ProSoft® TECHNOLOGY

Where Automation Connects.

# MVI56E-GEC

**ControlLogix® Platform**

Generic ASCII Ethernet
Communication Module

August 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-GEC User Manual
For Public Use.

August 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.

© 2025 ProSoft Technology. 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.

**Warning** – Cancer and Reproductive Harm – www.P65Warnings.ca.gov

## Agency Approvals & 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

# 1 Start Here

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

- **Studio 5000 Logix Designer** ®: 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 Ethernet ASCII and ControlLogix devices to a power source and to the MVI56E-GEC module's Ethernet port

## 1.1 System Requirements

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

- Rockwell Automation ControlLogix® processor (firmware version 10 or higher), with compatible power supply, and one free slot in the rack for the MVI56E-GEC module. The module requires 800 mA of available 5 VDC power and 3 mA of available 24 VDC power.
- Rockwell Automation Studio 5000 Logix Designer 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)
- ProSoft Discovery Service (PDS) (included in PCB)
- Pentium® II 450 MHz minimum.
- 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

**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.

## 1.2 Package Contents

| Qty. | Part Name | Part Number | Part Description |
|---|---|---|---|
| 1 | MVI56E-GEC Module | MVI56E-GEC | Generic ASCII Ethernet Communication Module |

## 1.3 Installing ProSoft Configuration Builder Software

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

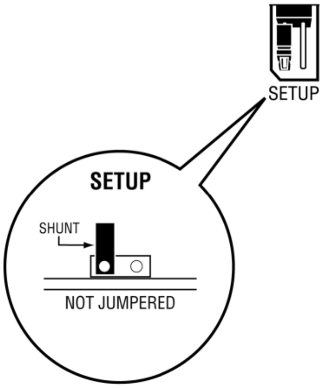
***Installing ProSoft Configuration Builder from the ProSoft website***

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

## 1.4 Setting Jumpers

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

The following illustration shows the MVI56E-GEC jumper configuration.



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

## 1.5 Installing the Module in the Rack

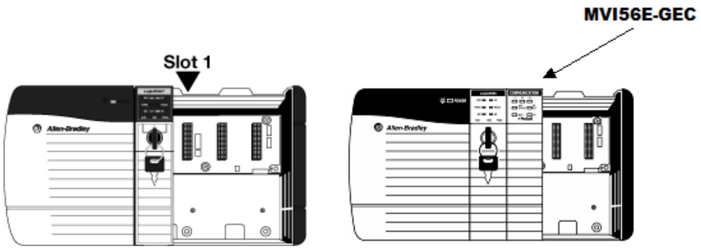
If you have not already installed and configured your ControlLogix processor and power supply, please do so before installing the MVI56E-GEC module. Refer to your Rockwell Automation product documentation for installation instructions.

**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 MVI56E-GEC into the ControlLogix chassis. Use the same technique recommended by Rockwell Automation to remove and install ControlLogix modules.

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

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



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

## 1.6 Opening the Sample Ladder Logic

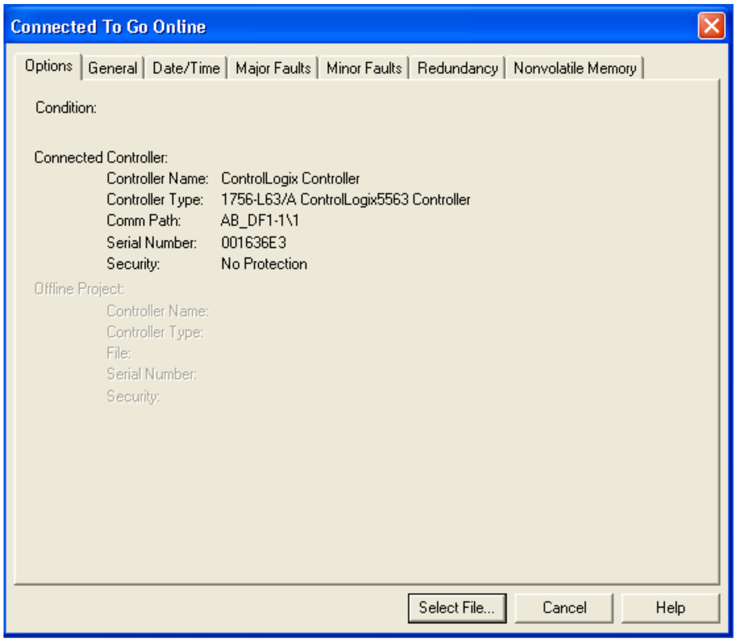
The sample program for your MVI56E-GEC module includes custom tags, data types and ladder logic for data I/O and status monitoring. For most applications, you can run the sample ladder program without modification, or, for advanced applications, you can incorporate the sample program into your existing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

Please download the latest MVI56E-GEC ladder logic file from:
www.prosoft-technology.com.

### *1.6.1 Determining the Firmware Version of Your Processor*

**Important:** The RSLinx service must be installed and running on your computer in order for Studio 5000 Logix Designer to communicate with the processor. Refer to your RSLinx and Studio 5000 Logix Designer documentation for help configuring and troubleshooting these applications.

1 Start Studio 5000 Logix Designer® and close any existing project that may be loaded.
2 Open the **COMMUNICATIONS** menu and choose **GO ONLINE**. Studio 5000 Logix Designer will establish communication with the processor. This may take a few moments.
3 When Studio 5000 Logix Designer has established communication with the processor, the *Connected To Go Online* dialog box will open.

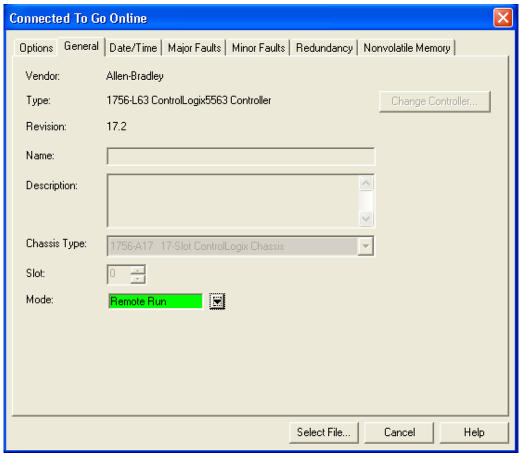


Condition:

Connected Controller:
Controller Name: ControlLogix Controller
Controller Type: 1756-L63/A ControlLogix5563 Controller
Comm Path: AB_DF1-1\1
Serial Number: 001636E3
Security: No Protection

Offline Project:
Controller Name:
Controller Type:
File:
Serial Number:
Security:

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



Vendor: Allen-Bradley
Type: 1756-L63 ControlLogix5563 Controller
Revision: 17.2
Name:
Description:
Chassis Type: 1756-A17 17-Slot ControlLogix Chassis
Slot: 0
Mode: Remote Run

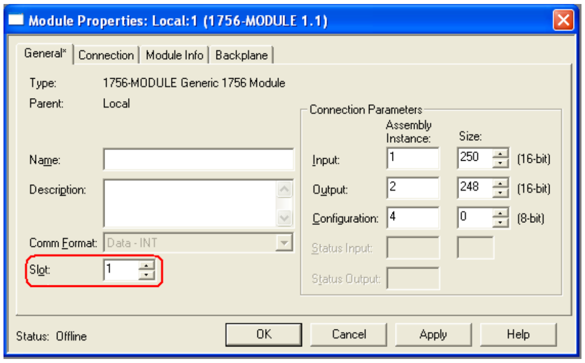
### 1.6.2 Selecting the Slot Number for the Module

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

#### ***Changing the slot number***

1 In the *Controller Organization* list, select the module and then click the right mouse button to open a shortcut menu.

2 On the shortcut menu, choose **PROPERTIES**. This action opens the *Module Properties* dialog box.



Type: 1756-MODULE Generic 1756 Module
Parent: Local
Name:
Description:
Comm Format: Data - INT
Slot: 1

Connection Parameters

| | Assembly Instance: | Size: | |
|---|---|---|---|
| Input: | 1 | 250 | (16-bit) |
| Output: | 2 | 248 | (16-bit) |
| Configuration: | 4 | 0 | (8-bit) |
| Status Input: | | | |
| Status Output: | | | |

Status: Offline

3 In the *Slot* field, adjust the slot location and then click **OK**.

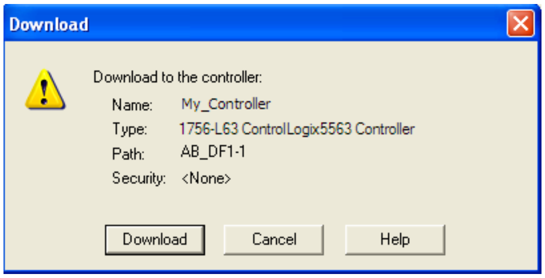
Studio 5000 Logix Designer will automatically apply the slot number change to all tags, variables and ladder logic rungs that use the MVI56E-GEC slot number for computation.

## 1.7 Downloading the Sample Program to the Processor

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

**1** If you are not already online with the processor, open the *Communications* menu, and then choose **DOWNLOAD**. Studio 5000 Logix Designer will establish communication with the processor.

**2** When communication is established, Studio 5000 Logix Designer will open a confirmation dialog box. Click the **DOWNLOAD** button to transfer the sample program to the processor.



Download

Download to the controller:

Name: My_Controller

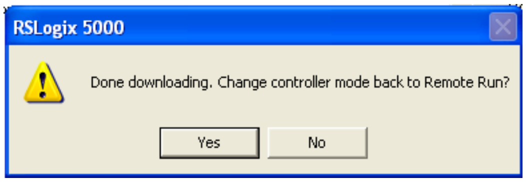
Type: 1756-L63 ControlLogix5563 Controller

Path: AB_DF1-1

Security: <None>

**3** Studio 5000 Logix Designer will compile the program and transfer it to the processor. This process may take a few minutes.

**4** When the download is complete, Studio 5000 Logix Designer 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.



RSLogix 5000

Done downloading. Change controller mode back to Remote Run?

**Note:** If you receive an error message during these steps, refer to your Studio 5000 Logix Designer documentation to interpret and correct the error.

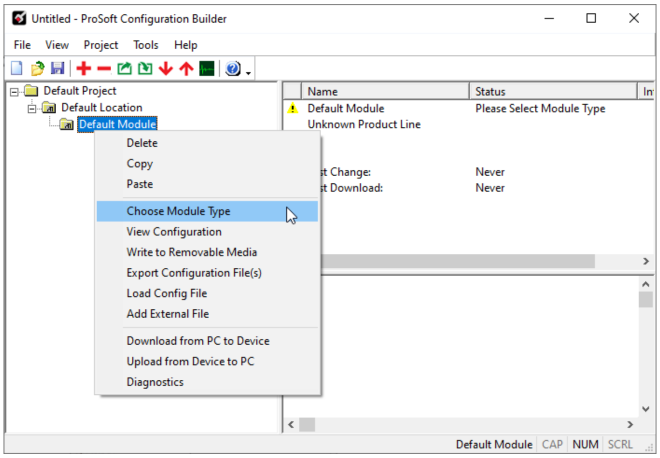
# 2 Using ProSoft Configuration Builder

## 2.1 ProSoft Configuration Builder

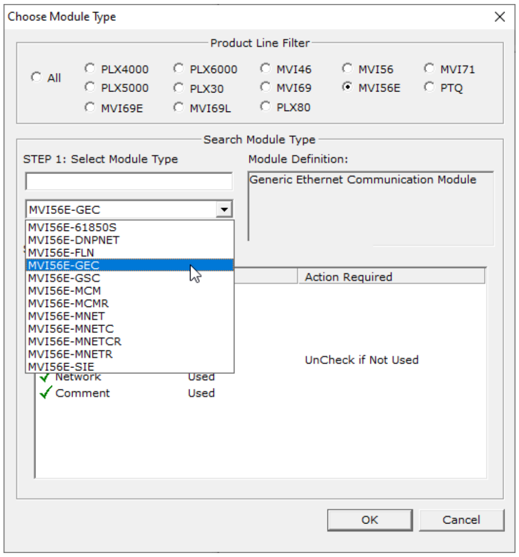
*ProSoft Configuration Builder (PCB)* provides a quick and easy way to manage module configuration files customized to meet your application needs. PCB also allows you to import information from previously installed (known working) configurations to new projects.

### 2.1.1 *Adding the MVI56E-GEC Module to the Project*

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



2 In the *Product Line Filter* area of the dialog box, select **MVI56E**. In the *Select Module Type* dropdown list, select **MVI56E-GEC**, and then click **OK**.

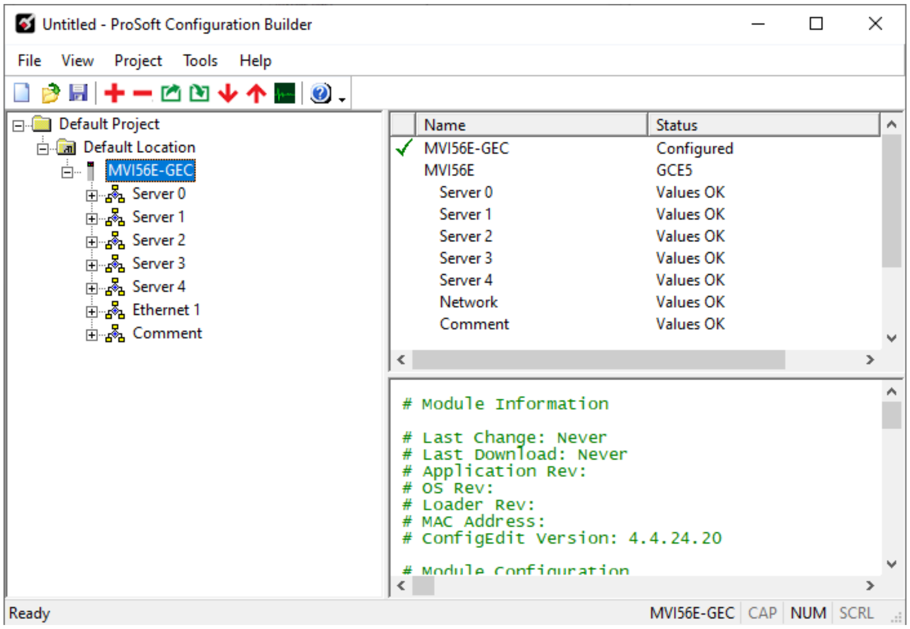


STEP 1: Select Module Type: MVI56E-GEC

Module Definition: Generic Ethernet Communication Module

### 2.1.2 Renaming PCB Objects

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



| | Name | Status |
|---|---|---|
| ✓ | MVI56E-GEC | Configured |
| | MVI56E | GCE5 |
| | Server 0 | Values OK |
| | Server 1 | Values OK |
| | Server 2 | Values OK |
| | Server 3 | Values OK |
| | Server 4 | Values OK |
| | Network | Values OK |
| | Comment | Values OK |

```
# Module Information

# Last Change: Never
# Last Download: Never
# Application Rev:
# OS Rev:
# Loader Rev:
# MAC Address:
# ConfigEdit Version: 4.4.24.20

# Module Configuration
```

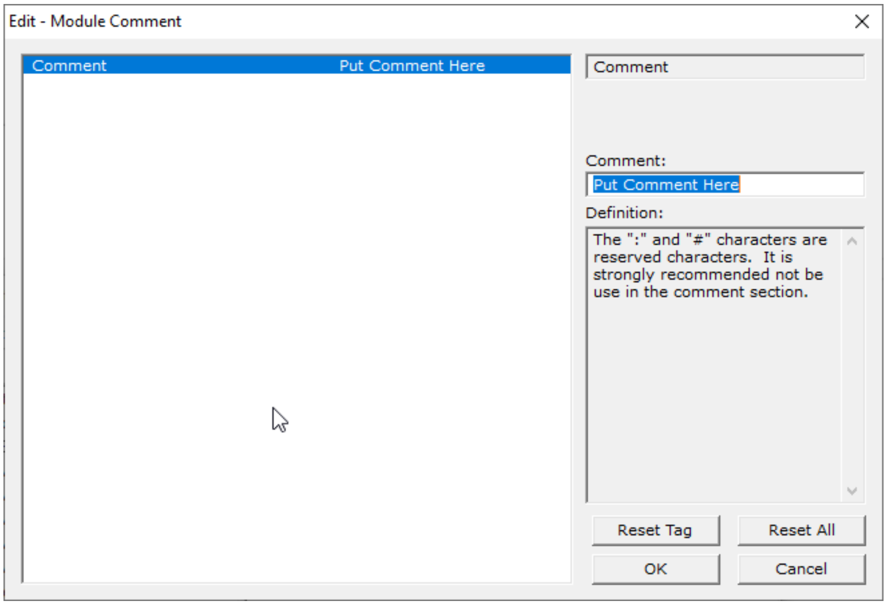
1 Select the object, and then click the right mouse button to open a shortcut menu. From the shortcut menu, choose **RENAME.**
2 Type the name to assign to the object.
3 Click *away* from the object to save the new name.

<u>*Configuring Module Parameters*</u>

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

### *Creating Optional Comment Entries*

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



| Comment | Put Comment Here |
|---|---|

Comment

Comment: Put Comment Here

Definition:
The ":" and "#" characters are reserved characters. It is strongly recommended not be use in the comment section.

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

### *Printing a Configuration File*

1 Select the module icon, and then click the right mouse button to open a shortcut menu.
2 On the shortcut menu, choose **VIEW CONFIGURATION.** This action opens the *View Configuration* window.
3 In the *View Configuration* window, open the **FILE** menu, and choose **PRINT.** This action opens the *Print* dialog box.
4 In the *Print* dialog box, choose the printer to use from the drop-down list, select printing options, and then click **OK.**

## 2.2 [Server x]

You can configure up to five servers ([Server 0] through [Server 4]). The configuration section for each server contains the same set of parameters. You can configure the parameters for each server to meet the requirements of your application.

### *2.2.1 Enabled*

Yes or No

This parameter determines if the server will be utilized by the module. If a value of "Yes" is entered, the server will be used. Any other value will disable the server.

### *2.2.2 Service Port Number*

1 to 65535

This parameter sets the TCP/IP service port for this server. Each server can have its own unique service port or can share the same number with other servers.

### *2.2.3 Connection Timeout*

0 or 5000 to 65535

This parameter specifies the number of milliseconds the server will permit the server to be inactive after a connection is made before closing the socket. This timeout period is reset on each read or write packet. If the parameter is set to 0, the connection will not timeout.

### *2.2.4 Connection Close Type*

0, 1 or 2

This coded parameter defines the personality of the server after a connection is made. If the parameter is set to 0, the socket will only be closed when a request from the client is received or the connection timeout is exceeded. If a value of 1 is selected, the server will close the socket after it transmits a single message. If a value of 2 is selected, the server will close the socket after it receives a message.

### *2.2.5 Swap Rx Data Bytes*

Yes or No

This parameter determines if the data received by the server will have the byte order of the data swapped. If the parameter is set to No, no byte swapping will occur. If the parameter is set to Yes, the odd byte will be swapped with the even byte in each word of data received.

### *2.2.6 Swap Tx Data Bytes*

Yes or No

This parameter determines if the data to be transmitted by the module will have the byte order of the data swapped. If the parameter is set to No, no byte swapping will occur. If the parameter is set to Yes, the odd byte will be swapped with the even byte in each word of data received.

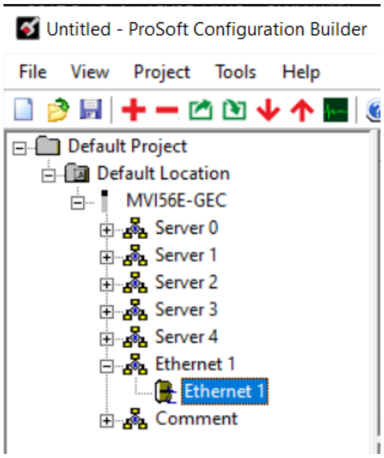
## 2.3 Ethernet 1 Configuration

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

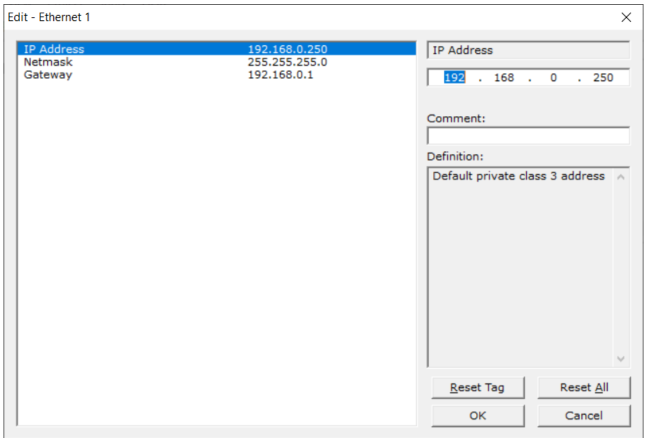
**1** Determine the network settings for your module, with the help of your network administrator if necessary. You will need the following information: IP address (fixed IP required), Subnet mask, Gateway address

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

**2** Double-click the **ETHERNET 1** icon.



**3** This action opens the *Edit – Ethernet 1* dialog box.


Edit - Ethernet 1

| | |
|---|---|
| IP Address | 192.168.0.250 |
| Netmask | 255.255.255.0 |
| Gateway | 192.168.0.1 |

IP Address

192 . 168 . 0 . 250

Comment:

Definition:

Default private class 3 address

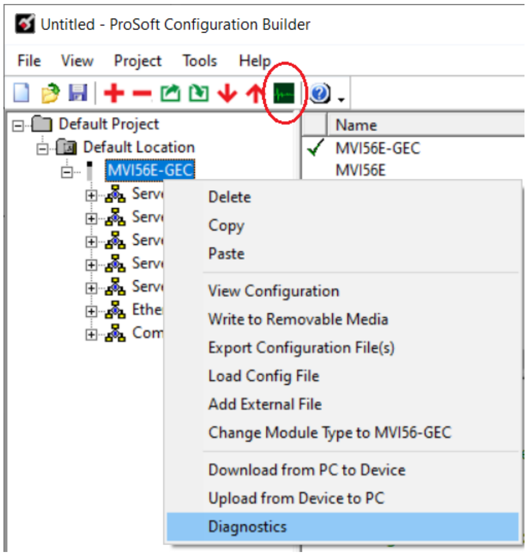
**4** Edit the values for *my_ip*, *netmask* (subnet mask) and *gateway* (default gateway).

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

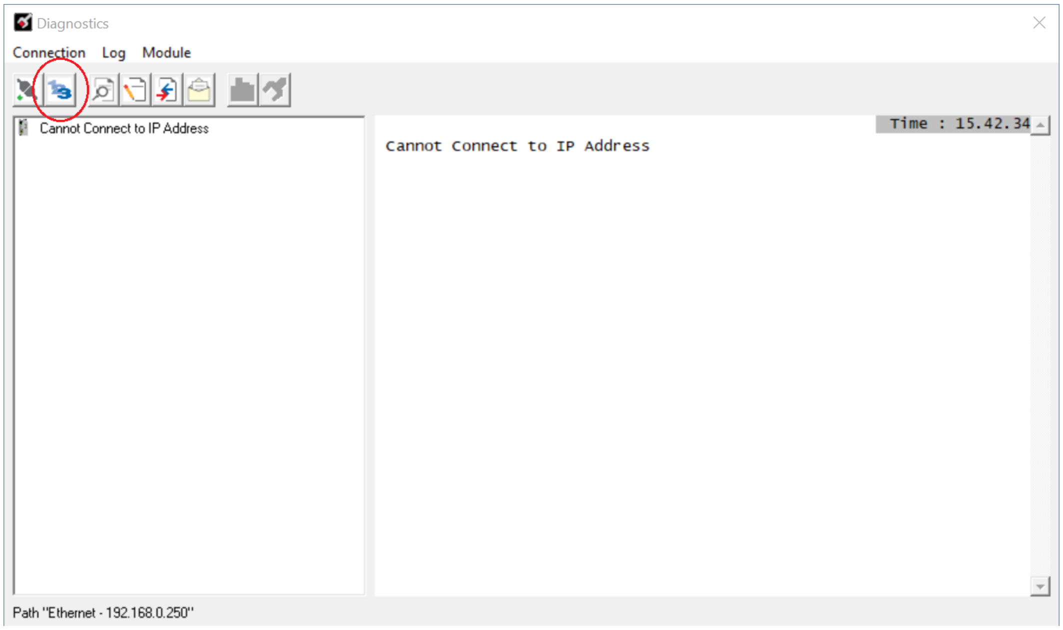
## 2.4 Assigning a Temporary IP Address

You can assign a temporary IP address to the module using the ProSoft Discovery Service utility. This utility is installed within ProSoft Configuration Builder (PCB).

1 In PCB, click on the *Diagnostics* icon in the shortcuts menu, or right-click on the module's icon and select **DIAGNOSTICS**.



2 The *Diagnostics* dialog opens. Click on the *Connection Setup* icon, or select **CONNECTION > CONNECTION SETUP**.

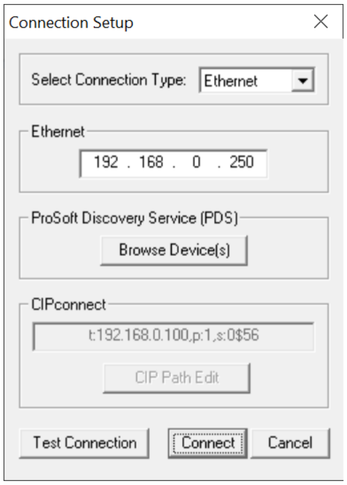


Cannot Connect to IP Address

```
Time : 15.42.34
Cannot Connect to IP Address
```

Path "Ethernet - 192.168.0.250"

**3** In the *Connection Setup* dialog box, click **BROWSE DEVICE(S)** to start *ProSoft Discovery Service***.**

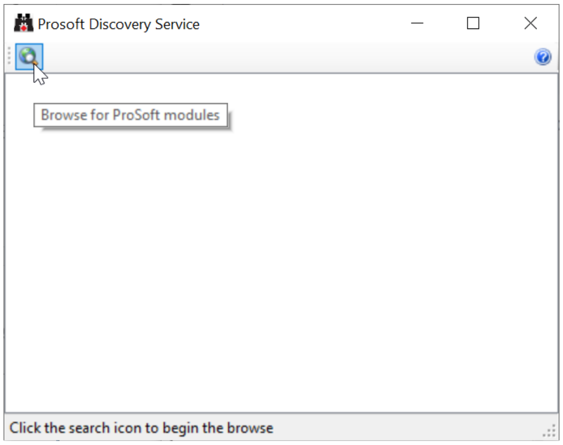


Select Connection Type: Ethernet

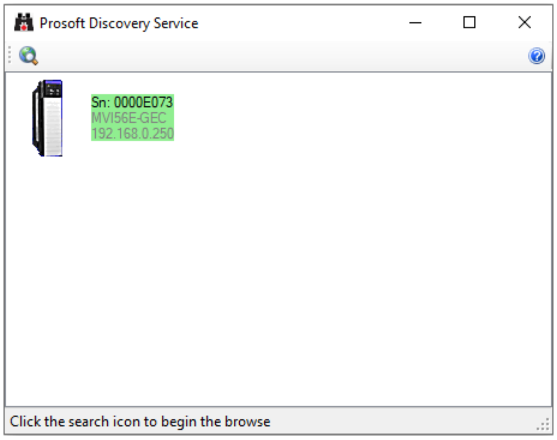
Ethernet: 192 . 168 . 0 . 250

CIPconnect: t:192.168.0.100,p:1,s:0$56

**4** ProSoft Discovery Service scans the network for ProSoft devices. Click on the icon to manually initiate a search.



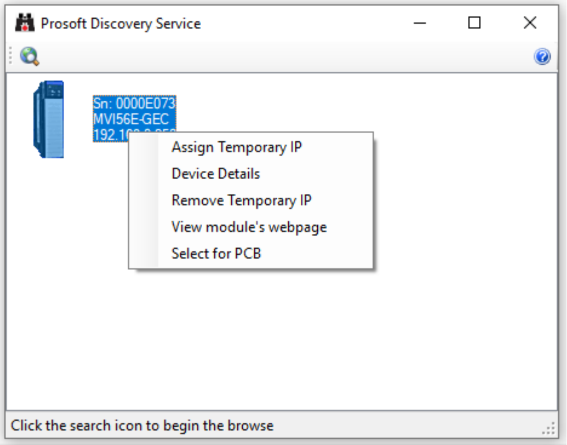
Click the search icon to begin the browse



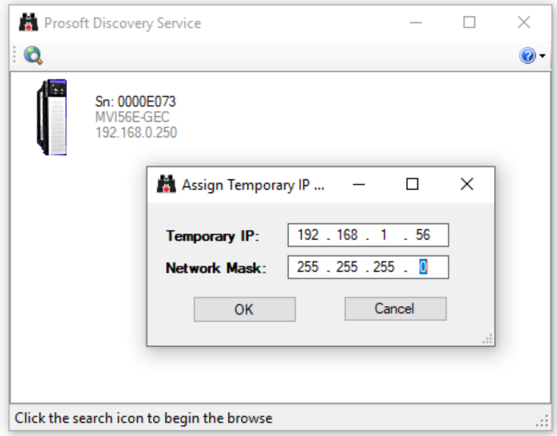
Sn: 0000E073
MVI56E-GEC
192.168.0.250

Click the search icon to begin the browse

**5** Right-click the module icon and select **ASSIGN TEMPORARY IP**.



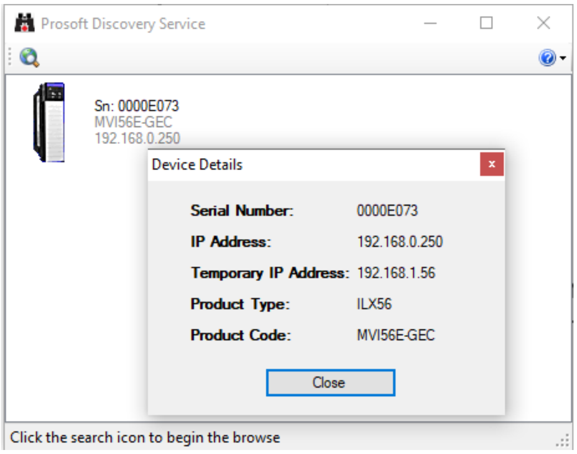
**6** Enter the Temporary IP and Network Mask.


Prosoft Discovery Service
Sn: 0000E073
MVI56E-GEC
192.168.0.250
Assign Temporary IP ...
OK
Cancel
Click the search icon to begin the browse

Temporary IP: 192 . 168 . 1 . 56

Network Mask: 255 . 255 . 255 . 0

**7** To view the current parameters of the MVI56E-GEC, right-click on the module icon and select *Device Details*.


Prosoft Discovery Service
Sn: 0000E073
MVI56E-GEC
192.168.0.250
Device Details
Close
Click the search icon to begin the browse

Serial Number: 0000E073

IP Address: 192.168.0.250

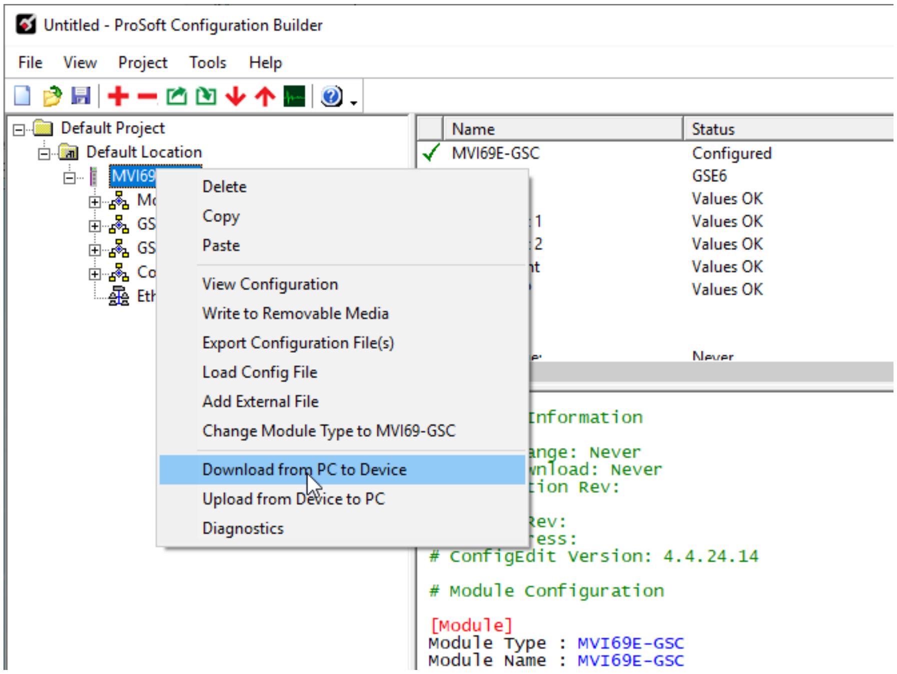
Temporary IP Address: 192.168.1.56

Product Type: ILX56

Product Code: MVI56E-GEC

## 2.5 Downloading the Configuration File to the Module

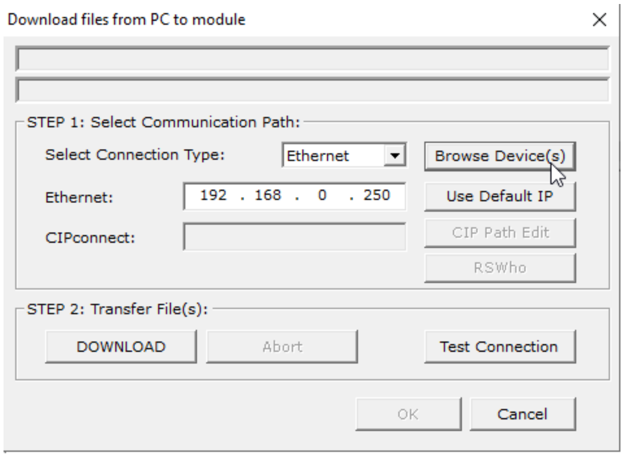
1 In the ProSoft Configuration Builder tree view, right-click the module icon and choose **DOWNLOAD FROM PC TO DEVICE**.



| Name | Status |
|---|---|
| ✓ MVI69E-GSC | Configured |
| | GSE6 |
| | Values OK |
| 1 | Values OK |
| 2 | Values OK |
| t | Values OK |
| | Values OK |
| e: | Never |

```
Information

ange: Never
wnload: Never
tion Rev:

Rev:
ress:
# ConfigEdit Version: 4.4.24.14

# Module Configuration

[Module]
Module Type : MVI69E-GSC
Module Name : MVI69E-GSC
```

2 In the *Download Files from PC to Module* dialog box, select **BROWSE DEVICES**.

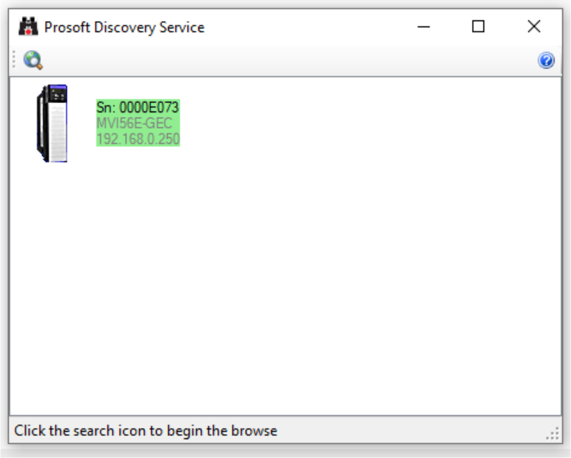

Download files from PC to module
STEP 1: Select Communication Path:
Browse Device(s)
Use Default IP
CIP Path Edit
RSWho
STEP 2: Transfer File(s):
DOWNLOAD
Abort
Test Connection
OK
Cancel

Select Connection Type: Ethernet

Ethernet: 192 . 168 . 0 . 250

CIPconnect:

**3** In the *ProSoft Discovery Service* dialog, double-click on the **MVI56E-GEC** icon:

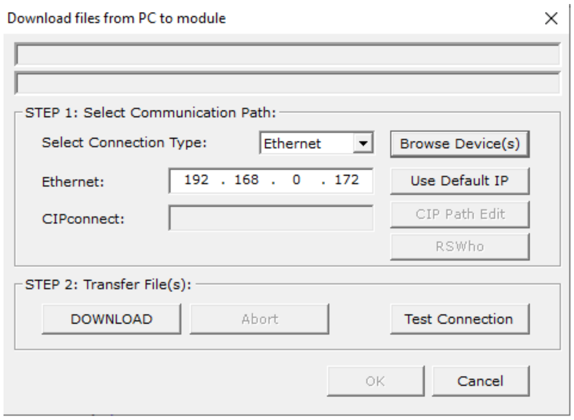


Prosoft Discovery Service

Sn: 0000E073
MVI56E-GEC
192.168.0.250

Click the search icon to begin the browse

**4** The *IP address* field is populated with the module's IP address. Click **DOWNLOAD** to confirm the operation:



Download files from PC to module

STEP 1: Select Communication Path:

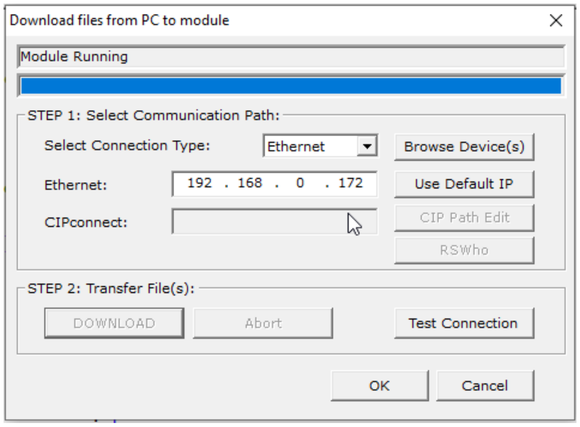
Select Connection Type: Ethernet

Ethernet: 192 . 168 . 0 . 172

CIPconnect:

STEP 2: Transfer File(s):

**5** Once the download is complete, the status field shows 'Module Running'. Click **OK**.



Download files from PC to module

Module Running

STEP 1: Select Communication Path:

Select Connection Type: Ethernet

Ethernet: 192 . 168 . 0 . 172

CIPconnect:

STEP 2: Transfer File(s):

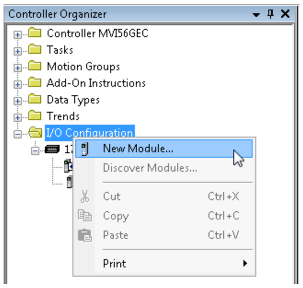
# 3 Configuring the Module in Studio 5000 Logix Designer

Ladder logic is required for application of the MVI56E-GEC module. Tasks that must be handled by the ladder logic are module data transfer, special block handling, and status data receipt. Additionally, a power-up handler may be needed to handle the initialization of the module's data and to clear any processor fault conditions.

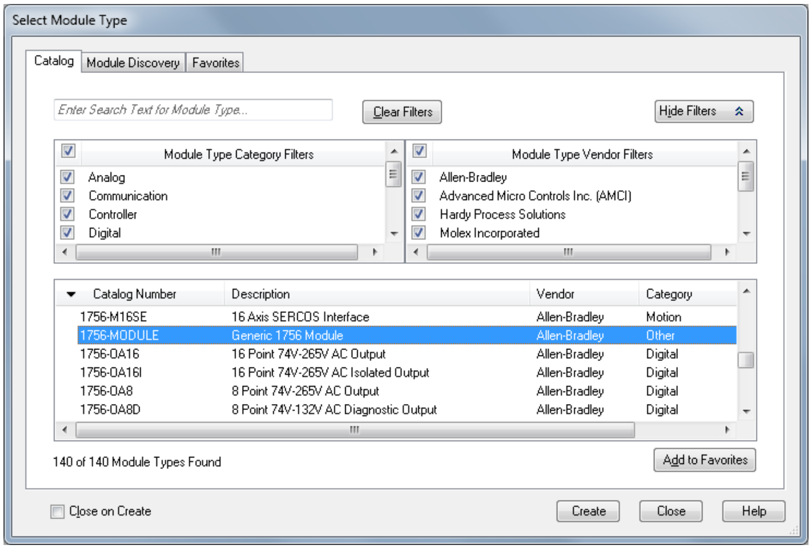
The sample ladder logic is extensively commented, to provide information on the purpose and function of each rung. For most applications, the sample ladder will work without modification.

## 3.1 Configuring the MVI56E-GEC in Studio 5000 Logix Designer

1 Right-click the mouse button on the *I/O Configuration* option in the *Controller Organizer* window to display a pop-up menu. Select the **NEW MODULE** option.



2 In the *Select Module Type* dialog, select the **1756-MODULE** option and click the **CREATE** button.



☑ Module Type Category Filters
☑ Analog
☑ Communication
☑ Controller
☑ Digital

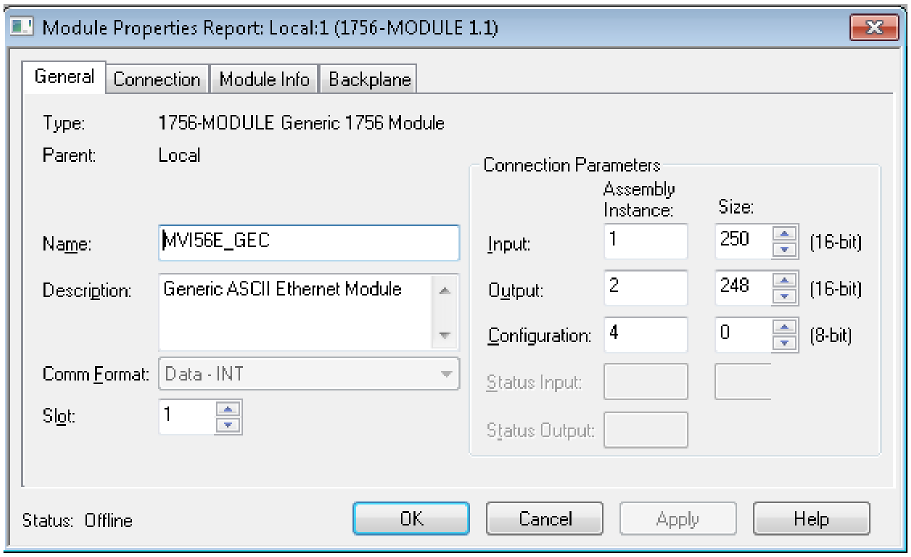
☑ Module Type Vendor Filters
☑ Allen-Bradley
☑ Advanced Micro Controls Inc. (AMCI)
☑ Hardy Process Solutions
☑ Molex Incorporated

| Catalog Number | Description | Vendor | Category |
|---|---|---|---|
| 1756-M16SE | 16 Axis SERCOS Interface | Allen-Bradley | Motion |
| 1756-MODULE | Generic 1756 Module | Allen-Bradley | Other |
| 1756-OA16 | 16 Point 74V-265V AC Output | Allen-Bradley | Digital |
| 1756-OA16I | 16 Point 74V-265V AC Isolated Output | Allen-Bradley | Digital |
| 1756-OA8 | 8 Point 74V-265V AC Output | Allen-Bradley | Digital |
| 1756-OA8D | 8 Point 74V-132V AC Diagnostic Output | Allen-Bradley | Digital |

140 of 140 Module Types Found

☐ Close on Create

**3** In the *New Module* dialog, enter the *Name*, *Description*, and *Slot* options for your application, then click on the *Connection* tab.



Type: 1756-MODULE Generic 1756 Module
Parent: Local
Name: MVI56E_GEC
Description: Generic ASCII Ethernet Module
Comm Format: Data - INT
Slot: 1

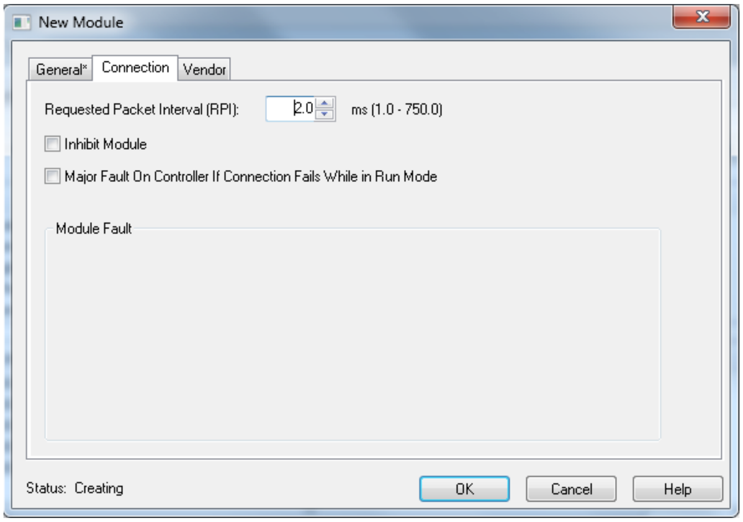
Connection Parameters

| | Assembly Instance: | Size: | |
|---|---|---|---|
| Input: | 1 | 250 | (16-bit) |
| Output: | 2 | 248 | (16-bit) |
| Configuration: | 4 | 0 | (8-bit) |
| Status Input: | | | |
| Status Output: | | | |

Status: Offline

| Parameter | Value |
|---|---|
| **NAME** | MVI56E_GEC |
| **DESCRIPTION** | Enter a description for the module. Example: Generic ASCII Ethernet Communication Module |
| **COMM FORMAT** | Select **DATA-INT** |
| **SLOT** | Enter the slot number in the rack where the MVI56E-GEC module is located |
| **INPUT ASSEMBLY INSTANCE** | 1 |
| **INPUT SIZE** | 250 |
| **OUTPUT ASSEMBLY INSTANCE** | 2 |
| **OUTPUT SIZE** | 248 |
| **CONFIGURATION ASSEMBLY INSTANCE** | 4 |
| **CONFIGURATION SIZE** | 0 |

**4** In the *Connection* tab, select the *Request Packet Interval* value for scanning the I/O on the module. This value represents the minimum frequency the module will handle scheduled events. This value should not be set to less than 1 millisecond. Values between 1 and 10 milliseconds should work with most applications. Click **OK**.



New Module

General* | Connection | Vendor

Requested Packet Interval (RPI): 2.0 ms (1.0 - 750.0)

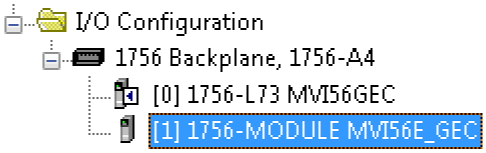
□ Inhibit Module

□ Major Fault On Controller If Connection Fails While in Run Mode

Module Fault

Status: Creating

**5** The *I/O Configuration* now displays the module's presence.



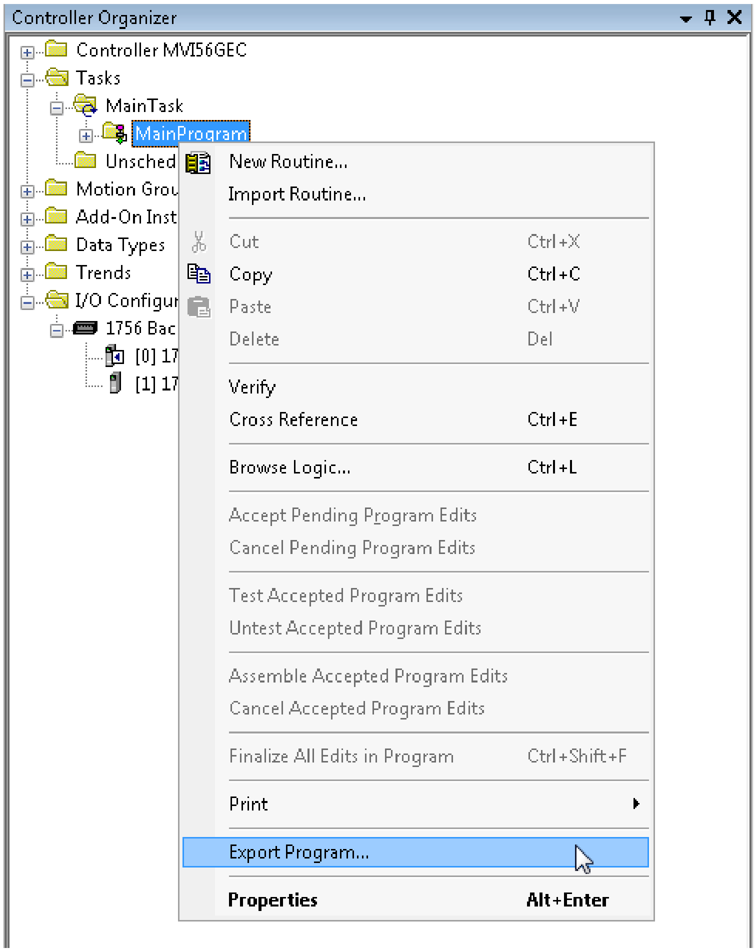
I/O Configuration
1756 Backplane, 1756-A4
[0] 1756-L73 MVI56GEC
[1] 1756-MODULE MVI56E_GEC

## 3.2 Ladder Logic

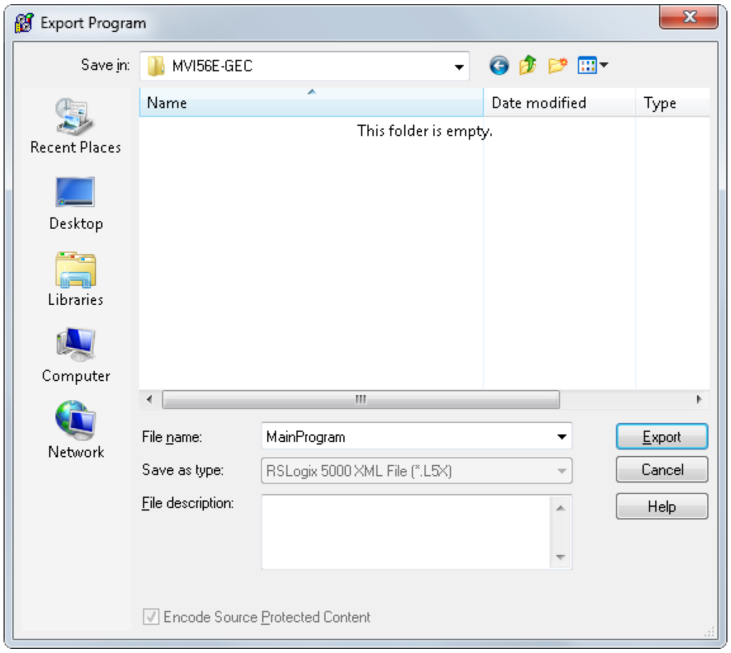
The MVI56E-GEC ladder logic can be used as-is. If you need to add the MVI56E-GEC ladder logic to an existing ControlLogix application, you can export the sample ladder logic components to a .L5X file. This allows you to import the .L5X file into the existing application.

### 3.2.1 Exporting the MVI56E-GEC .L5X File

1 Download the sample ladder logic program (*MVI56EGEC.ACD*) from www.prosoft-technology.com.
2 Open the *MVI56EGEC.ACD* file.
3 In the *Controller Organizer* window, right-click on the *Tasks* > *Main Task* > *MainProgram* folder. Click on the **EXPORT PROGRAM** option.



4 Select a storage location to save the *MainProgram.L5X* file. Click the **EXPORT** button.



Save in: MVI56E-GEC

This folder is empty.

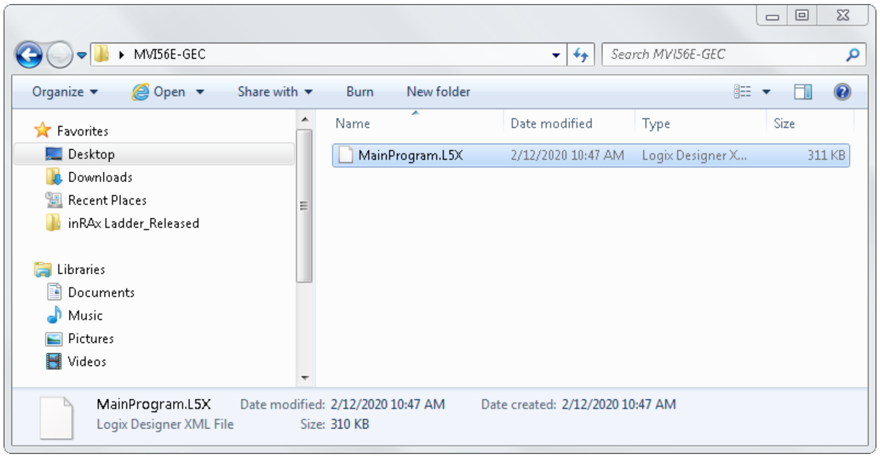
File name: MainProgram

Save as type: RSLogix 5000 XML File (*.L5X)

File description:

☑ Encode Source Protected Content

5 Verify the *MainProgram.L5X* file has been successfully exported.

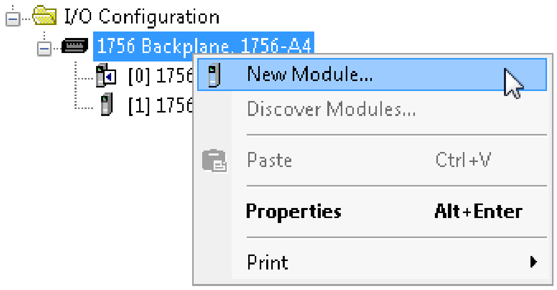


| Name | Date modified | Type | Size |
|---|---|---|---|
| MainProgram.L5X | 2/12/2020 10:47 AM | Logix Designer X... | 311 KB |

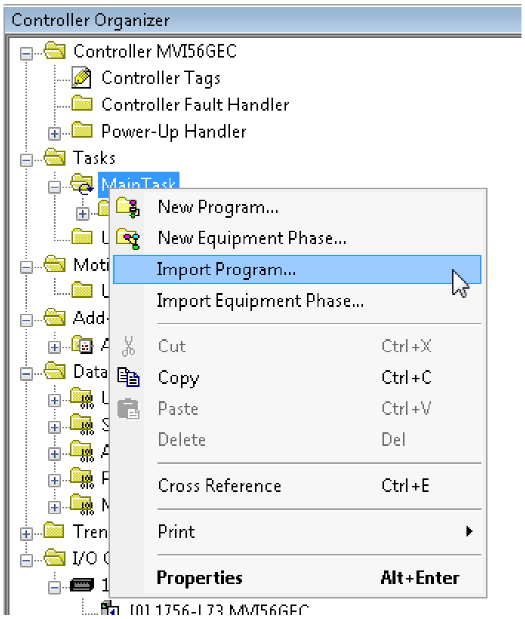
MainProgram.L5X Date modified: 2/12/2020 10:47 AM Date created: 2/12/2020 10:47 AM
Logix Designer XML File Size: 310 KB

### 3.2.2 Importing the MVI56E-GEC .L5X file Into an Studio 5000 Project

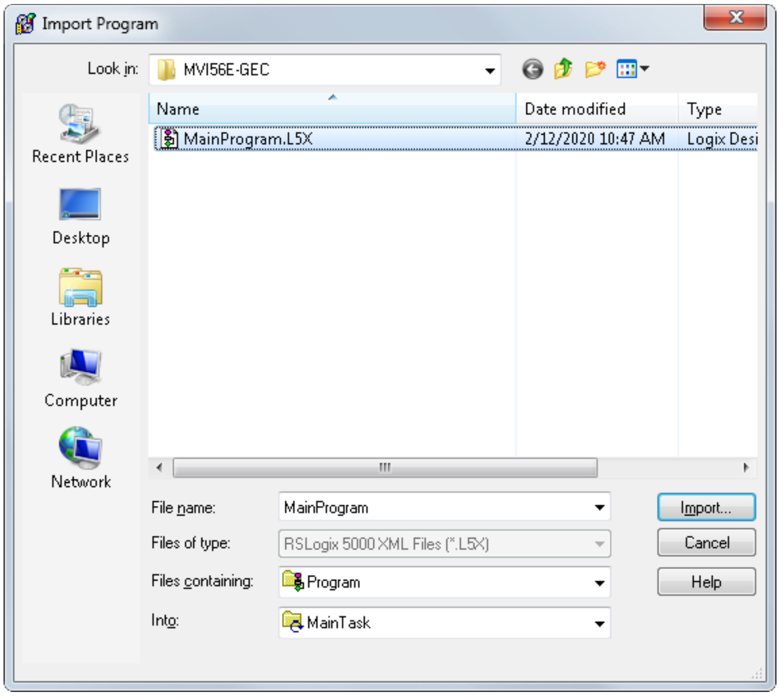
**1** Within the existing Studio 5000 Logix Designer Project, add the MVI56E-GEC module to the *I/O Configuration*. (For further instructions, see Configuring the MVI56E-GEC in Studio 5000 Logix Designer on page 23).



**2** In the *Controller Organizer* window, right-click on the *Tasks* > *Main Task* folder. Select the **IMPORT PROGRAM** option.



**3** Select the (previously exported) MVI56E-GEC *MainProgram.L5X* file and click the **IMPORT** button.



Look in: MVI56E-GEC

| Name | Date modified | Type |
|---|---|---|
| MainProgram.L5X | 2/12/2020 10:47 AM | Logix Desi |

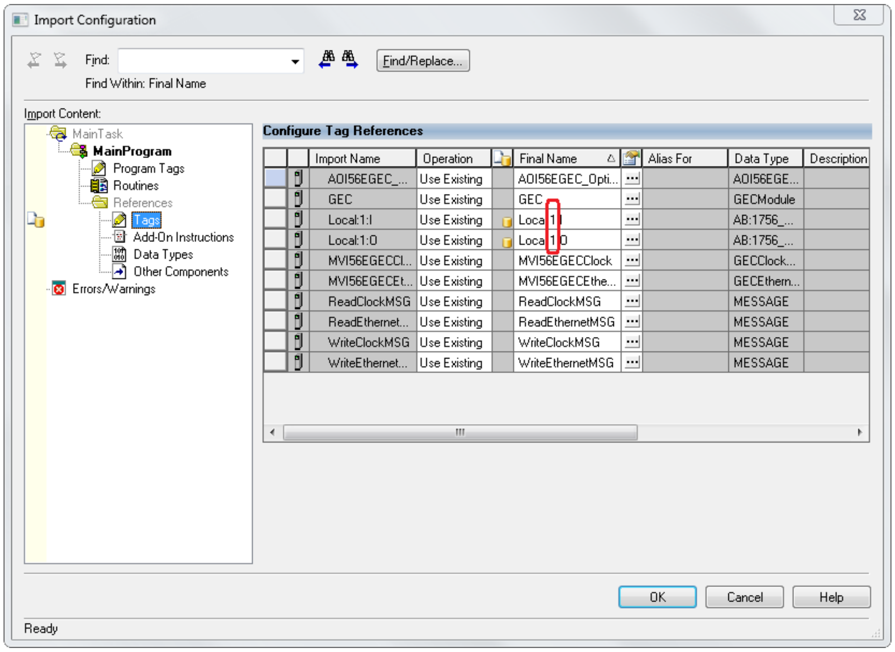
File name: MainProgram

Files of type: RSLogix 5000 XML Files (*.L5X)

Files containing: Program

Into: MainTask

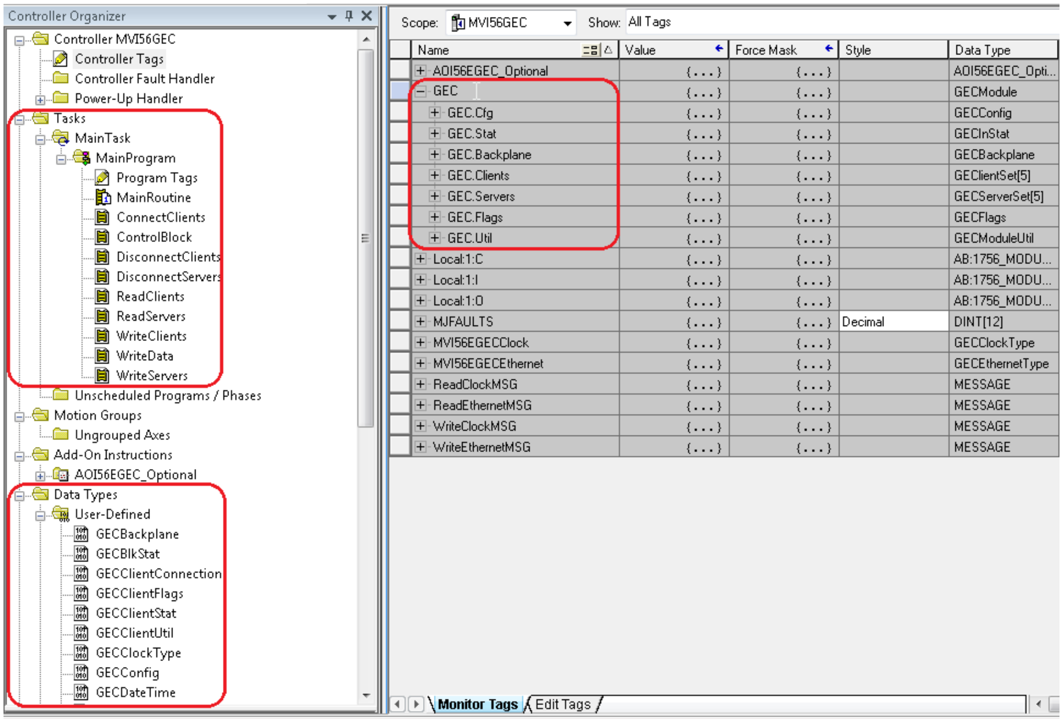
**4** In the *Import Configuration* dialog, edit the **Local:x:I** and **Local:x:O** tags to correspond to the slot position of the MVI56E-GEC. Then click **OK**.



Configure Tag References

| Import Name | Operation | Final Name | Alias For | Data Type | Description |
|---|---|---|---|---|---|
| AOI56EGEC_... | Use Existing | AOI56EGEC_Opti... | | AOI56EGE... | |
| GEC | Use Existing | GEC | | GECModule | |
| Local:1:I | Use Existing | Local:1:I | | AB:1756_... | |
| Local:1:O | Use Existing | Local:1:O | | AB:1756_... | |
| MVI56EGECCl... | Use Existing | MVI56EGECClock | | GECClock... | |
| MVI56EGECEt... | Use Existing | MVI56EGECEthe... | | GECEthern... | |
| ReadClockMSG | Use Existing | ReadClockMSG | | MESSAGE | |
| ReadEthernet... | Use Existing | ReadEthernetMSG | | MESSAGE | |
| WriteClockMSG | Use Existing | WriteClockMSG | | MESSAGE | |
| WriteEthernet... | Use Existing | WriteEthernetMSG | | MESSAGE | |

**5** Upon successful import, the MVI56E-GEC ladder logic, Controller Tags, and UDT's are now visible in the Studio 5000 Logix Designer project.



Controller Organizer

- Controller MVI56GEC
  - Controller Tags
  - Controller Fault Handler
  - Power-Up Handler
- Tasks
  - MainTask
    - MainProgram
      - Program Tags
      - MainRoutine
      - ConnectClients
      - ControlBlock
      - DisconnectClients
      - DisconnectServers
      - ReadClients
      - ReadServers
      - WriteClients
      - WriteData
      - WriteServers
  - Unscheduled Programs / Phases
- Motion Groups
  - Ungrouped Axes
- Add-On Instructions
  - AOI56EGEC_Optional
- Data Types
  - User-Defined
    - GECBackplane
    - GECBlkStat
    - GECClientConnection
    - GECClientFlags
    - GECClientStat
    - GECClientUtil
    - GECClockType
    - GECConfig
    - GECDateTime

Scope: MVI56GEC Show: All Tags

| Name | Value | Force Mask | Style | Data Type |
|---|---|---|---|---|
| + AOI56EGEC_Optional | {...} | {...} | | AOI56EGEC_Opti... |
| − GEC | {...} | {...} | | GECModule |
| + GEC.Cfg | {...} | {...} | | GECConfig |
| + GEC.Stat | {...} | {...} | | GECInStat |
| + GEC.Backplane | {...} | {...} | | GECBackplane |
| + GEC.Clients | {...} | {...} | | GECClientSet[5] |
| + GEC.Servers | {...} | {...} | | GECServerSet[5] |
| + GEC.Flags | {...} | {...} | | GECFlags |
| + GEC.Util | {...} | {...} | | GECModuleUtil |
| + Local:1:C | {...} | {...} | | AB:1756_MODU... |
| + Local:1:I | {...} | {...} | | AB:1756_MODU... |
| + Local:1:O | {...} | {...} | | AB:1756_MODU... |
| + MJFAULTS | {...} | {...} | Decimal | DINT[12] |
| + MVI56EGECClock | {...} | {...} | | GECClockType |
| + MVI56EGECEthernet | {...} | {...} | | GECEthernetType |
| + ReadClockMSG | {...} | {...} | | MESSAGE |
| + ReadEthernetMSG | {...} | {...} | | MESSAGE |
| + WriteClockMSG | {...} | {...} | | MESSAGE |
| + WriteEthernetMSG | {...} | {...} | | MESSAGE |

## 3.3 Optional Add-On Instruction

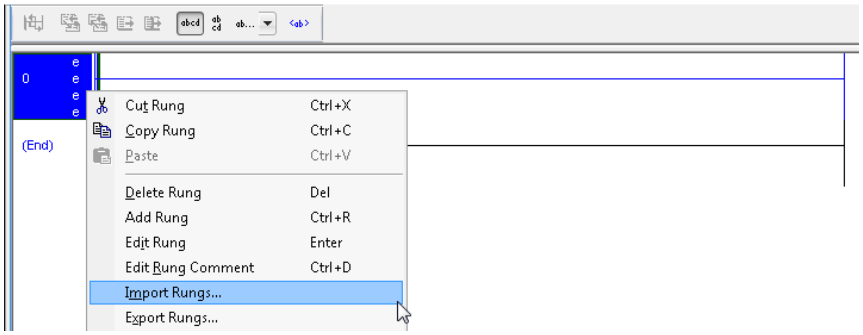
The Optional AOI supports the following optional features:

- Read/Write IP Address
- Read/Write Date Time

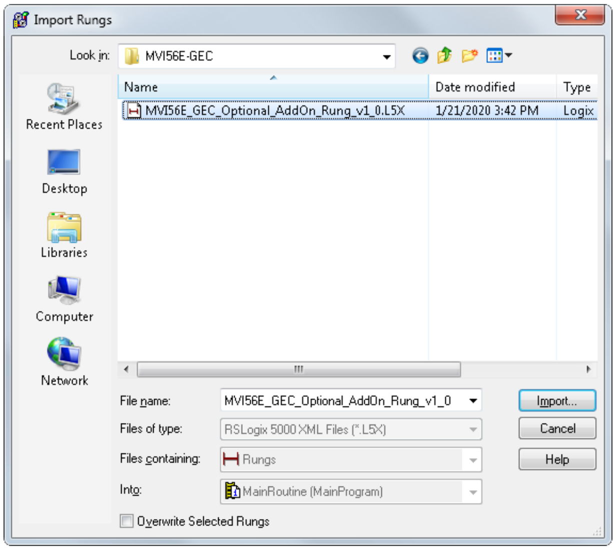
Using controller tags, the Optional AOI allows you to request and set the module's IP address, date, and time. These optional features are not supported by the MVI56-GEC legacy module.

> **Note:** The Optional AOI may be added to an existing legacy MVI56-GEC application to add the new functionality during a module replacement.

**1** Add a new rung to the existing processor ladder logic. Right-click on the new rung and select *Import Rungs…*



**2** In the *Import Rungs* dialog, select the Optional AOI file: *MVI56E_GEC_Optional_AddOn_Rung.L5X* and click on the **IMPORT** button.

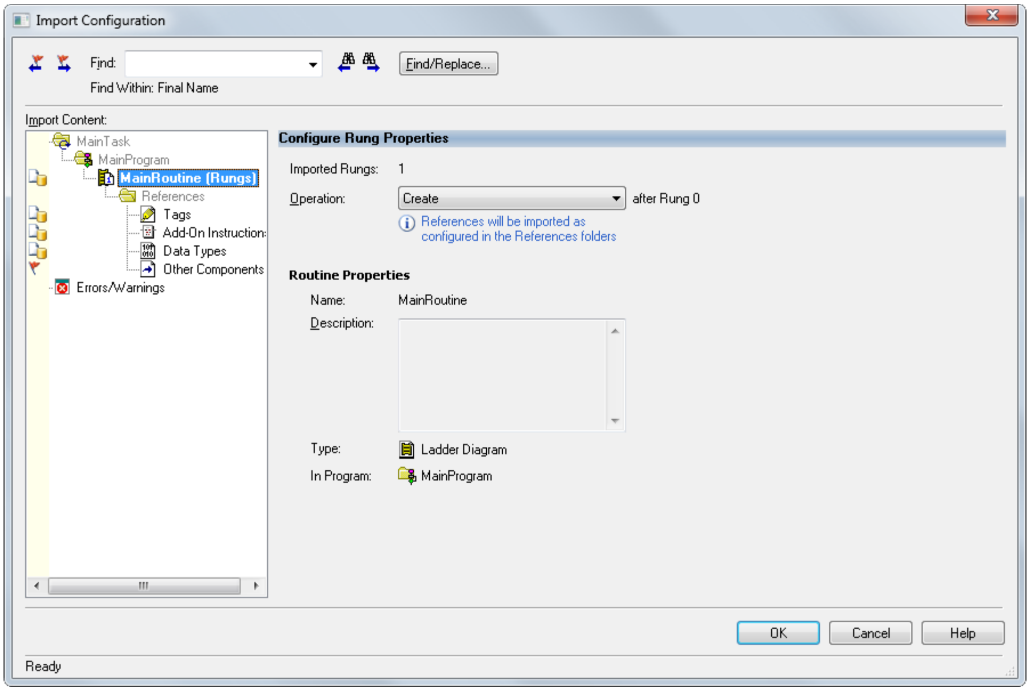


Look in: MVI56E-GEC

| Name | Date modified | Type |
|---|---|---|
| MVI56E_GEC_Optional_AddOn_Rung_v1_0.L5X | 1/21/2020 3:42 PM | Logix |

File name: MVI56E_GEC_Optional_AddOn_Rung_v1_0
Files of type: RSLogix 5000 XML Files (*.L5X)
Files containing: Rungs
Into: MainRoutine (MainProgram)
☐ Overwrite Selected Rungs

3 At the *Import Configuration* window, select the *Operation* parameter to **CREATE**. Then click **OK**.



Configure Rung Properties

Imported Rungs: 1

Operation: Create after Rung 0

References will be imported as configured in the References folders

Routine Properties

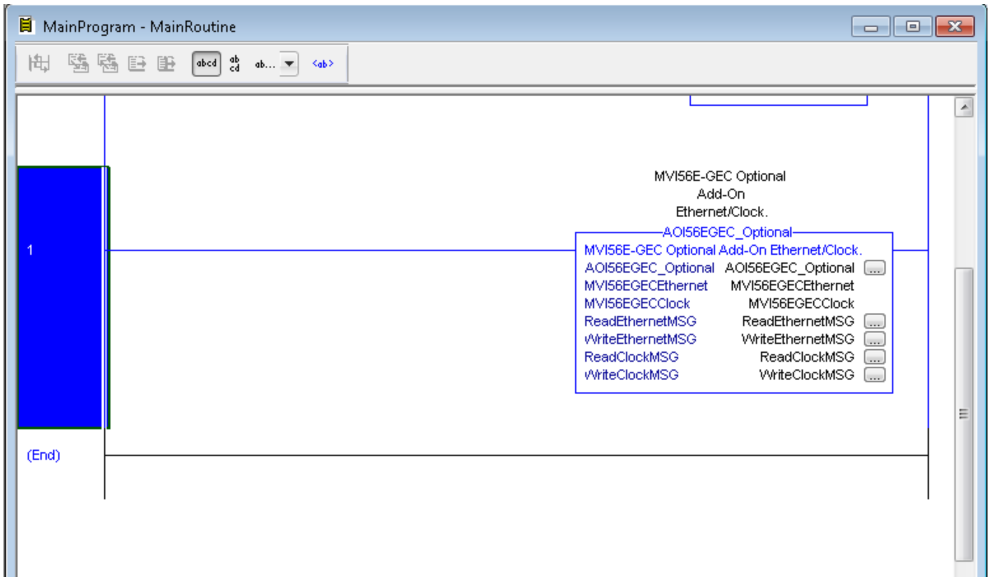
Name: MainRoutine

Description:

Type: Ladder Diagram

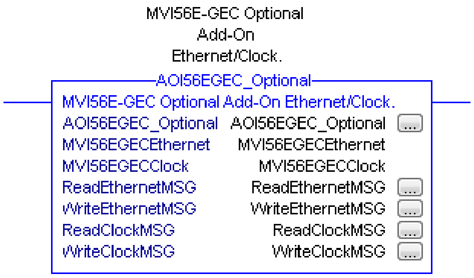
In Program: MainProgram

4 The imported AOI rung is now in place.


MainProgram - MainRoutine
1
(End)
MVI56E-GEC Optional
Add-On
Ethernet/Clock.
AOI56EGEC_Optional
MVI56E-GEC Optional Add-On Ethernet/Clock.
AOI56EGEC_Optional AOI56EGEC_Optional
MVI56EGECEthernet MVI56EGECEthernet
MVI56EGECClock MVI56EGECClock
ReadEthernetMSG ReadEthernetMSG
WriteEthernetMSG WriteEthernetMSG
ReadClockMSG ReadClockMSG
WriteClockMSG WriteClockMSG

### 3.3.1 *Setting Up the Optional AOI*

1 Click on the *ReadEthernetMSG* … icon to configure the message route:



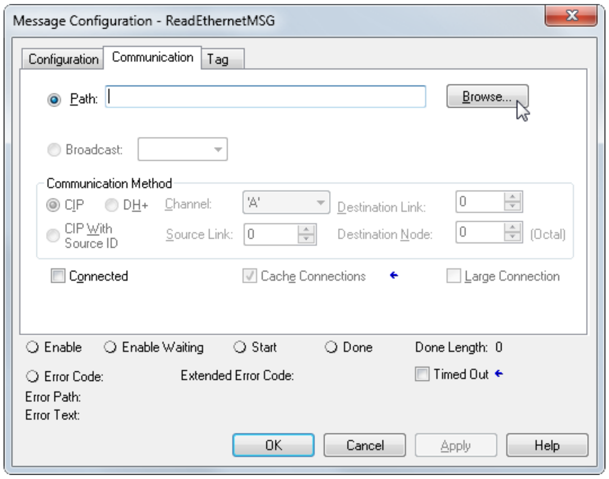
MVI56E-GEC Optional
Add-On
Ethernet/Clock.

AOI56EGEC_Optional

MVI56E-GEC Optional Add-On Ethernet/Clock.

| | |
|---|---|
| AOI56EGEC_Optional | AOI56EGEC_Optional … |
| MVI56EGECEthernet | MVI56EGECEthernet |
| MVI56EGECClock | MVI56EGECClock |
| ReadEthernetMSG | ReadEthernetMSG … |
| WriteEthernetMSG | WriteEthernetMSG … |
| ReadClockMSG | ReadClockMSG … |
| WriteClockMSG | WriteClockMSG … |

2 In the *Message Configuration* dialog, under the *Communication* tab, select the **BROWSE** button.



Message Configuration - ReadEthernetMSG

● Path:

○ Broadcast:

○ CIP ○ DH+ Channel: 'A' Destination Link: 0

○ CIP With Source ID Source Link: 0 Destination Node: 0 (Octal)

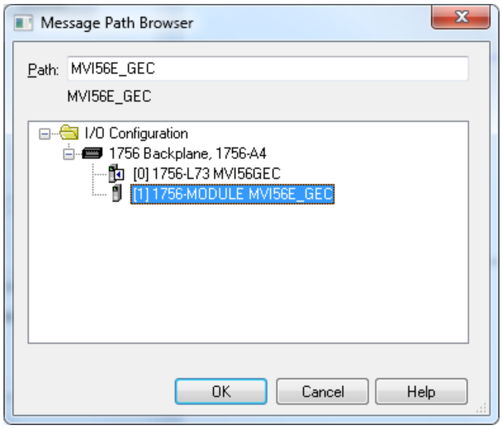
□ Connected ☑ Cache Connections □ Large Connection

○ Enable ○ Enable Waiting ○ Start ○ Done Done Length: 0

○ Error Code: Extended Error Code: □ Timed Out

Error Path:
Error Text:

3 In the *Message Path Browser* dialog, select the MVI56E-GEC module and click **OK**.



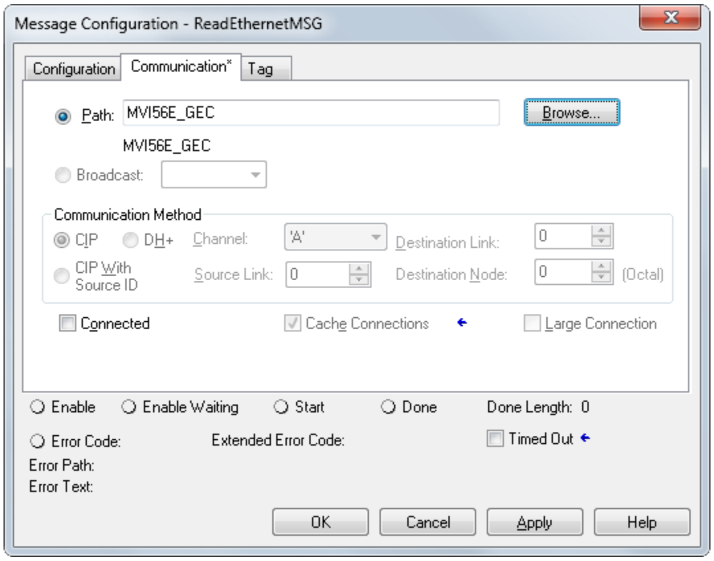
Message Path Browser

Path: MVI56E_GEC

MVI56E_GEC

I/O Configuration
1756 Backplane, 1756-A4
[0] 1756-L73 MVI56GEC
[1] 1756-MODULE MVI56E_GEC

**4** The module name is now displayed in the *Path* field. Click **OK** to confirm the route configuration.



Message Configuration - ReadEthernetMSG

Configuration | Communication* | Tag

● Path: MVI56E_GEC Browse...

MVI56E_GEC

○ Broadcast:

Communication Method

● CIP ○ DH+ Channel: 'A' Destination Link: 0

○ CIP With Source ID Source Link: 0 Destination Node: 0 (Octal)

□ Connected ☑ Cache Connections □ Large Connection

○ Enable ○ Enable Waiting ○ Start ○ Done Done Length: 0

○ Error Code: Extended Error Code: □ Timed Out

Error Path:

Error Text:

**5** Repeat the same procedure to set the route for the remaining messages:

- *WriteEthernetMSG* …
- *ReadClockMSG* …
- *WriteClockMSG* …

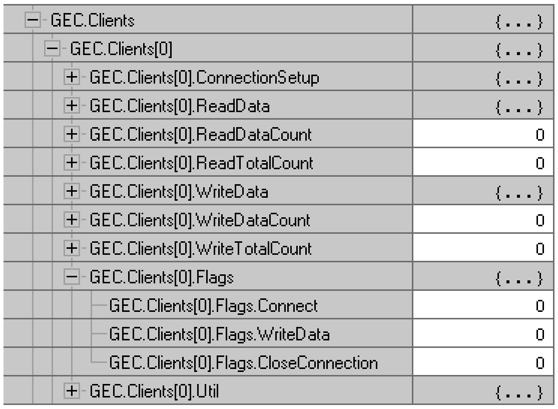
# 4 Sending and Receiving ASCII Data

## 4.1 Sending ASCII Data

The MVI56E-GEC, whether used as a client or server, can send ASCII texts to a remote device.

### *4.1.1 Sending ASCII Text as a Client*

Use the following steps to configure the MVI56E-GEC as a client to send an ASCII string to a remote device (server). The MVI56E-GEC can simultaneously connect and send data to up to five servers.



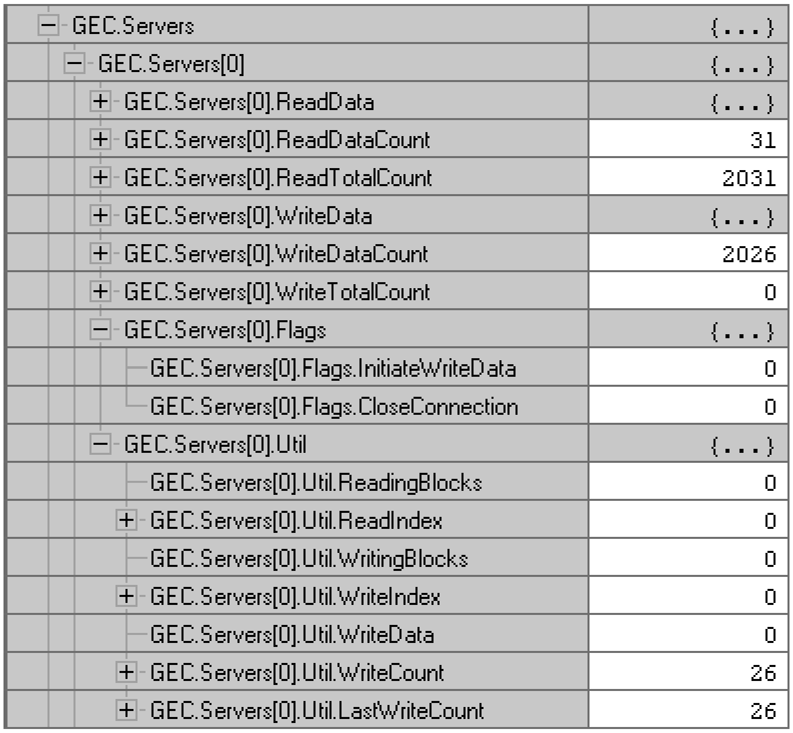
| | |
|---|---|
| ⊟ GEC.Clients | {...} |
| ⊟ GEC.Clients[0] | {...} |
| ⊞ GEC.Clients[0].ConnectionSetup | {...} |
| ⊞ GEC.Clients[0].ReadData | {...} |
| ⊞ GEC.Clients[0].ReadDataCount | 0 |
| ⊞ GEC.Clients[0].ReadTotalCount | 0 |
| ⊞ GEC.Clients[0].WriteData | {...} |
| ⊞ GEC.Clients[0].WriteDataCount | 0 |
| ⊞ GEC.Clients[0].WriteTotalCount | 0 |
| ⊟ GEC.Clients[0].Flags | {...} |
| GEC.Clients[0].Flags.Connect | 0 |
| GEC.Clients[0].Flags.WriteData | 0 |
| GEC.Clients[0].Flags.CloseConnection | 0 |
| ⊞ GEC.Clients[0].Util | {...} |

1. Enter the IP address of the server at **GEC.CLIENTS[X].CONNECTIONSETUP.SERVERIP**.
2. Enter the service port of the server at **GEC.CLIENTS[X].CONNECTIONSETUP.SERVICEPORT**.
3. Enter the text to be sent at **GEC.CLIENTS[X].WRITEDATA**.
4. Enter the number of characters to be sent at **GEC.CLIENTS[X].WRITEDATACOUNT**
5. Set the bit at **GEC.CLIENTS[X].FLAGS.CONNECT** to open the connection to the server.
6. Set the bit at **GEC.CLIENTS[X].FLAGS.WRITEDATA** to send the message.
7. (Optional) Set the bit at **GEC.CLIENTS[X].FLAGS.CLOSECONNECTION** to close the connection.
8. **GEC.STAT.CLIENT[X].TXCOUNT** increments by 1 on every transmission.

### 4.1.2 *Sending ASCII Text as a Server*

The remote Client must initiate a socket connection before the Server can transmit data to the Client.

1 Enter the data to be sent at **GEC.SERVERS[X].WRITEDATA**.
2 Enter the number of characters to be sent at **GEC.SERVERS[X].WRITEDATACOUNT.**
3 Set the bit at **GEC.SERVERS[X].FLAGS.WRITEDATA** to send the message.
4 (Optional) Set the bit at **GEC.SERVERS[X].FLAGS.CLOSECONNECTION** to close the connection.
5 **GEC. SERVERS[X].UTIL.WRITECOUNT** increments by 1 on every transmission.



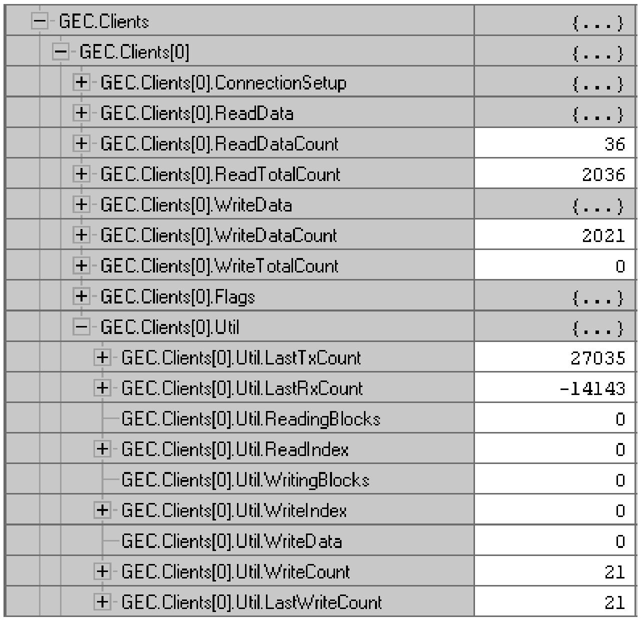
| | |
|---|---|
| ⊟ GEC.Servers | {...} |
| ⊟ GEC.Servers[0] | {...} |
| ⊞ GEC.Servers[0].ReadData | {...} |
| ⊞ GEC.Servers[0].ReadDataCount | 31 |
| ⊞ GEC.Servers[0].ReadTotalCount | 2031 |
| ⊞ GEC.Servers[0].WriteData | {...} |
| ⊞ GEC.Servers[0].WriteDataCount | 2026 |
| ⊞ GEC.Servers[0].WriteTotalCount | 0 |
| ⊟ GEC.Servers[0].Flags | {...} |
| GEC.Servers[0].Flags.InitiateWriteData | 0 |
| GEC.Servers[0].Flags.CloseConnection | 0 |
| ⊟ GEC.Servers[0].Util | {...} |
| GEC.Servers[0].Util.ReadingBlocks | 0 |
| ⊞ GEC.Servers[0].Util.ReadIndex | 0 |
| GEC.Servers[0].Util.WritingBlocks | 0 |
| ⊞ GEC.Servers[0].Util.WriteIndex | 0 |
| GEC.Servers[0].Util.WriteData | 0 |
| ⊞ GEC.Servers[0].Util.WriteCount | 26 |
| ⊞ GEC.Servers[0].Util.LastWriteCount | 26 |

## 4.2 Receiving ASCII Data

The MVI56E-GEC, whether used as a client or server, can receive incoming ASCII texts from a remote device.

### *4.2.1 Receiving ASCII Text as a Client*

The MVI56E-GEC can receive ASCII strings from the same server it sends to. Since the client socket connection has already been established with the server, the incoming data will be stored in the **GEC.CLIENTS[X].READDATA** array.



| | |
|---|---|
| − GEC.Clients | {...} |
| − GEC.Clients[0] | {...} |
| + GEC.Clients[0].ConnectionSetup | {...} |
| + GEC.Clients[0].ReadData | {...} |
| + GEC.Clients[0].ReadDataCount | 36 |
| + GEC.Clients[0].ReadTotalCount | 2036 |
| + GEC.Clients[0].WriteData | {...} |
| + GEC.Clients[0].WriteDataCount | 2021 |
| + GEC.Clients[0].WriteTotalCount | 0 |
| + GEC.Clients[0].Flags | {...} |
| − GEC.Clients[0].Util | {...} |
| + GEC.Clients[0].Util.LastTxCount | 27035 |
| + GEC.Clients[0].Util.LastRxCount | -14143 |
| GEC.Clients[0].Util.ReadingBlocks | 0 |
| + GEC.Clients[0].Util.ReadIndex | 0 |
| GEC.Clients[0].Util.WritingBlocks | 0 |
| + GEC.Clients[0].Util.WriteIndex | 0 |
| GEC.Clients[0].Util.WriteData | 0 |
| + GEC.Clients[0].Util.WriteCount | 21 |
| + GEC.Clients[0].Util.LastWriteCount | 21 |

**1** When the MVI56E-GEC receives an ASCII string from a server, the **GEC.STAT.CLIENT[X].RXCOUNT** controller tag increments by 1. You will need to monitor this tag to determine a new message was received.

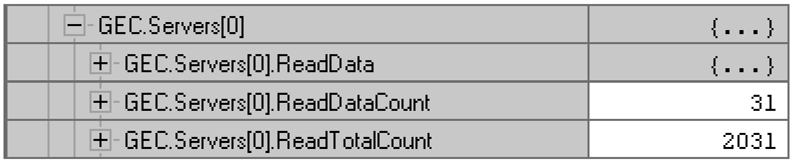
**2** The **GEC.CLIENT[X].READDATA** array contains the ASCII text of the new message. This array is overwritten every time a new string is received. You will need to create logic that monitors when a new message is received (**GEC.STAT.CLIENT[X].RXCOUNT** increases by 1), and copies the text out of the **GEC.CLIENTS[X].READDATA ARRAY**.

**3** The number of characters received in the new message is located at **GEC.CLIENTS[X].READDATACOUNT**.

**4** The accumulated total number of characters received is located at **GEC.CLIENTS[X].READTOTALCOUNT**.

### 4.2.2 Receiving ASCII Text as a Server

When a server port of the MVI56E-GEC is set up, it will accept incoming ASCII text from a client only.



| | |
|---|---|
| ⊟ GEC.Servers[0] | {...} |
| ⊞ GEC.Servers[0].ReadData | {...} |
| ⊞ GEC.Servers[0].ReadDataCount | 31 |
| ⊞ GEC.Servers[0].ReadTotalCount | 2031 |

1 When the MVI56E-GEC receives an ASCII string from a client, the **GEC.STAT.SERVER[X].RX** controller tag increments by 1. You will need to monitor this tag to determine a new message was received.
2 The **GEC.SERVERS[X].READDATA** array contains the ASCII text of the new message. This array is overwritten every time a new string is received. You will need to create logic that monitors when a new message is received (**GEC.STAT.SERVER[X].RX** increases by 1), and copies the text out of the **GEC.SERVERS[X].READDATA** array.
3 The number of characters received in the new message is located at **GEC.SERVERS[X].READDATACOUNT**.
4 The accumulated total number of characters received is located at **GEC.SERVERS[X].READTOTALCOUNT**.

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

## 5.1 LED Indicators

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

| LED | Color | Status | Indication |
|---|---|---|---|
| APP | Red or Green | Off | No connections established over Ethernet (Client or Server) |
| | | Green | At least one connection is established (Client or Server) |
| | | Red | During operation, this LED will remain Red in case all connections (Client or Server) previously established have been terminated. In case a single connection is established, the LED will be Green.<br>This LED will also be temporarily set as Red during one of the following conditions:<br>- The firmware is initializing during startup.<br>- The module is rebooting due to a Cold or Warmboot request from the ladder logic of Debug Menu. |
| OK | Red or Green | Off | The module is not receiving adequate power or is not securely fastened into the rack. |
| | | Green | The module is operating normally. |
| | | Red | The module has detected an error or is being initialized. If the LED remains red for over 10 seconds, the program has probably halted. Remove the card from the rack and re-insert the card to restart the module's program. |
| ERR | Red or Green | Off | Not Used |

### *5.1.1 Ethernet Port LED Indicators*

| 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. |

### 5.1.2 Scrolling LED

The MVI56E-GEC scrolling LED is be implemented as follows:

**Initialization**

| Code | Message |
|---|---|
| Boot | Module is initializing |
| Waiting for Processor Connection | Module is waiting to connect to processor. |
| MVI56E-GEC vXX.YY.ZZZ Last Config: <date> | Indicates the last date when the module changed its IP address. |

**After Initialization, the following pattern will be repeated:**

<Backplane Status> <IP Address>

| Code | Message |
|---|---|
| Backplane Status | **OK**: Module is communicating with processor<br>**ERR**: Module is unable to communicate with processor<br>**Processor faulted**: PLC is faulted.<br>**Processor in program mode**: PLC is in program mode.<br>**Processor faulted or is in program mode** |
| IP Address | MVI56E-GEC's IP Address |

### 5.1.3 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.

### 5.1.4 Troubleshooting

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

#### Processor Errors

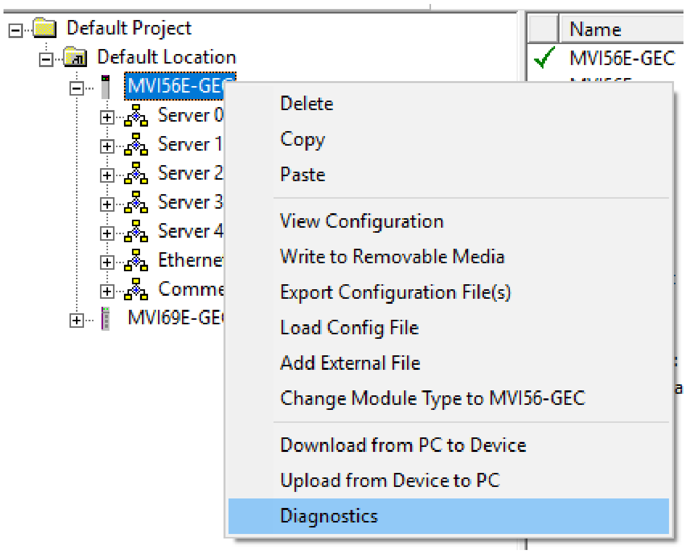
| Problem description | Steps to take |
|---|---|
| Processor fault | Verify that the module is plugged into the slot that has been configured for the module in the I/O Configuration of Studio 5000 Logix Designer.<br>Verify that the slot location in the rack has been configured correctly in the ladder 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-GEC. Verify that all modules in the rack are correctly configured in the ladder logic. |

#### Module Errors

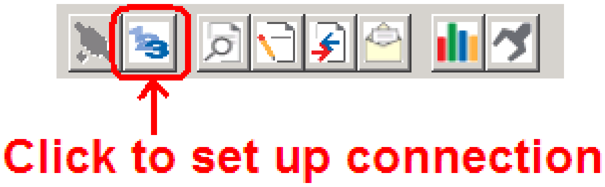
| Problem description | Steps to take |
|---|---|
| Scrolling LED display: *<Backplane Status>* condition reads ERR | This indicates that backplane transfer operations are failing. Connect to the module's Configuration/Debug port to check this.<br>To establish backplane communications, verify the following items:<br>▪ The processor is in RUN or REM RUN mode.<br>▪ The backplane driver is loaded in the module.<br>▪ The module is configured for read and write data block transfer.<br>▪ The ladder logic handles all read and write block situations.<br>▪ 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 it, and then restore power to the rack. |

## 5.2 ProSoft Configuration Builder Diagnostics Menu

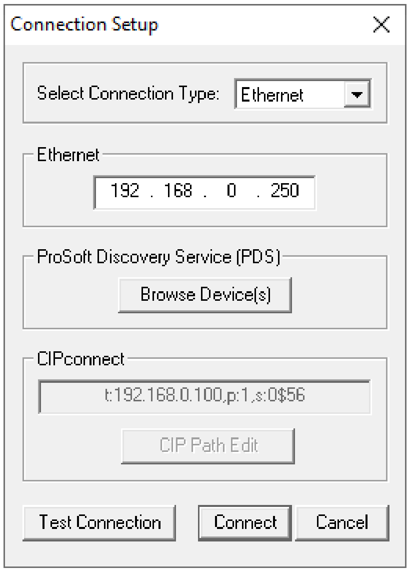
**1** In the tree view in ProSoft Configuration Builder, right-click the **MVI56E-GEC** module and then choose **DIAGNOSTICS**.



**2** After the *Diagnostics* window opens, click the **SET UP CONNECTION** button to browse for the module's IP address.


Click to set up connection

**3** In the *Ethernet* field of the *Connection Setup* dialog box, enter the current IP address, whether it is temporary or permanent. Click **TEST CONNECTION** to verify that the module is accessible with the current settings.

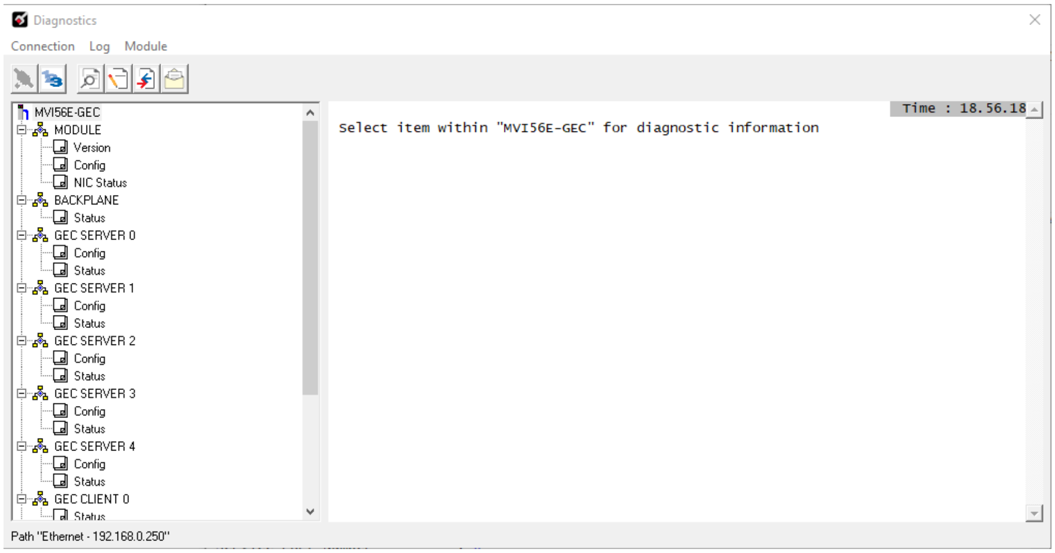


Select Connection Type: Ethernet

Ethernet: 192 . 168 . 0 . 250

CIPconnect: t:192.168.0.100,p:1,s:0$56

**4** If the **TEST CONNECTION** is successful, click **CONNECT**. The *Diagnostics* window is now visible.



```
Time : 18.56.18
Select item within "MVI56E-GEC" for diagnostic information
```

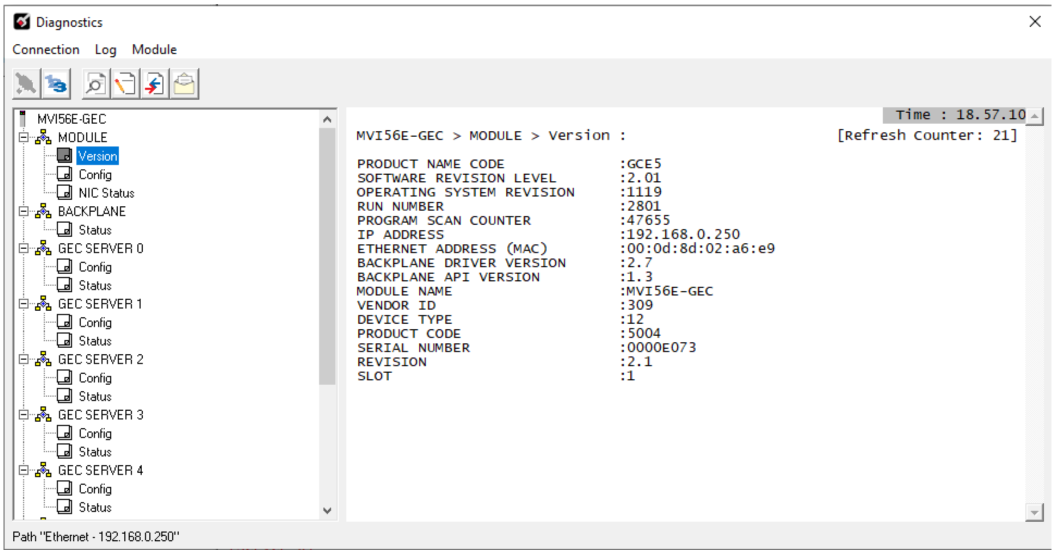
Path "Ethernet - 192.168.0.250"

### 5.2.1 Diagnostics Menu Navigation

In the *Diagnostics* window in ProSoft Configuration Builder, the Diagnostics menu is available through the Ethernet configuration port. The menu is arranged as a tree structure.

### 5.2.2 Viewing Module Information

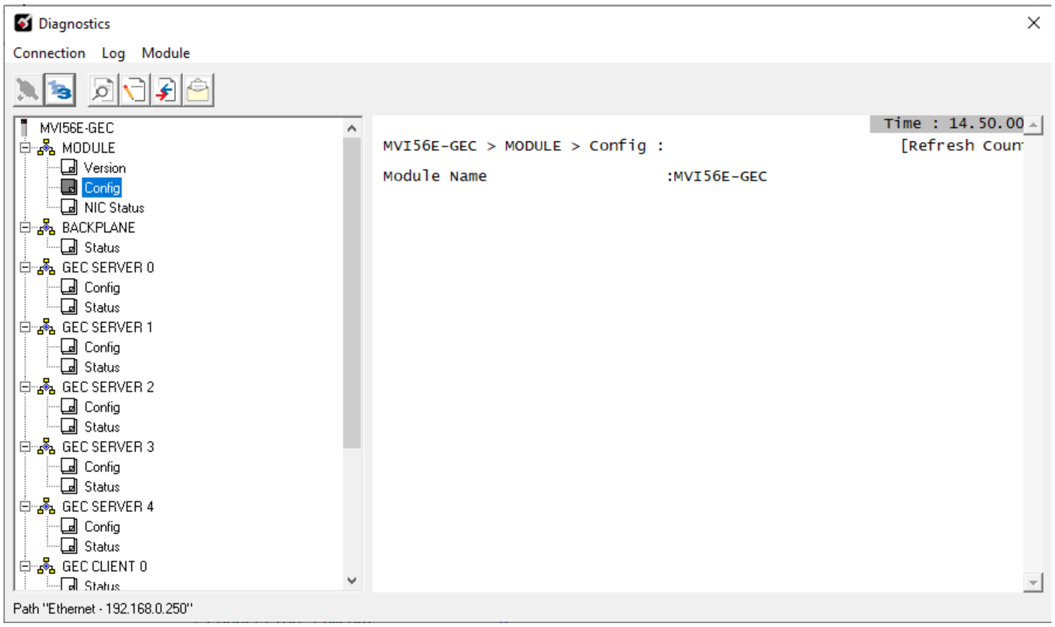
*MODULE > Version*



```
                                                      Time : 18.57.10
MVI56E-GEC > MODULE > Version :                   [Refresh Counter: 21]

PRODUCT NAME CODE              :GCE5
SOFTWARE REVISION LEVEL        :2.01
OPERATING SYSTEM REVISION      :1119
RUN NUMBER                     :2801
PROGRAM SCAN COUNTER           :47655
IP ADDRESS                     :192.168.0.250
ETHERNET ADDRESS (MAC)         :00:0d:8d:02:a6:e9
BACKPLANE DRIVER VERSION       :2.7
BACKPLANE API VERSION          :1.3
MODULE NAME                    :MVI56E-GEC
VENDOR ID                      :309
DEVICE TYPE                    :12
PRODUCT CODE                   :5004
SERIAL NUMBER                  :0000E073
REVISION                       :2.1
SLOT                           :1
```

Path "Ethernet - 192.168.0.250"

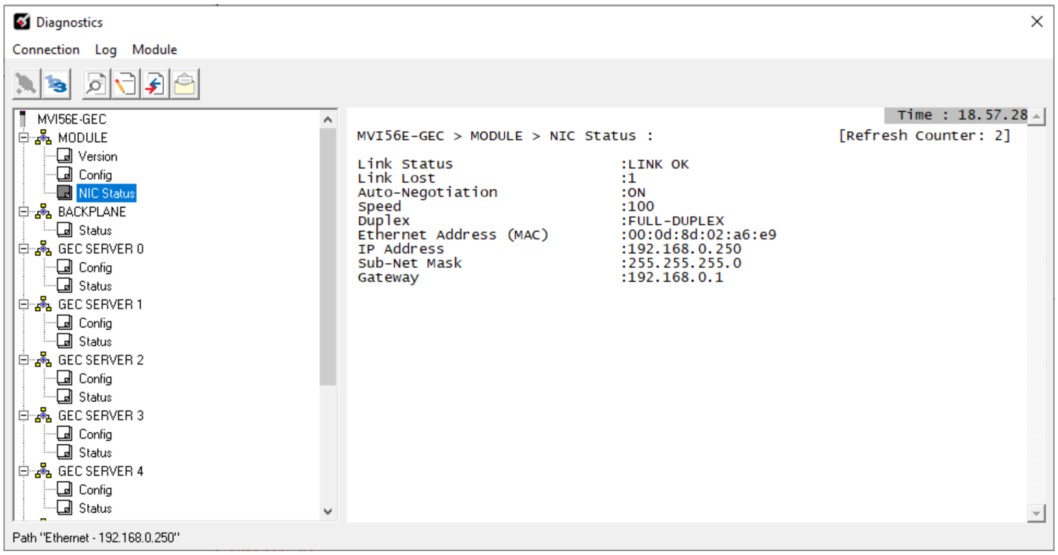
*MODULE > Config*



```
                                                 Time : 14.50.00
MVI56E-GEC > MODULE > Config :                   [Refresh Coun

Module Name                    :MVI56E-GEC
```

Path "Ethernet - 192.168.0.250"

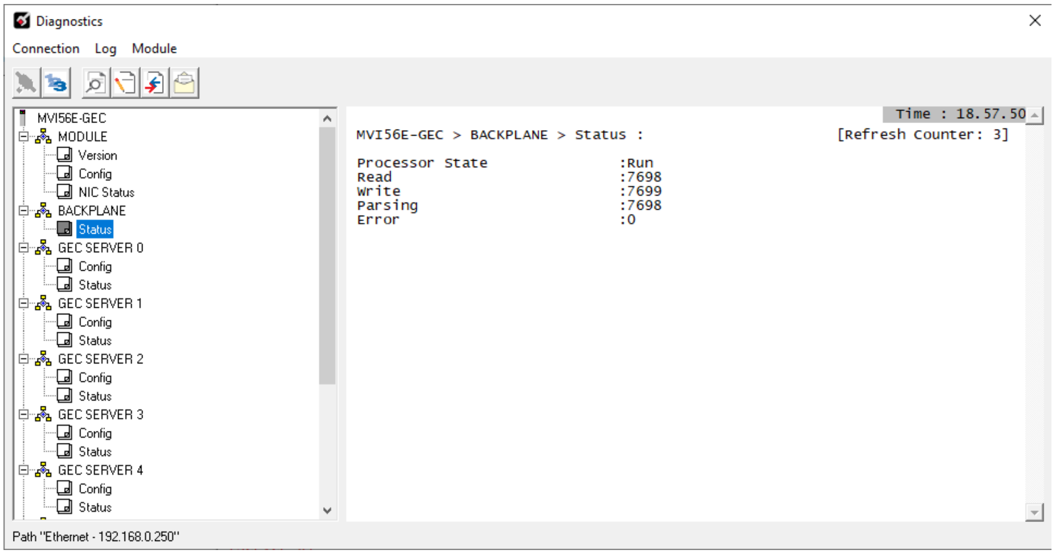
*MODULE > NIC Status*



```
                                                          Time : 18.57.28
MVI56E-GEC > MODULE > NIC Status :                     [Refresh Counter: 2]

Link Status                         :LINK OK
Link Lost                           :1
Auto-Negotiation                    :ON
Speed                               :100
Duplex                              :FULL-DUPLEX
Ethernet Address (MAC)              :00:0d:8d:02:a6:e9
IP Address                          :192.168.0.250
Sub-Net Mask                        :255.255.255.0
Gateway                             :192.168.0.1
```

### 5.2.3 Monitoring Backplane Information

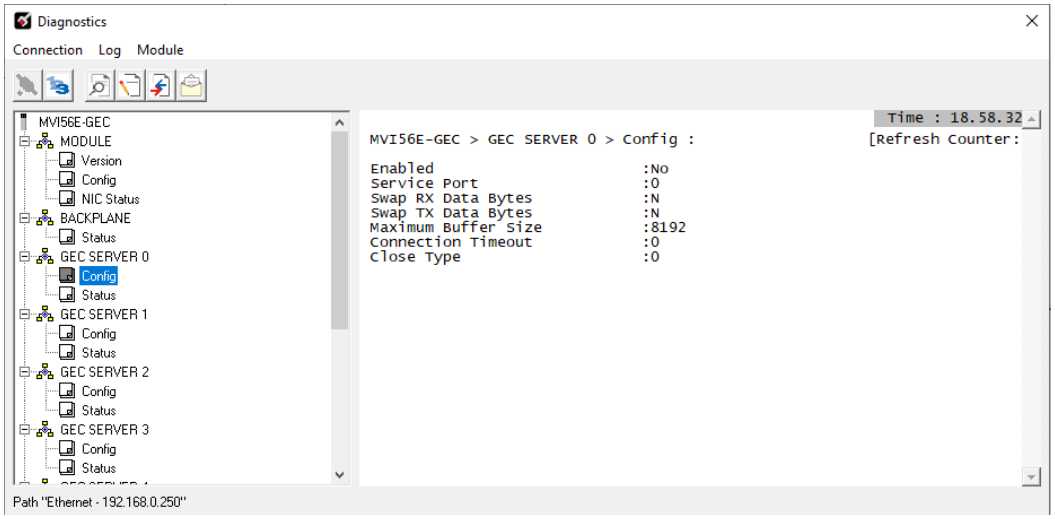
*BACKPLANE > Status*



```
                                                          Time : 18.57.50
MVI56E-GEC > BACKPLANE > Status :                      [Refresh Counter: 3]

Processor State                     :Run
Read                                :7698
Write                               :7699
Parsing                             :7698
Error                               :0
```

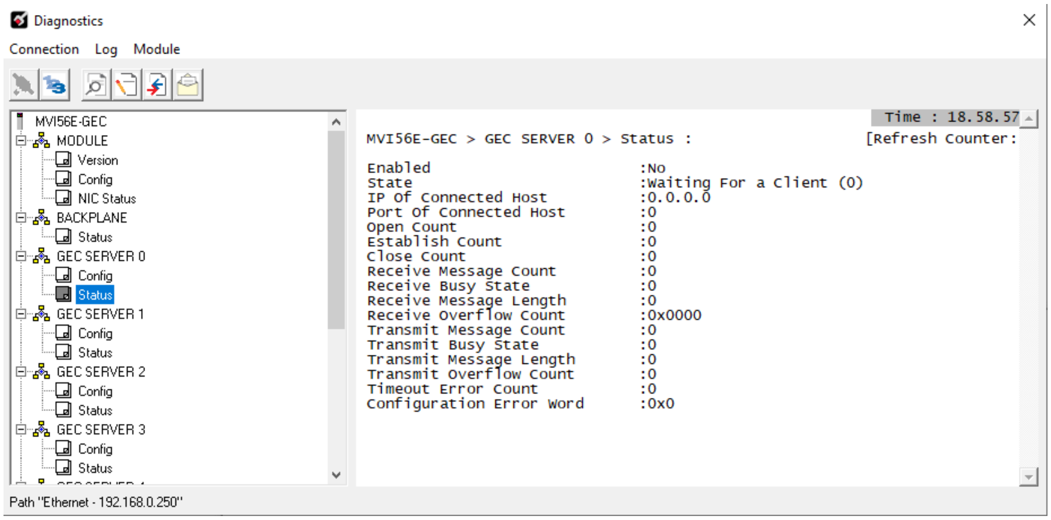
### 5.2.4 Monitoring MVI56E-GEC Server X Information

*GEC SERVER X > Config*



```
Time : 18.58.32
MVI56E-GEC > GEC SERVER 0 > Config :                  [Refresh Counter:

Enabled                        :No
Service Port                   :0
Swap RX Data Bytes             :N
Swap TX Data Bytes             :N
Maximum Buffer Size            :8192
Connection Timeout             :0
Close Type                     :0
```

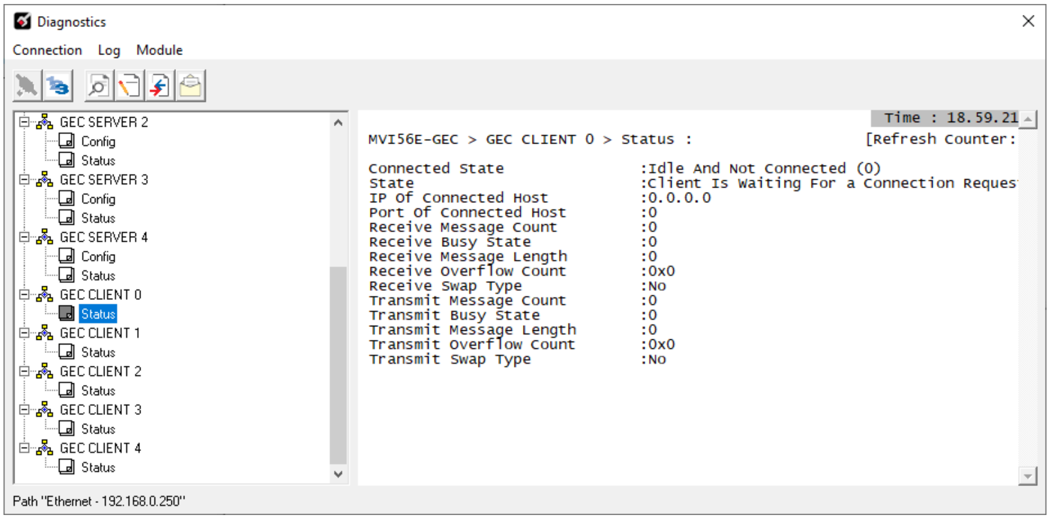
*GEC SERVER X > Status*



```
Time : 18.58.57
MVI56E-GEC > GEC SERVER 0 > Status :                  [Refresh Counter:

Enabled                        :No
State                          :Waiting For a Client (0)
IP Of Connected Host           :0.0.0.0
Port Of Connected Host         :0
Open Count                     :0
Establish Count                :0
Close Count                    :0
Receive Message Count          :0
Receive Busy State             :0
Receive Message Length         :0
Receive Overflow Count         :0x0000
Transmit Message Count         :0
Transmit Busy State            :0
Transmit Message Length        :0
Transmit Overflow Count        :0
Timeout Error Count            :0
Configuration Error Word       :0x0
```

### 5.2.5 Monitoring MVI56E-GEC Client X Information

*GEC CLIENT X > Status*



```
                                                    Time : 18.59.21
MVI56E-GEC > GEC CLIENT 0 > Status :               [Refresh Counter:

Connected State              :Idle And Not Connected (0)
State                        :Client Is Waiting For a Connection Reques
IP Of Connected Host         :0.0.0.0
Port Of Connected Host       :0
Receive Message Count        :0
Receive Busy State           :0
Receive Message Length       :0
Receive Overflow Count       :0x0
Receive Swap Type            :No
Transmit Message Count       :0
Transmit Busy State          :0
Transmit Message Length      :0
Transmit Overflow Count      :0x0
Transmit Swap Type           :No
```

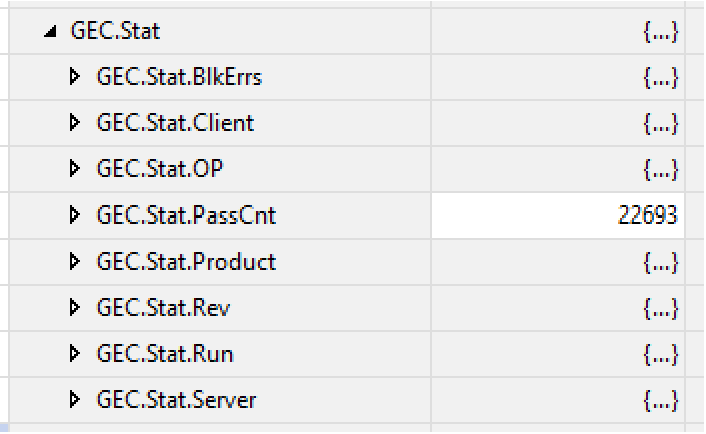
## 5.3 MVI56E-GEC Status Controller Tags

The MVI56E-GEC module returns status data to the ControlLogix processor in each read block. This data is transferred to the ControlLogix processor continuously with each read block.

The status data includes:

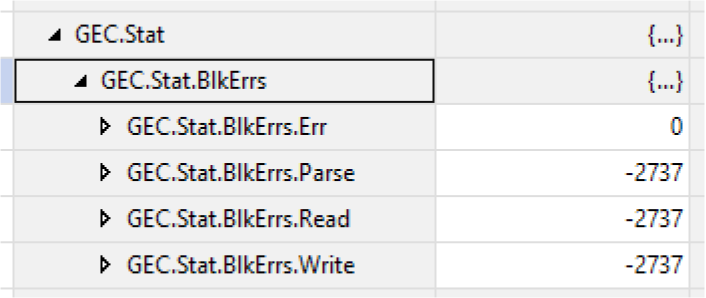
- Module version, product name, Operating Systems version
- Server Configuration and Status
- Client Configuration and Status

General status:



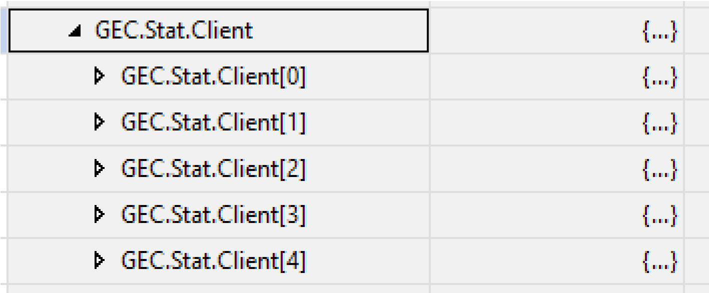
| | |
|---|---|
| ◢ GEC.Stat | {...} |
| ▷ GEC.Stat.BlkErrs | {...} |
| ▷ GEC.Stat.Client | {...} |
| ▷ GEC.Stat.OP | {...} |
| ▷ GEC.Stat.PassCnt | 22693 |
| ▷ GEC.Stat.Product | {...} |
| ▷ GEC.Stat.Rev | {...} |
| ▷ GEC.Stat.Run | {...} |
| ▷ GEC.Stat.Server | {...} |

Backplane status:

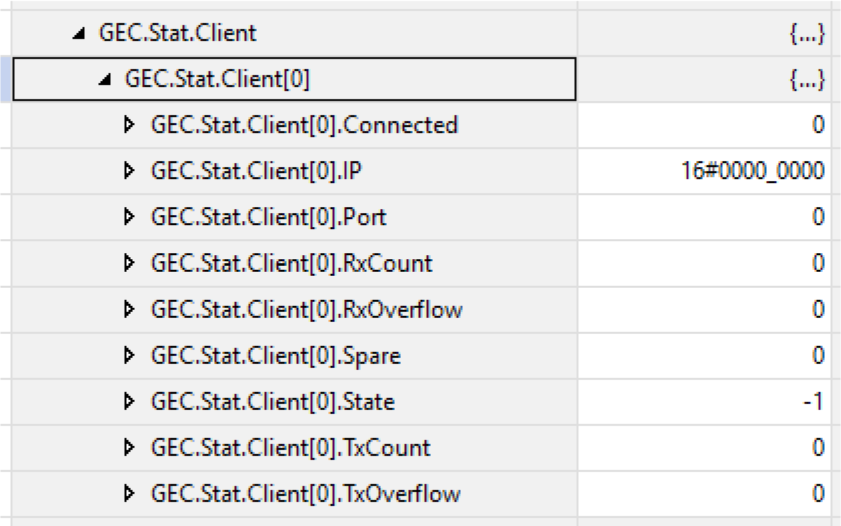


| | |
|---|---|
| ◢ GEC.Stat | {...} |
| ◢ GEC.Stat.BlkErrs | {...} |
| ▷ GEC.Stat.BlkErrs.Err | 0 |
| ▷ GEC.Stat.BlkErrs.Parse | -2737 |
| ▷ GEC.Stat.BlkErrs.Read | -2737 |
| ▷ GEC.Stat.BlkErrs.Write | -2737 |

Client Status:

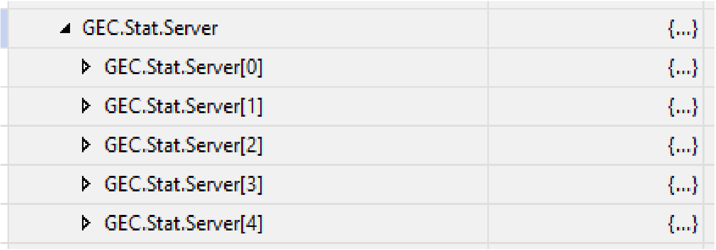


| | |
|---|---|
| ◢ GEC.Stat.Client | {...} |
| ▷ GEC.Stat.Client[0] | {...} |
| ▷ GEC.Stat.Client[1] | {...} |
| ▷ GEC.Stat.Client[2] | {...} |
| ▷ GEC.Stat.Client[3] | {...} |
| ▷ GEC.Stat.Client[4] | {...} |

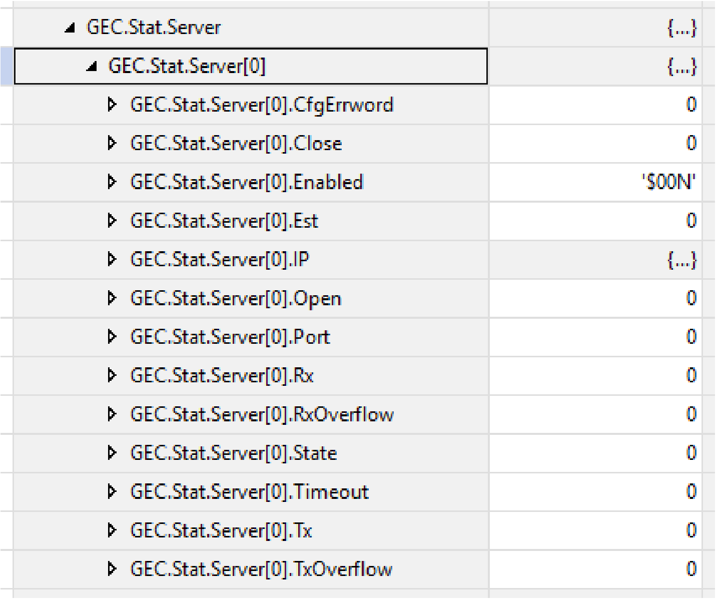


| | |
|---|---|
| ◢ GEC.Stat.Client | {...} |
| ◢ GEC.Stat.Client[0] | {...} |
| ▷ GEC.Stat.Client[0].Connected | 0 |
| ▷ GEC.Stat.Client[0].IP | 16#0000_0000 |
| ▷ GEC.Stat.Client[0].Port | 0 |
| ▷ GEC.Stat.Client[0].RxCount | 0 |
| ▷ GEC.Stat.Client[0].RxOverflow | 0 |
| ▷ GEC.Stat.Client[0].Spare | 0 |
| ▷ GEC.Stat.Client[0].State | -1 |
| ▷ GEC.Stat.Client[0].TxCount | 0 |
| ▷ GEC.Stat.Client[0].TxOverflow | 0 |

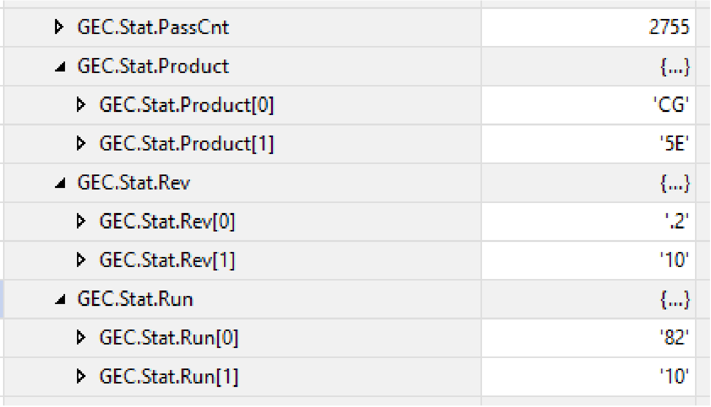
Server Status:



| | |
|---|---|
| ◢ GEC.Stat.Server | {...} |
| ▷ GEC.Stat.Server[0] | {...} |
| ▷ GEC.Stat.Server[1] | {...} |
| ▷ GEC.Stat.Server[2] | {...} |
| ▷ GEC.Stat.Server[3] | {...} |
| ▷ GEC.Stat.Server[4] | {...} |



| | |
|---|---|
| ◢ GEC.Stat.Server | {...} |
| ◢ GEC.Stat.Server[0] | {...} |
| ▷ GEC.Stat.Server[0].CfgErrword | 0 |
| ▷ GEC.Stat.Server[0].Close | 0 |
| ▷ GEC.Stat.Server[0].Enabled | '$00N' |
| ▷ GEC.Stat.Server[0].Est | 0 |
| ▷ GEC.Stat.Server[0].IP | {...} |
| ▷ GEC.Stat.Server[0].Open | 0 |
| ▷ GEC.Stat.Server[0].Port | 0 |
| ▷ GEC.Stat.Server[0].Rx | 0 |
| ▷ GEC.Stat.Server[0].RxOverflow | 0 |
| ▷ GEC.Stat.Server[0].State | 0 |
| ▷ GEC.Stat.Server[0].Timeout | 0 |
| ▷ GEC.Stat.Server[0].Tx | 0 |
| ▷ GEC.Stat.Server[0].TxOverflow | 0 |

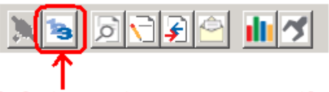


| | |
|---|---|
| ▷ GEC.Stat.PassCnt | 2755 |
| ◢ GEC.Stat.Product | {...} |
| ▷ GEC.Stat.Product[0] | 'CG' |
| ▷ GEC.Stat.Product[1] | '5E' |
| ◢ GEC.Stat.Rev | {...} |
| ▷ GEC.Stat.Rev[0] | '.2' |
| ▷ GEC.Stat.Rev[1] | '10' |
| ◢ GEC.Stat.Run | {...} |
| ▷ GEC.Stat.Run[0] | '82' |
| ▷ GEC.Stat.Run[1] | '10' |

## 5.4 Connecting to the Module's Webpage

The module's internal web server provides access to module version and status information, as well as the ability to set the date and time, reboot the module, and download firmware upgrade to the module. Enter the assigned IP address of the module into a web browser or use the following steps in PCB.

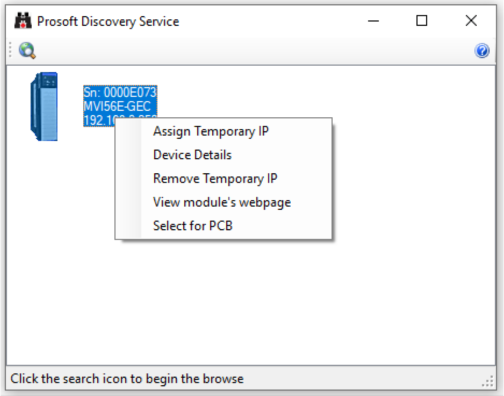
1 In the PCB Diagnostics window, click the **SET UP CONNECTION** button.



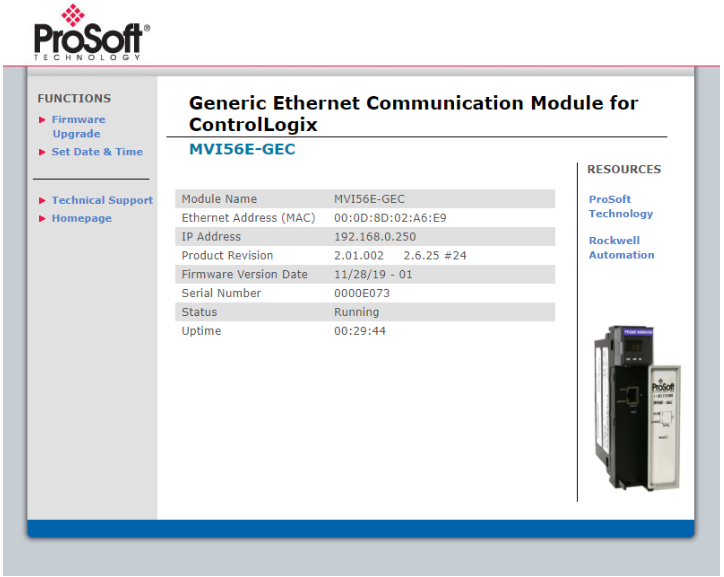
Click to set up connection

2 In the *Connection Setup* dialog box, click **BROWSE DEVICE(S)** to start *ProSoft Discovery Service*.

3 Right-click the module icon and choose **VIEW MODULE'S WEBPAGE** to launch your default browser and display the module's webpage.



Click the search icon to begin the browse



**Generic Ethernet Communication Module for ControlLogix**

**MVI56E-GEC**

| | |
|---|---|
| Module Name | MVI56E-GEC |
| Ethernet Address (MAC) | 00:0D:8D:02:A6:E9 |
| IP Address | 192.168.0.250 |
| Product Revision | 2.01.002 2.6.25 #24 |
| Firmware Version Date | 11/28/19 - 01 |
| Serial Number | 0000E073 |
| Status | Running |
| Uptime | 00:29:44 |

# 6 Reference

## 6.1 Product Specifications

The MVI56E-GEC Generic ASCII Ethernet Interface module is designed to allow ControlLogix processors to interface easily with ASCII devices using the TCP/IP protocol and the ControlLogix processor. Compatible devices may be either ASCII instruments with Ethernet built-in or Ethernet connection via a thin server to the existing ASCII device.

Five servers and clients are present on the module permitting both the reception and transmission of data between the Rockwell Automation processor and attached devices.

The MVI56E-GEC module is a powerful communication interface for ControlLogix processors. Developed under license from Rockwell Automation, the module incorporates proprietary backplane technology that enables powerful data access between the module and the ControlLogix processor.

Five servers and clients are present on the module permitting both the reception and transmission of data between the ControlLogix processor and attached devices.

The MVI56E-GEC module is a powerful communication interface for ControlLogix processors. Developed under license from Rockwell Automation, the module incorporates proprietary backplane technology that enables powerful data access between the module and the ControlLogix processor.

### *6.1.1 General Specifications*

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

### 6.1.2 Hardware Specifications

| Specification | Description |
|---|---|
| Backplane Current Load | 800 mA @ 5 Vdc<br>3 mA @ 24 Vdc |
| Operating Temperature | 0°C to 60°C (32°F to 140°F) |
| Storage Temperature | -40°C to 85°C (-40°F to 185°F) |
| Shock | 30 g operational<br>50 g non-operational<br>Vibration: 5 g from 10 Hz to 150 Hz |
| Relative Humidity | 5% to 95% (without condensation) |
| LED Indicators | Application Status (APP)<br>Module Status (OK) |
| 4-Character, Scrolling, Alpha-Numeric LED Display | Shows Module, Version, IP, Application Port Setting, Port Status, and Error Information |

| Ethernet Port (E1) | Description |
|---|---|
| Ethernet Port | 10/100 Base-T, RJ45 Connector, for CAT5 cable<br>Link and Activity LED indicators<br>Auto-crossover cable detection |

### 6.1.3 Functional Specifications

- Five Servers and five Clients to receive and/or transmit data
- 10/100 Base-T Ethernet-compatible interface
- Configurable parameters
    - Service port number
    - Connection timeout
    - Close type
- Simple ladder logic operation
- Setup and monitoring through RS-Logix 5000 software
- ControlLogix backplane interface via I/O access
- Each Server monitors
    - State
    - IP and port number of connected Client
    - Error codes
- Each Client monitors
    - State
    - IP and port number of connected Server
    - Message related parameters
- ASCII character strings up to 2048 characters in length supported
- Module error and status conditions returned to processor for diagnostic purposes
    - Module status
    - Port error status word (bit mapped)
    - Port receive state
    - Port receive character count
    - Port transmit state
    - Port transmit character count
- All data related to the module is contained in a single controller tag with defined objects to simplify the configuration, monitoring, and interfacing with the module
- Module configuration and communication configuration data is transferred to the MVI56E-GEC via a pre-defined user data type in the processor

## 6.2 Functional Overview

### *6.2.1 General Concepts*

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

*Module Power Up*

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

**1** Initialize hardware components

- Initialize ControlLogix backplane driver
- Test and clear all RAM
- Initialize the TCP/IP stack and Ethernet interface

**2** Initialize servers and clients

After the module has received the configuration, the module will begin receiving and transmitting messages with clients and servers on the Ethernet network.

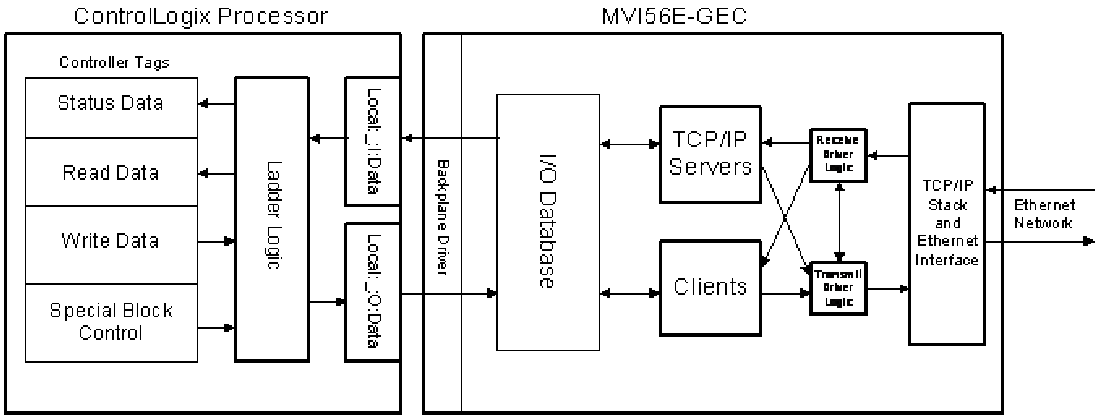
*Backplane Data Transfer*

The MVI56E-GEC module communicates directly over the ControlLogix backplane. Data travels between the module and the ControlLogix processor across the backplane using the module's input and output images. The update frequency of the data is determined by the user-defined scan rate for the module, and the communication load on the module. Typical updates are in the range of 2 to 10 milliseconds.

Data received by the module's clients and servers is placed in the module's Database. This data is read and then processed by the ladder logic in the ControlLogix processor through the Input Image (Local Input connection). The input image parameter in Studio 5000 Logix Designer for the module is set to 250 registers (500 bytes).

The processor writes data through the Output Image (Local Output connection) for transfer into the module's Database. The module's program extracts the data from the module's Database and transmits the data out to the Ethernet network. Each message is directed to a client or server that is connected to a remote host. The Output image parameter in Studio 5000 Logix Designer for the module is set to 248 registers (496 bytes). This large I/O area permits fast throughput of data between the module and the processor.

The following illustration shows the data transfer method used to move data between the ControlLogix processor, the MVI56E-GEC module, and the Ethernet network.



All data transferred between the module and the processor over the backplane is through the input and output images. Ladder logic must be written in the ControlLogix processor to interface the input and output image data defined in the controller tags.

You must configure the module properly for accurate reception of data through its ports, and you must program the ladder logic to interpret the data from the module and send the correct response to the module.

Use the sample ladder logic program to establish data transfer between the module and the processor.

*Normal Data Transfer*

Normal data transfer includes data transmitted between clients and servers, and status data. Data transfer takes place through read (input image) and write (output image) blocks.

Refer to Installing and Configuring the Module for a description of the data objects used with the blocks, and the ladder logic required.

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

### Read Block

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

| | Word Offset | Description |
|---|---|---|
| Received Data | 0 | Server/Client Number for data received. If the word contains a value of -1, no receive data is present. If the word contains a value from 0 to 4, the block contains data for one of the servers in the module. If the word contains a value of 10 to 14, the block contains data for one of the clients. |
| | 1 | Number of characters (0 to 200) in receive block (2 to 101). 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. |
| | 2 to 101 | 200 bytes of data received for specified server or client. |
| Module Status | 102 to 234 | This data area contains the status data block. Each block transferred to the processor contains this set of information. |
| IDs for Servers, Clients and Message Blocks | 235 to 246 | Reserved for future use. |
| | 247 | Value for designated Server or Client number during writing process (0-4) for Servers (10-14) for clients. |
| | 248 | Number of characters processed from last write block (0 to 400). |
| | 249 | Block Sequence Number (Bumped each scan by module) |

The Block Sequence Number (word offset 249) is an index value that signals to the ControlLogix processor that a new block is ready for processing. The ladder logic must recognize a change in this value, and then process the data encapsulated in the input image. If data is available for a server, the module passes a block containing data received and the complete status data area to the processor. The value at word 0 in the block contains one of the following values:

| Value | Description |
|---|---|
| -1 | No receive data in block |
| 0 | Server 0 data in block |
| 1 | Server 1 data in block |
| 2 | Server 2 data in block |
| 3 | Server 3 data in block |
| 4 | Server 4 data in block |
| 10 | Client 0 data in block |
| 11 | Client 1 data in block |
| 12 | Client 2 data in block |
| 13 | Client 3 data in block |
| 14 | Client 4 data in block |

Word 1 of the input image determines the number of bytes in the data area of the block. This word only has significance if word 0 of the block is set to a valid server index value (0 to 4), or client index value (10 to 14).

### Multiple Blocks

If a server or client receives a message longer than 200 bytes, it must send the received message in multiple blocks to the processor. In this case, the byte count field of the block will be set to -1 for each block where the server is sending more than 200 characters. Each block with a byte count field of -1 contains 200 bytes of data. The data set is located in the block starting at word offset 2. When the last block of data to send by the server is less than or equal to 200 bytes, the byte count field will be set to a number from 1 to 200. This signifies to the processor that this is the last block. The ladder logic must handle data received on each server or client enabled in the module.

The module status data begins at word offset 102. Refer to Installing and Configuring the Module for a full discussion of this data set**.** The ladder logic should use the state value of each server to determine which servers have open connections (state value of 1). Ladder logic can send messages to any open connection.

Word 248 of the message informs the ladder logic of the number of bytes sent in the last write block that were processed by a server. Ladder logic should check to make sure all bytes sent to a server are processed. If not, data may be lost.

The last word of the input image (word offset 249) is the block sequence number. This word's value changes each time the module has processed the previous output image or is sending new information to the processor. This value should be copied to the first word of the output image (write block sequence number). Both the processor and the module can easily recognize the arrival of new data by the sequence number.

### Write Block

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

| Output Image | Word Offset | Description |
|---|---|---|
| | 0 | Block Sequence Number (Read block number as set by module) |
| Transmit Data | 1 | Server/Client Number for data to transmit. If the word contains a value of -1, no transmit data is present. If the word contains a value from 0 to maximum number of servers -1, the block contains data to send from the specified server in the module. If the word contains a value from 10 to 14, then the block contains data to send from one of the clients. |
| | 2 | Number of characters to transmit to a server or client. Each block can transmit up to 400 bytes of data. If this word contains a value of -1, the block contains 400 bytes of data and more blocks of data are to follow. When the module received the last block containing a positive value representing the number of bytes in the block, the server or client will transmit the data to the end device. |
| | 3 to 202 | Data to transmit on specified server or client. |
| Transmit Control | 203 to 212 | Client connection request data. |
| | 213 to 235 | Reserved for future use |
| | 236 | Server[0] Control Word (1=Abort) |
| | 237 | Server[1] Control Word (1=Abort) |
| | 238 | Server[2] Control Word (1=Abort) |
| | 239 | Server[3] Control Word (1=Abort) |
| | 240 | Server[4] Control Word (1=Abort) |
| | 241 | Client[0] Control Word |
| | 242 | Client[1] Control Word |
| | 243 | Client[2] Control Word |
| | 244 | Client[3] Control Word |
| | 245 | Client[4] Control Word |
| | 246 | Server Control Word |
| | 247 | Module Control Word contains a value of 0, -1, -2 or -3. |

The Block Sequence Number is received on the last read block transfer through the input image on the module. The ladder logic should copy this value from word 249 of the input image to word 0 of output image in the ladder logic.

| Block ID | Description |
|---|---|
| 0 to 65535 | Normal data blocks for read and write |

**Note:** This is the last operation performed when constructing the write block. The module's program will trigger the Process Write Block function when it recognizes a new value in word 0 of the output image.

Words 236 to 240 are used for server socket control. If the server socket is established and the user sets the value in the appropriate word to 1, the server state will be changed to 1003 (new server state) and will abort the socket immediately. It will then reopen the socket to wait for a new connection. These words should be set to zero when no operation is required.

Word 1 of the block contains one of the following values:

| Value | Description |
|---|---|
| -1 | No transmit data in block |
| 0 | Server 0 data in block |
| 1 | Server 1 data in block |
| 2 | Server 2 data in block |
| 3 | Server 3 data in block |
| 4 | Server 4 data in block |
| 10 | Client 0 data in block |
| 11 | Client 1 data in block |
| 12 | Client 2 data in block |
| 13 | Client 3 data in block |
| 14 | Client 4 data in block |

If the word is set to a value of -1, there is no data in the message. A value of 0 to 4 will cause the enclosed message to be sent to the server if the message length is set to a value other than zero and the server has an open connection. A value of 10 to 14 will cause the enclosed message to be sent to the client if a message length is set to a value other than zero. Word 2 of the block defines the length of data in the block.

Up to 400 bytes of data can be sent in each block starting at word offset 3 in the block. If the message to be sent to a server is longer than 400 bytes, multiple blocks are required. If more than 400 bytes are required to send, the message length field (Word Offset 2) should be set to a value of -1 and 400 bytes of data should be placed in the block. When 400 or fewer bytes remain to be sent, the message length field should be set to that value and the remaining data placed in the block.

The client or server will accept messages up to a length of 2048 bytes. If a message over 2048 bytes is sent, the data in the bytes over the max size roll over and are placed at the beginning of the *GEC.CLIENTS[X].READDATA* array.

For example, a message of 2050 bytes will result in the 2049th byte in *GEC.CLIENTS[X].READDATA[0]* and the 2050th byte in *GEC.CLIENTS[X].READDATA[1]*.

Word offsets 203 and 212 request a new connection for one of the five clients in the module. The format of this data area is shown in the following table.

| Word | Description |
|---|---|
| 203 | Client to utilized for connection (10 to 14) |
| 204 | Reserved for future use. |
| 205 to 208 | IP address of server to which connection will be made. Each word contains one of the digits of a dotted notation IP address. |
| 209 | Service port in server to which connection will be made. This service must be available in the server for the connection to succeed. |
| 210 | Swap Rx data bytes (0 = No, not 0 = Yes) |
| 211 | Swap Tx data bytes (0 = No, not 0 = Yes) |
| 212 | Client TimeOut value in milliseconds. The client will close the connection with remote Server after the specified milliseconds once data transfer ceases between the client and the remote Server. A value of 0 will keep the connection open indefinitely. |

Ladder logic is required to transfer this data to the block when a connection is required. The ladder logic should also clear the area after the connection is requested. The example ladder logic present in this document presents code to accomplish these tasks.

### Client Control Word Codes

Words 241 to 245 close the connection on one of the clients. The following table lists the values recognized by the module for these words:

| Code | Definition |
|---|---|
| 0 | No operation to perform |
| 1 | Close the client socket |
| 2 | Abort the client socket |

### Server Control Word Codes

The last two words of the image control the server or the module. The server control word (word 246) controls the server. The following table lists the values recognized by the module:

| Code | Definition |
|---|---|
| 0 | No operation to perform |
| 1 | Close socket after transmit operation |
| 2 | Abort socket after transmit operation |

If a value of 0 is present in the field, the server takes no action. If a value of 1 is present in the field, the server will gently close the socket after it transmits the message contained in the block. If a value of 2 is present, the server will send the message contained in the block and then force the connection closed by sending a reset message to the client.

### Module Control Word Codes

Word 247 is used to control the module. The following table lists the commands recognized by the module.

| Code | Definition |
|---|---|
| 0 | No operation to perform |
| -1 | Warm boot module |
| -2 | Cold boot module |
| -3 | Build configuration files |

The module will perform the action specified in the command control word.

### Configuration Block

In order to support module configuration from the ladder logic, a write block format is required when the Control Word is set to -3. The following table shows the general structure of the block:

| | Word Offset | Description |
|---|---|---|
| | 0 | Block Sequence Number (Read block number as set by module) |
| [Module] | 1 to 40 | Module Name (80 characters of data with null termination) |
| | 41 to 50 | Password (20 characters with null termination) |
| [Server 0] | 51 | Enabled (0 = No, not 0 = Yes) |
| | 52 | Service port |
| | 53 | Connection Timeout |
| | 54 | Connection Close Type (0, 1 or 2) |
| | 55 | Swap Rx Data Bytes (0 = No, not 0 = Yes) |
| | 56 | Swap Tx Data Bytes (0 = No, not 0 = Yes) |
| [Server 1] | 57 | Enabled (0 = No, not 0 = Yes) |
| | 58 | Service port |
| | 59 | Connection Timeout |
| | 60 | Connection Close Type (0, 1 or 2) |
| | 61 | Swap Rx Data Bytes (0 = No, not 0 = Yes) |
| | 62 | Swap Tx Data Bytes (0 = No, not 0 = Yes) |
| [Server 2] | 63 | Enabled (0 = No, not 0 = Yes) |
| | 64 | Service port |
| | 65 | Connection Timeout |
| | 66 | Connection Close Type (0, 1 or 2) |
| | 67 | Swap Rx Data Bytes (0 = No, not 0 = Yes) |
| | 68 | Swap Tx Data Bytes (0 = No, not 0 = Yes) |
| [Server 3] | 69 | Enabled (0 = No, not 0 = Yes) |
| | 70 | Service port |
| | 71 | Connection Timeout |
| | 72 | Connection Close Type (0, 1 or 2) |
| | 73 | Swap Rx Data Bytes (0 = No, not 0 = Yes) |
| | 74 | Swap Tx Data Bytes (0 = No, not 0 = Yes) |
| [Server 4] | 75 | Enabled (0 = No, not 0 = Yes) |
| | 76 | Service port |
| | 77 | Connection Timeout |
| | 78 | Connection Close Type (0, 1 or 2) |
| | 79 | Swap Rx Data Bytes (0 = No, not 0 = Yes) |
| | 80 | Swap Tx Data Bytes (0 = No, not 0 = Yes) |
| My_IP | 81 to 84 | The IP address for the module. Each word contains one of the four values that comprise the IP address. For example, 192.168.0.100. |
| Netmask | 85 to 88 | Network mask. Each word contains one of the values for the network mask. |
| Gateway | 89 to 92 | Default gateway |
| | 93 to 96 | Default network |
| | 97 to 100 | Default subnet mask |
| | 101 to 246 | Reserved for future use |
| | 247 | Module Control Word contains a value of -3 to indicate build configuration file option. |

When configuring the module from the ladder logic, use numeric values to select options (0 for "No" and non-zero for "Yes") rather than text values. After the module receives this block, it will build the two configuration files and perform a reboot operation. The module will then use the new configuration.

### Handling Multiple Blocks

It is important to understand how to handle multiple blocks. The module can only send 200 bytes at each scan to the processor. For example, if a device sends a message that contains 1000 bytes to the module, the module will break the message down into 5 blocks of 200 bytes. The first four blocks will set the number of characters parameter as -1, indicating that each block is part of the same message. The last block sets the number of characters to 200, indicating that there are no more blocks from that message.

The concept is the same for writing data from the processor to the module, except that you can write up to 400 bytes at a time to the module. The module buffers all data until it receives a block that contains the "number of characters" parameter set to >=0. In this case, it sends all data to the client that is connected to the server, or the server connects to the client.

**Important:** Messages are usually broken down into smaller frames by the IP layer in a specific LAN or WAN according to the Maximum Transmit Unit (MTU) of the network.

For example, a message that contains 2000 bytes can be broken down into 2 messages by the IP layer in the network (after it is sent to the module). The same issue is applied when a client sends data to the server; although a client sends a single message to the module, it could be broken down into smaller fragments before it gets to the module. In this case, the module would interpret it as two different messages.

The application layer defines when a message is finished. This is the reason why the user should consider using some sort of control so the ladder could identify different messages as actually part of one single message. This could be accomplished either by using a specific character at the end of each message or by using a fixed length for each message.

### *Network Data Transfer*

In order for data to be transferred between the module and another device, a TCP/IP connection must be made between a client and a server on the module. The MVI56E-GEC module contains five servers that listen on the user assigned service ports waiting for a connection. When a client wishes to send data to the module, it must open a TCP/IP connection to the module. After the connection is established, either device can send and receive data. When either device is finished with the connection, the connection must be closed. This operation can be initiated from either end device.

The MVI56E-GEC module servers and clients are configured to handle their TCP/IP session independently. The user parameter Connection Timeout is utilized to determine the amount of time a connection can remain idle before the server will close the connection. If the parameter is set to 0, the server will not perform the timeout logic and the socket will never be closed by the server on an idle condition. If this feature is utilized, it can prevent connections that may be lost and were not properly closed. After the connection is established, the ladder logic should verify that the client had not been communicating for some time and close the connection.

Each server on the module is assigned its own server port number. This does not mean that two or more servers cannot share the same port number. In fact this might be desirable in some instances. It is up to the ladder logic to keep track of each message and to insure that a request/response transaction is associated with the correct connection. Information to keep track of each connection is passed in each input image. The status data set provides the IP address and TCP port address for the connection on each server. Each message transferred between the module and the processor has a server index word. This word associates the message with a server, which is associated with a connection to a specific IP address and TCP port address. Therefore, each connection is specified to the processor by the server index. The following illustration shows a snapshot of the modules status data:

| Server | Status Data | Description of Server |
|---|---|---|
| Server 0 | IP of Host (192.168.0.100)<br>Port of Host (1243)<br>State = 1 | This server is connected (State=1) to IP address 192.168.0.100 on TCP port 1243. |
| Server 1 | IP of Host (192.168.0.100)<br>Port of Host (1244)<br>State = 1 | This server is connected (State=1) to IP address 192.168.0.100 on TCP port 1244. |
| Server 2 | IP of Host (192.168.0.101)<br>Port of Host (56443)<br>State = 1 | This server is connected (State=1) to IP address 192.168.0.101 on TCP port 56443. |
| Server 3 | IP of Host (192.168.0.102)<br>Port of Host (7943)<br>State = 1 | This server is connected (State=1) to IP address 192.168.0.102 on TCP port 7943. |
| Server 4 | IP of Host (0.0.0.0)<br>Port of Host (0)<br>State = 0 | This server is not connected (State not equal to 1) and is waiting for a connection. |

Ladder logic can send messages to the clients connected to servers 0 to 3. Messages sent to server 4 will not be sent from the module because there is no connection active on that server.

Each server has a state value in the status data area. This value is utilized by the ladder logic to determine if a connection is present on server. The following table defines the state status values used by each server:

### Server state list

| State Value | Definition |
|---|---|
| -1 | Server is initializing and is being set up to listen. |
| 0 | The server is waiting for a client to establish a connection. |
| 1 | The server has established a connection with a client and can send or receive data. |
| 1000 | The server has initiated a close operation on the connection. |
| 1001 | The server is waiting for the close on the connection to complete. |
| 1002 | The server is issuing an abort (reset) on the connection. The socket is forced closed. |
| 1003 | The server is issuing an abort (reset) on the connection due to control from user. |

### Client state list

| State Value | Definition |
|---|---|
| -1 | Client is waiting for a connection request. |
| 0 | The client is waiting to establish the connection with the server. |
| 1 | The client has established a connection with the server and can send and receive data. |
| 1000 | The client has initiated a close operation on the connection. |
| 1001 | The client is waiting for the close on the connection to complete. |
| 1002 | The client is issuing an abort (reset) on the connection. The socket is forced closed. |

### Client Connection States

| State Value | Definition |
|---|---|
| -4 | The Client is idle and not connected. It was disconnected by time value entered in the tag for clients' connection setup Time-Out register. |
| -3 | Server closed connection for client or server is not available. |
| -2 | Unable to open connection with specified server. |
| -1 | Unable to open connection with specified server because of invalid IP address. |
| 0 | The client is idle and not connected. Disconnected by the ladder logic timer. |
| 1 | The client set to connect to the server and waiting for the connection to establish. |
| 2 | The client is connected to the server and can transfer data. |
| 3 | The connection is being closed for the client. |

Ladder logic should only direct messages to servers that have a state status value of 1. The module will ignore all messages sent to servers with any other state value.

When the ladder logic sends a message to a server, it can request that the socket be closed after the message is sent. The server control word in the output image is used for this purpose. Place a value of 1 in this register to gently close the connection after the message is sent. If a value of 2 is placed in the register, the server will abort the connection to force the socket closed (send a message with the Reset Flag set to the client). Most applications will have the client close the socket.

## 6.3 Status Data

This section contains a listing of the data contained in the MVI56E-GEC status data object.

| Object in GSCInStat | Block Offset Start | Description |
|---|---|---|
| PassCnt | 102 | Program cycle counter |
| Product | 103 | Product name as ASCII string |
| Rev | 105 | Revision level as ASCII string |
| OP | 107 | Operating system level as ASCII string |
| Run | 109 | Run number as ASCII string |
| BlkErrs.Read | 111 | Number of blocks transferred from module to processor |
| BlkErrs.Write | 112 | Number of blocks transferred from processor to module |
| BlkErrs.Parse | 113 | Number of blocks parsed by module |
| BlkErrs.Err | 114 | Number of block errors in module |
| Server[0].Enabled | 115 | This flag defines if the server is utilized. A value of 0 indicates the server is not used. Any other value indicates the server is used. |
| Server[0].State | 116 | This flag defines the current state of the server. |
| Server[0].IP | 117 | This double-word value contains the IP address of the client connected to the server. |
| Server[0].Port | 119 | This word value contains the port address for the client connected to the server. |
| Server[0].Open | 120 | Indicates that the module has opened the server port and is waiting for a connection to a client. |
| Server[0].Established | 121 | This status value contains the total number of times a connection was established on the socket. |
| Server[0].Closed | 122 | This status value contains the total number of times a close operation was performed on the socket. |
| Server[0].RxCount | 123 | This status value contains the total number of messages received by the server. |
| Server[0].RxOverflow | 124 | This status value contains the total number of messages received that exceed the user specified buffer size for the server. |
| Server[0].TxCount | 125 | This status value contains the total number of messages transmitted by the server. |
| Server[0].TxOverflow | 126 | This status value contains the total number of transmit messages that exceeded the user specified maximum buffer size for the server. |
| Server[0].Timeout | 127 | This status value contains the total number of times a connection timeout occurred on the socket. This status value is incremented each time a connection timeout or a socket close timeout (2 seconds) occurs. |
| Server[0].CfgErrWord | 128 | This bit mapped word defines the configuration errors for the server. |
| Server[1].Enabled | 129 | This flag defines if the server is utilized. A value of 0 indicates the server is not used. Any other value indicates the server is used. |
| Server[1].State | 130 | This flag defines the current state of the server. |
| Server[1].IP | 131 | This double-word value contains the IP address of the client connected to the server. |
| Server[1].Port | 133 | This word value contains the port address for the client connected to the server. |

| Object in GSCInStat | Block Offset Start | Description |
|---|---|---|
| Server[1].Open | 134 | Indicates that the module has opened the server port and is waiting for a connection to a client. |
| Server[1].Established | 135 | This status value contains the total number of times a connection was established on the socket. |
| Server[1].Closed | 136 | This status value contains the total number of times a close operation was performed on the socket. |
| Server[1].RxCount | 137 | This status value contains the total number of messages received by the server. |
| Server[1].RxOverflow | 138 | This status value contains the total number of messages received that exceed the user specified buffer size for the server. |
| Server[1].TxCount | 139 | This status value contains the total number of messages transmitted by the server. |
| Server[1].TxOverflow | 140 | This status value contains the total number of transmit messages that exceeded the user specified maximum buffer size for the server. |
| Server[1].Timeout | 141 | This status value contains the total number of times a connection timeout occurred on the socket. This status value is incremented each time a connection timeout or a socket close timeout (2 seconds) occurs. |
| Server[1].CfgErrWord | 142 | This bit mapped word defines the configuration errors for the server. |
| Server[2].Enabled | 143 | This flag defines if the server is utilized. A value of 0 indicates the server is not used. Any other value indicates the server is used. |
| Server[2].State | 144 | This flag defines the current state of the server. |
| Server[2].IP | 145 | This double-word value contains the IP address of the client connected to the server. |
| Server[2].Port | 147 | This word value contains the port address for the client connected to the server. |
| Server[2].Open | 148 | Indicates that the module has opened the server port and is waiting for a connection to a client. |
| Server[2].Established | 149 | This status value contains the total number of times a connection was established on the socket. |
| Server[2].Closed | 150 | This status value contains the total number of times a close operation was performed on the socket. |
| Server[2].RxCount | 151 | This status value contains the total number of messages received by the server. |
| Server[2].RxOverflow | 152 | This status value contains the total number of messages received that exceed the user specified buffer size for the server. |
| Server[2].TxCount | 153 | This status value contains the total number of messages transmitted by the server. |
| Server[2].TxOverflow | 154 | This status value contains the total number of transmit messages that exceeded the user specified maximum buffer size for the server. |
| Server[2].Timeout | 155 | This status value contains the total number of times a connection timeout occurred on the socket. This status value is incremented each time a connection timeout or a socket close timeout (2 seconds) occurs. |
| Server[2].CfgErrWord | 156 | This bit mapped word defines the configuration errors for the server. |

| Object in GSCInStat | Block Offset Start | Description |
|---|---|---|
| Server[3].Enabled | 157 | This flag defines if the server is utilized. A value of 0 indicates the server is not used. Any other value indicates the server is used. |
| Server[3].State | 158 | This flag defines the current state of the server. |
| Server[3].IP | 159 | This double-word value contains the IP address of the client connected to the server. |
| Server[3].Port | 161 | This word value contains the port address for the client connected to the server. |
| Server[3].Open | 162 | Indicates that the module has opened the server port and is waiting for a connection to a client. |
| Server[3].Established | 163 | This status value contains the total number of times a connection was established on the socket. |
| Server[3].Closed | 164 | This status value contains the total number of times a close operation was performed on the socket. |
| Server[3].RxCount | 165 | This status value contains the total number of messages received by the server. |
| Server[3].RxOverflow | 166 | This status value contains the total number of messages received that exceed the user specified buffer size for the server. |
| Server[3].TxCount | 167 | This status value contains the total number of messages transmitted by the server. |
| Server[3].TxOverflow | 168 | This status value contains the total number of transmit messages that exceeded the user specified maximum buffer size for the server. |
| Server[3].Timeout | 169 | This status value contains the total number of times a connection timeout occurred on the socket. This status value is incremented each time a connection timeout or a socket close timeout (2 seconds) occurs. |
| Server[3].CfgErrWord | 170 | This bit mapped word defines the configuration errors for the server. |
| Server[4].Enabled | 171 | This flag defines if the server is utilized. A value of 0 indicates the server is not used. Any other value indicates the server is used. |
| Server[4].State | 172 | This flag defines the current state of the server. |
| Server[4].IP | 173 | This double-word value contains the IP address of the client connected to the server. |
| Server[4].Port | 175 | This word value contains the port address for the client connected to the server. |
| Server[4].Open | 176 | Indicates that the module has opened the server port and is waiting for a connection to a client. |
| Server[4].Established | 177 | This status value contains the total number of times a connection was established on the socket. |
| Server[4].Closed | 178 | This status value contains the total number of times a close operation was performed on the socket. |
| Server[4].RxCount | 179 | This status value contains the total number of messages received by the server. |
| Server[4].RxOverflow | 180 | This status value contains the total number of messages received that exceed the user specified buffer size for the server. |
| Server[4].TxCount | 181 | This status value contains the total number of messages transmitted by the server. |

| Object in GSCInStat | Block Offset Start | Description |
|---|---|---|
| Server[4].TxOverflow | 182 | This status value contains the total number of transmit messages that exceeded the user specified maximum buffer size for the server. |
| Server[4].Timeout | 183 | This status value contains the total number of times a connection timeout occurred on the socket. This status value is incremented each time a connection timeout or a socket close timeout (2 seconds) occurs. |
| Server[4].CfgErrWord | 184 | This bit mapped word defines the configuration errors for the server. |
| Client[0].Connected | 185 | This flag defines if the client is utilized and connected to a server. A value of 0 indicates the client is not connected and can be utilized for a connection. Any other value indicates the client is connected and being used. |
| Client[0].State | 186 | This flag defines the current state of the client. |
| Client[0].IP | 187 | This double-word value contains the IP address of the server connected to the client. |
| Client[0].Port | 189 | This word value contains the port address for the server connected to the client. |
| Client[0].RxCount | 190 | This status value contains the total number of messages received by the client. |
| Client[0].RxOverflow | 191 | This status value contains the total number of messages received that exceed the user specified buffer size for the client. |
| Client[0].TxCount | 192 | This status value contains the total number of messages transmitted by the client. |
| Client[0].TxOverflow | 193 | This status value contains the total number of transmit messages that exceeded the user specified maximum buffer size for the client. |
| Client[0].spare | 194 | Reserved for future use |
| Client[1].Connected | 195 | This flag defines if the client is utilized and connected to a server. A value of 0 indicates the client is not connected and can be utilized for a connection. Any other value indicates the client is connected and being used. |
| Client[1].State | 196 | This flag defines the current state of the client. |
| Client[1].IP | 197 | This double-word value contains the IP address of the server connected to the client. |
| Client[1].Port | 199 | This word value contains the port address for the server connected to the client. |
| Client[1].RxCount | 200 | This status value contains the total number of messages received by the client. |
| Client[1].RxOverflow | 201 | This status value contains the total number of messages received that exceed the user specified buffer size for the client. |
| Client[1].TxCount | 202 | This status value contains the total number of messages transmitted by the client. |
| Client[1].TxOverflow | 203 | This status value contains the total number of transmit messages that exceeded the user specified maximum buffer size for the client. |
| Client[1].spare | 204 | Reserved for future use |

| Object in GSCInStat | Block Offset Start | Description |
|---|---|---|
| Client[2].Connected | 205 | This flag defines if the client is utilized and connected to a server. A value of 0 indicates the client is not connected and can be utilized for a connection. Any other value indicates the client is connected and being used. |
| Client[2].State | 206 | This flag defines the current state of the client. |
| Client[2].IP | 207 | This double-word value contains the IP address of the server connected to the client. |
| Client[2].Port | 209 | This word value contains the port address for the server connected to the client. |
| Client[2].RxCount | 210 | This status value contains the total number of messages received by the client. |
| Client[2].RxOverflow | 211 | This status value contains the total number of messages received that exceed the user specified buffer size for the client. |
| Client[2].TxCount | 212 | This status value contains the total number of messages transmitted by the client. |
| Client[2].TxOverflow | 213 | This status value contains the total number of transmit messages that exceeded the user specified maximum buffer size for the client. |
| Client[2].spare | 214 | Reserved for future use |
| Client[3].Connected | 215 | This flag defines if the client is utilized and connected to a server. A value of 0 indicates the client is not connected and can be utilized for a connection. Any other value indicates the client is connected and being used. |
| Client[3].State | 216 | This flag defines the current state of the client. |
| Client[3].IP | 217 | This double-word value contains the IP address of the server connected to the client. |
| Client[3].Port | 219 | This word value contains the port address for the server connected to the client. |
| Client[3].RxCount | 220 | This status value contains the total number of messages received by the client. |
| Client[3].RxOverflow | 221 | This status value contains the total number of messages received that exceed the user specified buffer size for the client. |
| Client[3].TxCount | 222 | This status value contains the total number of messages transmitted by the client. |
| Client[3].TxOverflow | 223 | This status value contains the total number of transmit messages that exceeded the user specified maximum buffer size for the client. |
| Client[3].spare | 224 | Reserved for future use |
| Client[4].Connected | 225 | This flag defines if the client is utilized and connected to a server. A value of 0 indicates the client is not connected and can be utilized for a connection. Any other value indicates the client is connected and being used. |
| Client[4].State | 226 | This flag defines the current state of the client. |
| Client[4].IP | 227 | This double-word value contains the IP address of the server connected to the client. |
| Client[4].Port | 229 | This word value contains the port address for the server connected to the client. |

| Object in GSClnStat | Block Offset Start | Description |
|---|---|---|
| Client[4].RxCount | 230 | This status value contains the total number of messages received by the client. |
| Client[4].RxOverflow | 231 | This status value contains the total number of messages received that exceed the user specified buffer size for the client. |
| Client[4].TxCount | 232 | This status value contains the total number of messages transmitted by the client. |
| Client[4].TxOverflow | 233 | This status value contains the total number of transmit messages that exceeded the user specified maximum buffer size for the client. |
| Client[4].spare | 234 | Reserved for future use |

The following table describes the format format of the server configuration error word.

### *6.3.1 Configuration Error Word Bits*

| Bit Position | | Description |
|---|---|---|
| Bit 0 | 0x01 | A value other than Y or N was entered for the server enabled parameter. |
| Bit 1 | 0x02 | Not utilized |
| Bit 2 | 0x04 | A value less than 5000 is entered for the Connection Timeout parameter other than 0. |
| Bit 3 | 0x08 | An invalid value is entered for the Connection Close Type parameter. The program only accepts value of 0 to 2. |

## 6.4 Configuration Data

This section contains a listing of the parameters and their definitions for the MVI56E-GEC module configuration.

| [Section]/Item | Range | Description |
|---|---|---|
| [MODULE] |  | Module parameter definitions |
| Module Name: | 0 to 80 characters | This parameter assigns a name to the module that can be viewed using the configuration/debug port. Use this parameter to identify the module and the configuration file. |
| Password: | Up to 20 chars | Not supported with the latest release |

| [Section]/Item | Range | Description |
|---|---|---|
| [Server 0] |  | Server definitions |
| Enabled: | Y or N | This parameter determines if the server will be utilized by the module. If a value of "Y" is entered, the server will be used. Any other value will disable the server. |
| Service Port Number: | 1 to 65535 | This parameter sets the TCP/IP service port for this server. Each server can have its own unique service port or can share the same number with other servers. |
| Connection Timeout: | 0 or 5000 to 65535 | This parameter specifies the number of milliseconds the server will permit the server to be inactive after a connection is made before closing the socket. This timeout period is reset on each read or write packet. If the parameter is set to 0, the connection will not timeout. |
| Connection Close Type: | 0, 1 or 2 | This coded parameter defines the personality of the server after a connection is made. If the parameter is set to 0, the socket will only be closed when a request from the client is received or the connection timeout is exceeded. If a value of 1 is selected, the server will close the socket after it transmits a single message. If a value of 2 is selected, the server will close the socket after it receives a message. |
| Swap Rx Data Bytes: | Y or N | This parameter is determines if the data received by the server will have the byte order of the data swapped. If the parameter is set to N, no byte swapping will occur. If the parameter is set to Y, the odd byte will be swapped with the even byte in each word of data received. |
| Swap Tx Data Bytes: | Y or N | This parameter is determines if the data to be transmitted by the server will have the byte order of the data swapped. If the parameter is set to N, no byte swapping will occur. If the parameter is set to Y, the odd byte will be swapped with the even byte in each word of data received. |

**Note:** Add other [Server n] sections for each server to be utilized.

| [Section]/Item | Range | Description |
|---|---|---|
| [Server n] | [redacted] | Server definitions |
| Enabled: | Y or N | This parameter determines if the server will be utilized by the module. If a value of "Y" is entered, the server will be used. Any other value will disable the server. |
| Service Port Number: | 1 to 65535 | This parameter sets the TCP/IP service port for this server. Each server can have its own unique service port or can share the same number with other servers. |
| Connection Timeout: | 0 or 5000 to 65535 | This parameter specifies the number of milliseconds the server will permit the server to be inactive after a connection is made before closing the socket. This timeout period is reset on each read or write packet. If the parameter is set to 0, the connection will not timeout. |
| Connection Close Type: | 0, 1 or 2 | This coded parameter defines the personality of the server after a connection is made. If the parameter is set to 0, the socket will only be closed when a request from the client is received or the connection timeout is exceeded. If a value of 1 is selected, the server will close the socket after it transmits a single message. If a value of 2 is selected, the server will close the socket after it receives a message. |
| Swap Rx Data Bytes: | Y or N | This parameter is determines if the data received by the server will have the byte order of the data swapped. If the parameter is set to N, no byte swapping will occur. If the parameter is set to Y, the odd byte will be swapped with the even byte in each word of data received. |
| Swap Tx Data Bytes: | Y or N | This parameter is determines if the data to be transmitted by the server will have the byte order of the data swapped. If the parameter is set to N, no byte swapping will occur. If the parameter is set to Y, the odd byte will be swapped with the even byte in each word of data received. |

# 7 Support, Service & Warranty

## 7.1 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<br>ps.prosofttechnology@belden.com<br>Languages spoken: English, Spanish<br><br>REGIONAL TECH SUPPORT<br>ps.support@belden.com | Phone: +33.(0)5.34.36.87.20<br>ps.europe@belden.com<br>Languages spoken: English, French, Hindi, Italian<br><br>REGIONAL TECH SUPPORT<br>ps.support.emea@belden.com |
| **Latin America Regional Office** | **Asia Pacific Regional Office** |
| Phone: +52.222.264.1814<br>ps.latinam@belden.com<br>Languages spoken: English, Spanish, Portuguese<br><br>REGIONAL TECH SUPPORT<br>ps.support.la@belden.com | Phone: +60.3.2247.1898<br>ps.asiapc@belden.com<br>Languages spoken: Bahasa, Chinese, English, Hindi, Japanese, Korean, Malay<br><br>REGIONAL TECH SUPPORT<br>ps.support.ap@belden.com |

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

## 7.2 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/RMA