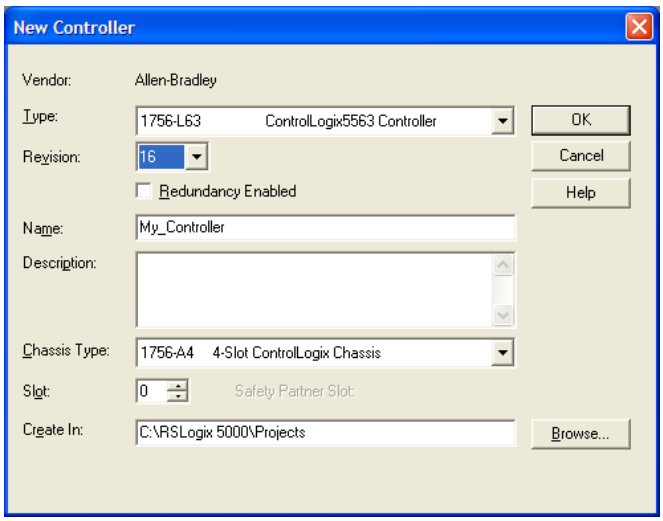
Important: The Optional Add-On Instruction supports only the two features listed above. You must use the regular MVI56E-GSC Add-On Instruction for all other features including backplane transfer and Modbus data communication.

1.9 Creating a New RSLogix 5000 Project

1 Open the **FILE** menu, and then choose **NEW**.



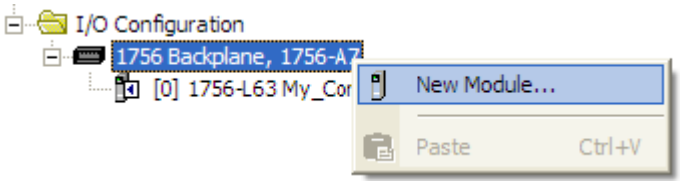
2 Select your ControlLogix controller model.
3 Select **REVISION 16**.
4 Enter a name for your controller, such as *My_Controller*.
5 Select your ControlLogix chassis type.
6 Select **SLOT 0** for the controller.



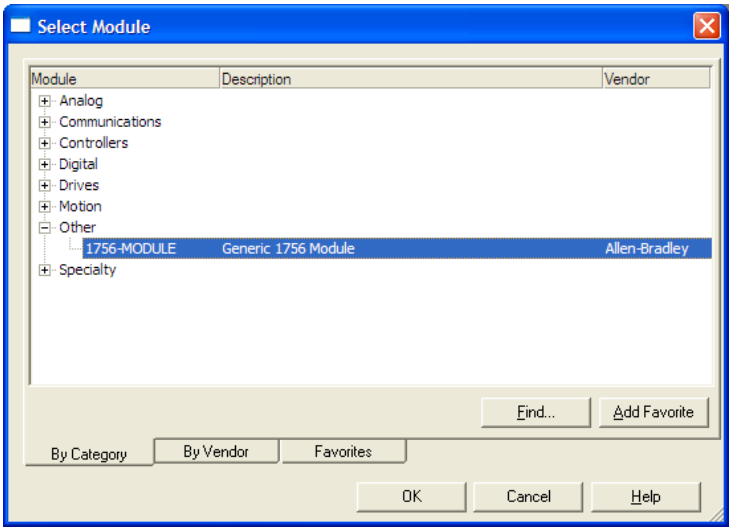
1.9.1 Creating the Module

1 Add the MVI56E-GSC module to the project.

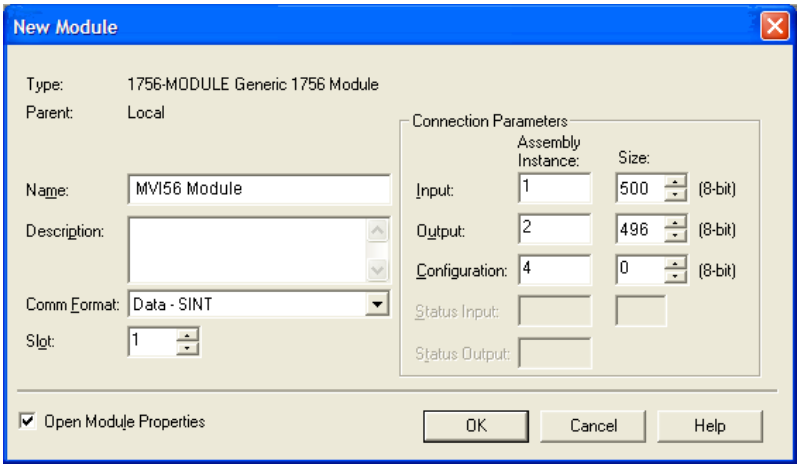
In the **CONTROLLER ORGANIZATION** window, select **I/O CONFIGURATION** and click the right mouse button to open a shortcut menu. On the shortcut menu, choose **NEW MODULE...**



This action opens the **SELECT MODULE** dialog box.



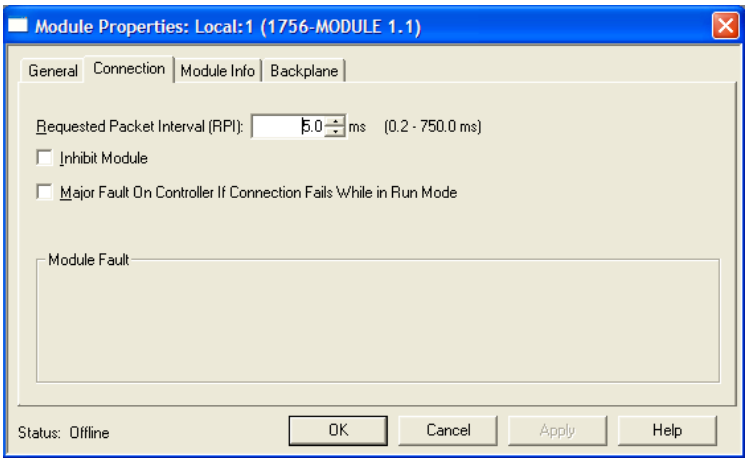
2 Select the **1756-MODULE (GENERIC 1756 MODULE)** from the list and click **OK.** This action opens the **NEW MODULE** dialog box.



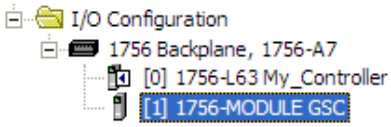
3 Set the Module Properties values as follows:

Parameter	Value
Name	Enter a module identification string. Example: GSC
Description	Enter a description for the module. Example: Enhanced Generic ASCII Serial Communication Module.
Comm Format	Select DATA-SINT.
Slot	Enter the slot number in the rack where the MVI56E-GSC module is located.
Input Assembly Instance	1
Input Size	500
Output Assembly Instance	2
Output Size	496
Configuration Assembly Instance	4
Configuration Size	0

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

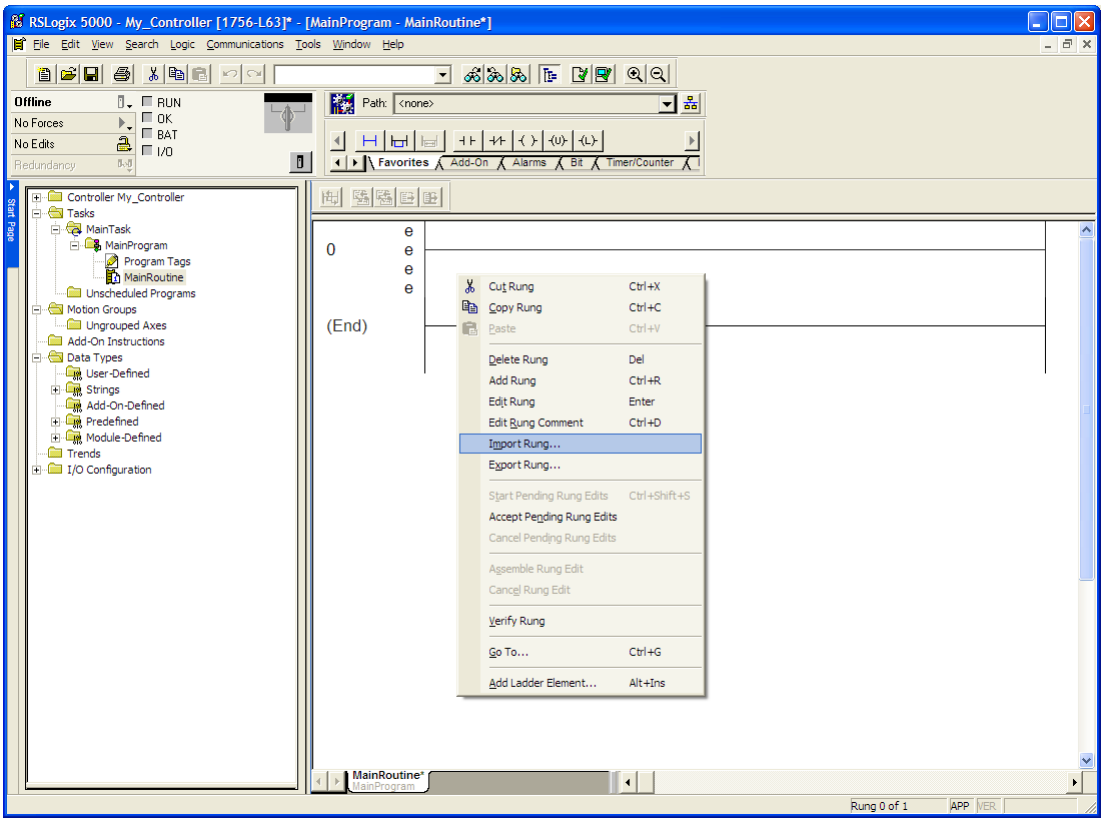


The **MVI56E-GSC** module is now visible in the **I/O CONFIGURATION** section.

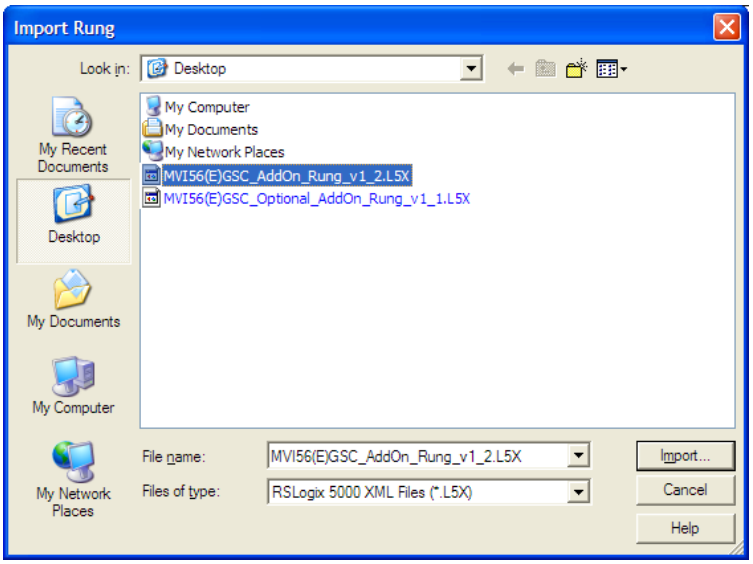


1.9.2 Importing the Ladder Rung

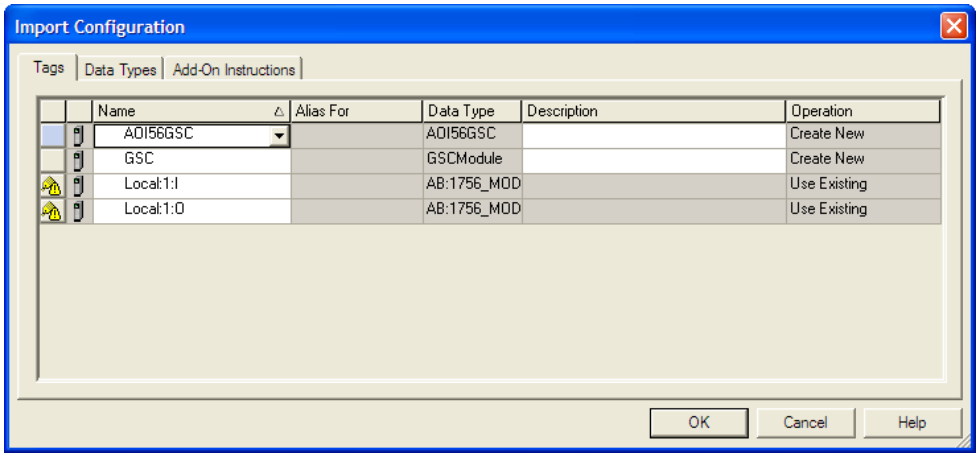
1. In the **CONTROLLER ORGANIZATION** window, expand the **TASKS** folder and subfolder until you reach the **MAINPROGRAM** folder.
2. In the **MAINPROGRAM** folder, double-click to open the **MAINROUTINE** ladder.
3. Select an empty rung in the new routine, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose **IMPORT RUNG…**



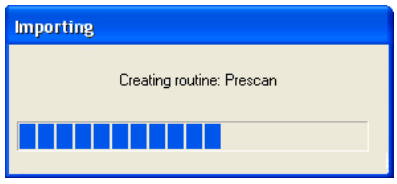
4 Navigate to the location on your PC where you Before You Begin (page 32) the Add-On Instruction (for example, "My Documents" or "Desktop"). Select the **MVI56EGSC_ADDON_RUNG_V1_3.L5X** file



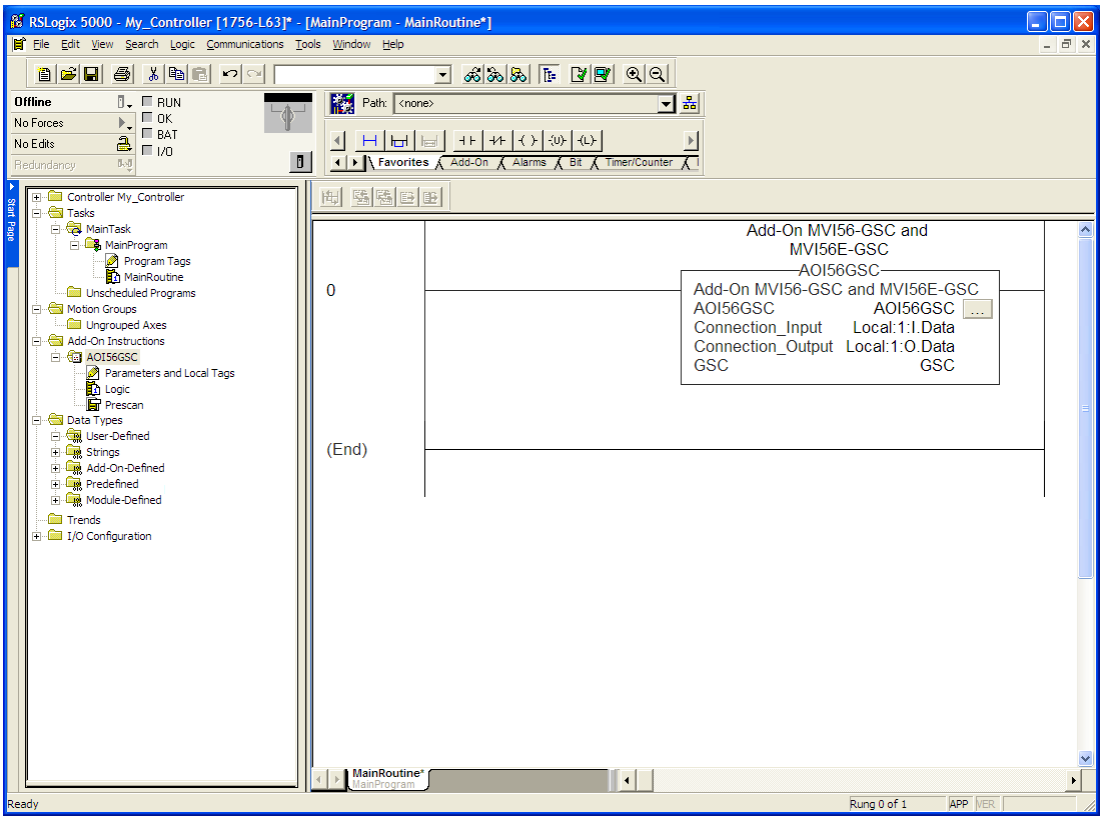
This action opens the **IMPORT CONFIGURATION** dialog box, showing the controller tags that will be created.



5 Click **OK** to confirm the import. RSLogix will indicate that the import is in progress:



When the import is complete, you will see the new Add-On Instruction rung in the ladder.

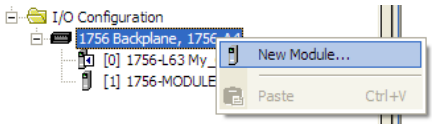


The procedure has also imported new User Defined Data Types, data objects and the Add-On instruction for your project.

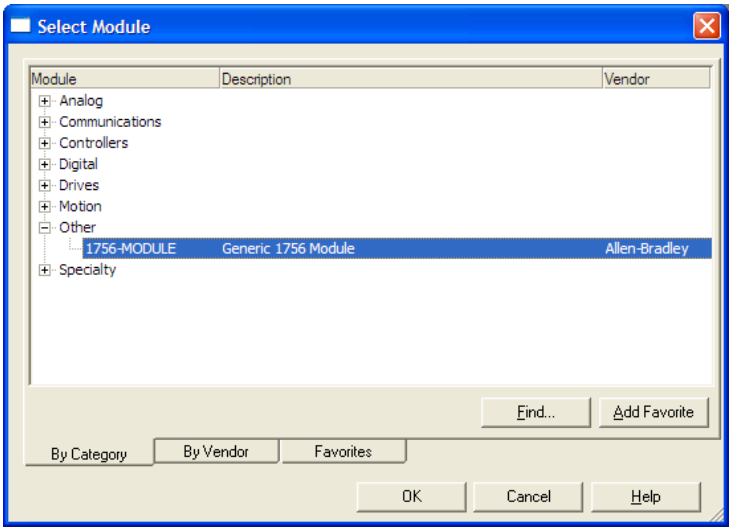


Adding Multiple Modules (Optional)

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



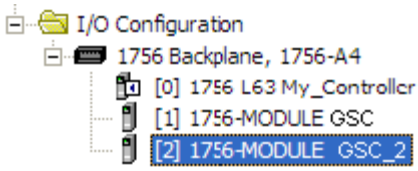
2 Select **1756-MODULE**



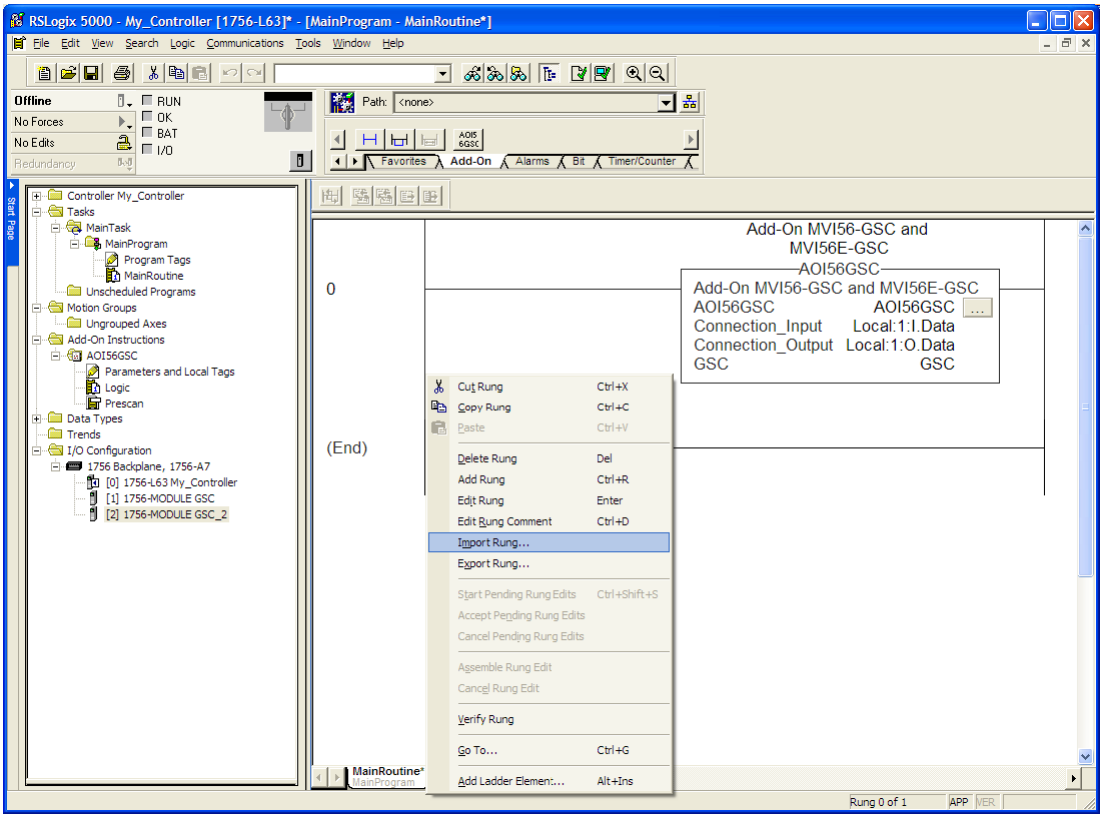
3 Set the Module Properties values as follows:

Parameter	Value
Name	Enter a module identification string. Example: GSC_2.
Description	Enter a description for the module. Example: ProSoft Enhanced Generic ASCII Serial Communication Module.
Comm Format	Select DATA-SINT.
Slot	Enter the slot number in the rack where the MVI56E-GSC module is located.
Input Assembly Instance	1
Input Size	500
Output Assembly Instance	2
Output Size	496
Configuration Assembly Instance	4
Configuration Size	0

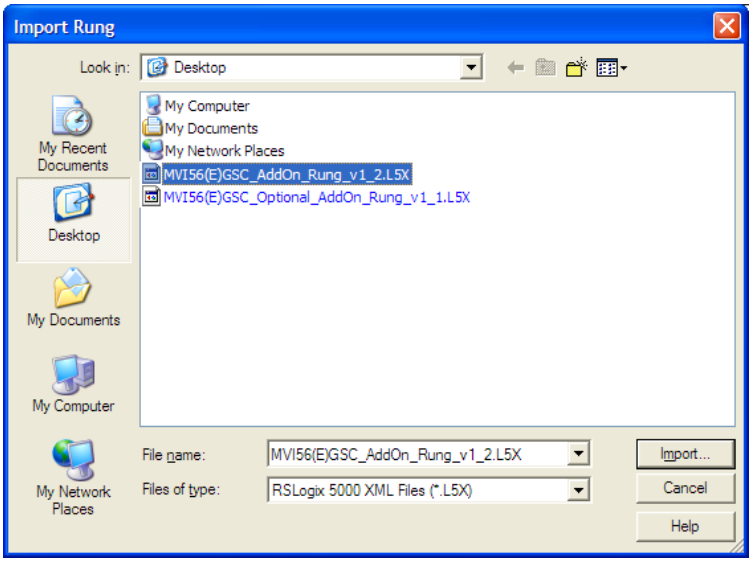
4 Click **OK** to confirm. The new module is now visible:



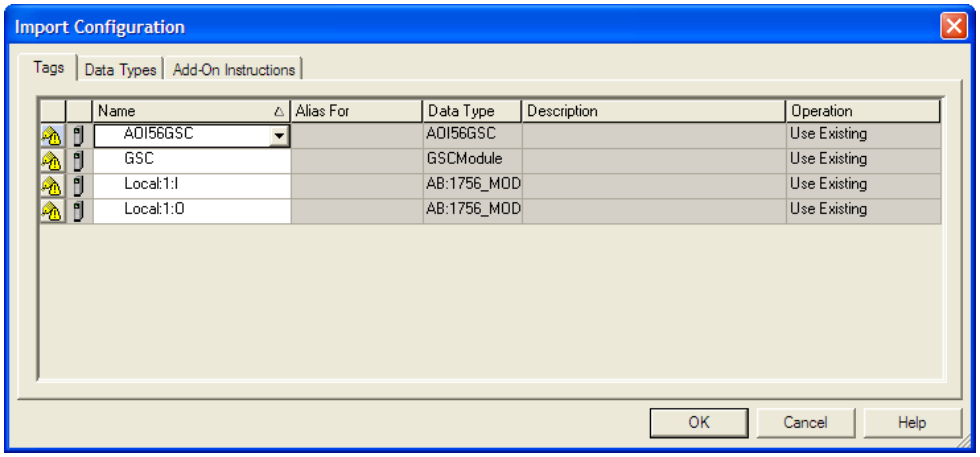
5 Expand the **TASKS** folder, and then expand the **MAINTASK** folder.
6 On the **MAINPROGRAM** folder, click the right mouse button to open a shortcut menu. On the shortcut menu, choose **NEW ROUTINE**. As an alternative to creating a separate New Routine, you could skip to Step 8 and import the AOI for the second module into the same routine you created for the first module.
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 or an existing routine, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose **IMPORT RUNG…**



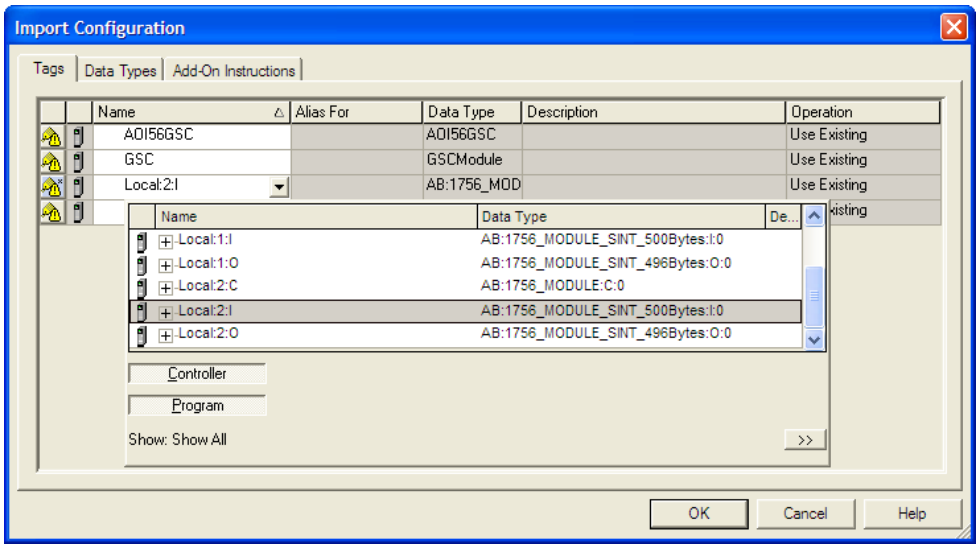
9 Select the file **MVI56EGSC_ADDON_RUNG_V1_3.L5X**

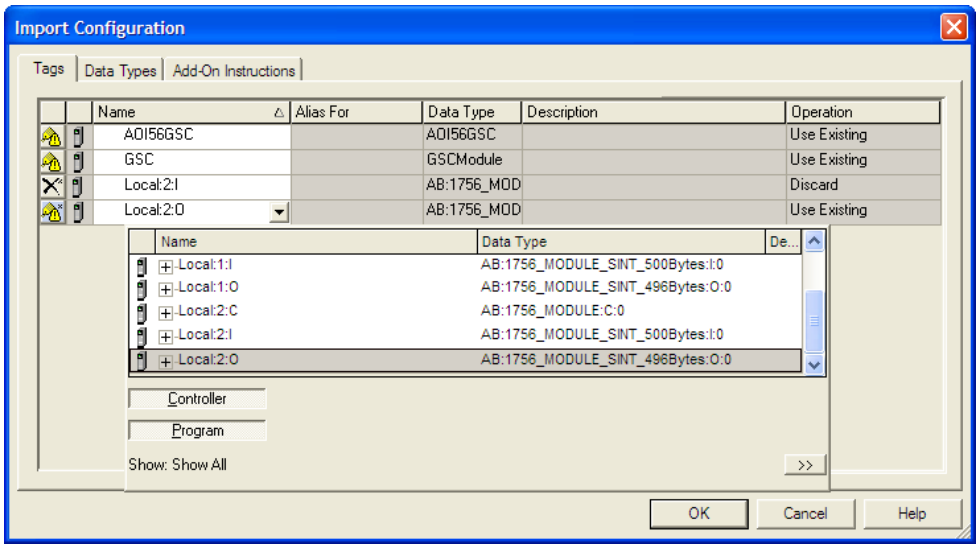


10 The following window will be displayed showing the tags to be imported:

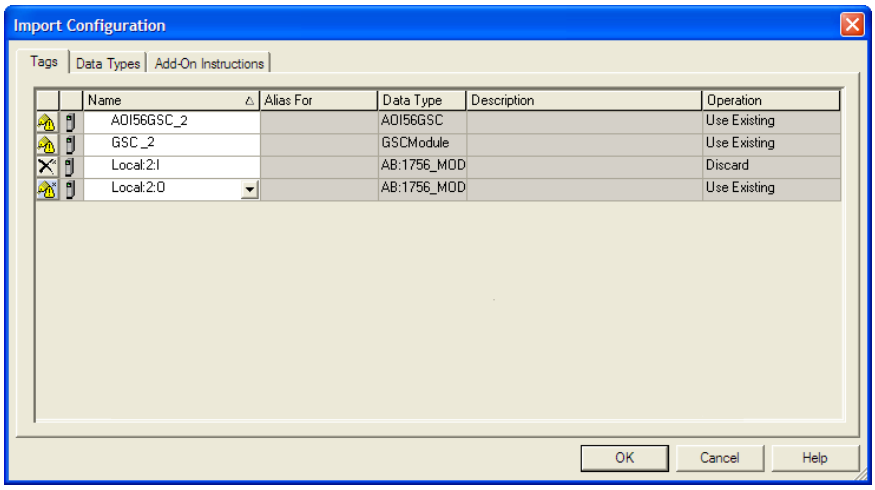


11 Associate the I/O connection variables to the correct module. The default values are Local:1:I and Local:1:O so these require change.

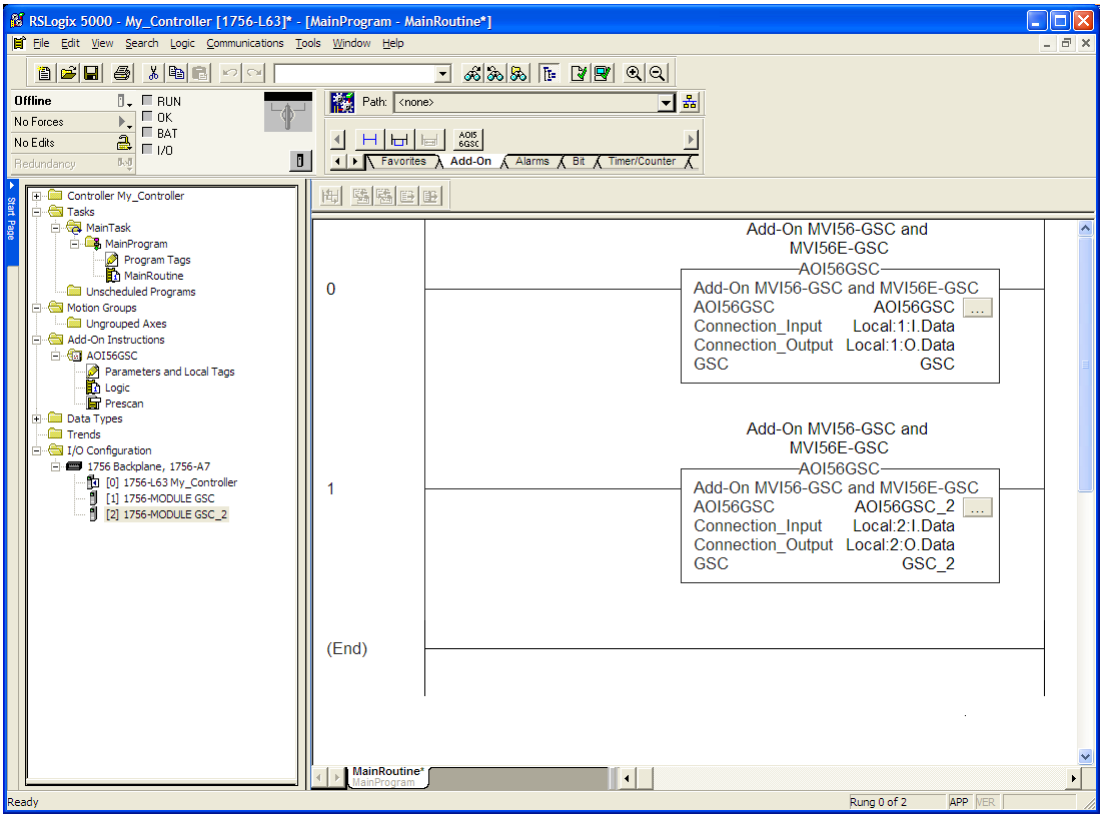




Change the default tag names **AOI56GSC** and **GSC** to avoid conflict with existing tags. In this step, you should append a string to the default tag names, such as "_2", as shown in the following illustration.



12 Click **OK** to confirm.



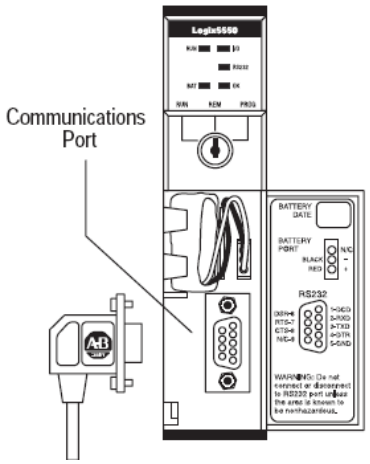
The setup procedure is now complete. Save the project and download the application to your ControlLogix processor.

1.10 Connecting Your PC to the ControlLogix Processor

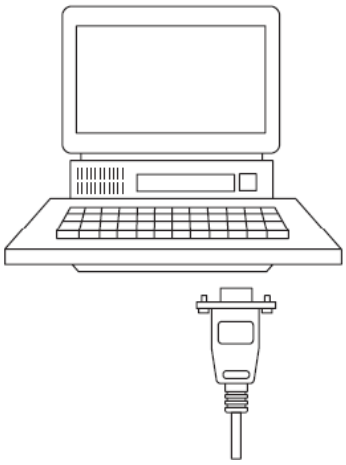
There are several ways to establish communication between your PC and the ControlLogix processor. The following steps show how to establish communication through the serial interface.

Note: It is not mandatory that you use the processor's serial interface. You may access the processor through whatever network interface is available on your system. Refer to your Rockwell Automation documentation for information on other connection methods

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



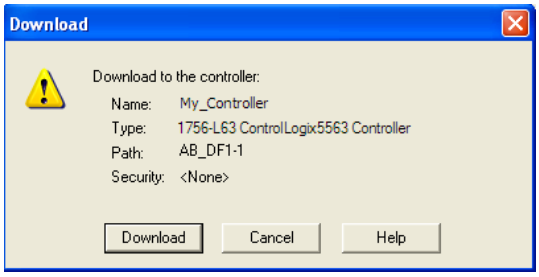
2 Connect the straight connector end of the cable to the serial port on your computer.



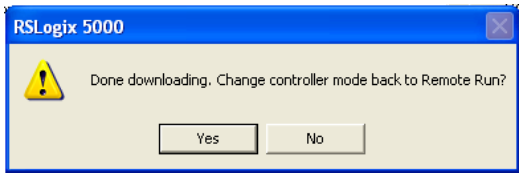
1.11 Downloading the Sample Program to the Processor

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

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



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



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

2 MVI56E-GSC Configuration

All module configuration settings, data to be exchanged, status, and error data, except for Debug Port Ethernet settings, are contained in the RSLogix 5000 controller tag arrays.

⊟-GSC	{...}		GSCMODULEDEF
⊞-GSC.CONFIG	{...}		GSCPORTCONFIG
⊞-GSC.DATA	{...}		GSCDATA
⊞-GSC.STATUS	{...}		GSCSTATUS
⊞-GSC.CONTROL	{...}		GSCCONTROL
⊞-GSC.UTIL	{...}		GSCUTIL

Only the following RSLogix controller tags need to be configured for the module to work as needed for your application.

1 The **GSC.UTIL.BACKPLANEFAIL** controller tag sets up a backplane communication failure counter to monitor the health of communication between the module and the ControlLogix processor across the ControlLogix backplane.

⊟-GSC	{...}		GSCMODULEDEF
⊞-GSC.CONFIG	{...}		GSCPORTCONFIG
⊞-GSC.DATA	{...}		GSCDATA
⊞-GSC.STATUS	{...}		GSCSTATUS
⊞-GSC.CONTROL	{...}		GSCCONTROL
⊟-GSC.UTIL	{...}		GSCUTIL
⊞-GSC.UTIL.LastRead	114	Decimal	SINT
⊞-GSC.UTIL.LastWrite	0	Decimal	SINT
⊞-GSC.UTIL.BackplaneFail	10	Decimal	INT
⊞-GSC.UTIL.P1WriteLast	0	Decimal	INT
⊞-GSC.UTIL.P2Writelast	-2	Decimal	INT
⊞-GSC.UTIL.P1ArrayIndex	0	Decimal	INT
⊞-GSC.UTIL.P2ArrayIndex	0	Decimal	INT
⊞-GSC.UTIL.P1RSindex	0	Decimal	INT
⊞-GSC.UTIL.P2RSindex	0	Decimal	INT

2 The **GSC.CONFIG.PORT1** and **GSC.CONFIG.PORT2** controller tag arrays contain tags to configure the application serial ports. These sets of controller tags allow you to configure typical serial port parameters, such as baud rate, data bits, and stop bits.

GSC	{...}		GSCMODULEDEF
GSC.CONFIG	{...}		GSCPORTCONFIG
GSC.CONFIG.Port1	{...}		GSCPORTCONFIGPARA
GSC.CONFIG.Port1.Enabled	1	Decimal	INT
GSC.CONFIG.Port1.Type	1	Decimal	INT
GSC.CONFIG.Port1.Baudrate	9600	Decimal	INT
GSC.CONFIG.Port1.Parity	0	Decimal	INT
GSC.CONFIG.Port1.DataBits	8	Decimal	INT
GSC.CONFIG.Port1.StopBits	1	Decimal	INT
GSC.CONFIG.Port1.RTSOn	0	Decimal	INT
GSC.CONFIG.Port1.RTSOff	0	Decimal	INT
GSC.CONFIG.Port1.Handshaking	0	Decimal	INT
GSC.CONFIG.Port1.RTermCnt	2	Decimal	INT
GSC.CONFIG.Port1.RTermChar	{...}	Hex	SINT[12]
GSC.CONFIG.Port1.RPacketLen	240	Decimal	INT
GSC.CONFIG.Port1.RTimeout	2000	Decimal	INT
GSC.CONFIG.Port1.RDelay	0	Decimal	INT
GSC.CONFIG.Port1.WTermCnt	0	Decimal	INT
GSC.CONFIG.Port1.WTermChar	{...}	Hex	SINT[12]
GSC.CONFIG.Port1.WPacketLen	0	Decimal	INT
GSC.CONFIG.Port1.WTimeout	500	Decimal	INT
GSC.CONFIG.Port1.WDelay	0	Decimal	INT
GSC.CONFIG.Port1.WMinDelay	0	Decimal	INT

2.1 GSC.UTIL.BackplaneFail

GSC.UTIL.BackplaneFail	10	Decimal	INT

A "backplane communication failure" is any failed attempt by the module to communicate with the ControlLogix processor. For local rack applications, where the module is installed in the same chassis as the processor, backplane failures can occur due to a hardware problem or Input/Output (I/O) configuration problem. For remote rack applications, where the module is installed in a different chassis from the processor and linked by communication adapters, such as ControlNet or EtherNet/IP, in addition to hardware and I/O configuration problems, any failure to communicate with the ControlLogix processor across a process network will be considered a backplane communication failure.

The **GSC.UTIL.BACKPLANEFAIL** parameter specifies the number of successive ControlLogix backplane transfer failures that must occur before the ASCII serial communication ports are shut down. If the parameter is set to zero, the communication ports will continue to operate under even if the module has no communication with the processor. If the value is set greater than 0 (1 to 65535), serial port communication will cease if the specified number of successive communication failures occur.

The only ways to restart serial port communication after a *Backplane Fail* shutdown are by Cold Boot from ladder logic, by removing and then reinserting the module in the chassis, or by turning power to the module or chassis off and then back on (Cold Start).

2.2 GSC.CONFIG.PortX

All configuration of the two ASCII application serial ports is done using the tags in the two **GSC.CONFIG.PORTX** arrays.

⊟ GSC	{...}		GSCMODULEDEF
⊟ GSC.CONFIG	{...}		GSCPORTCONFIG
⊟ GSC.CONFIG.Port1	{...}		GSCPORTCONFIGPARA
⊞ GSC.CONFIG.Port1.Enabled	1	Decimal	INT
⊞ GSC.CONFIG.Port1.Type	1	Decimal	INT
⊞ GSC.CONFIG.Port1.Baudrate	9600	Decimal	INT
⊞ GSC.CONFIG.Port1.Parity	0	Decimal	INT
⊞ GSC.CONFIG.Port1.DataBits	8	Decimal	INT
⊞ GSC.CONFIG.Port1.StopBits	1	Decimal	INT
⊞ GSC.CONFIG.Port1.RTSOn	0	Decimal	INT
⊞ GSC.CONFIG.Port1.RTSOff	0	Decimal	INT
⊞ GSC.CONFIG.Port1.Handshaking	0	Decimal	INT
⊞ GSC.CONFIG.Port1.RTermCnt	2	Decimal	INT
⊞ GSC.CONFIG.Port1.RTermChar	{...}	Hex	SINT[12]
⊞ GSC.CONFIG.Port1.RPacketLen	240	Decimal	INT
⊞ GSC.CONFIG.Port1.RTimeout	2000	Decimal	INT
⊞ GSC.CONFIG.Port1.RDelay	0	Decimal	INT
⊞ GSC.CONFIG.Port1.WTermCnt	0	Decimal	INT
⊞ GSC.CONFIG.Port1.WTermChar	{...}	Hex	SINT[12]
⊞ GSC.CONFIG.Port1.WPacketLen	0	Decimal	INT
⊞ GSC.CONFIG.Port1.WTimeout	500	Decimal	INT
⊞ GSC.CONFIG.Port1.WDelay	0	Decimal	INT
⊞ GSC.CONFIG.Port1.WMinDelay	0	Decimal	INT
⊞ GSC.CONFIG.Port2	{...}		GSCPORTCONFIGPARA
⊞ GSC.DATA	{...}		GSCDATA
⊞ GSC.STATUS	{...}		GSCSTATUS
⊞ GSC.CONTROL	{...}		GSCCONTROL
⊞ GSC.UTIL	{...}		GSCUTIL

2.2.1 Port[x].Enabled

0 = Disable port, 1 = Enable the port.

This parameter enables or disables the protocol port.

2.2.2 Port[x].Type

0 through 15

You can use any combination of one or more termination types. When multiple termination types are selected, the module will determine the end of the string to be as soon as any of the selected terminating conditions becomes true. After that, any new characters received will be considered part of a new string until the next time a terminating condition is true.

Decimal Value of Bits Selected	0 = No Bits Set **(S)** Streaming Mode Selected	1 = Bit 0 Set **(T)** Terminating Characters Selected	2 = Bit 1 Set **(M)** Message Timeout Selected	4 = Bit 2 Set **(D)** Intercharacter Delay Selected	8 = Bit 3 Set **(P)** Packet Length Selected	Port Types Selected by Bitmap
0	**0**					**S**
1		**1**				**T**
2			**2**			**M**
3		**1**	**2**			**M, T**
4				**4**		**D**
5		**1**		**4**		**D, T**
6			**2**	**4**		**D, M**
7		**1**	**2**	**4**		**D, M, T**
8					**8**	**P**
9		**1**			**8**	**P, T**
10			**2**		**8**	**P, M**
11		**1**	**2**		**8**	**P, M, T**
12				**4**	**8**	**P, D**
13		**1**		**4**	**8**	**P, D, T**
14			**2**	**4**	**8**	**P ,D, M**
15		**1**	**2**	**4**	**8**	**P, D, M, T**

For each termination type, you must also enter values in the following parameter configuration tags. Streaming mode is not recommended for general use, as it creates heavy demands on the ControlLogix processor and requires time-critical programming logic to properly process.

For Termination Type	Receive Parameters to Set
Terminating Characters	GSC.PortX.RTermCnt GSC.PortX.RTermChar
Message Timeout	GSC.PortX.RTimeout
Intercharacter Delay	GSC.PortX.RDelay
Packet (String) Length	GSC.PortX.RPacketLen

2.2.3 Port[x].Baudrate

This is the baud rate to use on the port. Enter the baud rate (bits per second) as a value. All devices on this port must communicate at the same baud rate. For example, to select 19K baud, enter 19200.

The following table describes the valid parameters for this configuration entry.

Value	Baud Rate (bits per second)
110	110 Baud
150	150 Baud
300	300 Baud
600	600 Baud
1200	1200 Baud
2400	2400 Baud
4800	4800 Baud
9600	9600 Baud
19200	19200 Baud
384	38400 Baud
576	57600 Baud
115	115200 Baud

2.2.4 Port[x].Parity

Parity is a simple error checking algorithm used in serial communication. This parameter specifies the type of parity checking to use. All devices connected to the port must use the same parity.

Value	Description
0	None
1	Odd
2	Even
3	Mark
4	Space

2.2.5 Port[x].DataBits

5, 6, 7 or 8

This parameter sets the number of data bits for each word used by the protocol. All devices communicating through this port must use the same number of data bits.

2.2.6 Port[x].StopBits

1 or **2**

Stop bits signal the end of a character in the data stream. For most applications, use one stop bit. For slower devices that require more time to re-synchronize, use two stop bits.

All devices communicating through this port must use the same number of stop bits.

2.2.7 Port[x].RTSOn

0 to **65535** milliseconds

This parameter sets the number of milliseconds to delay after *Ready To Send* (RTS) is asserted before data will be transmitted.

2.2.8 Port[x].RTSOff

0 to **65535** milliseconds

This parameter sets the number of milliseconds to delay after the last byte of data is sent before the RTS modem signal will be set low.

2.2.9 Port[x].Handshaking

Handshaking is a negotiation process between devices that establishes a data connection. Select the handshaking type that best matches the needs of the devices connected to the port.

Value	Description
0	No hardware or software handshaking
1	RTS/CTS hardware handshaking
2	DTR/DSR hardware handshaking
3	XON/XOFF software handshaking

2.2.10 Port[x].RTermCnt

0 to 12 characters

This parameter specifies the number of termination characters that define the end of a received message. Use this parameter if you set the **PORT[X].TYPE** Port[x].Type (page 48) value to 1, 3, 5, 7, 9, 11, 13 or 15.

2.2.11 Port[x].RTermChar

This array of up to 12 characters specifies the termination characters that define the end of a received message. Each character occupies one position in the array. The number of termination characters you enter here must match the value in the **PORT[X].RTERMCNT** tag.

2.2.12 Port[x].RPacketLen

This parameter specifies the number of bytes of data to receive on the port before considering an incoming message complete and transferring the data to the processor. Use this parameter if you set the **PORT[X].TYPE**Port[x].Type (page 48) value to 8, 9, 10, 11, 12, 13, 14, or 15.

2.2.13 Port[x].RTimeout

This parameter specifies the number milliseconds to wait after receiving the first character on the port before considering an incoming message complete and automatically sending the data to the processor. Use this parameter if you set the **PORT[X].TYPE** Port[x].Type (page 48) value to 2, 3, 6, 7, 10, 11, 14, or 15.

2.2.14 Port[x].RDelay

This parameter specifies the maximum number milliseconds to wait between each character received on the port to see if more characters are coming before considering an incoming message complete and automatically sending the data to the processor. In practice, this can be thought of as a period of time in which no characters are received after receiving a string of characters. Use this parameter if you set the **PORT[X].TYPE** Port[x].Type (page 48) value to 4, 5, 6, 7, 10, 11, 14, or 15.

2.2.15 Port[x].WTermCnt

Not used in current release of product.

2.2.16 Port[x].WTermChar

Not used in current release of product.

2.2.17 Port[x].WPacketLen

Not used in current release of product.

2.2.18 Port[x].WTimeout

This parameter specifies the timeout period to transmit a message out the port. A message must be transmitted out the port within the specified timeout period. Message transmission will be aborted if the timeout is exceeded. Use this parameter if you set the **PORT[X].TYPE** Port[x].Type (page 48) to 2, 3, 6, 7, 10, 11, 14, or 15.

Note: If this parameter is left at zero, then the Port Configuration Error Word will show a value of 0400. In order to clear this error, this parameter has to be set to any non-zero value.

2.2.19 Port[x].Spare

Not used in current release of product.

2.2.20 Port[x].WMinDelay

This parameter specifies the minimum number of milliseconds to delay before transmitting a message out the port. This pre-send delay is applied before the RTS On time. This may be required when communicating with slow devices. Use this parameter if you set the **PORT[X].TYPE** Port[x].Type (page 48) to 8, 9, 10, 11, 12, 13, 14, or 15.

2.3 Changing Parameters During Operation

When you change the configuration parameters in RSLogix, you must reboot or cycle power to the module off and on before the new configuration takes effect. You may also change the value of the GSC.ColdBoot or GSC.WarmBoot controller tags from 0 to 1 to reboot the module from logic.

3 Diagnostics and Troubleshooting

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

- LED status indicators on the front of the module provide information on the module's status.
- Status data contained in the module can be viewed in *ProSoft Configuration Builder* through the Ethernet port.
- Status data values are transferred from the module to the processor.

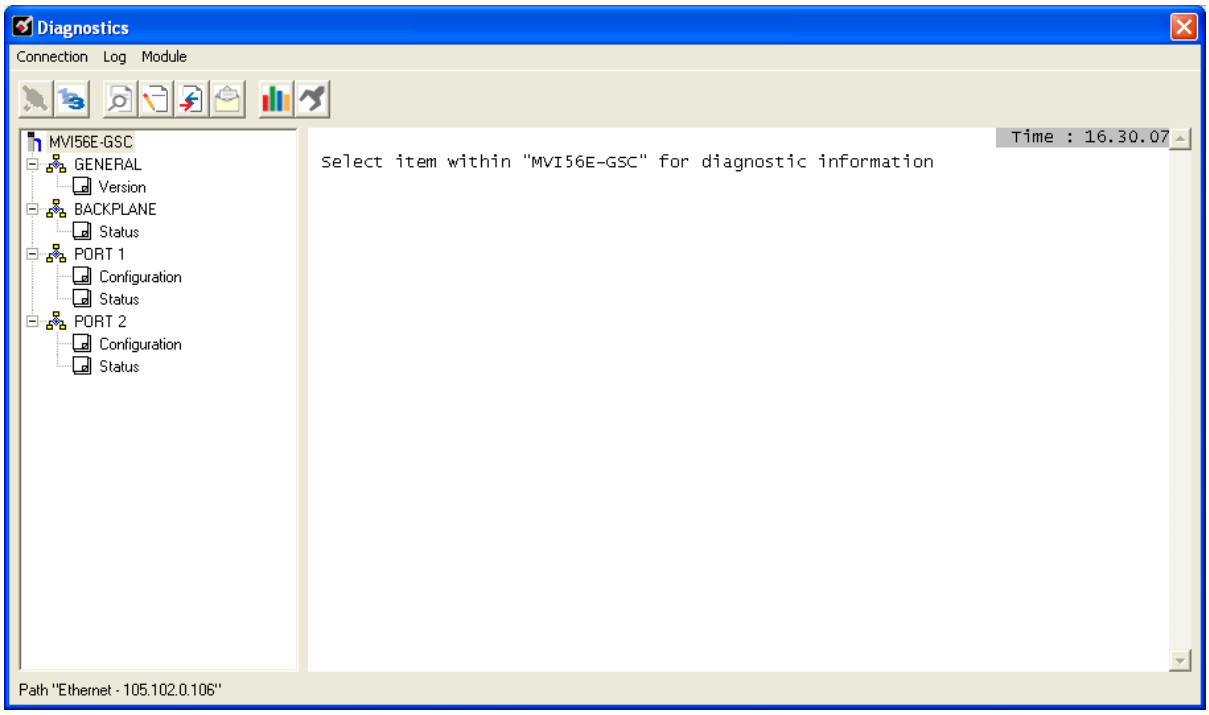
3.1 Reading Status Data from the Module

The MVI56E-GSC module returns three separate status data areas to the ControlLogix processor in each read block. This data is transferred to the ControlLogix processor continuously with each read block. For a complete listing of the status data object, refer to the Module Configuration section of this manual.

This guide also includes example ladder logic showing how to extract this data from the input image and place it in the module's Controller Logic Tag. Refer to Diagnostics and Troubleshooting for a discussion of the features available through the use of this utility.

3.2 The Diagnostics Menu

The *Diagnostics* menu, available through the Ethernet configuration port for this module, is arranged as a tree structure, with the *Main* menu at the top of the tree, and one or more submenus for each menu command. The first menu you see when you connect to the module is the *Main* menu.

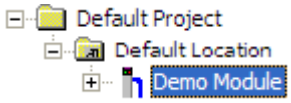


3.2.1 Using the Diagnostics Menu in ProSoft Configuration Builder

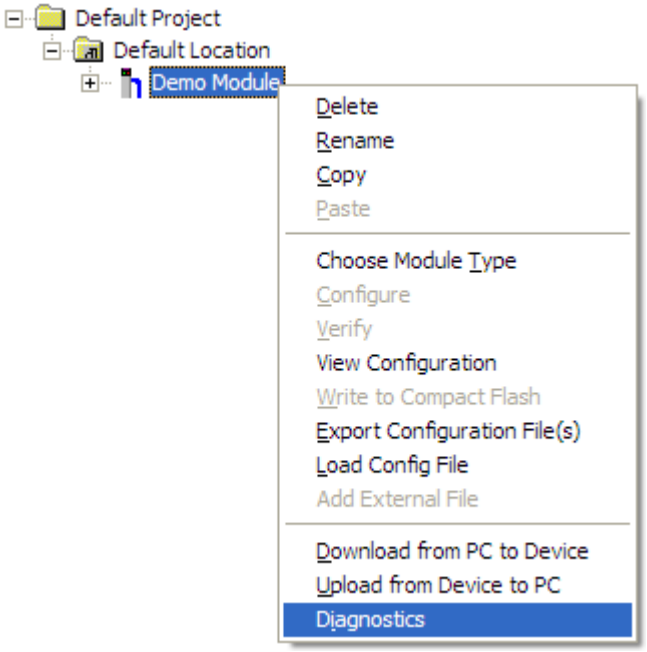
Tip: You can have a ProSoft Configuration Builder *Diagnostics* window open for more than one module at a time.

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

1 In *ProSoft Configuration Builder*, select the module, and then click the right mouse button to open a shortcut menu.



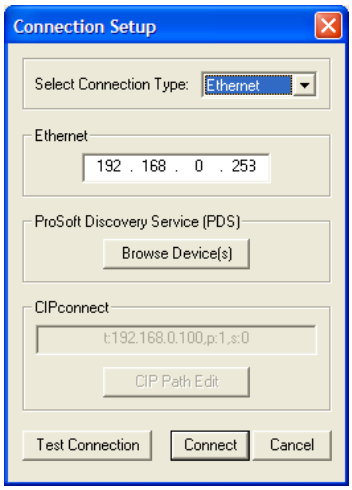
2 On the shortcut menu, choose **DIAGNOSTICS.**



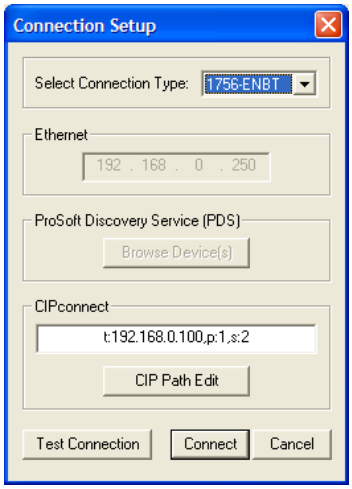
3 In the *Diagnostics* window, click the **SET UP CONNECTION** button to browse for the module's IP address.



4 In the *Connection Setup* dialog box, click the **TEST CONNECTION** button to verify that the module is accessible with the current settings.



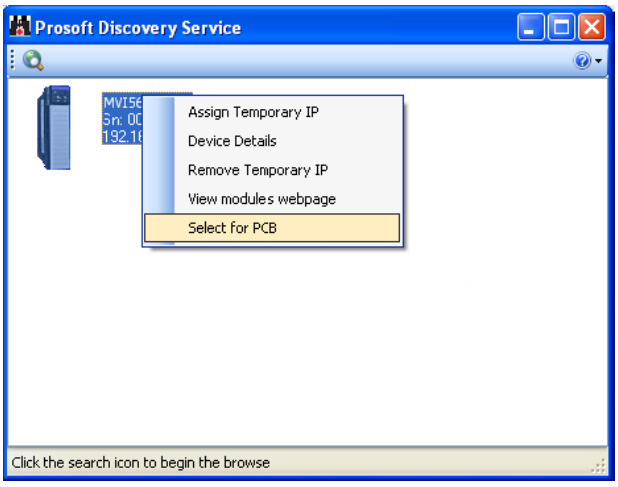
You can also use CIPconnect® to connect to the module through a 1756-ENBT card. Refer to Using CIPconnect® to Connect to the Module (page 22) for information on how to construct a CIP path.



5 If the *Test Connection* is successful, click **CONNECT**.

If *PCB* is unable to connect to the module:

1 Click the **BROWSE DEVICE(S)** button to open the *ProSoft Discovery Service*. Select the module, then right-click and choose **SELECT FOR PCB**.



2 Close *ProSoft Discovery Service*, and click the **CONNECT** button again.

3 If these troubleshooting steps fail, verify that the Ethernet cable is connected properly between your computer and the module, either through a hub or switch (using the grey cable) or directly between your computer and the module (using the red cable).

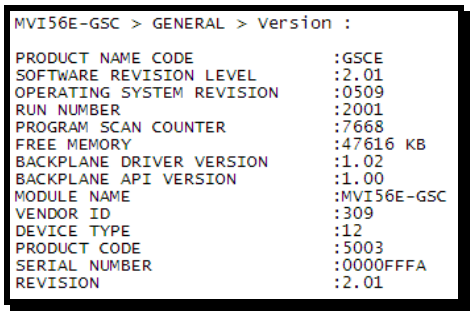
If you are still not able to establish a connection, contact ProSoft Technology for assistance.

3.3 Monitoring Module Information

Use the *MODULE* menu to view configuration and hardware information for the MVI56E-GSC module's backplane and Ethernet application port.

3.3.1 Version Menu

Use the **VERSION** menu to view module hardware and firmware information.



3.3.2 Config

Use the *Configuration* menu to view backplane configuration settings for the MVI56E-GSC module.

The information on this menu corresponds with the configuration information in the *Module* settings in *ProSoft Configuration Builder*.

3.3.3 NIC Status

Use the *NIC Status* (Network Interface Card) menu to view configuration and status information for the MVI56E-GSC module's Ethernet application port.

The information on this menu is useful for troubleshooting Ethernet network connectivity problems.

3.4 Monitoring Backplane Information

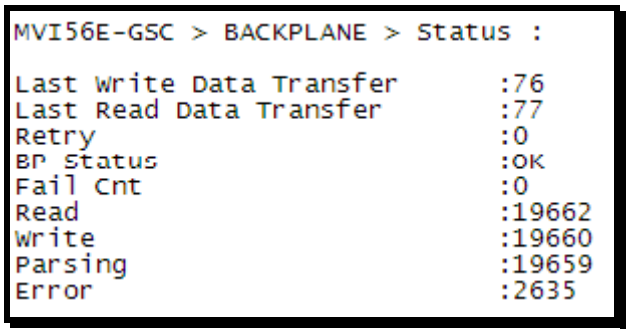
Use the *BACKPLANE* menu to view the backplane status information for the MVI56E-GSC module.

3.4.1 Backplane Status Menu

Click **STATUS** to view current backplane status, including

- Number of retries
- Backplane Status
- Fail Count
- Number of words read
- Number of words written
- Number of words parsed
- Error count

During normal operation, the Read, Write, and Parsing values should increment continuously, while the error value should not increment.



The status values on this menu correspond with the members of the MVI56E-GSC Status object.

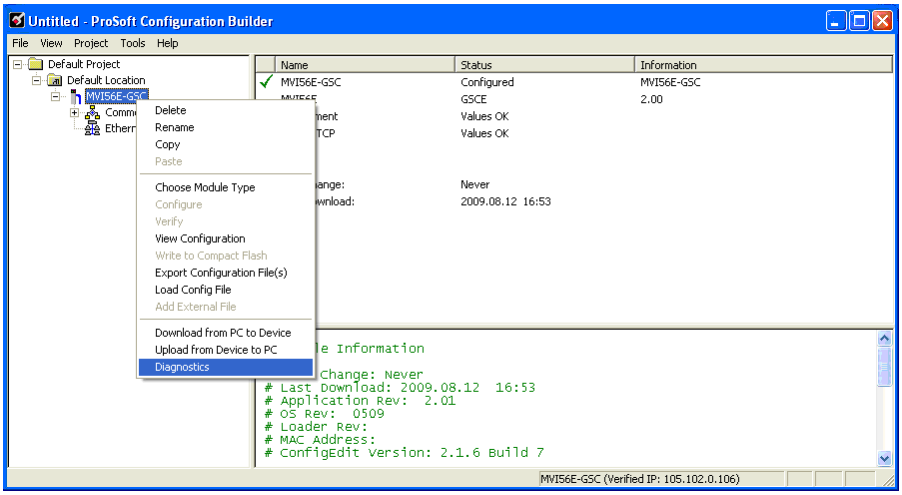
3.5 Data Analyzer

The Data Analyzer mode allows you to view all bytes of data transferred on each port. Both the transmitted and received data bytes are displayed. Use of this feature is limited without a thorough understanding of the protocol.

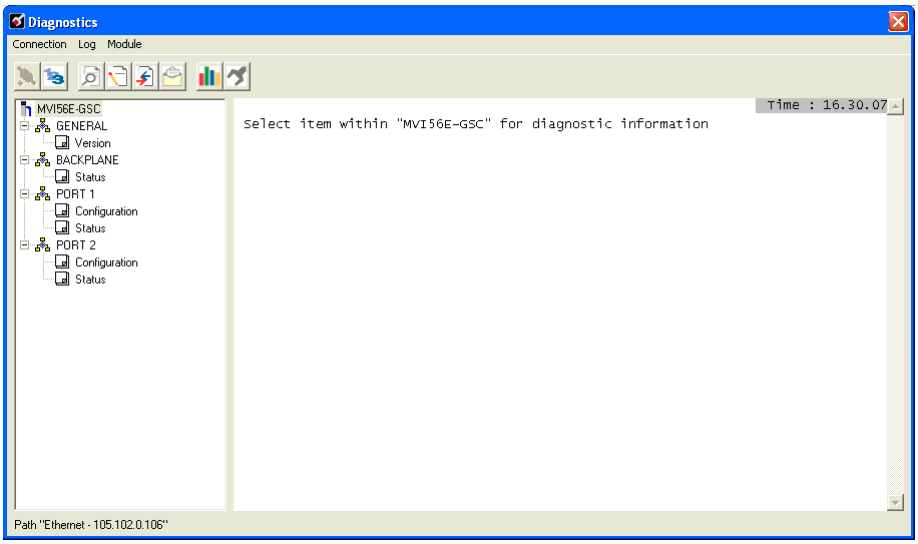
3.5.1 Starting the Data Analyzer

Turn on the *Data Analyzer* feature so you can monitor the data exchanged between the processor and the module. Start ProSoft Configuration Builder (PCB) and perform the following sequence.

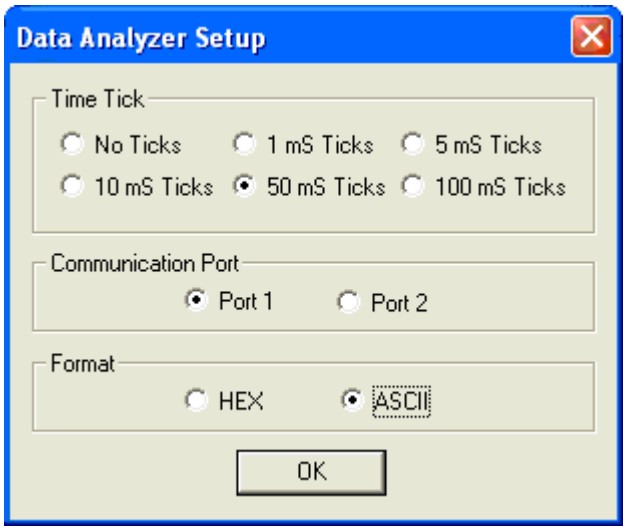
1 Start by right-clicking on the module name and left-clicking **DIAGNOSTICS** on the context menu.



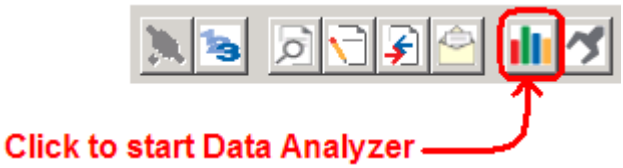
This opens the main Diagnostics screen.



2 Left-click on the button to setup the *Data Analyzer* feature as shown.

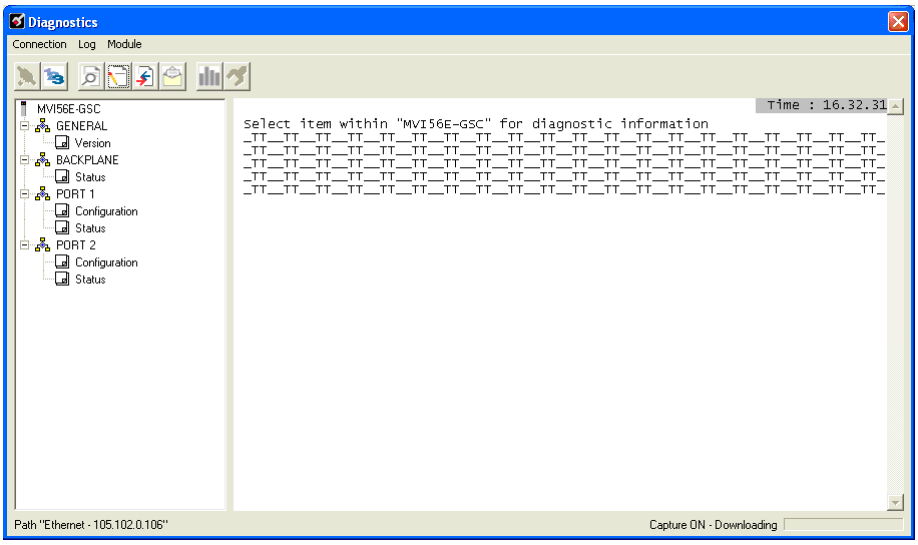


3 Left-click on the button to start the *Data Analyzer* feature.



The following illustration shows an example of the Data Analyzer output with no traffic being passed on the serial network. With no transmitted data to display, the Data Analyzer shows only the "_TT_" timing markers, called "Time Ticks". The time between Time Ticks is adjustable and can be set in the Data Analyzer Setup dialog box, as previously seen.

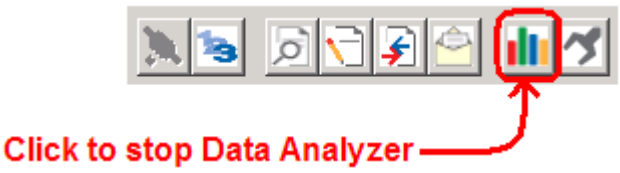
In this example the Time Ticks have been set to 50 milliseconds. During actual transmission of data, the Time Ticks will appear interspersed within the data bytes and indicate the configured time interval. This information can be valuable for troubleshooting certain kinds of communication problems.



The Data Analyzer can display the following special characters.

Character	Definition
[ ]	Data enclosed in [square brackets] represent data received on the port.
< >	Data enclosed in <angle brackets> represent data transmitted on the port.
<R+>	These characters are inserted when the Ready To Send (RTS) line is driven high on the port, just before data transmission begins.
<R->	These characters are inserted when the RTS line is dropped low on the port, indicating the end of transmission.
<CS>	These characters are displayed when the Clear to Send (CTS) line is recognized high. These characters will appear only when hardware handshaking is enabled in the port configuration (Use CTS = 1). Most applications do not require the use of hardware handshaking.
_TT_	These characters are displayed when the "Time Tick" is set to any value other than "No Ticks". Time Ticks will be displayed at the interval selected in the Data Analyzer Setup dialog box and will be interspersed with any data received on the port. ASCII Characters or Decimal byte values shown between any two Time Ticks will indicate the characters received on the port during that time interval.

3.5.2 Stopping the Data Analyzer



Important: When in analyzer mode, program execution will slow down. Only use this tool during a troubleshooting session. Before disconnecting from the Config/Debug port, please stop the data analyzer. This action will allow the module to resume its normal high speed operating mode.

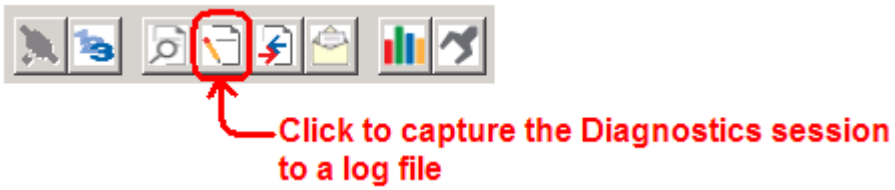
3.5.3 Data Analyzer Tips

For most applications, HEX is the best format to view the data, and this does include ASCII based messages (because some characters will not display in the Diagnostics window in ASCII mode, and, by capturing the data in HEX, you can figure out what the corresponding ASCII characters are supposed to be).

The Time Tick value is a timing mark. The module will print a _TT_ every so many milliseconds. The Time Tick setting is adjustable in the Data Analyzer Setup dialog box. Usually 10 milliseconds works best for most applications.

To save a capture file of your Diagnostics session:

1 After you have selected the Port, Format, and Tick, you are now ready to start a capture of this data.



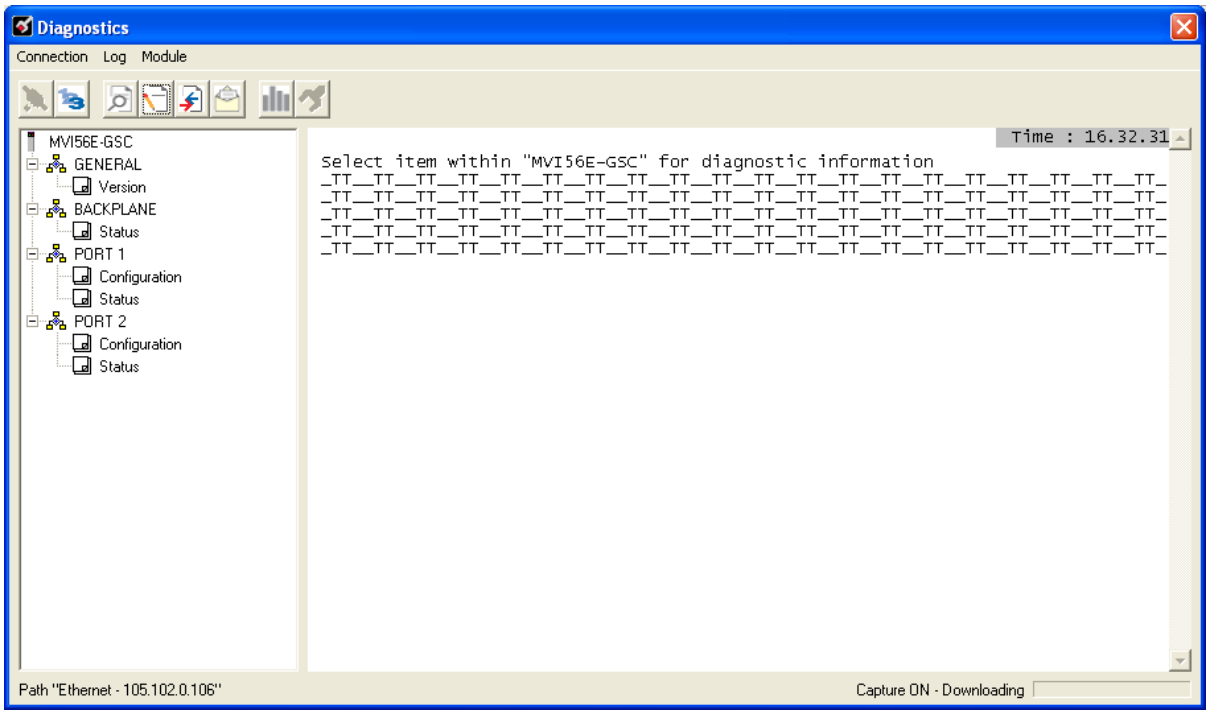
2 When you have captured the data you want to save, click again to stop capturing data.



You have now captured and saved the data to a file on your PC. This file can now be used in analyzing the communication traffic on the line and assist in determining communication errors. The log file name is PCB-Log.txt, located in the root directory of your hard drive (normally Drive C).

Once you have everything that shows up on the Diagnostics screen being logged to a file called PCB-Log.txt, you can email this file to ProSoft Technical Support for help with the analysis of communication problems.

To begin the display of communications data, start the Data Analyzer. When the Data Analyzer is running, you should see something like this.



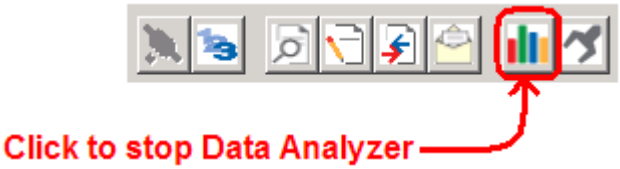
The <R+> means that the module is transitioning the communications line to a transmit state.

All characters shown in <> brackets are characters being sent out by the module.

The <R-> shows when the module is done transmitting data and is now ready to receive information.

All characters shown in [ ] braces are information being received from another device by the module.

After capturing traffic for a minute or two, stop the Data Analyzer.



3.6 Scrolling LED Status Indicators

The scrolling LED display indicates the module's operating status as follows:

3.6.1 Initialization Messages

Message	Meaning
"Boot"	Module is initializing its operating system
"Waiting for Processor Connection..."	Module is not able to establish a backplane connection with the ControlLogix PAC. Possible causes: ▪ Ladder logic or AOI is not loaded on processor ▪ Module is located in a different slot than the one configured in the ladder logic/AOI ▪ Module I/O properties are invalid ▪ Processor is not in RUN or REM RUN mode
"INIT"	Module is beginning its firmware initialization
"Ladder Logic Configuration Required"	Module is waiting for valid module configuration data from ladder logic. Check that module configuration parameters are set to valid values in the ladder logic controller tags
"UPDATING"	Module is performing a firmware update, initiated from the web page.
"STOP"	Module firmware is stopping after receiving a "Cold Boot" request, which forces an automatic re-start, same as from power-up
"MVI-56E-GSC <Version#> Last Config: <LCfgDate> Config P1: <TermType> <BaudRate>, <Parity>, <DataBits>, <StopBits>, <RS_Type>, P2: <TermType> <BaudRate>, <Parity>, <DataBits>, <StopBits>, <RS_Type>"	After power up and every reconfiguration reboot, the module will display the configuration of both ports. This message will scroll through once and not be repeated until the next reboot. The message contains: ▪ <Version#> Firmware revision number, as in "V2.01" ▪ <LCfgDate> Date of last configuration change (reboot) ▪ <TermType>: Termination Type; options are: STREAM - Streaming TERM - Terminating Characters DELAY - Intercharacter Delay PACKET - Packet Length MSGTO - Message Timeout ▪ <BaudRate>: 115200 / 57600 / 38400 / 19200 / 9600/ 4800 / 2400 / 1200 / 600 / 300 ▪ <Parity>: None / Even / Odd ▪ <DataBits>: 7 / 8 ▪ <StopBits>: 1 / 2 ▪ <RS_Type>: RS-232 / RS-422 / RS-485

3.6.2 Initialization Error Messages

During initialization, if the module detects a unrecoverable fault that will prevent communication, an error message will be displayed. If any of these errors persist after repeated reboot attempts, contact ProSoft Technology Technical Support for further assistance.

Message	Meaning
"E003"	Startup sequence failed, module failed to initialize
"E005"	Displayed during startup if there is an error opening application serial Port 1(P1)
"E006"	Displayed during startup if there is an error opening application serial Port 2 (P2)
"E008"	Displayed during startup if either of the two application serial ports fails to open after a second attempt
"E009"	Displayed during startup if there is an error setting up the signal handlers)

3.6.3 Operation Message

After the initialization step, the following message pattern will be repeated on the scrolling LED display.

```
<Backplane Status> <IP Address> <Backplane Status> <Port1 Status> <Port2 Status>
```

Message Component	Meaning
<Backplane Status>	"OK": Module is communicating with processor
<IP Address>	Module IP address
<Port1 Status>	"Enabled": Port is enabled, sending and receiving possible
<Port2 Status>	"Disabled": Port is disabled, no sending or receiving possible

3.7 Ethernet LED Indicators

The Ethernet LEDs indicate the module's Ethernet port status as follows:

LED	State	Description
10/100	Off	No activity on the Ethernet port.
	Green Flash	The Ethernet port is actively transmitting or receiving data.
LINK/ACT	Off	No physical network connection is detected. No Ethernet communication is possible. Check wiring and cables.
	Green Solid	Physical network connection detected. This LED must be On solid for Ethernet communication to be possible.

3.8 Non-Scrolling LED Status Indicators

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

LED Label	Color	Status	Indication
APP	Red or Green	OFF	The module is not receiving adequate power or is not securely plugged into the rack. May also be OFF during configuration download.
		GREEN	The MVI56E-GSC is working normally.
		RED	The most common cause is that the module has detected a communication error during operation of an application port. The following conditions may also cause a RED LED: ▪ The firmware is initializing during startup ▪ The firmware detects an on-board hardware problem during startup ▪ Failure of application port hardware during startup ▪ The module is shutting down ▪ The module is rebooting due to a ColdBoot or WarmBoot request from the ladder logic or Debug Menu
OK	Red or Green	OFF	The module is not receiving adequate power or is not securely plugged into the rack.
		GREEN	The module is operating normally.
		RED	The module has detected an internal error or is being initialized. If the LED remains RED for over 10 seconds, the module is not working. Remove it from the rack and re-insert it to restart its internal program.
ERR	Red		Not used.

3.9 ControlLogix Processor Not in RUN or REM RUN

Whenever the module detects that the processor has gone out of the RUN mode (that is, the processors faults or is set to PGM (Program mode), the application ports can be shut down as prescribed in the user configuration. When the processor is returned to a running state, the module will resume communication on the serial networks.

3.10 Clearing a Fault Condition

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

To clear the condition, follow these steps:

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

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

3.11 Troubleshooting

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

3.11.1 Processor Errors

Problem Description	Steps to take
Processor Fault	Verify that the module is plugged into the slot that has been configured for the module in the I/O Configuration of RSLogix. Verify that the slot location in the rack has been configured correctly in the ladder logic.
Processor I/O LED flashes	This indicates a problem with backplane communications. A problem could exist between the processor and any installed I/O module, not just the MVI56E-GSC. Verify that all modules in the rack are correctly configured in the ladder logic.

3.11.2 Module Errors

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

4 Reference

4.1 Product Specifications

The Enhanced Generic ASCII Serial Communication Modules allow Rockwell Automation® ControlLogix® Programmable Automation Controllers (PACs) to easily interface with serial devices using non-specific ASCII character text string or byte value serial communication protocols.

MVI56E-GSC enhancements include local and remote diagnostics through the module's Ethernet port, and CIPconnect® technology for bridging though Rockwell Automation ControlNet™and EtherNet/IP™ networks.

The MVI56E-GSC module is a fast and easy way to add two fully configurable serial communication ports to the ControlLogix platform, eliminating the need to use the front port of the processor, or consume valuable processor time sending and receiving serial messages.

The module is a single-slot, backplane-compatible solution. Each port is capable of sending and receiving large ASCII character strings or byte streams of up to 4096 characters or bytes. Many different serial communication devices can be integrated into the ControlLogix platform by building upon the sample ladder logic provided for this module.

4.1.1 General Specifications

- Backward-compatible with previous MVI56-GSC version
- Single Slot - 1756 ControlLogix® backplane compatible
- 10/100 MB Ethernet port for network configuration and diagnostics with Auto Cable Crossover Detection
- User-definable module data memory mapping of up to 10,000 16-bit registers
- CIPconnect®-enabled network diagnostics and monitoring using ControlLogix 1756-ENxT modules and EtherNet/IP® pass-thru communications
- Sample Ladder Logic or Add-On Instruction (AOI) used for data transfers between module and processor and for module configuration
- 4-character, scrolling, alphanumeric LED display of status and diagnostic data in plain English
- ProSoft Discovery Service (PDS) software finds the module on the network and assigns a temporary IP address to facilitate module access

4.1.2 Functional Specifications

- The MVI56E-GSC and MVI56E-GSCXT are functionally identical. The MVI56E-GSC is for normal process and control environments. The MVI56E-GSCXT is conformal coated for extra protection in harsh or caustic environments and operates in extreme high or low temperature environments.
- Both modules transfer data in the largest possible I/O image block sizes, which optimizes data through-put and update time.
- Both modules appear to the ControlLogix processor as input/output (I/O) modules, rather than communication modules.
- Two ASCII serial communication ports:
  - Can independently transmit and/or receive ASCII character strings and serial byte streams
  - Each port is individually configurable
- Receive ASCII strings or byte streams up to 4096 characters in length
- Received packet termination types:
  - Stream mode (no packet termination)
  - Receipt of specified character or characters
  - Message length timeout
  - Intercharacter spacing timeout
  - Packet size limit (number of received characters/bytes)
- Module configuration and communication configuration data is transferred to the module via predefined sample ladder logic
- Module error and status conditions returned to processor for diagnostics
  - Module status
  - Port error status word (bit mapped)
  - Port receive state
  - Port receive character count
  - Port receive block count
  - Port transmit state
  - Port transmit character count
  - Port transmit block count

4.1.3 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) - MVI56E-GSC -25°C to 70°C (-13°F to 158°F) - MVI56E-GSCXT
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 to 150 Hz
Relative Humidity	5% to 95% (without condensation)
LED Indicators	Application Status (APP) Module Status (OK)
4-Character, Scrolling, Alpha-Numeric LED Display	Shows Module, Version, IP, Port Status, P1 and P2 Settings, and Error Information
Debug/Configuration Ethernet port (E1 - Config)	
Ethernet Port	10/100 Base-T, RJ45 Connector, for CAT5 cable Link and Activity LED indicators Auto-crossover cable detection
Serial Application ports (P1 & P2)	
Software configurable communication parameters	Baud rate: 110 baud to 115.2 kbps RS-232, RS-485, and RS-422 Parity: none, odd or even Data bits: 5, 6, 7, or 8 Stop bits: 1 or 2 RTS on/off delay: 0 to 65535 milliseconds Full hardware handshaking control (optional) Radio and modem support
App Ports (P1, P2)	RJ45 (DB-9M with supplied adapter cable) Configurable RS-232 hardware handshaking 500V Optical isolation from backplane RS-232, RS-422, RS-485 jumper-select, per port RX (Receive) and TX (Transmit) LEDs, each port
Shipped with Unit	RJ45 to DB-9M cables for each serial port

4.2 General Concepts

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

4.2.1 Backplane Data Transfer

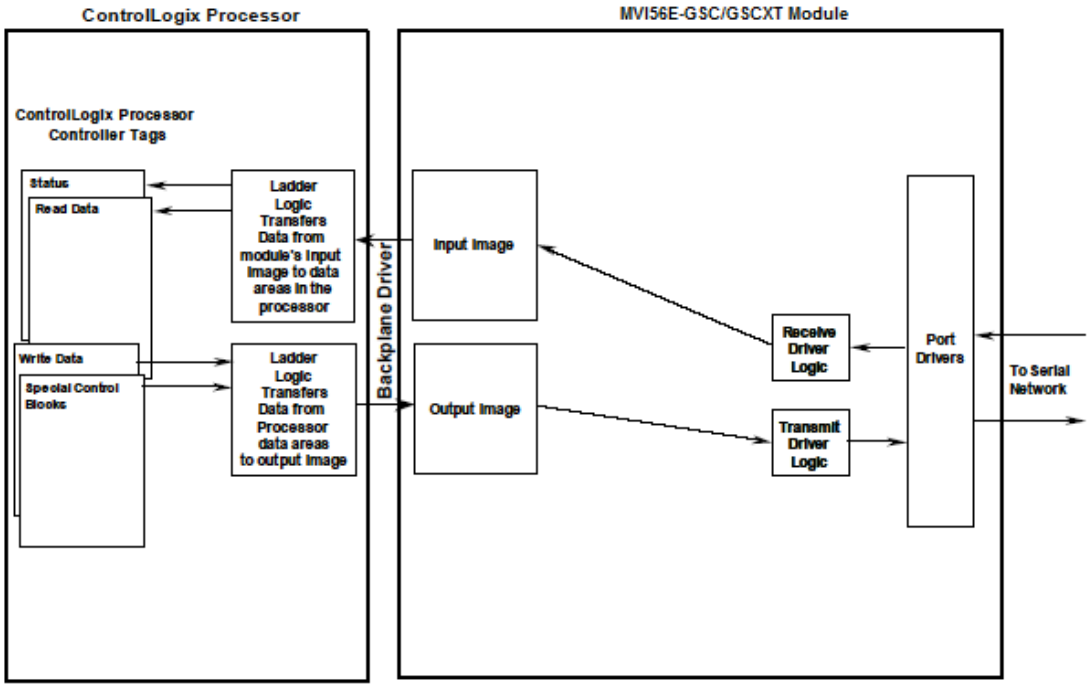
The MVI56E-GSC module communicates directly with the ControlLogix either through the backplane, when in a local chassis, or over a process I/O network (like ControlNet or EtherNet/IP) when in a remote chassis. Data travels between the module and the ControlLogix processor across the backplane or I/O network using the module's input and output images. The I/O image update frequency is determined by the scheduled Requested Packet Interval (RPI) time set by the user for the module, as well as the overall communication load on the module. Typical updates are in the range of 1 to 10 milliseconds.

Data received on the application ports is placed in the module's input image. This data is processed by the logic in the ControlLogix processor. The input image size for the module is set to 500 bytes. This large data area permits fast throughput of data between the module and the processor.

The processor inserts data in the module's output image to transfer that data to the module. The module's program extracts the data and transmits it out the communication port or ports. The output image size for the module is set to 496 bytes. This large data area permits fast throughput of data from the processor to the module.

The following illustration shows the data transfer method used to move data between the ControlLogix processor, the MVI56E-GSC module and the serial devices.

MVI56E-GSC Module



As shown in the illustration above, all data transferred between the module and the processor over the backplane is through the input and output images. Ladder logic must be written in the ControlLogix processor to interface the input and output image data with data defined in the Controller Tags. Your ladder logic must handle and interpret all data received on the application ports and transferred into the input image. You must also construct messages to be transferred out of the application ports by building the messages in the appropriate controller tags for transfer to the output image of the module.

4.2.2 Data Flow between MVI56E-GSC Module and ControlLogix Processor

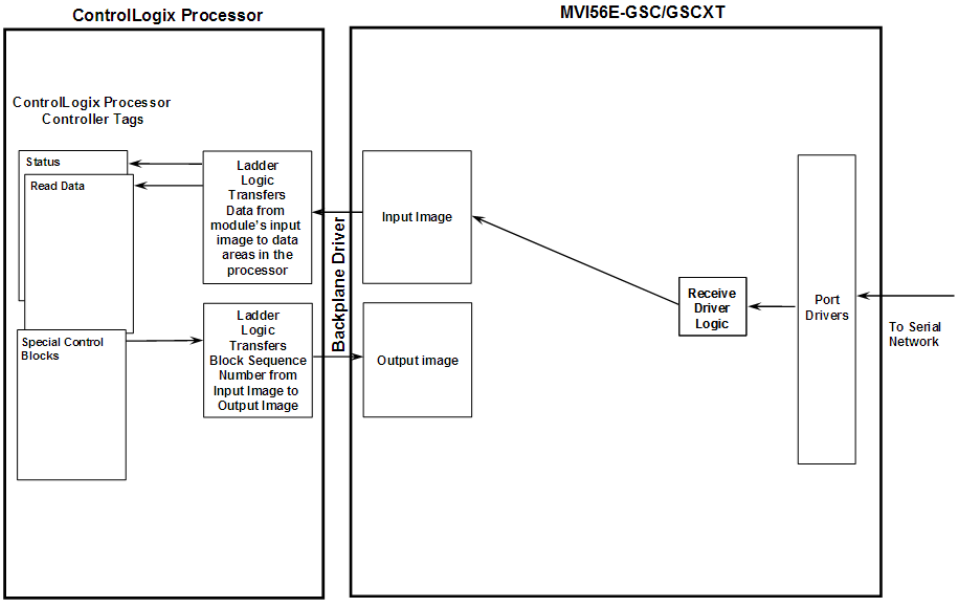
The following topics describe the flow of data between the two pieces of hardware (ControlLogix processor and MVI56E-GSC module) and other devices attached to the application ports. Each application port on the module is configured independently to interface with serial communication devices. The sections below show the three possible types of communication devices that can be attached to the application ports: write-only, read-only and read-write.

Write-Only Devices

Write-only devices are those that only send data to the module. An example of this type of device is a barcode reader which has been configured to only send data and is not expecting to receive data.

In this situation, the application port on the MVI56E-GSC module will never have to transmit data. All data received from the barcode reader will be passed from the module to the ControlLogix processor through the module's input image. Ladder logic in the processor must handle the data received from the module. The output image on the module will only be used to inform the module when the input image has been processed. This is accomplished by copying the Byte 499 in the input image to Byte 0 of the output image.

The data flow diagram for a write-only device is shown below:

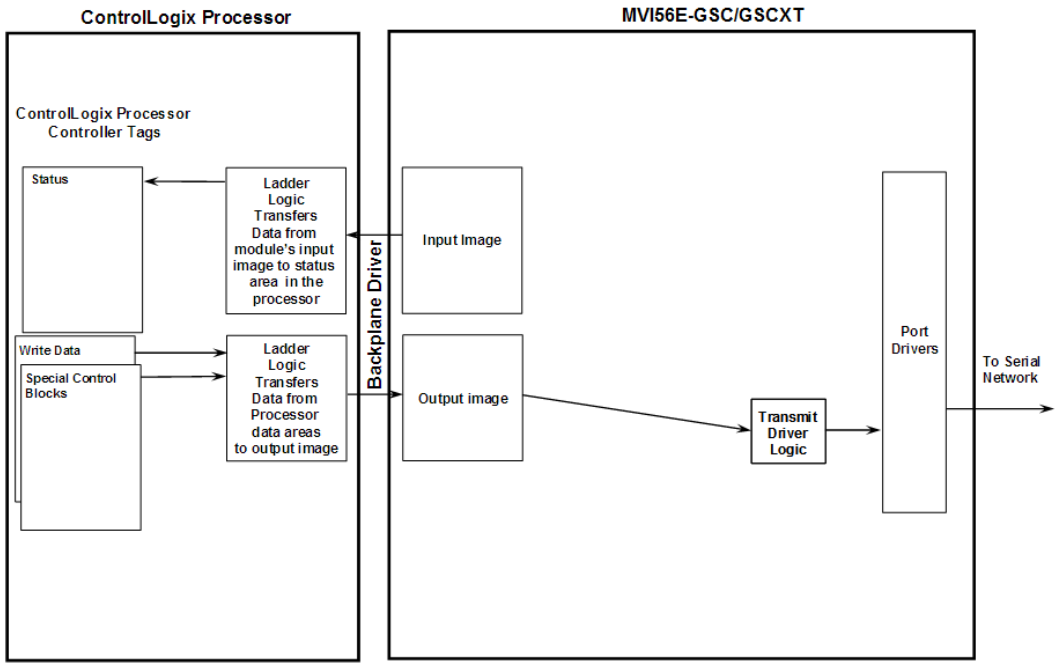


Read-Only Devices

Read-Only devices are those that only receive data from the module. An example of this type of device is a printer. The printer will generate output or be controlled based on the data it receives on its communication port.

Ladder logic is used to construct the write blocks to be sent to the module. When the module receives a new write block containing data, it will transmit the data out the port. The Block Sequence Number used in the write block should be the one received on the last read block.

The data flow diagram for a read-only device is shown below:

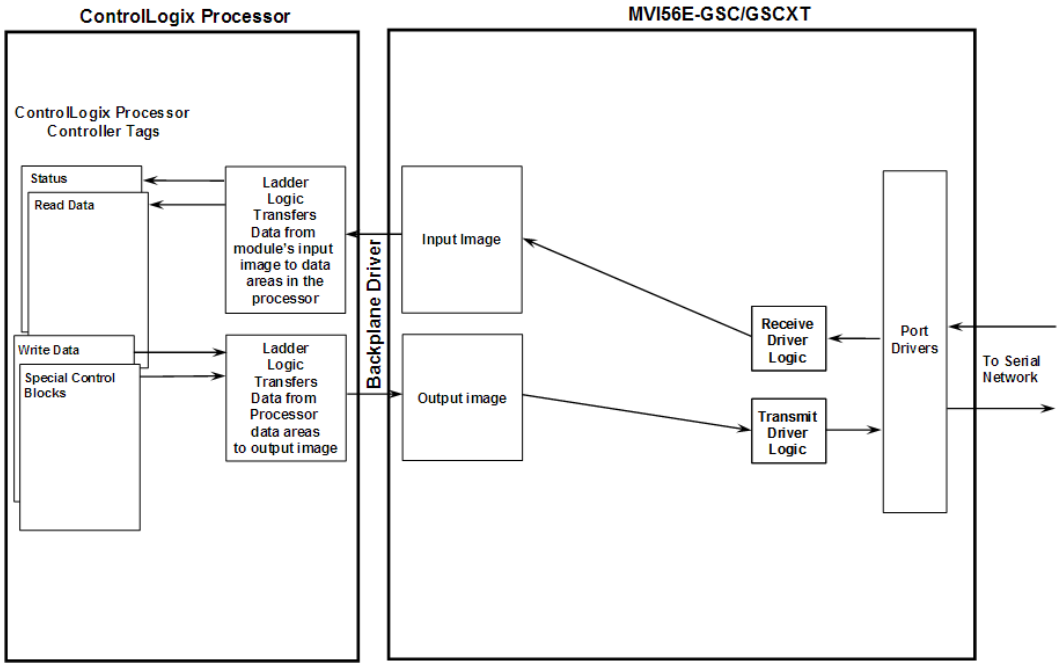

ControlLogix Processor
MVI56E-GSC/GSCXT
ControlLogix Processor
Controller Tags
Status
Ladder
Logic
Transfers
Data from
module's input
image to status
area in the
processor
Write Data
Special Control
Blocks
Ladder
Logic
Transfers
Data from
Processor
data areas
to output image
Backplane Driver
Input Image
Output image
Transmit
Driver
Logic
Port
Drivers
To Serial
Network

Read-Write Devices

Read-write devices are those that both send and receive data. An example of this type of device is a computer terminal. A terminal will send data entered on the keyboard out its serial port and display any data received on its port on the monitor.

All data received from the terminal keyboard will be routed to the ControlLogix processor through the MVI56E-GSC's input image. Data to be written to the terminal screen will be sent to the module using the output image. The module will send new data from the output image out the application port to the terminal. The example shipped with the module can be used to interface with a terminal to echo back all characters received.

The data flow diagram for a read-write device is shown below:



4.2.3 Termination of Received Data

When data is received on either of the application ports, you must define in the configuration when this data will be transferred to the ControlLogix processor. Within the module, this is known as the termination type for port. When the termination condition is met, the data will be sent from the port's receive buffer (data area of 4096 bytes) to the processor using the input image. This termination type is set using a bit-mapped value, entered into the **GSC.CONFIG.PORTX.TYPE** field of the configuration controller tags.

Termination Mode Selection Chart

Decimal Value of Bits Selected	0 = No Bits Set **(S)** Streaming Mode Selected	1 = Bit 0 Set **(T)** Terminating Characters Selected	2 = Bit 1 Set **(M)** Message Timeout Selected	4 = Bit 2 Set **(D)** Intercharacter Delay Selected	8 = Bit 3 Set **(P)** Packet Length Selected	Port Types Selected by Bitmap
0	**0**					**S**
1		**1**				**T**
2			**2**			**M**
3		**1**	**2**			**M, T**
4				**4**		**D**
5		**1**		**4**		**D, T**
6			**2**	**4**		**D, M**
7		**1**	**2**	**4**		**D, M, T**
8					**8**	**P**
9		**1**			**8**	**P, T**
10			**2**		**8**	**P, M**
11		**1**	**2**		**8**	**P, M, T**
12				**4**	**8**	**P, D**
13		**1**		**4**	**8**	**P, D, T**
14			**2**	**4**	**8**	**P ,D, M**
15		**1**	**2**	**4**	**8**	**P, D, M, T**

Note: For each termination type, you must also enter the corresponding parameters in the following configuration tags:

GSC.PortX.RtermCnt
GSC.PortX.RtermCnt
GSC.PortX.RpacketLen
GSC.PortX.Rtimeout
GSC.PortX.Rdelay
GSC.PortX.WtermCnt
GSC.PortX.WtermChar
GSC.PortX.WpacketLen
GSC.PortX.Wtimeout
GSC.PortX.Wdelay
GSC.PortX.WMinDelay

Termination Type Field

If none of the bits are set (Type=0), the port will be configured for stream mode. Any characters received on the port are immediately sent to the processor. The processor must buffer and assemble a packet of information if this mode is selected. If the data can be handled by the processor in this mode and it is appropriate for your application, this is the fastest method of communication between the device and the processor. However, stream mode tends to be processor-intensive and does not work well in larger applications due to the potential for loss of data. For larger applications, consider using one of the other modes that allows the module to buffer incoming data from complete messages before transferring it to the processor.

Any combination of termination types/bit settings is acceptable to the module and should be set to match the device on the specific port. An example of each termination type is given below.

Termination character(s) used

Termination character(s) used

Settings:
Count = 1 (RTermCnt=1)
Termination on 0x0d (carriage return character) (RTermChar = 0d 00 00 00 …)

Data Received on port:

A B C 0x0d D E

Comment:
The characters "ABC" will be sent along with the 0x0d character to the controller after the 0x0d character is received. The characters "DE" will not be sent until the 0x0d character is received.

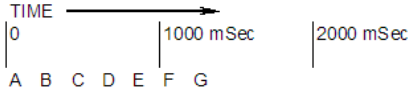
Message timeout used

Message timeout used

Settings:

Message timeout = 1000 mSec (Rtimeout=1000)

Data Received on port:



Comment:

After the 'A' character is received on the port, the message timeout is started. The characters "ABCDE" will be sent to the controller in one block. The characters "FG" will follow in the second block one second later.

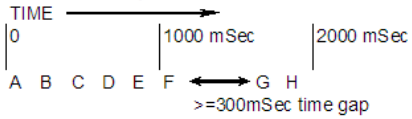
Intercharacter delay timeout used

Intercharacter delay timeout used

Settings:

Intercharacter delay timeout = 300 mSec (Rdelay=300)

Data Received on port:



Comment:

After each character is received, the intercharacter delay timer is reset. The characters "ABCDEF" will be sent to the controller in one block because the delay timer expires. The characters "GH" will follow in the second block when the next time gap is recognized.

Packet size limit used

Packet size limit used

Settings:

Packet size = 4 (RPacketLen=4)

Data Received on port:

A B C D E F G H I J

Comment:

The first block sent to the controller will contain the characters "ABCD", and the second block will contain the characters "EFGH". The characters "IJ" will not be sent until two more characters are received on the port.

Termination Mode Selection Chart

Decimal Value of Bits Selected	0 = No Bits Set **(S)** Streaming Mode Selected	1 = Bit 0 Set **(T)** Terminating Characters Selected	2 = Bit 1 Set **(M)** Message Timeout Selected	4 = Bit 2 Set **(D)** Intercharacter Delay Selected	8 = Bit 3 Set **(P)** Packet Length Selected	Port Types Selected by Bitmap
0	**0**					**S**
1		**1**				**T**
2			**2**			**M**
3		**1**	**2**			**M, T**
4				**4**		**D**
5		**1**		**4**		**D, T**
6			**2**	**4**		**D, M**
7		**1**	**2**	**4**		**D, M, T**
8					**8**	**P**
9		**1**			**8**	**P, T**
10			**2**		**8**	**P, M**
11		**1**	**2**		**8**	**P, M, T**
12				**4**	**8**	**P, D**
13		**1**		**4**	**8**	**P, D, T**
14			**2**	**4**	**8**	**P ,D, M**
15		**1**	**2**	**4**	**8**	**P, D, M, T**

Note: For each termination type, you must also enter values in the following parameter configuration tags:

For Termination Type	Receive Parameters to Set
Terminating Characters	GSC.PortX.RtermCnt GSC.PortX.RTermChar
Message Timeout	GSC.PortX.RTimeout
Intercharacter Delay	GSC.PortX.RDelay
Packet (String) Length	GSC.PortX.RPacketLen

Streaming mode is not recommended for general use, as it creates heavy demands on the ControlLogix processor and requires time-critical programming logic to properly process.

4.3 Normal Data Transfer

Normal data transfer includes the transferring of data received or to be transmitted on the ports and the status data. These data are transferred through read (input image) and write (output image) blocks. Refer to Sample Logic Program for a description of the data objects used with the blocks and the ladder logic required. The following topics describe the function and structure of each block.

4.3.1 Block Request from the Processor to the Module

These blocks of data transfer information from the ControlLogix processor to the module. The structure of the output image used to transfer this data is shown below:

Port 1

Byte Offset	Description
0	Block Sequence Number (Read Block Byte 499 value sent by module)
1 to 2	Intercharacter delay for this message (milliseconds between characters)
3 to 4	Number of characters to transmit on Port 1 (0 to 200)
5 to 204	Port 1 data to transmit
205 to 250	Reserved

Port 2

Byte Offset	Description
251 to 252	Intercharacter delay for this message (milliseconds between characters)
253 to 254	Number of characters to transmit on Port 2 (0 to 200)
255 to 454	Port 2 data to transmit
455 to 495	Reserved

To set up a message to be transmitted, the simple example ladder expects the user to do the following in this order, either manually or by writing additional process logic:

- Load the character codes for the outgoing message string into the controller tag *GSC.Px_Data.WriteString* (where x = 1 for Port 1 or 2 for Port 2)
- Set the tag *GSC.Px_Data.WriteLength* to the number of characters to send

Once the number of characters to transmit in the write block is set greater than zero (value in word at bytes 3 & 4 and/or 253 & 254), the ladder logic program will trigger the Process Write Block function and move a new sequence number value into Byte 0 of the output image. This Block Sequence Number will be the value received in the most recently received Read Block. If the selected port is not already busy, the data in the block will be moved to the port's transmit buffer and sent out the port.

For devices that do not buffer received data, when interfacing with a modem in command mode, or when simulating keyboard or keypad entry, inter-character delays may be required. In order to pace the characters for the write operation, an inter-character delay value can be sent with each write message.

For example, if the port is tied to a device that expects input with delays of 200 milliseconds between each character, set the Inter-character Delay word (Bytes 1 & 2 for Port 1 or Bytes 251 & 252 for Port 2) to a value of 200 in the module's output image using processor ladder logic. The message will be transmitted with a 200-millisecond wait period between each character. Because this delay value is sent from the processor for each write message, the inter-character delay can be set independently for each message.

For example, when writing AT commands to a dial-up modem, an inter-character delay of 100 milliseconds may be required for dialing. But, when the modem has made its connection and is ready for data transfer, the Inter-character Delay might need to be set to 0. When the delay is set to 0, the whole packet of data will be placed in the module's transmit buffer at one time and all characters will be sent one after the other with no delay in between.

4.3.2 Read Block

The module buffers the data received on its application serial ports in individual port receive buffers until one of the specified termination condition is recognized. The module will then transfer the received block of data from the port buffer to the controller. Read data blocks transfer information from the module to the ControlLogix processor.

The structure of the input image used to transfer this data is shown below.

Port 1

Byte Offset	Description
0	Reserved
1 to 2	Number of characters (0 to 200) in Port 1 receive block (5 to 204). If the receive data in the module is larger than 200 bytes, multiple blocks will be transferred. Any block with a value of -1 in this field represents the first or continuation block and the block contains 200 bytes of data. The last block of data will contain a positive number in this field that represents the number of characters in the last block.
3 to 4	Number of characters transmitted (0 to 200) from last block write Port 1
5 to 204	Port 1 data received
205 to 209	Reserved

Module Status

Byte Offset	Description
210 to 211	Program cycle counter
212 to 215	Product name as ASCII string
216 to 219	Revision level as ASCII string
220 to 223	Operating system level as ASCII string
224 to 227	Run number as ASCII string
228 to 229	Number of blocks transferred from module to processor
230 to 231	Number of blocks transferred from processor to module
232 to 233	Number of blocks parsed by module
234 to 235	Number of block errors in module

Port 1 Status

Byte Offset	Description
236 to 237	Port 1 receive state: -1 = Listening for data 1 = Receiving Port Data 2 = Waiting for Backplane transfer
238 to 239	Port 1 receive character count
240 to 241	Port 1 receive block count
242 to 243	Port 1 transmit state: 0 = Waiting for Data to Send 1 = RTS On 2 = RTS Timeout 3 = Sending data 4 = Waiting for RTS Off 5 = RTS turned off 30 = Intercharacter Delay 31 = Intercharacter Delay 32 = Intercharacter Delay 100 = Message Delay before Transmit 101 = Message Delay before Transmit
244 to 245	Port 1 transmit character count
246 to 247	Port 1 transmit block count
248 to 249	Port 1 error word
250	Reserved

Port 2

Byte Offset	Description
251 to 252	Number of characters (0 to 200) in Port 2 receive block (255 to 454). If the receive data in the module is larger than 200 bytes, multiple blocks will be transferred. Any block with a value of -1 in this field represents the first or continuation block and the block contains 200 bytes of data. The last block of data will contain a positive number in this field that represents the number of characters in the last block.
253 to 254	Number of characters transmitted (0 to 200) from last block write Port 2
255 to 454	Port 2 data received
455	Reserved

Port 2 Status

Byte Offset	Description
456 to 457	Port 2 receive state: -1 = Listening for data 1 = Receiving Port Data 2 = Waiting for Backplane transfer
458 to 459	Port 2 receive character count
460 to 461	Port 2 receive block count

Byte Offset	Description
462 to 463	Port 2 transmit state: 0 = Waiting for Data to Send 1 = RTS On 2 = RTS Timeout 3 = Sending data 4 = Waiting for RTS Off 5 = RTS turned off 30 = Intercharacter Delay 31 = Intercharacter Delay 32 = Intercharacter Delay 100 = Message Delay before Transmit 101 = Message Delay before Transmit
464 to 465	Port 2 transmit character count
466 to 467	Port 2 transmit block count
468 to 469	Port 2 error word
470 to 498	Reserved
499	Block Sequence Number (Bumped each scan by module)

The Block Sequence Number (byte 499) is an index value used to signal to the ControlLogix processor that a new block is ready for ladder logic processing. The ladder logic must recognize a change in this value and process the data encapsulated in the input image.

The read block contains the data received on each port and status data for each port. The two word-length values at bytes 1 & 2 (Port 1 Receive Length) and 251 & 252 (Port 2 Receive Length) define the number of bytes in this input image to be processed by the ladder logic for each port. Received data for the ports are found starting at byte 5 for Port 1and 255 for Port 2.

The ladder logic is required to handle all read data transferred from the communication port buffers to the processor. The simple example ladder logic assumes the received data block contains no more than 200 bytes (200 ASCII character codes) of data for each port. Since the receive buffer for each application port in the module can hold up to 4096 bytes (characters), the ladder logic to handle incoming strings larger than 200 bytes is more complex than that shown in the simple example logic.

If more than two hundred bytes are present in the buffer to be sent to the controller, the length field will be set to a value of -1 for all Read Blocks transferred, except for the last block. Each new block with the next 200 bytes from the buffer will arrive with a new Block Sequence Number and a length of -1. When 200 or fewer bytes remain in the buffer, the module will send the last block with a positive number in the length field. The value passed is the remaining number of valid bytes present in the data area that completes the long received message.

If you expect to receive incoming messages that will contain more than 200 bytes (characters) per message, your ladder logic must be modified to process sequential 200-byte blocks of data and recognize the final partial block in order to successfully handle those larger messages.

The two word values at bytes 3 & 4 (Port 1 Transmit Count) and bytes 253 & 254 (Port 2 Transmit Count) inform the processor of the number of bytes transferred from the last write block to the respective port transmit buffers. If a value of zero is returned in one of these words and data was sent in the last write block, the ladder logic must re-send the data in the next write block as the port was in a busy state and could not buffer the new data to be transmitted. If a value is returned in one of these words, the value represents the number of bytes from the last write block moved into the port's transmit buffer.

The status information transferred in the Read Block can be used by the processor to determine the state and "health" of the module and the device or devices attached to each application port. An important member of the value in the status object is the Error Word for each port. This value contains the configuration error flags for each port and the receive buffer overflow error flag.

4.4 Special Function Blocks

Special function blocks are special blocks used to control the module or request special data from the module. The current version of the software supports three special function blocks:

- Warm Boot block
- Cold Boot block
- Configuration Data block

4.4.1 Block 9998: Warm Boot

This block is sent from the ControlLogix processor to the module (output image) when the module is required to perform a warm-boot (software reset) operation. This block is commonly sent to the module any time configuration data modifications are made in the controller tags data area. This will cause the module to read the new configuration information and to restart. To Warm Boot the module, place a value of - 2 in the first byte of the output image (where the Block Sequence Number would normally go).

4.4.2 Block 9999: Cold Boot

This block is sent from the ControlLogix processor to the module (output image) when the module is required to perform the cold boot (hardware reset) operation. This block is sent to the module when a hardware problem is detected by the ladder logic that requires a hardware reset. To Cold Boot the module, place a value of - 3 in the first byte of the output image (where the Block Sequence Number would normally go).

4.4.3 Configuration Data Transfer Block

When the module performs a restart operation, it will request configuration information from the ControlLogix processor. This data is transferred to the module in a specially formatted Write Block (output image). The module will request the configuration block by setting the Block Sequence Number in the Read Block (input image) to a value of -1. Refer to the MVI56E-GSC Configuration (page 46) section of this manual for a description of the data objects used with the blocks and the ladder logic required. The format of the configuration block is shown below:

Port 1

Byte Offset	Description
2 to 3	BPFail
4 to 5	Enabled
6 to 7	Termination Type
8 to 9	Baud Rate
10 to 11	Parity
12 to 13	Data Bits
14 to 15	Stop Bits
16 to 17	RTS On Delay

Byte Offset	Description
18 to 19	RTS Off Delay
20 to 21	Handshaking
22 to 23	Receive Termination Character Count
24 to 35	Receive Termination Characters
36 to 27	Receive Packet Length
38 to 39	Receive Message Timeout
40 to 41	Receive Intercharacter Delay Timeout
42 to 43	Transmit Termination Character Count (Not Used)
44 to 55	Transmit Termination Characters (Not Used)
56 to 57	Transmit Packet Length (Not Used)
58 to 59	Transmit Message Timeout
60 to 61	Transmit Intercharacter Delay Timeout (Not Used)
62 to 63	Minimum Message Transmit Delay

Port 2

Byte Offset	Description
64 to 65	Enabled
66 to 67	Termination Type
68 to 69	Baud Rate
70 to 71	Parity
72 to 73	Data Bits
74 to 75	Stop Bits
76 to 77	RTS On Delay
78 to 79	RTS Off Delay
80 to 81	Handshaking
82 to 83	Receive Termination Character Count
84 to 95	Receive Termination Characters
96 to 97	Receive Packet Length
98 to 99	Receive Message Timeout
100 to 101	Receive Intercharacter Delay Timeout
102 to 103	Transmit Termination Character Count (Not Used)
104 to 115	Transmit Termination Characters (Not Used)
116 to 117	Transmit Packet Length (Not Used)
118 to 119	Transmit Message Timeout
120 to 121	Transmit Intercharacter Delay Timeout (Not Used)
122 to 123	Minimum Message Transmit Delay

4.5 Using the Sample Add-On Instruction

Ladder logic is required to use the MVI56E-GSC module. Tasks that must be handled by the ladder logic are:

- Module backplane data transfer
- Application serial port data handling
- Special block handling
- Status data handling
- Power-up Handler (may be needed to initialize the module's database and to clear any processor fault conditions.)

The sample Import Rung with Add-On Instruction is extensively commented to provide information on the purpose and function of each user-defined data type and controller tag. For most applications, the Add-On Instruction will work without modification for all of the tasks listed above, except any application-specific parsing or compiling that may be needed to control input or output data and any process-specific decision-making logic that may be needed based on input data.

In other words, the sample AOI will give you all the logic any application will need to receive data on the module's ports, send data on the module's ports, manage input and output image data transfers between the module and the processor, and handle status data sent by the module to the processor. The only logic you may need to add would be whatever you need to make use of incoming data strings and whatever you need to create any outgoing data strings.

The sample logic consists of the following:

- I/O Configuration and Module Properties
- User-Defined Data Types
- Controller Tags
- Add-On Defined Data Types
- Add-On Instruction Logic

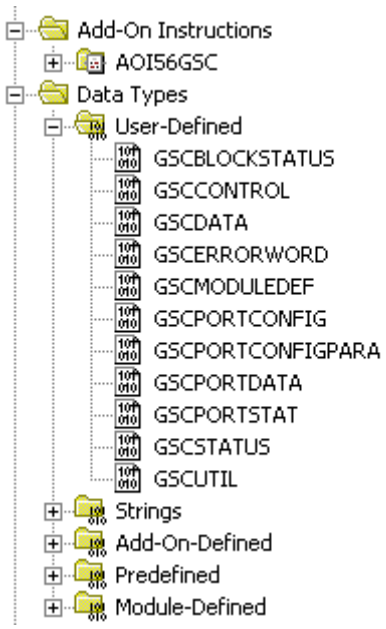
Each of these items will be covered in detail in the following sections.

4.5.1 Input/Output (I/O) Configuration and Module Properties

The I/O configurations and module properties control backplane data transfers between the module and the ControlLogix processor. All of the parameters and settings required have been discussed in an earlier section of the first chapter. For additional details, please refer to Chapter 1, Creating the Module (page 34).

4.5.2 User-Defined Data Types

When you import the rung with the Add-On Instruction (AOI), several new User-defined Data Types (UDTs) are created for you. These UDTs form the basis for creating many of the Controller Tags and Tag Arrays used in the sample AOI logic.



GSCBLOCKSTATUS

This low-level data type creates tags to hold module status values that show input and output image (I/O) block transfer activity. These block transfers occur between the module and the ControlLogix processor. This low-level structure is used to create the higher-level structure, *GSCSTATUS*.

Members: Data Type Size: 8 byte(s)

	Name	Data Type	Style	Description
	Read	INT	Decimal	Number of blocks read by the module
	Write	INT	Decimal	Number of blocks written by the module
	Parse	INT	Decimal	Number of blocks parsed by the module
	Err	INT	Decimal	Number of block transfer errors

These tags hold values of module internal counters that keep a running total of how many input and output image blocks have been transferred between the module and the ControlLogix processor. During normal operation, the values displayed for *Read, Write*, and *Parse* should increment together and be very close to the same values, unless backplane errors are occurring. If backplane errors are occurring, the *Err* value will increment and the *Parse* value will not.

As long as *Read, Write*, and *Parse* continue to increment and *Err* does not change, this indicates normal backplane activity. An occasional backplane error may occur, even in normally functioning systems. So, if the value in the *Err* counter slowly increases over time, this is generally not a cause for concern. However, if normal backplane transfers appear sluggish or non-functional, if input data is being lost or output data is not being sent, especially when the *Err* counter is incrementing frequently, this could indicate a hardware failure, I/O configuration problem, or, for remote rack installations only, a process network problem that may need to be corrected.

GSCCLOCKTYPE

This data type allows the clock value to be exchanged between the module and processor.

Name	Data Type	Style	Description	External Access
Read	BOOL	Decimal	Reads current time from the module	Read/Write
Write	BOOL	Decimal	Writes current time from the module	Read/Write
Config	GSCDATETIME		Date and time configuration read or written between the module and processor	Read/Write

GSCCONTROL

This data type creates the Boolean (single-bit binary) tags to hold control bits for special functions.

Members: Data Type Size: 4 byte(s)

Name	Data Type	Style	Description
Warmboot	BOOL	Decimal	Performs a warm boot operation on the module (software reset).
Coldboot	BOOL	Decimal	Performs a cold boot operation on the module (hardware and software reset).
ConfigDone	BOOL	Decimal	This to confirm the module reveived its configuration from ladder logic

GSCDATA

This data type is a mid-level data structure that creates Port structures to organize and hold incoming and outgoing string data for both ports. This data type uses one iteration of the *GSCPORTDATA* (page 96) UDT data structure for each of the two application serial ports.

Members: Data Type Size: 16392 byte(s)

Name	Data Type	Style	Description
Port1	GSCPORTDATA		Port 1 data
ReadString	SINT[4096]	Decimal	Where received data will be displayed
WriteLength	INT	Decimal	The length of character the port is transmitting
WriteString	SINT[4096]	Decimal	Where sent data will be placed before ii is sent
Port2	GSCPORTDATA		Port 2 data
ReadString	SINT[4096]	Decimal	Where received data will be displayed
WriteLength	INT	Decimal	The length of character the port is transmitting
WriteString	SINT[4096]	Decimal	Where sent data will be placed before ii is sent

GSCDATETIME

This data type stores the time and date values.

	Name	Data Type	Style	Description	External Access
	Year	DINT	Decimal		Read/Write
	Month	DINT	Decimal		Read/Write
	Day	DINT	Decimal		Read/Write
	Hour	DINT	Decimal		Read/Write
	Minute	DINT	Decimal		Read/Write
	Seconds	DINT	Decimal		Read/Write

GSCERRORWORD

This data type creates the Boolean (single-bit binary) tags to help identify the various errors present in the *Error/Configuration Word*. Whenever the module detects an invalid value in one of the provided configuration parameters, it will set one or more of these bit tags to a value of 1 to indicate the parameter or parameters with the incorrect value. This low-level data structure is used to create the higher-level structure, *GSCPORTSTAT* (page 97).

Members: Data Type Size: 4 byte(s)

	Name	Data Type	Style	Description
	Cfg_type	BOOL	Decimal	Bit 0, type field configuration error
	Cfg_Baud	BOOL	Decimal	Bit 1, baud rate field configuration error
	Cfg_Parity	BOOL	Decimal	Bit 2, parity field configuration error
	Cfg_DataBits	BOOL	Decimal	Bit 3, data bit field configuration error
	Cfg_StopBits	BOOL	Decimal	Bit 4, stop bit field configuration error
	Cfg_Handshake	BOOL	Decimal	Bit 5, handshaking field configuration error
	Cfg_RTermcount	BOOL	Decimal	Bit 6, receive termination character count field configuration error
	Cfg_RPacketLen	BOOL	Decimal	Bit 7, receive packet length field configuration error
	Cfg_RTimeout	BOOL	Decimal	Bit 8, receive message timeout field configuration error
	Cfg_RDelay	BOOL	Decimal	Bit 9, receive intercharacter delay field configuration error
	Cfg_WTimeout	BOOL	Decimal	Bit 10, write message timeout field configuration error
	Resv_Bit11	BOOL	Decimal	Bit 11
	Resv_Bit12	BOOL	Decimal	Bit 12
	Resv_Bit13	BOOL	Decimal	Bit 13
	Resv_Bit14	BOOL	Decimal	Bit 14
	Err_ROverflow	BOOL	Decimal	Bit 15, receive overflow error

GSCETHERNET

This data type stores the IP settings of the MVI56E-GSC.

	Name	Data Type	Style	Description	External Access
	IP	INT[4]	Decimal		Read/Write
	Netmask	INT[4]	Decimal		Read/Write
	Gateway	INT[4]	Decimal		Read/Write

GSCETHERNETTYPE

This data type allows the MVI56E-GSC Ethernet configuration to be exchanged between the module and processor.

Members: Data Type Size: 4 byte(s)

	Name	Data Type	Style	Description
	Warmboot	BOOL	Decimal	Performs a warm boot operation on the module (software reset).
	Coldboot	BOOL	Decimal	Performs a cold boot operation on the module (hardware and software reset).
	ConfigDone	BOOL	Decimal	This to confirm the module reveived its configuration from ladder logic

GSCMODULEDEF

This data type is the top-level data structure that provides overall organization to the rest of the lower-level data structures, tag arrays, and tags.

Members: Data Type Size: 16592 byte(s)

	Name	Data Type	Style	Description
	⊞ CONFIG	GSCPORTCONFIG		Port configuration parameters
	⊞ DATA	GSCDATA		Port sent and received data
	⊞ STATUS	GSCSTATUS		Module status data and individual port status
	⊞ CONTROL	GSCCONTROL		Module control parameters
	⊞ UTIL	GSCUTIL		Tags used by the module

GSCPORTCONFIG

This data type is a mid-level data structure that creates Port structures to organize and hold port configuration data for both ports. This data type uses one iteration of the *GSCPORTCONFIGPARA* (page 96) UDT data structure for each of the two application serial ports.

Members: Data Type Size: 120 byte(s)

Name	Data Type	Style	Description
Port1	GSCPORTCONFIGPARA		Port 1 configuration data
Enabled	INT	Decimal	0=Disable port, 1=Enable port
Type	INT	Decimal	Type of control used to flush data received on port
Baudrate	INT	Decimal	Baud rate for communications
Parity	INT	Decimal	Parity (0=none, 1=even, 2=odd)
DataBits	INT	Decimal	Data bits (5, 6, 7, or 8)
StopBits	INT	Decimal	Stop bits (1 or 2)
RTSOn	INT	Decimal	RTS delay before sending data
RTSOff	INT	Decimal	RTS delay after sending message
Handshaking	INT	Decimal	Type of Handshaking (0=none, 1=RTSCTS, 2=DTRDSR,3=XONXOFF)
RTermCnt	INT	Decimal	Number of termination characters for received message for control
RTermChar	SINT[12]	Hex	Characters for termination control
RPacketLen	INT	Decimal	Fixed packet length value
RTimeout	INT	Decimal	Message timeout value for control
RDelay	INT	Decimal	Intercharacter timeout for control
WTermCnt	INT	Decimal	Not used
WTermChar	SINT[12]	Hex	Not used
WPacketLen	INT	Decimal	Not used
WTimeout	INT	Decimal	Timeout for sending a message out the port
WDelay	INT	Decimal	Not used
WMinDelay	INT	Decimal	Minimum delay before transmitting message out port
Port2	GSCPORTCONFIGPARA		Port 2 configuration data

GSCPORTCONFIGPARA

This low-level data type creates the tags and tag arrays required to organize and hold configuration values for an application serial port. The values entered into these tags will control the setup and operation of the port. This low-level structure is used to create the higher-level structure, *GSCPORTCONFIG* (page 95).

Members: Data Type Size: 60 byte(s)

Name	Data Type	Style	Description
Enabled	INT	Decimal	0=Disable port, 1=Enable port
Type	INT	Decimal	Type of control used to flush data received on port
Baudrate	INT	Decimal	Baud rate for communications
Parity	INT	Decimal	Parity (0=none, 1=even, 2=odd)
DataBits	INT	Decimal	Data bits (5, 6, 7, or 8)
StopBits	INT	Decimal	Stop bits (1 or 2)
RTSOn	INT	Decimal	RTS delay before sending data
RTSOff	INT	Decimal	RTS delay after sending message
Handshaking	INT	Decimal	Type of Handshaking (0=none, 1=RTSCTS, 2=DTRDSR,3=XONXOFF)
RTermCnt	INT	Decimal	Number of termination characters for received message for control
RTermChar	SINT[12]	Hex	Characters for termination control
RPacketLen	INT	Decimal	Fixed packet length value
RTimeout	INT	Decimal	Message timeout value for control
RDelay	INT	Decimal	Intercharacter timeout for control
WTermCnt	INT	Decimal	Not used
WTermChar	SINT[12]	Hex	Not used
WPacketLen	INT	Decimal	Not used
WTimeout	INT	Decimal	Timeout for sending a message out the port
WDelay	INT	Decimal	Not used
WMinDelay	INT	Decimal	Minimum delay before transmitting message out port

GSCPORTDATA

This low-level data type creates the tags and tag arrays required to organize hold incoming and outgoing data string values for an application serial port. The values in these tags will be the data received or data to be transmitted on the port. This low-level structure is used to create the higher-level structure, *GSCDATA* (page 92).

Members: Data Type Size: 8196 byte(s)

Name	Data Type	Style	Description
ReadString	SINT[4096]	Decimal	Where received data will be displayed
WriteLength	INT	Decimal	The length of character the port is transmitting
WriteString	SINT[4096]	Decimal	Where sent data will be placed before ii is sent

GSCPORTSTAT

This mid-level data structure creates the tags to organize and hold all port status data available from each port of the module. This data structure includes the bit-mapped *Error/Configuration Word* tags created by the *GSCDATA* (page 92) UDT. This mid-level structure is used to create the higher-level structure, *GSCSTATUS*.

Members: Data Type Size: 16 byte(s)

Name	Data Type	Style	Description
RxState	INT	Decimal	State of receive task
RxCharCnt	INT	Decimal	Number of characters received on port
RxMsgCnt	INT	Decimal	Number of messages received on port
TxState	INT	Decimal	State of transmit task
TxCharCnt	INT	Decimal	Number of characters transmitted on port
TxMsgCnt	INT	Decimal	Number of messages transmitted on port
ErrorWord	GSCERRORWORD		Error/Configuration Word for port
Cfg_type	BOOL	Decimal	Bit 0, type field configuration error
Cfg_Baud	BOOL	Decimal	Bit 1, baud rate field configuration error
Cfg_Parity	BOOL	Decimal	Bit 2, parity field configuration error
Cfg_DataBits	BOOL	Decimal	Bit 3, data bit field configuration error
Cfg_StopBits	BOOL	Decimal	Bit 4, stop bit field configuration error
Cfg_Handshake	BOOL	Decimal	Bit 5, handshaking field configuration error
Cfg_RTermcount	BOOL	Decimal	Bit 6, receive termination character count field configuration error
Cfg_RPacketLen	BOOL	Decimal	Bit 7, receive packet length field configuration error
Cfg_RTimeout	BOOL	Decimal	Bit 8, receive message timeout field configuration error
Cfg_RDelay	BOOL	Decimal	Bit 9, receive intercharacter delay field configuration error
Cfg_WTimeout	BOOL	Decimal	Bit 10, write message timeout field configuration error
Resv_Bit11	BOOL	Decimal	Bit 11
Resv_Bit12	BOOL	Decimal	Bit 12
Resv_Bit13	BOOL	Decimal	Bit 13
Resv_Bit14	BOOL	Decimal	Bit 14
Err_ROverflow	BOOL	Decimal	Bit 15, receive overflow error

GSCSTATUS

This upper-mid-level data structure creates the tags and structure needed to display all module status and error information, including general module status, block transfer status, and port status of each port. This higher-level data structure is a combination of new tags and the mid-level structures, *GSCBLOCKSTATUS* (page 91) and *GSCPORTSTAT* (page 97).

Members: Data Type Size: 60 byte(s)

Name	Data Type	Style	Description
PassCnt	INT	Decimal	Program cycle counter for module
Product	INT[2]	Hex	Product code for module (GSC)
Rev	INT[2]	Hex	Revision level of module's code
OP	INT[2]	Hex	Operating system version for module
Run	INT[2]	Hex	Run number for module
BlockStatus	GSCBLOCKSTATUS		Data block transfer statistics
Read	INT	Decimal	Number of blocks read by the module
Write	INT	Decimal	Number of blocks written by the module
Parse	INT	Decimal	Number of blocks parsed by the module
Err	INT	Decimal	Number of block transfer errors
Port1	GSCPORTSTAT		Status for port1
RxState	INT	Decimal	State of receive task
RxCharCnt	INT	Decimal	Number of characters received on port
RxMsgCnt	INT	Decimal	Number of messages received on port
TxState	INT	Decimal	State of transmit task
TxCharCnt	INT	Decimal	Number of characters transmitted on port
TxMsgCnt	INT	Decimal	Number of messages transmitted on port
ErrorWord	GSCERRORWORD		Error/Configuration Word for port
Port2	GSCPORTSTAT		Status for port 2
RxState	INT	Decimal	State of receive task
RxCharCnt	INT	Decimal	Number of characters received on port
RxMsgCnt	INT	Decimal	Number of messages received on port
TxState	INT	Decimal	State of transmit task
TxCharCnt	INT	Decimal	Number of characters transmitted on port
TxMsgCnt	INT	Decimal	Number of messages transmitted on port
ErrorWord	GSCERRORWORD		Error/Configuration Word for port

GSCUTIL

This data type creates tags needed for sample ladder logic sequencing and control. You can think of these tags mostly as a 'scratchpad' for the sample logic to keep track of where it is in the processing cycle. These tags also help control receiving strings longer than 200 characters.

Members: Data Type Size: 16 byte(s)

Name	Data Type	Style	Description
LastRead	SINT	Decimal	This is used to read the input image from the processor
LastWrite	SINT	Decimal	This is used to write the output image to the processor
BackplaneFail	INT	Decimal	This is used to show if there is any failed blocks
P1WriteLast	INT	Decimal	Port 1 last write data in the array
P2Writelast	INT	Decimal	Port 2 last data in the array
P1ArrayIndex	INT	Decimal	This is the index reference within the read or write data array for port1
P2ArrayIndex	INT	Decimal	This is the index reference within the read or write data array for port 2
P1RSindex	INT	Decimal	used when more than 200 characters are received on the port 1
P2RSindex	INT	Decimal	used when more than 200 characters are received on the port 2

4.5.3 Controller Tags

The controller tags associated with the sample ladder logic are all contained in and organized by the data structures and tag arrays created by the previously described User-Defined Data Types (page 91). Tags are arranged in a hierarchical structure and grouped by function.

⊟ GSC	{...}		GSCMODULEDEF
⊞ GSC.CONFIG	{...}		GSCPORTCONFIG
⊞ GSC.DATA	{...}		GSCDATA
⊞ GSC.STATUS	{...}		GSCSTATUS
⊞ GSC.CONTROL	{...}		GSCCONTROL
⊞ GSC.UTIL	{...}		GSCUTIL

GSC.CONFIG Array

The *GSC.CONFIG* array is composed of two major sub-arrays, *GSC.CONFIG.Port1* and *GSC.CONFIG.Port2*. Each sub-array contains all the tags necessary to set up and configure the actions of the application serial ports. Descriptions of the functions of tags in these arrays have been provided in the section, Creating the Module (page 34).

⊟ GSC	{...}	{...}		GSCMOD...
⊟ GSC.CONFIG	{...}	{...}		GSCPORT...
⊟ GSC.CONFIG.Port1	{...}	{...}		GSCPORT...
⊞ GSC.CONFIG.Port1.Enabled	1		Decimal	INT
⊞ GSC.CONFIG.Port1.Type	1		Decimal	INT
⊞ GSC.CONFIG.Port1.Baudrate	9600		Decimal	INT
⊞ GSC.CONFIG.Port1.Parity	0		Decimal	INT
⊞ GSC.CONFIG.Port1.DataBits	8		Decimal	INT
⊞ GSC.CONFIG.Port1.StopBits	1		Decimal	INT
⊞ GSC.CONFIG.Port1.RTSOn	0		Decimal	INT
⊞ GSC.CONFIG.Port1.RTSOff	0		Decimal	INT
⊞ GSC.CONFIG.Port1.Handshaking	0		Decimal	INT
⊞ GSC.CONFIG.Port1.RTermCnt	2		Decimal	INT
⊞ GSC.CONFIG.Port1.RTermChar	{...}	{...}	Hex	SINT[12]
⊞ GSC.CONFIG.Port1.RPacketLen	240		Decimal	INT
⊞ GSC.CONFIG.Port1.RTimeout	2000		Decimal	INT
⊞ GSC.CONFIG.Port1.RDelay	0		Decimal	INT
⊞ GSC.CONFIG.Port1.WTermCnt	0		Decimal	INT
⊞ GSC.CONFIG.Port1.WTermChar	{...}	{...}	Hex	SINT[12]
⊞ GSC.CONFIG.Port1.WPacketLen	0		Decimal	INT
⊞ GSC.CONFIG.Port1.WTimeout	500		Decimal	INT
⊞ GSC.CONFIG.Port1.Spare	0		Decimal	INT
⊞ GSC.CONFIG.Port1.WMinDelay	0		Decimal	INT

GSC.DATA Array

The *GSC.DATA* array is composed of two major sub-arrays, *GSC.DATA.Port1* and *GSC.DATA.Port2*. Each sub-array contains all the tags necessary to hold incoming and outgoing ASCII string data for the application serial ports. The values in these arrays can be displayed as byte values or ASCII characters. Setting the tag array Style to **ASCII** will show the array elements as alpha-numeric characters. Setting the tag Style to **DECIMAL** will allow you to see the raw byte values held in the array. This is useful to be able to verify non-printing ASCII code values which do not show as identifiable characters.

GSC	{...}	{...}		GSCMOD...
GSC.CONFIG	{...}	{...}		GSCPORT...
GSC.DATA	{...}	{...}		GSCDATA
GSC.DATA.Port1	{...}	{...}		GSCPORT...
GSC.DATA.Port1.ReadLength	0		Decimal	INT
GSC.DATA.Port1.ReadString	{...}	{...}	Decimal	SINT[4096]
GSC.DATA.Port1.WriteLength	0		Decimal	INT
GSC.DATA.Port1.WDelay	0		Decimal	INT
GSC.DATA.Port1.WriteString	{...}	{...}	Decimal	SINT[4096]
GSC.DATA.Port2	{...}	{...}		GSCPORT...
GSC.DATA.Port2.ReadLength	0		Decimal	INT
GSC.DATA.Port2.ReadString	{...}	{...}	Decimal	SINT[4096]
GSC.DATA.Port2.WriteLength	0		Decimal	INT
GSC.DATA.Port2.WDelay	0		Decimal	INT
GSC.DATA.Port2.WriteString	{...}	{...}	Decimal	SINT[4096]

GSC.DATA.PortX.ReadLength

This parameter reflects the number of characters received in the last string.

GSC.DATA.PortX.ReadString

The tag arrays, *GSC.DATA.PortX.ReadString*, are arrays of 4096 single-byte integer values, capable of holding binary byte values, which can also be interpreted as ASCII codes for alpha-numeric string characters. Since the port receive buffer can hold incoming strings of up to 4096 characters, this array has been sized to allow it to hold the largest possible received string. The ladder logic will place all application serial port data received in backplane transfers from the module into this array. If the received string is larger than 200 characters (bytes), multiple block transfers will be needed to get the entire string into the *ReadString* array.

GSC.DATA.PortX.WriteLength

This 16-bit integer tag serves a double purpose. Its primary purpose is to hold the number of ASCII characters (or number of single-byte integer values) that need to be transmitted from the application serial port. This tag is also used by the sample logic as a trigger to know when to send strings though the output data image block across the backplane (or process network, for remote rack applications) to the module to be transmitted by the serial port.

The idea is that your application-specific logic will first load the *GSC.DATA.PortX.WriteString* array with the values or ASCII string character that you wish to have sent. Then, to trigger the transmission, your application-specific logic will load into the *GSC.DATA.PortX.WriteLength* tag a value equal to the number of characters in the string. As soon as the sample ladder detects a non-zero value in the range of 1 to 4096 in this tag, it will begin to transfer data from the *GSC.DATA.PortX.WriteString* array into the output image and send it to the module. When the module receives data in the output image, it will transmit that data out the serial port.

GSC.DATA.PortX.WDelay

This parameter reflects the number of the milliseconds to wait between each character is sent from the port. Valid parameter for this parameter is from 1 to 32767.

GSC.DATA.PortX.WriteString

The tag arrays, *GSC.DATA.PortX.WriteString*, are arrays of 4096 single-byte integer values, capable of holding binary byte values, which can also be interpreted as ASCII codes for alpha-numeric string characters. The ladder logic will get all application serial port data to be sent to be transmitted out the port from this array. If the string to be transmitted is larger than 200 characters, multiple block transfers will be required to send the entire string to the module from the *WriteString* array.

GSC.STATUS Array

This tag array holds all the status and error information reported and updated by each input image block. These tags give valuable diagnostic and troubleshooting information and can be used to create application-specific alarm and control logic. For a more detailed description, see the section on Normal Data Transfer Read Block (page 84).

⊞ GSC.DATA	{...}		GSCDATA
⊟ GSC.STATUS	{...}		GSCSTATUS
⊞ GSC.STATUS.PassCnt	-13784	Decimal	INT
⊟ GSC.STATUS.Product	{...}	ASCII	INT[2]
⊞ GSC.STATUS.Product[0]	'SG'	ASCII	INT
⊞ GSC.STATUS.Product[1]	'EC'	ASCII	INT
⊟ GSC.STATUS.Rev	{...}	ASCII	INT[2]
⊞ GSC.STATUS.Rev[0]	'.2'	ASCII	INT
⊞ GSC.STATUS.Rev[1]	'10'	ASCII	INT
⊟ GSC.STATUS.OP	{...}	ASCII	INT[2]
⊞ GSC.STATUS.OP[0]	'01'	ASCII	INT
⊞ GSC.STATUS.OP[1]	'90'	ASCII	INT
⊟ GSC.STATUS.Run	{...}	ASCII	INT[2]
⊞ GSC.STATUS.Run[0]	'20'	ASCII	INT
⊞ GSC.STATUS.Run[1]	'10'	ASCII	INT
⊟ GSC.STATUS.BlockStatus	{...}		GSCBLOCKSTATUS
⊞ GSC.STATUS.BlockStatus.Read	-6542	Decimal	INT
⊞ GSC.STATUS.BlockStatus.Write	-6542	Decimal	INT
⊞ GSC.STATUS.BlockStatus.Parse	-6544	Decimal	INT
⊞ GSC.STATUS.BlockStatus.Err	1	Decimal	INT
⊟ GSC.STATUS.Port1	{...}		GSCPORTSTAT
⊞ GSC.STATUS.Port1.RxState	-1	Decimal	INT
⊞ GSC.STATUS.Port1.RxCharCnt	56	Decimal	INT
⊞ GSC.STATUS.Port1.RxMsgCnt	7	Decimal	INT
⊞ GSC.STATUS.Port1.TxState	0	Decimal	INT
⊞ GSC.STATUS.Port1.TxCharCnt	0	Decimal	INT
⊞ GSC.STATUS.Port1.TxMsgCnt	0	Decimal	INT
⊟ GSC.STATUS.Port1.ErrorWord	{...}		GSCERRORWORD
GSC.STATUS.Port1.ErrorWord.Cfg_type	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Cfg_Baud	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Cfg_Parity	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Cfg_DataBits	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Cfg_StopBits	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Cfg_Handshake	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Cfg_RTermco...	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Cfg_RPacketL...	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Cfg_RTimeout	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Cfg_RDelay	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Cfg_WTimeout	0	Decimal	BOOL
GSC.STATUS.Port1.ErrorWord.Resv_Bit11	0	Decimal	BOOL

GSC.CONTROL Array

This tag array holds the Boolean tags that allow the ladder logic to request that the module perform a *ColdBoot* or *WarmBoot*. When set to a value of 1 by logic, the logic will form a special output data block, which will transfer the reboot request to the module, where it will be executed.

As part of any restart operation, the module will send an input image block with special block code of -1 to request the logic to send the configuration values held in the *GSC.CONFIG* tag arrays. The logic will then form a special output image block that sends the configuration data to the module. Once the module receives and implements the configuration, it will set the GSC.CONTROL.ConfigDone bit to a value of 1. At this point, if the *Error/Configuration Word* (page 129) shows no errors (it shows a value of zero), the module has been successfully configured and is read for normal operation.

⊟ GSC	{...}		GSCMODULEDEF
⊞ GSC.CONFIG	{...}		GSCPORTCONFIG
⊞ GSC.DATA	{...}		GSCDATA
⊞ GSC.STATUS	{...}		GSCSTATUS
⊟ GSC.CONTROL	{...}		GSCCONTROL
GSC.CONTROL.Warmboot	0	Decimal	BOOL
GSC.CONTROL.Coldboot	0	Decimal	BOOL
GSC.CONTROL.ConfigDone	0	Decimal	BOOL
⊞ GSC.UTIL	{...}		GSCUTIL

GSC.UTIL Array

This tag array holds the tags needed for sample ladder logic sequencing and control. You can think of these tags mostly as 'scratchpad' memory for the sample logic to keep track of where it is in the processing cycle. These tags also help control receiving strings longer than 200 characters.

⊟ GSC	{...}	{...}		GSCMOD...
⊞ GSC.CONFIG	{...}	{...}		GSCPORT...
⊞ GSC.DATA	{...}	{...}		GSCDATA
⊞ GSC.STATUS	{...}	{...}		GSCSTAT...
⊞ GSC.CONTROL	{...}	{...}		GSCCONT...
⊟ GSC.UTIL	{...}	{...}		GSCUTIL
⊞ GSC.UTIL.LastRead	42		Decimal	SINT
⊞ GSC.UTIL.LastWrite	0		Decimal	SINT
⊞ GSC.UTIL.BackplaneFail	0		Decimal	INT
⊞ GSC.UTIL.P1WriteLast	0		Decimal	INT
⊞ GSC.UTIL.P2Writelast	-2		Decimal	INT
⊞ GSC.UTIL.P1ArrayIndex	0		Decimal	INT
⊞ GSC.UTIL.P2ArrayIndex	0		Decimal	INT
⊞ GSC.UTIL.P1RSindex	0		Decimal	INT
⊞ GSC.UTIL.P2RSindex	0		Decimal	INT
⊞ GSC.UTIL.P1Temp1	0		Decimal	INT
⊞ GSC.UTIL.P2Temp2	0		Decimal	INT
⊞ GSC.UTIL.Scratch1	0		Decimal	INT
⊞ GSC.UTIL.Scratch2	0		Decimal	INT

The ladder logic will change and update these tags as it works its way through the ladder process. Your application specific process and control logic can monitor the status or values in these tags as part of you control application; but should never write values to these tags, as this could cause serious adverse effects on the module's data processing activity. Incoming data might be lost or data that should be sent might not be transmitted.

However, the *GSC.UTIL.BackplaneFail* tag is the one exception to this read-only rule for these tags. The *GSC.UTIL.BackplaneFail* tag is used to hold a configuration value which can be used to control activity on application serial ports whenever a specified number of successive backplane transfer errors occur. You are expected to write a value to this tag as part of the module's configuration. For more details, see *GSC.UTIL.BackplaneFail* (page 47).

4.5.4 Add-On-Defined Data Types

When you import the Add-On Instruction (AOI) rung , several new Add-On-Defined Data Types (ADTs) are created for you, as well as the UDTs that were also created. The ADTs contain special data-types required by the AOI, either for special logic routines or to define the AOI interface with program logic. ADT information is organized and contained on several tabs in the *Data Types - Add-On-Defined* section of the RSLogix 5000 project tree.

- General tab
- Parameters tab
- Local Tags tab
- Scan Modes tab
- Change History tab
- Help tab

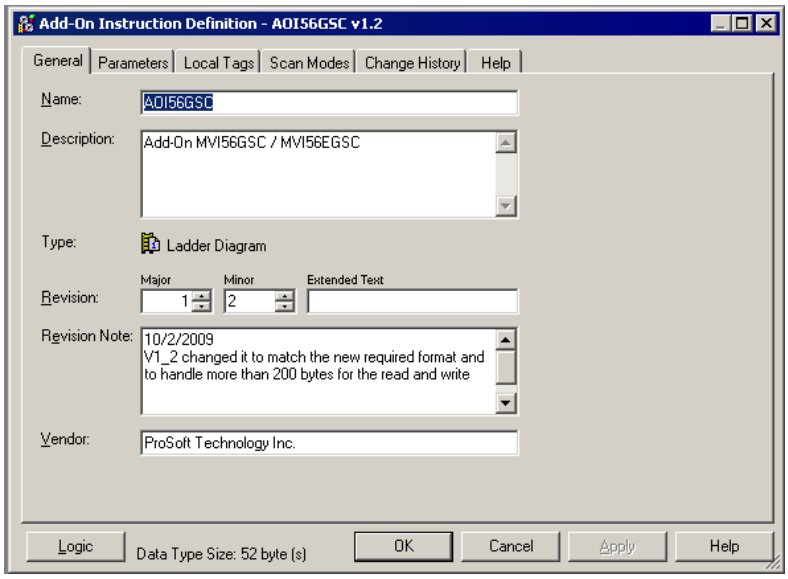
The content and use of each of these tabs will be discussed in more detail in the following sections.

AOI56GSC - General Tab

The General tab provides:

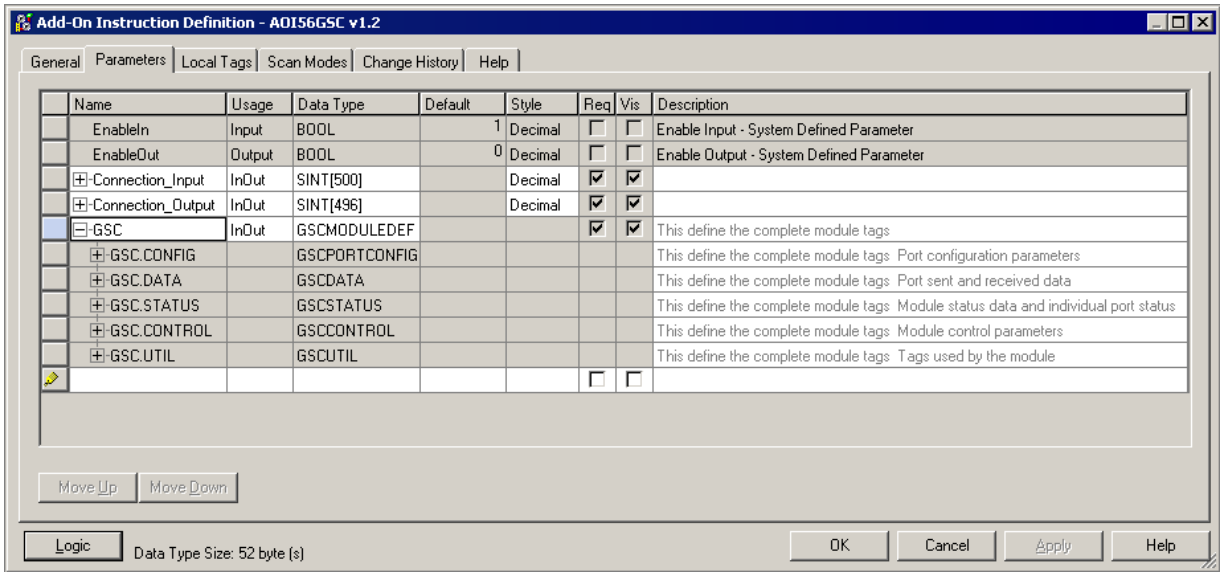
1 Name of the Add-On Instruction (AOI)
2 Description
3 Type of logic used
4 Revision numbers
5 Revision Notes
6 Vendor ID

This information will help you determine whether or not you are using the latest version of the AOI and will give you a few notes on what is different about this version when compared with previous version. It is recommended that you make no changes to the selections and entries on this tab.



AOI56GSC - Parameters Tab

This is the most important of all the tabs. This tab creates the tags and tag arrays needed by the AOI to interface with the rest of the RSLogix 5000 program. It is recommended that you make no changes to the selections and entries on this tab.



The **ENABLEIN** and **ENABLEOUT** Boolean tags allow you to enable or disable input to or output from the AOI to the rest of the program. For normal operation, these two tags must be set to a value of one (1). Setting either tag or both tags to zero (0) will cause loss of communication and backplane transfer errors.

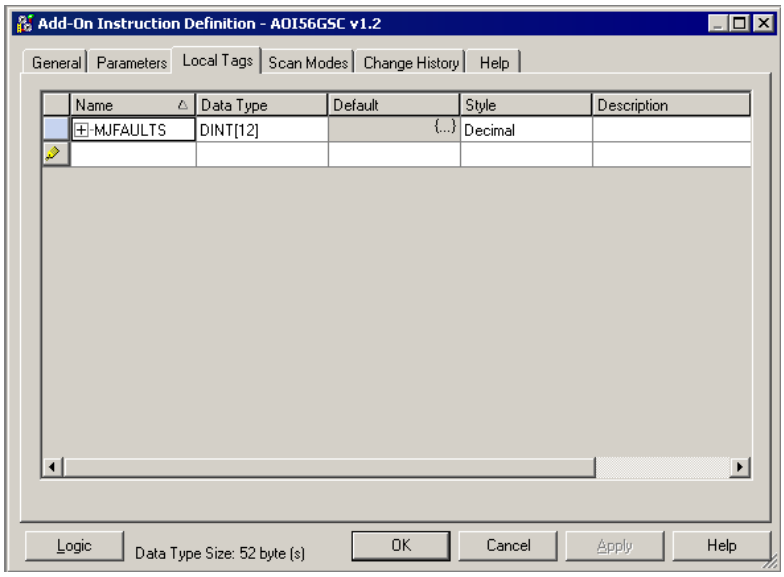
The **CONNECTION_INPUT** tag array is used to receive a copy of the data from the I/O input image data block and bring that data into the AOI logic for processing.

The **CONNECTION_OUTPUT** tag array is used to send data to the I/O output image data block after the AOI logic has finished its logic processing.

For information on the **GSC** array, see *Controller Tags* (page 99).

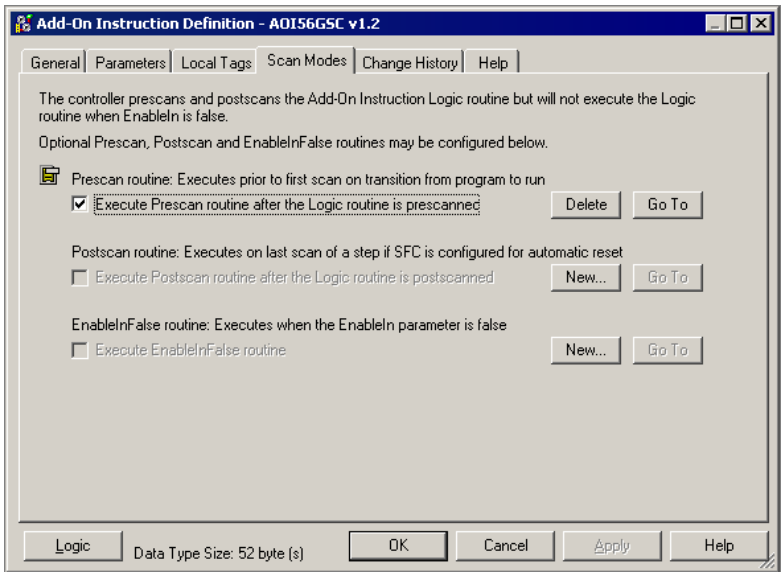
AOI56GSC Local Tags Tab

The **MJFAULTS** tag array defined in this tab is used by the AOI *Prescan* routine for module initialization. This tag array has little practical use for user-specific applications. It is recommended that you make no changes to the selections and entries on this tab.



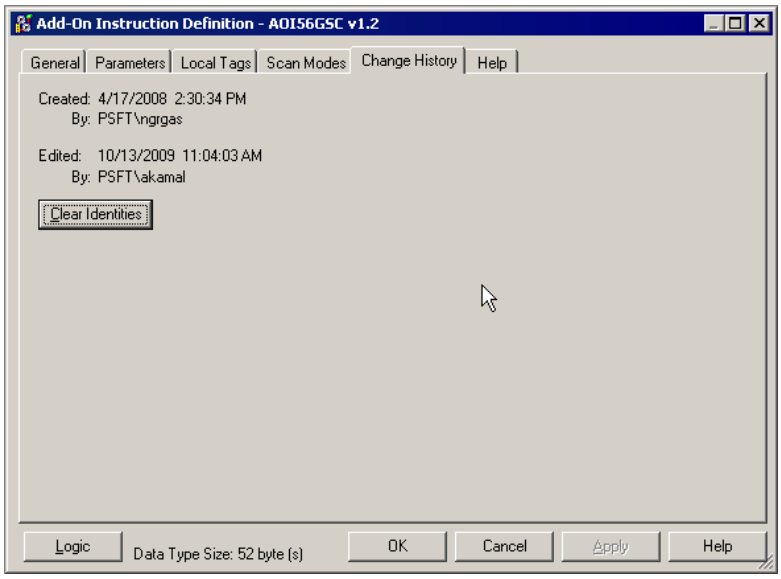
AOI56GSC Scan Modes Tab

This tab selects the use of available AOI logic options. You will notice the *Prescan* routine selection is checked. It is recommended that you make no changes to the selections and entries on this tab.



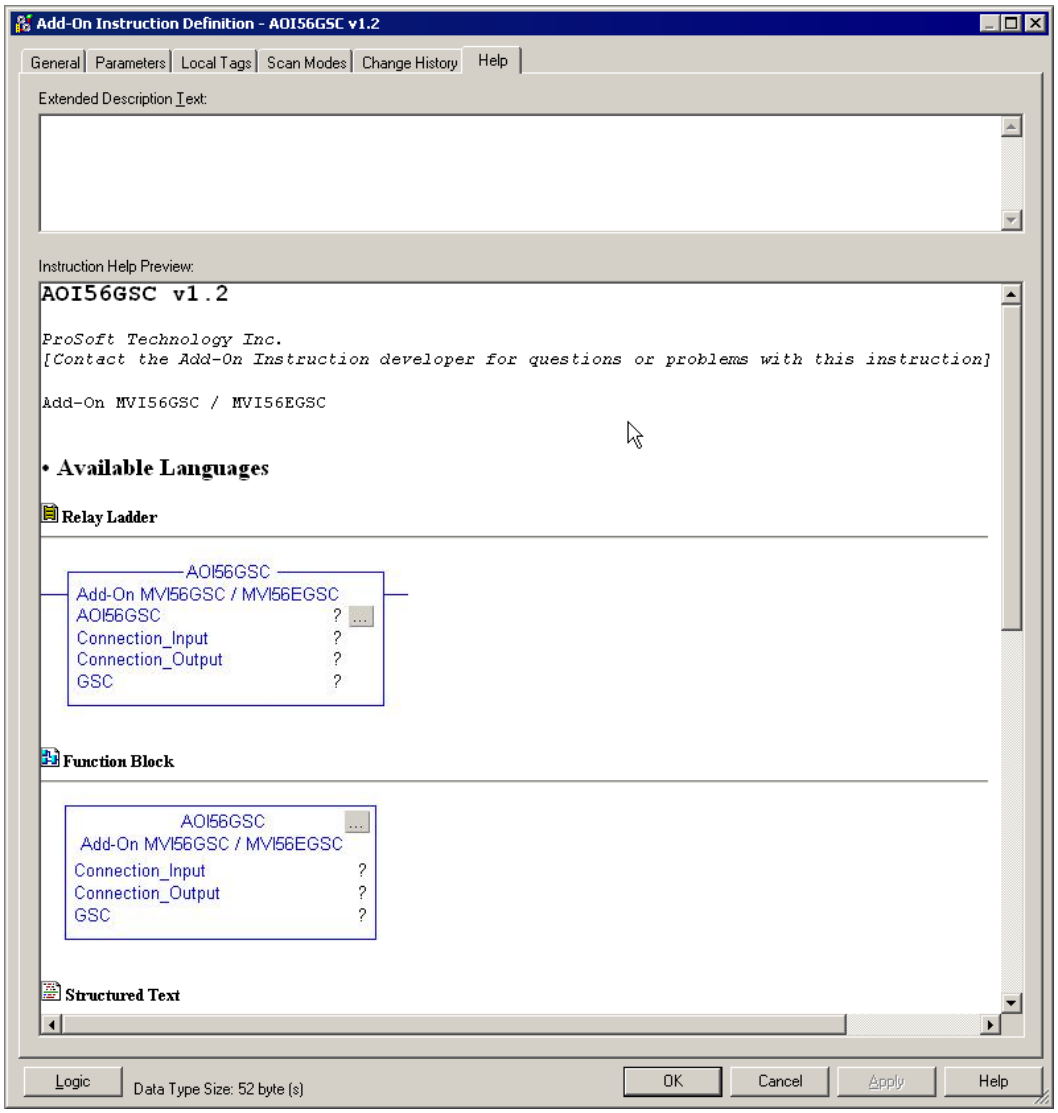
AOI56GSC Change History Tab

This tab shows the dates an times when the AOI was changed and who made the changes. It is recommended that you click on the **CLEAR IDENTITIES** button on this tab.



AOI56GSC Help Tab

This tab provides some additional information about the AOI. The information on this tab is limited and not meant to be a full and complete explanation of the AOI and how to use it.



4.6 Using the Optional Add-On Instruction

4.6.1 Before You Begin

- Make sure that you have installed RSLogix 5000 version 16 (or later).
- Download the Optional Add-On file *MVI56EGSC_Optional_Rung_v1_1.L5X* from www.prosoft-technology.com.
- Save a copy in a folder in your PC.

4.6.2 Overview

The Optional Add-On Instruction Rung Import contains optional logic for MVI56E-GSC applications to perform the following tasks.

- Read/Write Ethernet Configuration

  Allows the processor to read or write the module IP address, netmask and gateway values.

Note: This is an optional feature. You can perform the same task through PCB (ProSoft Configuration Builder). Even if your PC is in a different network group you can still access the module through PCB by setting a temporary IP address.

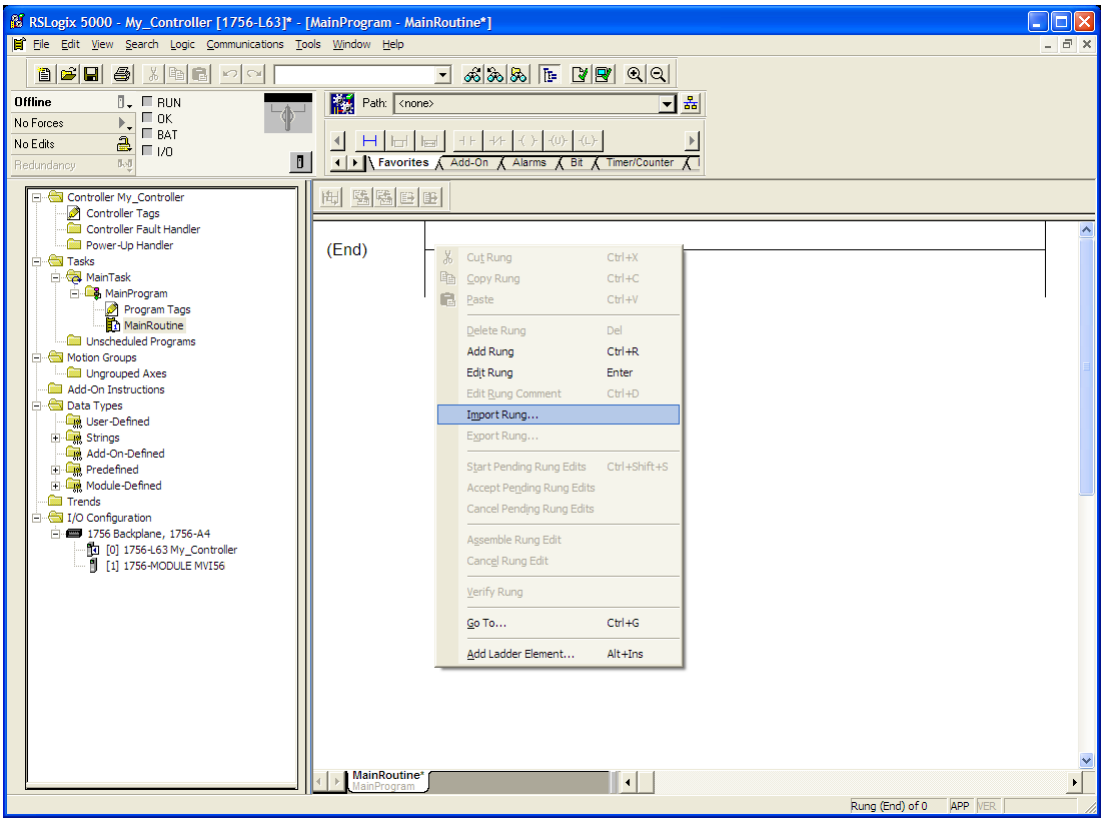
- Read/Write Module Clock Value

  Allows the processor to read and write the module clock settings. The module clock stores the last time that the Ethernet configuration was changed. The date and time of the last Ethernet configuration change is displayed in the scrolling LED during module power up.

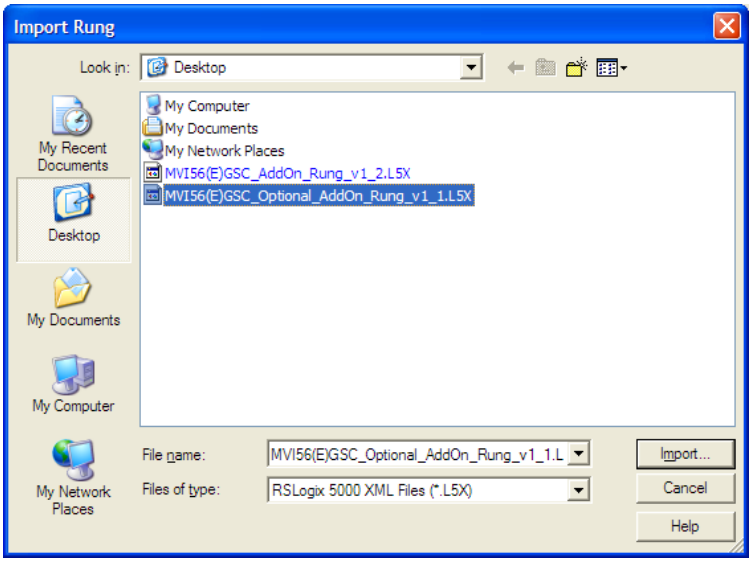
Important: The Optional Add-On Instruction only supports the two features listed above. You must use the sample ladder logic for all other features including backplane transfer of generic ASCII serial data.

4.6.3 Importing the Optional Add-On Instruction Rung

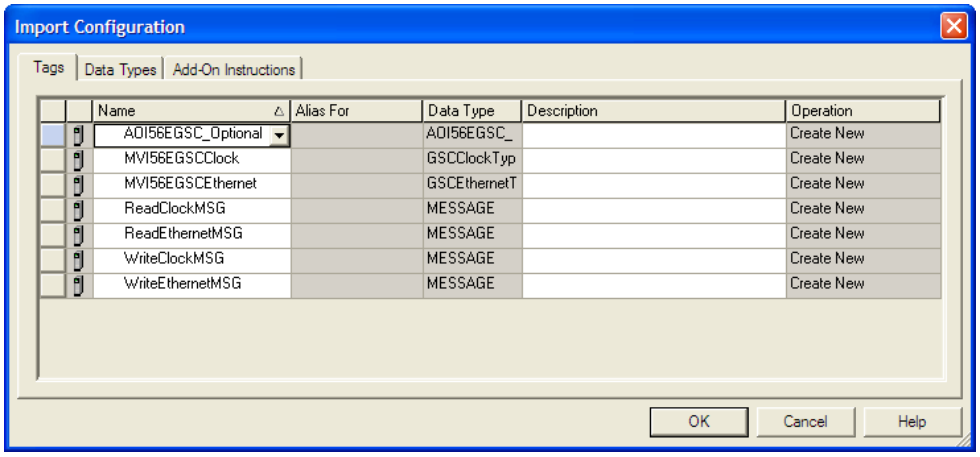
1 Right-click on an empty rung in the main routine of your existing ladder logic and choose **IMPORT RUNG…**



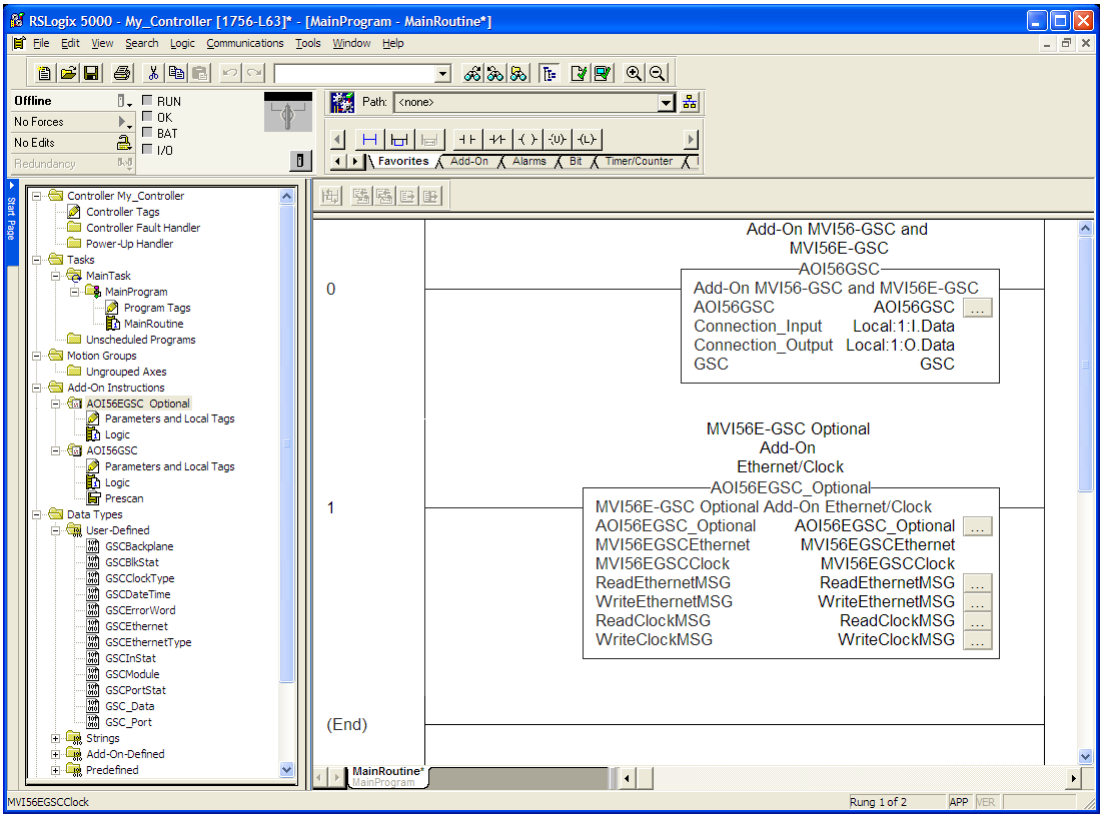
2 Navigate to the folder where you saved MVI56(E)GSC_Optional_AddOn_Rung_<version #>.L5X and select the file.



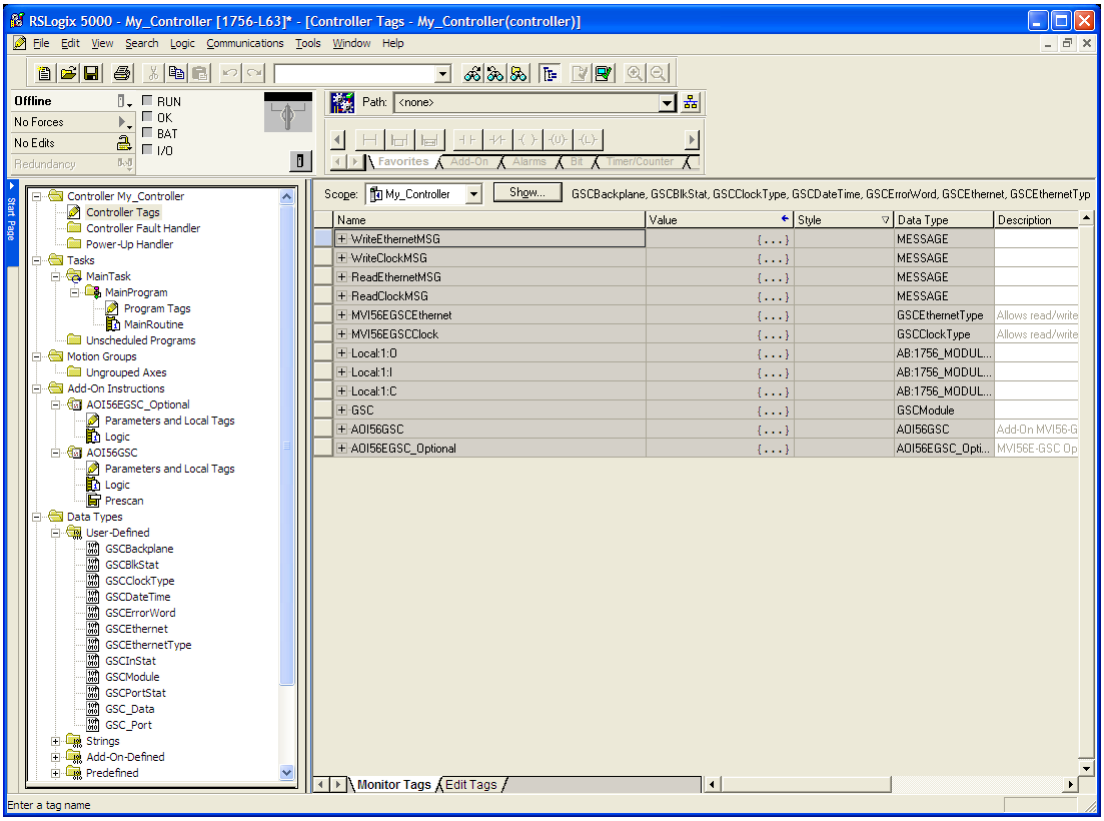
3 In the **IMPORT CONFIGURATION** window, click **OK.**



The Optional Add-On Instruction will be now visible in the ladder logic. Observe that the procedure has also imported data types and controller tags associated to the Optional Add-On Instruction.

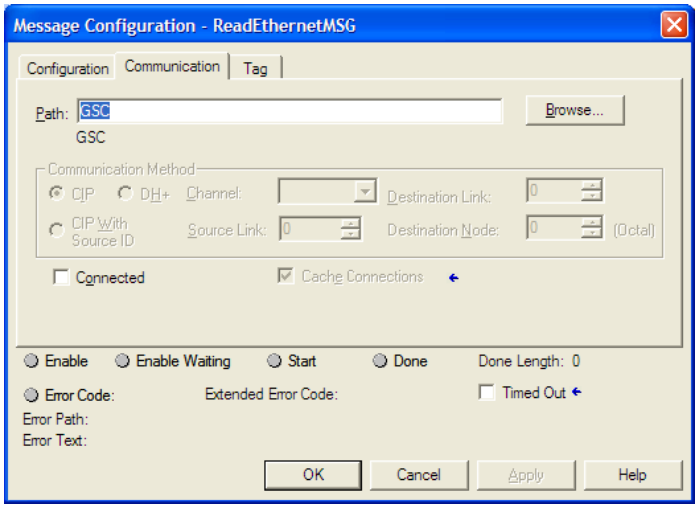


You will notice that new tags have been imported: four **MESSAGE** tags: **WRITEETHERNETMSG**, **WRITECLOCK.MSG**, **READCLOCKMSG,** and **READETHERNETMSG** tags.

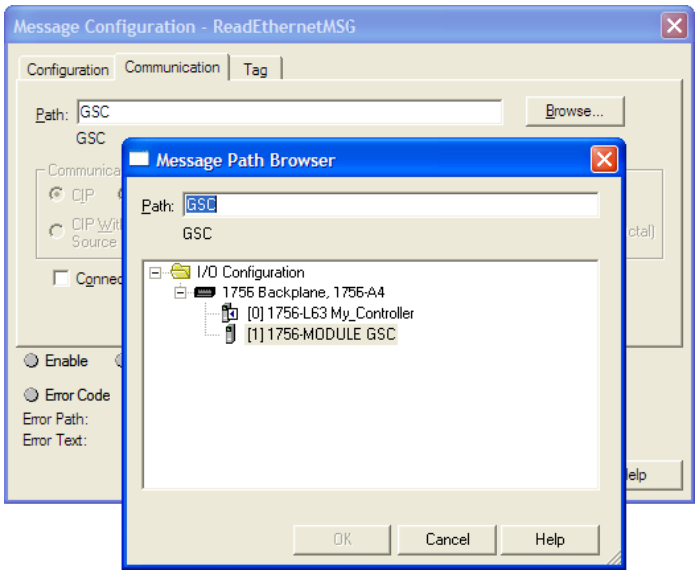


4 In the Optional Add-On Instruction, click the **[...]** button next to each **MSG** tag to open the **MESSAGE CONFIGURATION TAG**.

5 Click the **COMMUNICATION** tab and click the **BROWSE** button as follows.



6 Select the module to configure the message path.



4.6.4 Reading Ethernet Settings from the Module

1 Expand the **MVI56GSCETHERNET** controller tag and move a value of 1 to **MVI56GSCETHERNET.READ.**

MVI56EGSCEthernet	{...}
MVI56EGSCEthernet.Read	1
MVI56EGSCEthernet.Write	0
MVI56EGSCEthernet.Config	{...}
MVI56EGSCEthernet.Config.IP	{...}
MVI56EGSCEthernet.Config.IP[0]	105
MVI56EGSCEthernet.Config.IP[1]	102
MVI56EGSCEthernet.Config.IP[2]	0
MVI56EGSCEthernet.Config.IP[3]	216
MVI56EGSCEthernet.Config.Netmask	{...}
MVI56EGSCEthernet.Config.Netmask[0]	255
MVI56EGSCEthernet.Config.Netmask[1]	255
MVI56EGSCEthernet.Config.Netmask[2]	255
MVI56EGSCEthernet.Config.Netmask[3]	0
MVI56EGSCEthernet.Config.Gateway	{...}
MVI56EGSCEthernet.Config.Gateway[0]	105
MVI56EGSCEthernet.Config.Gateway[1]	102
MVI56EGSCEthernet.Config.Gateway[2]	0
MVI56EGSCEthernet.Config.Gateway[3]	1

2 The bit will be automatically reset and the current Ethernet settings will be copied to **MVI56GSCETHERNET** controller tag as follows.

MVI56EGSCEthernet	{...}
MVI56EGSCEthernet.Read	0
MVI56EGSCEthernet.Write	0
MVI56EGSCEthernet.Config	{...}
MVI56EGSCEthernet.Config.IP	{...}
MVI56EGSCEthernet.Config.IP[0]	105
MVI56EGSCEthernet.Config.IP[1]	102
MVI56EGSCEthernet.Config.IP[2]	0
MVI56EGSCEthernet.Config.IP[3]	216
MVI56EGSCEthernet.Config.Netmask	{...}
MVI56EGSCEthernet.Config.Netmask[0]	255
MVI56EGSCEthernet.Config.Netmask[1]	255
MVI56EGSCEthernet.Config.Netmask[2]	255
MVI56EGSCEthernet.Config.Netmask[3]	0
MVI56EGSCEthernet.Config.Gateway	{...}
MVI56EGSCEthernet.Config.Gateway[0]	105
MVI56EGSCEthernet.Config.Gateway[1]	102
MVI56EGSCEthernet.Config.Gateway[2]	0
MVI56EGSCEthernet.Config.Gateway[3]	1

3 To check the status of the message, refer to the **READETHERNETMSG** tag.

ReadEthernetMSG	{...}
ReadEthernetMSG.Flags	16#0200
ReadEthernetMSG.EW	0
ReadEthernetMSG.ER	0
ReadEthernetMSG.DN	0
ReadEthernetMSG.ST	0
ReadEthernetMSG.EN	0
ReadEthernetMSG.TO	0
ReadEthernetMSG.EN_CC	1
ReadEthernetMSG.ERR	16#0000
ReadEthernetMSG.EXERR	16#0000_0000
ReadEthernetMSG.ERR_SRC	0
ReadEthernetMSG.DN_LEN	0

4.6.5 Writing the Ethernet Settings to the Module

1 Expand the **MVI56GSCETHERNET** controller tag.
2 Set the new Ethernet configuration in **MVI56GSCETHERNET.CONFIG**
3 Move a value of 1 to **MVI56GSCETHERNET.WRITE**

MVI56EGSCEthernet	{...}
MVI56EGSCEthernet.Read	0
MVI56EGSCEthernet.Write	1
MVI56EGSCEthernet.Config	{...}
MVI56EGSCEthernet.Config.IP	{...}
MVI56EGSCEthernet.Config.IP[0]	105
MVI56EGSCEthernet.Config.IP[1]	102
MVI56EGSCEthernet.Config.IP[2]	0
MVI56EGSCEthernet.Config.IP[3]	216
MVI56EGSCEthernet.Config.Netmask	{...}
MVI56EGSCEthernet.Config.Netmask[0]	255
MVI56EGSCEthernet.Config.Netmask[1]	255
MVI56EGSCEthernet.Config.Netmask[2]	255
MVI56EGSCEthernet.Config.Netmask[3]	0
MVI56EGSCEthernet.Config.Gateway	{...}
MVI56EGSCEthernet.Config.Gateway[0]	105
MVI56EGSCEthernet.Config.Gateway[1]	102
MVI56EGSCEthernet.Config.Gateway[2]	0
MVI56EGSCEthernet.Config.Gateway[3]	1

4 After the message is executed, the **MVI56GSCETHERNET.WRITE** bit resets to 0.

MVI56EGSCEthernet	{...}
MVI56EGSCEthernet.Read	0
MVI56EGSCEthernet.Write	0
MVI56EGSCEthernet.Config	{...}
MVI56EGSCEthernet.Config.IP	{...}
MVI56EGSCEthernet.Config.IP[0]	105
MVI56EGSCEthernet.Config.IP[1]	102
MVI56EGSCEthernet.Config.IP[2]	0
MVI56EGSCEthernet.Config.IP[3]	216
MVI56EGSCEthernet.Config.Netmask	{...}
MVI56EGSCEthernet.Config.Netmask[0]	255
MVI56EGSCEthernet.Config.Netmask[1]	255
MVI56EGSCEthernet.Config.Netmask[2]	255
MVI56EGSCEthernet.Config.Netmask[3]	0
MVI56EGSCEthernet.Config.Gateway	{...}
MVI56EGSCEthernet.Config.Gateway[0]	105
MVI56EGSCEthernet.Config.Gateway[1]	102
MVI56EGSCEthernet.Config.Gateway[2]	0
MVI56EGSCEthernet.Config.Gateway[3]	1

5 To check the status of the message, refer to the **WRITEETHERNETMSG** tag.

WriteEthernetMSG	{...}
WriteEthernetMSG.Flags	16#0200
WriteEthernetMSG.EW	0
WriteEthernetMSG.ER	0
WriteEthernetMSG.DN	0
WriteEthernetMSG.ST	0
WriteEthernetMSG.EN	0
WriteEthernetMSG.TO	0
WriteEthernetMSG.EN_CC	1
WriteEthernetMSG.ERR	16#0000
WriteEthernetMSG.EXERR	16#0000_0000
WriteEthernetMSG.ERR_SRC	0
WriteEthernetMSG.DN_LEN	0
WriteEthernetMSG.REQ_LEN	24

4.6.6 Reading the Clock Value from the Module

1 Expand the **MVI56GSCCLOCK** controller tag and move a value of 1 to **MVI56GSCCLOCK.READ**

MVI56EGSCClock	{...}
MVI56EGSCClock.Read	1
MVI56EGSCClock.Write	0
MVI56EGSCClock.Config	{...}
MVI56EGSCClock.Config.Year	0
MVI56EGSCClock.Config.Month	0
MVI56EGSCClock.Config.Day	0
MVI56EGSCClock.Config.Hour	0
MVI56EGSCClock.Config.Minute	0
MVI56EGSCClock.Config.Seconds	0

2 The bit will be automatically reset and the current clock value will be copied to **MVI56GSCCLOCK.CONFIG** controller tag as follows.

MVI56EGSCClock	{...}
MVI56EGSCClock.Read	0
MVI56EGSCClock.Write	0
MVI56EGSCClock.Config	{...}
MVI56EGSCClock.Config.Year	2009
MVI56EGSCClock.Config.Month	9
MVI56EGSCClock.Config.Day	18
MVI56EGSCClock.Config.Hour	3
MVI56EGSCClock.Config.Minute	33
MVI56EGSCClock.Config.Seconds	52

3 To check the status of the message, refer to the **READCLOCKMSG** tag.

ReadClockMSG	{...}
ReadClockMSG.Flags	16#0200
ReadClockMSG.EW	0
ReadClockMSG.ER	0
ReadClockMSG.DN	0
ReadClockMSG.ST	0
ReadClockMSG.EN	0
ReadClockMSG.TO	0
ReadClockMSG.EN_CC	1
ReadClockMSG.ERR	16#0000
ReadClockMSG.EXERR	16#0000_0000
ReadClockMSG.ERR_SRC	0
ReadClockMSG.DN_LEN	0

4.6.7 Writing the Clock Value to the Module

1 Expand the **MVI56GSCCLOCK** controller tag.
2 Set the new Clock value in **MVI56GSCCLOCK.CONFIG**
3 Move a value of 1 to **MVI56GSCCLOCK.WRITE**

MVI56MCMClock	{...}
MVI56MCMClock.Read	0
MVI56MCMClock.Write	1
MVI56MCMClock.Config	{...}
MVI56MCMClock.Config.Year	2008
MVI56MCMClock.Config.Month	11
MVI56MCMClock.Config.Day	12
MVI56MCMClock.Config.Hour	15
MVI56MCMClock.Config.Minute	38
MVI56MCMClock.Config.Seconds	9

4 The bit will be automatically reset to 0.

MVI56MCMClock	{...}
MVI56MCMClock.Read	0
MVI56MCMClock.Write	0
MVI56MCMClock.Config	{...}
MVI56MCMClock.Config.Year	2008
MVI56MCMClock.Config.Month	11
MVI56MCMClock.Config.Day	12
MVI56MCMClock.Config.Hour	15
MVI56MCMClock.Config.Minute	38
MVI56MCMClock.Config.Seconds	9

5 To check the status of the message, refer to the **WRITECLOCKMSG** tag.

WriteClockMSG	{...}
WriteClockMSG.Flags	16#0220
WriteClockMSG.EW	0
WriteClockMSG.ER	0
WriteClockMSG.DN	1
WriteClockMSG.ST	0
WriteClockMSG.EN	0
WriteClockMSG.TO	0
WriteClockMSG.EN_CC	1
WriteClockMSG.ERR	16#0000
WriteClockMSG.EXERR	16#0000_0000
WriteClockMSG.ERR_SRC	0
WriteClockMSG.DN_LEN	0
WriteClockMSG.REQ_LEN	24

4.7 Using the Sample Program - RSLogix 5000 Version 15 and earlier

The sample program included with your MVI56E-GSC module contains predefined controller tags, configuration information, data types, and ladder logic that allow the module to communicate between the ControlLogix processor and a network of generic ASCII serial devices. For most applications, the sample program will work without modification.

4.7.1 Opening the Sample Program in RSLogix

The sample program for your MVI56E-GSC module includes custom tags, data types and ladder logic for data I/O, status and command control. For most applications, you can run the sample program without modification, or, for advanced applications, you can incorporate the sample program into your existing application.

Download the manuals and sample program from the ProSoft Technology web site

You can always download the latest version of the sample ladder logic and user manuals for the MVI56E-GSC module from the ProSoft Technology website, at www.prosoft-technology.com.

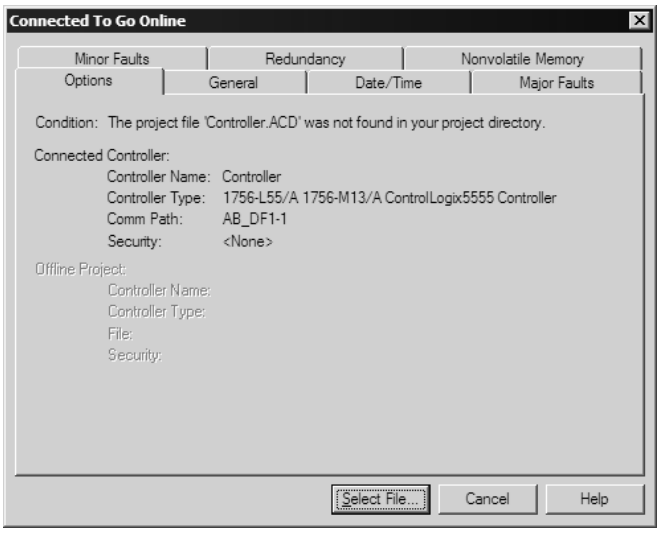
From that link, navigate to the download page for your module and choose the sample program to download for your version of RSLogix 5000 and your processor.

To determine the firmware version of your processor

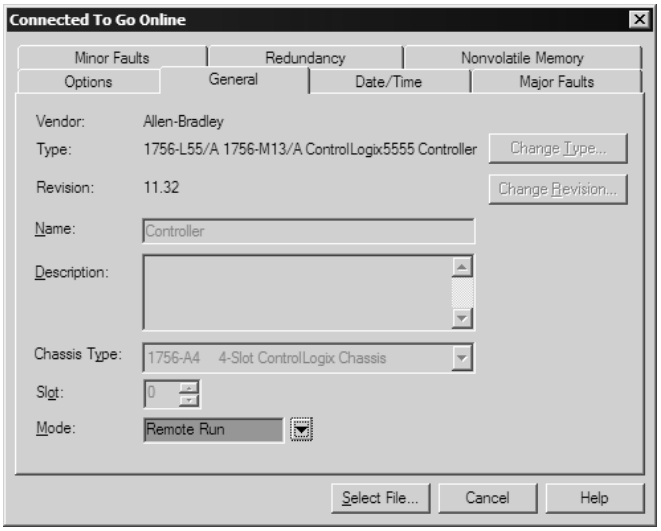
Important: The RSLinx service must be installed and running on your computer in order for RSLogix to communicate with the processor. Refer to your RSLinx and RSLogix documentation for help configuring and troubleshooting these applications.

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

4 When RSLogix has established communication with the processor, the *Connected To Go Online* dialog box will open.



5 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 11.32



6 Select the sample ladder logic file for your firmware version.

To open the sample program

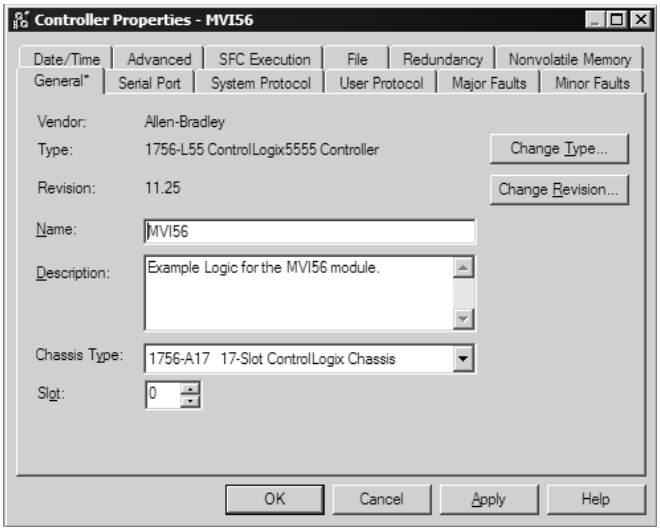
1. On the *Connected to Go Online* dialog box, click the **SELECT FILE** button.
2. Choose the sample program file that matches your firmware version, and then click the **SELECT** button.
3. RSLogix will load the sample program.

The next step is to configure the correct controller type and slot number for your application.

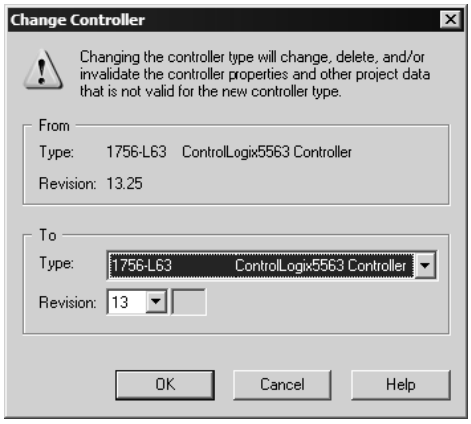
4.7.2 Choosing the Controller Type

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

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



3. Click the **CHANGE TYPE** or **CHANGE CONTROLLER** button. This action opens the *Change Controller* dialog box.



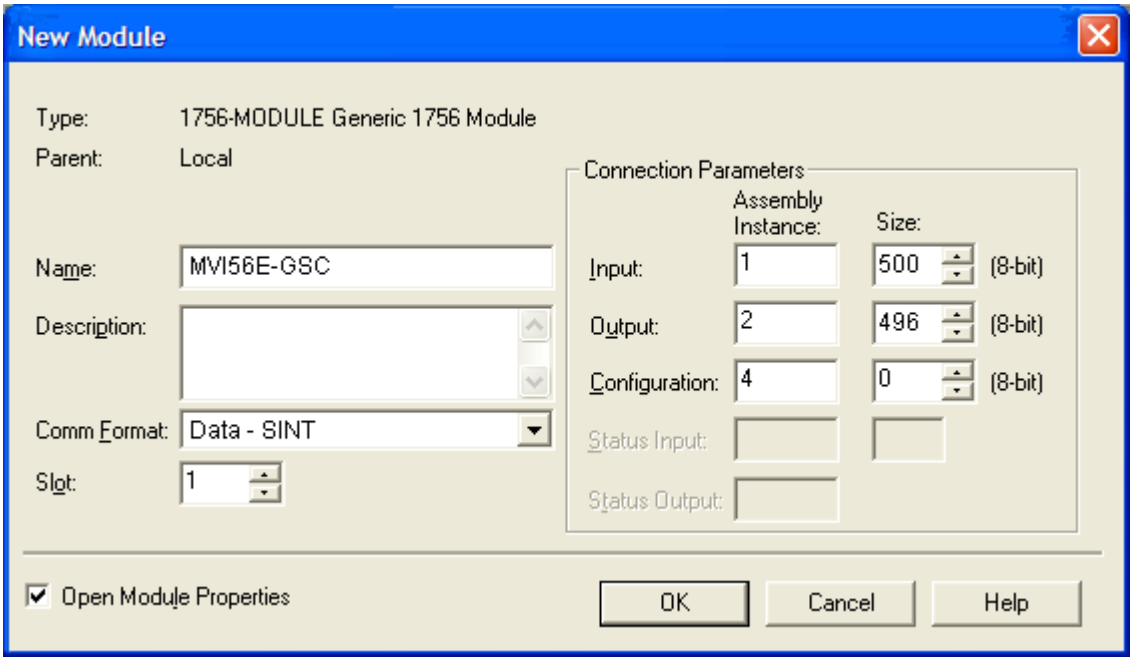
4. Open the **TYPE** dropdown list, and then select your ControlLogix controller.
5. Select the correct firmware revision for your controller, if necessary.
6. Click **OK** to save your changes and return to the previous window.

4.7.3 Select the Slot Number for the Module

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

To change the slot number

1. In the **CONTROLLER ORGANIZATION** list, select the module **[1] 1756-MODULE MVI56,** and then click the right mouse button to open a shortcut menu.
2. On the shortcut menu, choose **PROPERTIES.** This action opens the **MODULE PROPERTIES** dialog box.



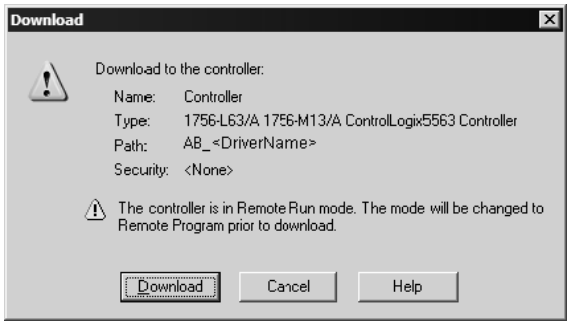
3. In the **SLOT** field, use the up and down arrows on the right side of the field to select the slot number where the module will reside in the rack, and then click **OK.**

RSLogix will automatically apply the slot number change to all tags, variables and ladder logic rungs that use the MVI56E-GSC slot number for computation.

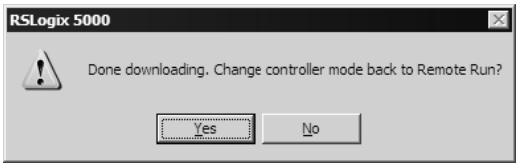
4.7.4 Downloading the Sample Program to the Processor

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

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



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

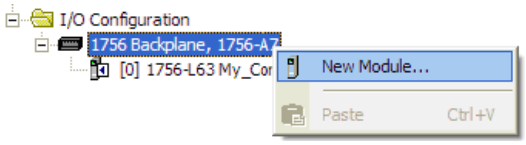


Note: If you receive an error message during these steps, refer to your RSLogix documentation to interpret and correct the error.

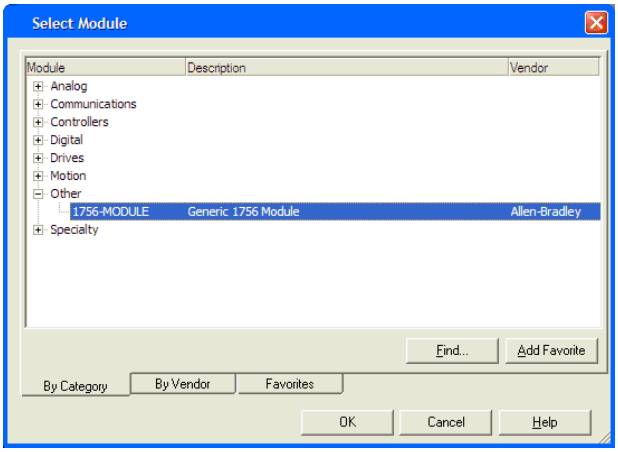
4.8 Adding the Sample Ladder to an Existing Application

Important: The following steps describe how to install and configure the MVI56E-GSC module with RSLogix 5000 version 15 or older. If you are using RSLogix 5000 version 16, please refer to Sample Add-On Instruction Import Procedure.

1 Add the MVI56E-GSC module to the project. Right-click the mouse button on the **I/O CONFIGURATION** option in the *Controller Organization* window to display a pop-up menu. Select the **NEW MODULE** option from the *I/O Configuration* menu.

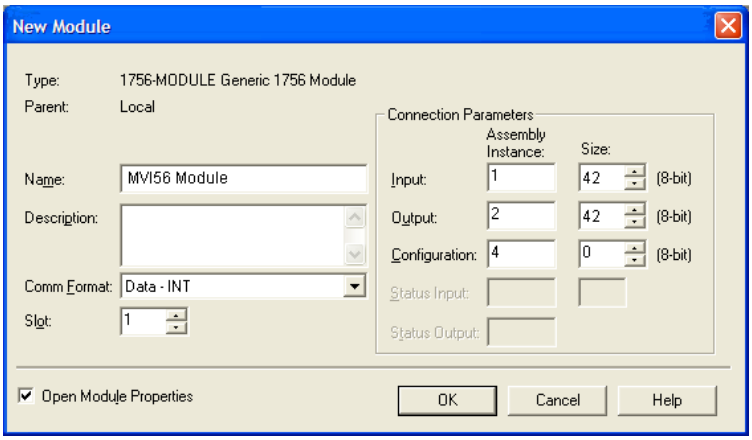


2 Select **1756-MODULE**.

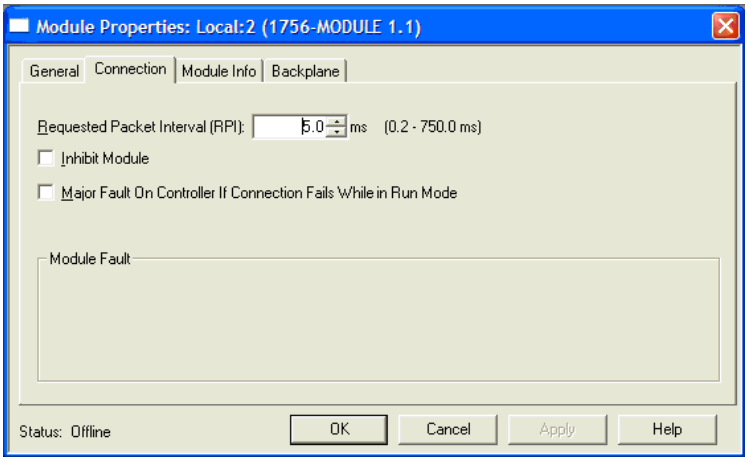


3 Set the *Module Properties* values as follows:

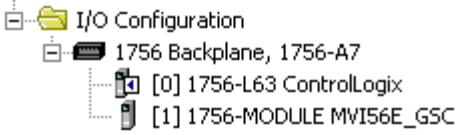
Parameter	Value
Name	Enter a module identification string. Example: MVI56GSC.
Description	Enter a description for the module. Example: ProSoft communication module for General Serial Communication.
Comm Format	Select DATA-SINT.
Slot	Enter the slot number in the rack where the MVI56-GSC module is located.
Input Assembly Instance	1
Input Size	500
Output Assembly Instance	2
Output Size	496
Configuration Assembly Instance	4
Configuration Size	0



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



The MVI56E-GSC module is now visible at the I/O Configuration section.



To complete the process, perform the following steps in this order:

1 Copy the User-defined Data Types from the sample program.
2 Copy the Controller Tags from the sample program.
3 Copy the Ladder Rungs from the sample program.
4 Save and Download the new application to the controller and place the processor in RUN mode.

4.9 Error/Configuration Word

If the module is configured correctly, the configuration error word should have a value of zero. Any other value indicates a configuration error. Use the value in the configuration error word to determine which set of parameters are invalid in the driver configuration area. The following table lists the bits associated with each configuration error in the word:

Bit	Code	Description
0	0x0001	Invalid Cfg_type
1	0x0002	Invalid Cfg_Baud
2	0x0004	Invalid Cfg_Parity
3	0x0008	Invalid Cfg_DataBits
4	0x0010	Invalid Cfg_StopBits
5	0x0020	Invalid Cfg_Handshake
6	0x0040	Invalid Cfg_RTermcount
7	0x0080	Invalid Cfg_RPacketLen
8	0x0100	Invalid Cfg_RTimeout
9	0x0200	Invalid Cfg_RDelay
10	0x0400	Invalid Cfg_WTimeout
11	0x0800	Invalid Resv_Bit11
12	0x1000	Invalid Resv_Bit12
13	0x2000	Invalid Resv_Bit13
14	0x4000	Invalid Resv_Bit14
15	0x8000	Invalid Err_ROverflow

4.10 Cable Connections

The application ports on the MVI56E-GSC module support RS-232, RS-422, and RS-485 interfaces. Please inspect the module to ensure that the jumpers are set correctly to correspond with the type of interface you are using.

Note: When using RS-232 with radio modem applications, some radios or modems require hardware handshaking (control and monitoring of modem signal lines). Enable this in the configuration of the module by setting the UseCTS parameter to 1.

4.10.1 Ethernet Cable Specifications

The recommended cable is Category 5 or better. A Category 5 cable has four twisted pairs of wires, which are color-coded and cannot be swapped. The module uses only two of the four pairs.

The Ethernet ports on the module are Auto-Sensing. You can use either a standard Ethernet straight-through cable or a crossover cable when connecting the module to an Ethernet hub, a 10/100 Base-T Ethernet switch, or directly to a PC. The module will detect the cable type and use the appropriate pins to send and receive Ethernet signals.

Ethernet cabling is like U.S. telephone cables, except that it has eight conductors. Some hubs have one input that can accept either a straight-through or crossover cable, depending on a switch position. In this case, you must ensure that the switch position and cable type agree.

Refer to Ethernet Cable Configuration (page 130) for a diagram of how to configure Ethernet cable.

4.10.2 Ethernet Performance

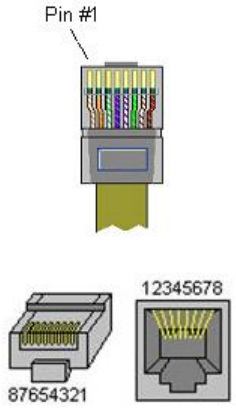
High Ethernet traffic may impact GSC performance, consider one of these options:

- Use managed switches to reduce traffic coming to module port
- Use CIPconnect for these applications and disconnect the module Ethernet port from the network

4.10.3 Ethernet Cable Configuration

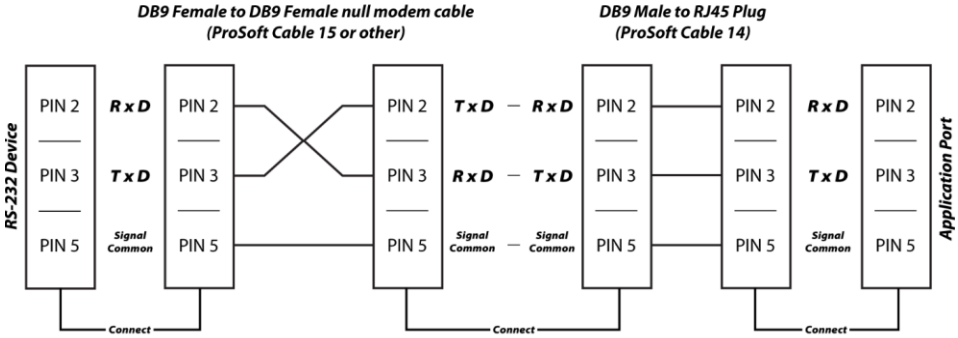
Note: The standard connector view shown is color-coded for a straight-through cable.

Crossover cable			Straight- through cable	
RJ-45 PIN	**RJ-45 PIN**		**RJ-45 PIN**	**RJ-45 PIN**
1 Rx+	3 Tx+		1 Rx+	1 Tx+
2 Rx-	6 Tx-		2 Rx-	2 Tx-
3 Tx+	1 Rx+		3 Tx+	3 Rx+
6 Tx-	2 Rx-		6 Tx-	6 Rx-



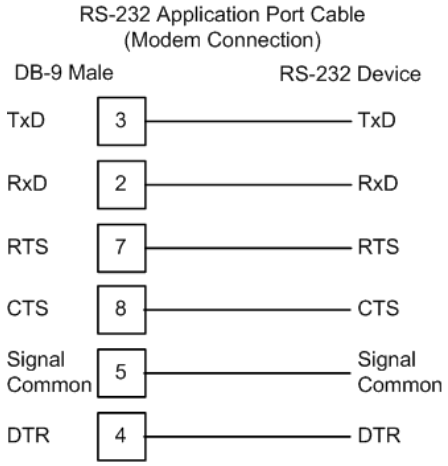
4.10.4 RS-232 Application Port(s)

When the RS-232 interface is selected, the use of hardware handshaking (control and monitoring of modem signal lines) is user definable. If no hardware handshaking will be used, here are the cable pinouts to connect to the port.



RS-232: Modem Connection (Hardware Handshaking Required)

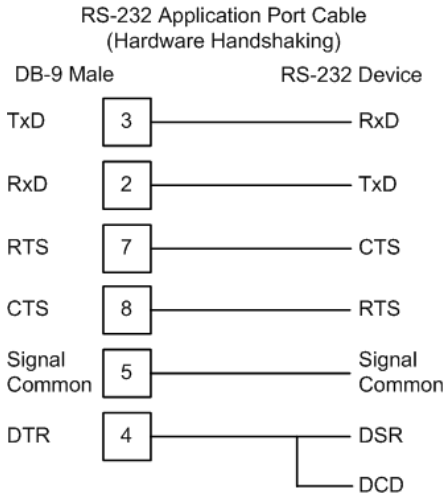
This type of connection is required between the module and a modem or other communication device.



The "Use CTS Line" parameter for the port configuration should be set to 'Y' for most modem applications.

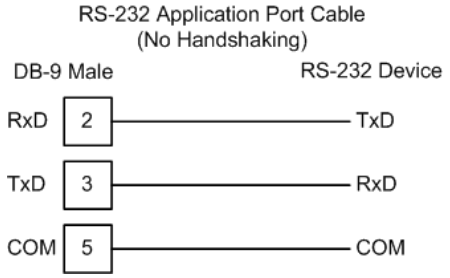
RS-232: Null Modem Connection (Hardware Handshaking)

This type of connection is used when the device connected to the module requires hardware handshaking (control and monitoring of modem signal lines).

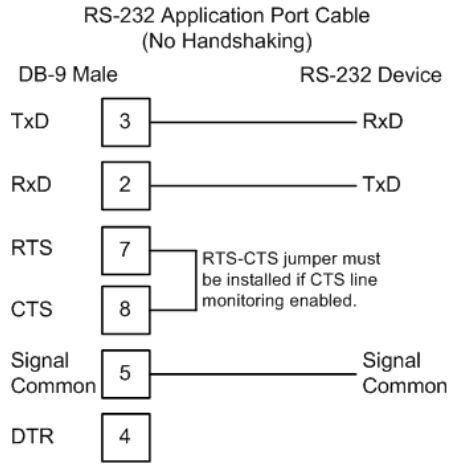


RS-232: Null Modem Connection (No Hardware Handshaking)

This type of connection can be used to connect the module to a computer or field device communication port.

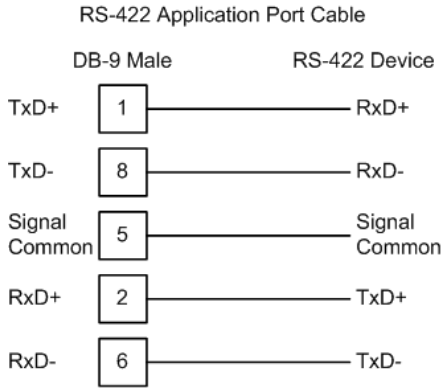


Note: For most null modem connections where hardware handshaking is not required, the *Use CTS Line* parameter should be set to **N** and no jumper will be required between Pins 7 (RTS) and 8 (CTS) on the connector. If the port is configured with the *Use CTS Line* set to **Y**, then a jumper is required between the RTS and the CTS lines on the port connection.



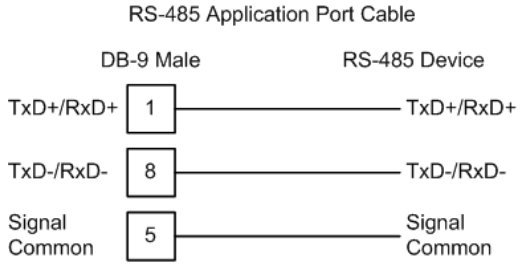
4.10.5 RS-422

The RS-422 interface requires a single four or five wire cable. The Common connection is optional, depending on the RS-422 network devices used.



4.10.6 RS-485 Application Port(s)

The RS-485 interface requires a single two or three wire cable. The Common connection is optional, depending on the RS-485 network devices used.



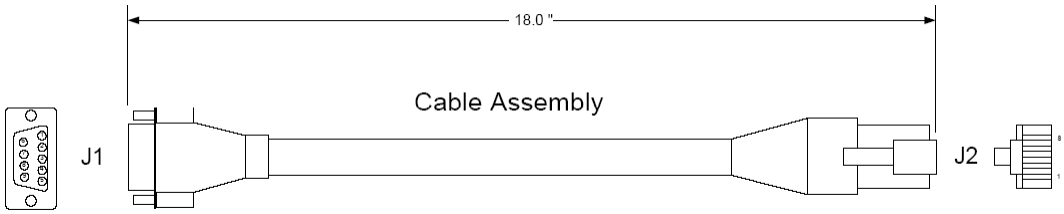
Note: This type of connection is commonly called a RS-485 half-duplex, 2-wire connection. If you have RS-485 4-wire, full-duplex devices, they can be connected to the gateway's serial ports by wiring together the TxD+ and RxD+ from the two pins of the full-duplex device to Pin 1 on the gateway and wiring together the TxD- and RxD- from the two pins of the full-duplex device to Pin 8 on the gateway. As an alternative, you could try setting the gateway to use the RS-422 interface and connect the full-duplex device according to the RS-422 wiring diagram. For additional assistance, please contact ProSoft Technical Support.

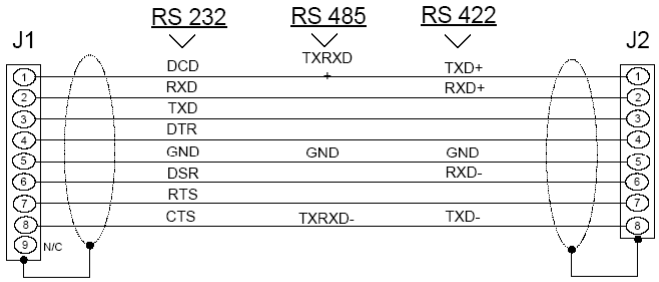
Note: Depending upon devices on the network, if there are problems in RS-485 communication that can be attributed to the signal echoes or reflections, then consider adding 120 OHM terminating resistors at both ends of the RS-485 line.

RS-485 and RS-422 Tip

If communication in the RS-422 or RS-485 mode does not work at first, despite all attempts, try switching termination polarities. Some manufacturers interpret + and -, or A and B, polarities differently.

4.10.7 DB9 to RJ45 Adaptor (Cable 14)





Wiring Diagram

5 Support, Service & Warranty

5.1 Contacting Technical Support

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

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

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

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

North America (Corporate Location)	Europe / Middle East / Africa Regional Office
Phone: +1 661-716-5100 ps.prosofttechnology@belden.com Languages spoken: English, Spanish REGIONAL TECH SUPPORT ps.support@belden.com	Phone: +33.(0)5.34.36.87.20 ps.europe@belden.com Languages spoken: English, French, Hindi, Italian REGIONAL TECH SUPPORT ps.support.emea@belden.com
Latin America Regional Office	**Asia Pacific Regional Office**
Phone: +52.222.264.1814 ps.latinam@belden.com Languages spoken: English, Spanish, Portuguese REGIONAL TECH SUPPORT ps.support.la@belden.com	Phone: +60.3.2247.1898 ps.asiapc@belden.com Languages spoken: Bahasa, Chinese, English, Hindi, Japanese, Korean, Malay REGIONAL TECH SUPPORT ps.support.ap@belden.com

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

5.2 Warranty Information

For details regarding ProSoft Technology's legal terms and conditions, please see:
www.prosoft-technology.com/ProSoft-Technology-Legal-Terms-and-Conditions

For Return Material Authorization information, please see:
www.prosoft-technology.com/Services-Support/Return-Material-Instructions