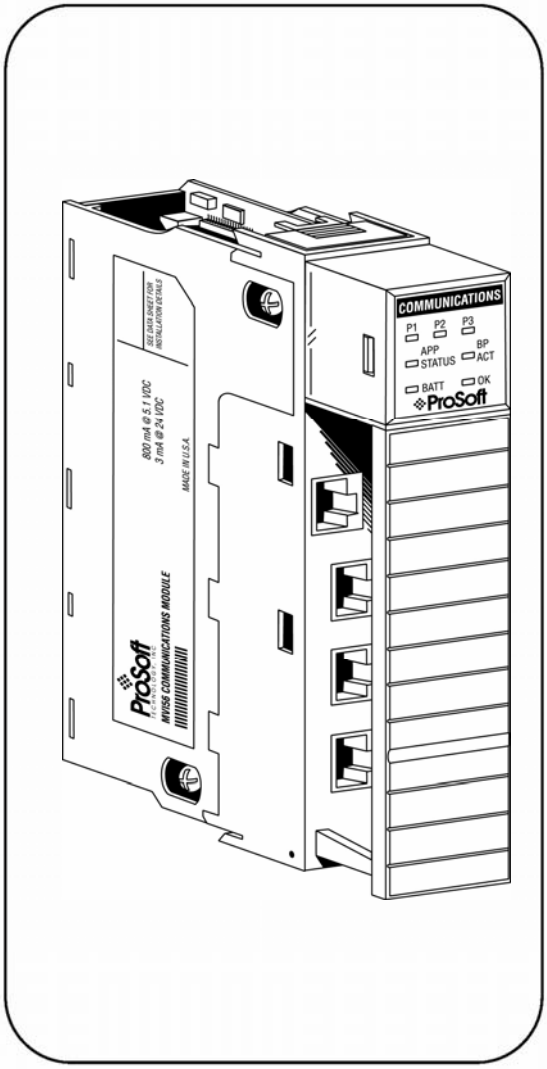
inRAx



MVI56-DH485

ControlLogix Platform

DH485 Interface Module

User Manual

November 12, 2004

ProSoft TECHNOLOGY

Please Read This Notice

Successful application of this module requires a reasonable working knowledge of the ControlLogix Platform DH485 Interface Module hardware and the application in which the combination is to be used. For this reason, it is important that those responsible for implementation satisfy themselves that the combination will meet the needs of the application without exposing personnel or equipment to unsafe or inappropriate working conditions.

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

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

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

ProSoft Technology, Inc.

1675 Chester Avenue, 2nd Floor

Bakersfield, CA 93301

(661) 716-5100

(661) 716-5101 (Fax)

www.prosoft-technology.com

MVI56-DH485 User Manual

November 12, 2004

Table of Contents

1 Introduction

The MVI56-DH485 (“DH485 Communication Module”) product allows Allen-Bradley ControlLogix I/O compatible processors to easily interface with other DH485 protocol compatible devices. Compatible devices include not only Allen-Bradley PLC’s and SLC's (which support the DH485 protocol) but also a wide assortment of end devices.

The MVI56-DH485 module acts as a gateway between the DH485 network and the Allen-Bradley backplane. The data transfer from the ControlLogix processor is asynchronous from the actions on the DH485 network. A 4000-word register space in the module is used to exchange data between the processor and the DH485 network.

The module can send and receive commands from other nodes on the DH485 network. Each port works as an independent DH485 node.

1.1 General Concepts

The following discussion covers several concepts that are key to understanding the operation of the MVI56-DH485 module.

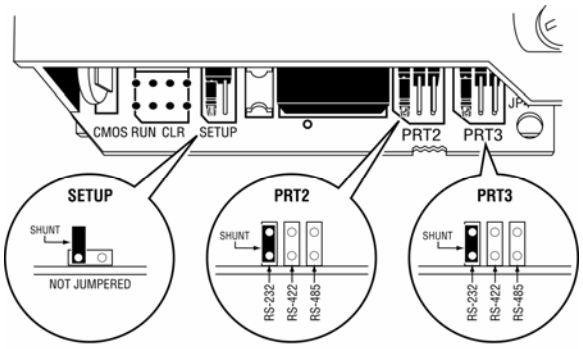
On power up the module begins performing the following logical functions:

1. Initialize hardware components
    a. Initialize ControlLogix backplane driver
    b. Test and Clear all RAM
    c. Initialize the serial communication ports
2. Reads configuration information from the compact flash disk
3. Initialize Module Register space
4. Enable Slave Driver
5. Enable Master Driver

Once the module has received the module configuration from the compact flash disk, the module will begin communicating with other nodes on the network, depending on the configuration.

1.2 Verifying the Jumper Settings

There are three jumpers located at the bottom of the module.



Verify that the jumpers on the module match the settings shown in the above illustration. The Setup Jumper should only be jumped when programming the module or when instructed to do so by ProSoft Technology, Inc.

The PRT2 and PRT3 jumpers are used to configure ports 2 and 3 for RS-232, RS-422, or RS-485 communications. The default jumper setting for both application ports is RS-232.

2 Understanding the Architecture

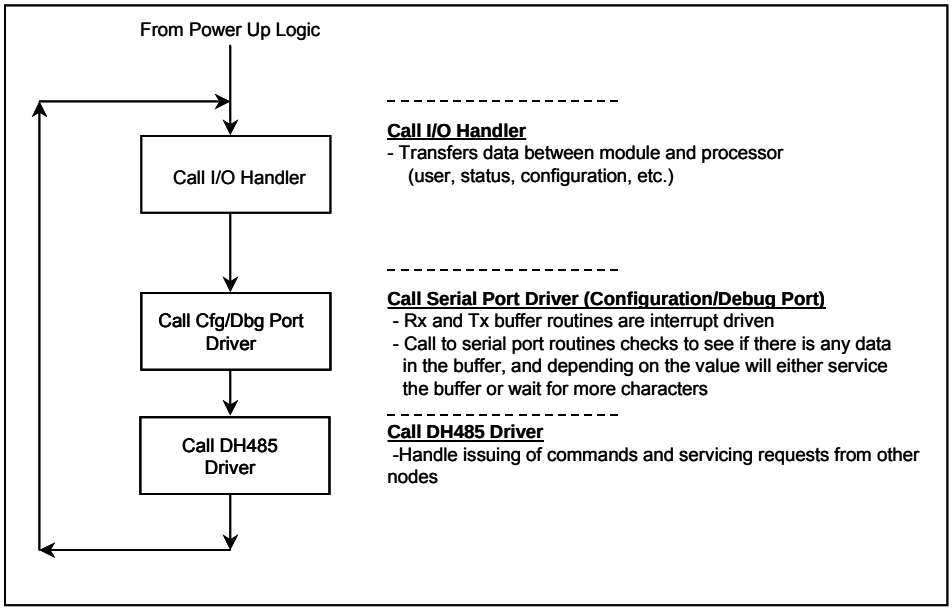
This section gives the reader a functional overview of the MVI56-DH485 module.

A thorough understanding of the information contained in this document is required for successful implementation of the module in a user application. If you already understand the content of this section, refer to the **Module Configuration** section to get the module up and running. If you are not familiar with the data transfer operation, read this section before setting up the module.

The module is designed to insulate the user from requiring a comprehensive knowledge of the DH485 protocol specification. Some knowledge is required when constructing commands and understanding the data types used in the protocol. This information is supplied in this manual where required.

2.1 Main Logic Loop

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



2.2 ControlLogix Processor Not in Run

Anytime the module detects that the processor has gone out of the Run mode (i.e., Fault or PGM), the DH485 ports can be shut down as prescribed in the user configuration. When the processor is returned to a running state, the module will resume communications on the network.

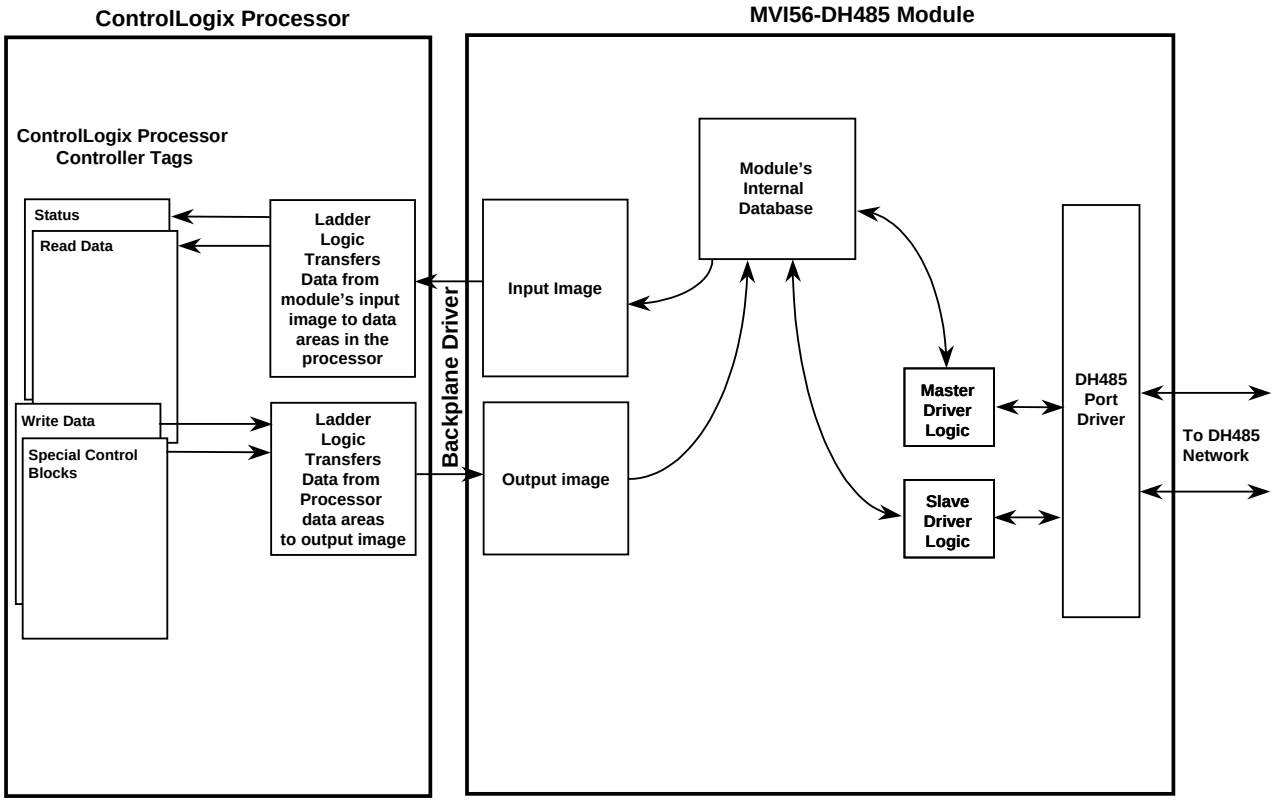
2.3 Backplane Data Transfer

The MVI56-DH485 module is unique in the way that the ControlLogix backplane is utilized. Data is paged between the module and the ControlLogix processor across the backplane using the module's input and output images. The update frequency of the images is determined by the scheduled scan rate defined by the user for the module and the communication load on the module for the input and output images. Typical updates are in the range of 1 to 10 milliseconds.

This bi-directional transference of data is accomplished by the module filling in data in the module's input image to send to the processor. Data in the input image is placed in the controller tags in the processor by the ladder logic. The input image for the module is set to 250 words. This large data area permits fast throughput of data between the module and the processor.

The processor inserts data in the module's output image to be transferred to the module. The module's program extracts the data and places it in the module's internal database. The output image for the module is set to 248 words. This large data area permits fast throughput of data from the processor to the module.

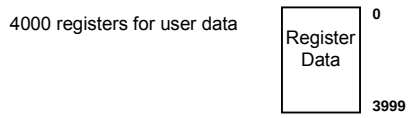
The following diagram displays the data transfer method used to move data between the ControlLogix processor, the MVI56-DH485 module, and the DH485 network.



As shown in the previous diagram, all data transferred between the module and the processor over the backplane is through the input and output images. Ladder logic must be written in the ControlLogix processor to interface the input and output image

data with data defined in the controller tags. All data used by the module is stored in its internal database. The following diagram shows the layout of the database.

Module's Internal Database Structure



Data contained in this database is paged through the input and output images by coordination of the ControlLogix ladder logic and the MVI56-DH485 module's program. Up to 250 words of data can be transferred from the module to the processor at once. Up to 248 words of data can be transferred from the processor to the module. The read and write block identification codes in each data block determine the function to be performed or the content of the data block. The block identification codes used by the module are listed in the following table:

Block Range	Descriptions
0 or -1	Status Block
1 to 20	Read or write data
1000 to 1019	Read data initialization blocks
3000	Port 0 command disable block
3001	Port 0 command enable block
3002	Port 0 command conditional block
3100	Port 1 command disable block
3101	Port 1 command enable block
3102	Port 1 command conditional block
9998	Warm-boot control block
9999	Cold-boot control block

Each image has a defined structure depending on the data content and the function of the data transfer as defined in the following sections.

2.3.1 Normal Data Transfer

Normal data transfer includes the transferring of data received by, or to be transmitted to, the DH485 drivers and the status data. This data is transferred through read (input image) and write (output image) blocks. The section entitled ***Module Configuration*** provides a description of the data objects used with the blocks and the ladder logic required. The following sections discuss the structure and function of each block.

2.3.1.1 Read Block

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

Offset	Description	Length
0	Reserved	1
1	Write Block ID	1
2-201	Read Data	200
202	Program Scan Counter	1
203-204	Product Code	2
205-206	Product Version	2
207-208	Operating System	2
209-210	Run Number	2
211-216	Data Transfer Status	6
217-248	Spare	32
249	Read Block ID	1

The data transfer status has the following structure (words 211 to 216):

Word	Description
211	Block Reads
212	Block Write
213	Block Parse
214	Block Events
215	Block Commands
216	Block Error

The Block Identification Code (word 249) is used to signal to the ControlLogix processor that a new block is ready for processing and informs the processor of the contents of the block. If the value of the code is set to 1, the block contains the first 200 words of data contained in the database of the module.

The block also contains the block identification code the module expects to receive from the processor (word 1 in the block). Under normal data transfer conditions, the ladder logic should use the code to build the appropriate block for the module in the output image.

Normal read data blocks also contain status information (words 202 to 216) that can be used to determine the health of the module. The example ladder logic displays how this information is stored in controller tags in the processor. This information is also passed to the processor for block identification codes 0 and –1 along with the status data for each port. The format of these blocks is as follows:

Offset	Description	Length
0	Reserved	1
1	Write Block ID	1
2-51	Port 0 Status	50
52-101	Port 1 Status	50
102-201	Reserved	100
202	Program Scan Counter	1
203-204	Product Code	2
205-206	Product Version	2
207-208	Operating System	2
209-210	Run Number	2
211-216	Data Transfer Status	6
217-248	Reserved	32
249	Read Block ID (0 or -1)	1

Refer to Appendix A for a detailed listing of the content of this data block.

2.3.1.2 Write Block

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

Offset	Description	Length
0	Write Block ID	1
1-200	Write Data	200
201-247	Spare	47

The Write Block ID is an index value used to determine the location in the module's database where the data will be placed. Each transfer can move up to 200 words.

2.3.2 Command Control Blocks

Command control blocks are special blocks used to control the module or perform special operations. The current version of the software supports four different command control blocks: initialize the read data, change command type control, warm boot, and cold boot:

2.3.2.1 Initialize The Read Data

Important: In order to use this feature, you must change the "Initialize Output Data" parameter to "Yes" in the configuration file.

Blocks 1000 to 1019 are used to initialize the read data when the module starts. This is the data passed from the module's internal database to the processor during normal operation. This optional function can be used to set the read data area to the values last read from the module before the first transfer of data from the module to the processor. The module will request this data from the processor using a block with the following format:

Offset	Description	Length
0	Reserved	1
1	1000 to 1019	1
2-248	Spare	247
249	1000 to 1019	1

The block number utilized represents the 200-word offset of data requested. Therefore, when the module request block 1000, the processor should return the first 200 words of read data. For block 1001, the processor should return the second 200 words of read data. The format of the response block built by the module is as follows:

Offset	Description	Length
0	1000 to 1019	1
1 to 200	Data to place in output area	200
201-247	Spare	47

The number of blocks requested by the module from the processor is dependent on the number of read registers configured for the module. If the module is configured for 600 registers, 3 blocks (1000 to 1002) will be requested. This data will be placed in the module's internal database starting at the read register configured for the module.

Use of this feature requires ladder logic to process the blocks requested. The example ladder logic displays a rung that will handle this optional feature.

2.3.2.2 Change Command Type Control Block

Blocks 3000, 3001, 3002, 3100, 3101 and 3102 are used to alter the command type for entries in the user command list. When commands are processed during the startup of the module, the command type is set to the value specified in the configuration file. These control blocks are used to alter the configured command type.

Each block contains a command index list. The values entered in this list correspond to the indices of the commands configured by the user. The first command in the user command list has an index of 0 and the second has in index of 1. Therefore, to alter the command type of commands 3 and 5, set the number of command indexes (word 1) to 2 and enter the values 3 and 5 in words 2 and 3. Each block can handle up to 60 commands in the list.

Block 3000 and 3100 requests are used to change the commands listed in the block to the disable type (type = 0). Block 3000 is used for Port 0 and 3100 is used for Port 1 commands. The format for the request block from the ladder logic is as follows:

Offset	Description	Length
0	3000 or 3100	1
1	Number of command indexes	1
2 to 62	Command index list	60
63-247	Spare	186

The response block from the module to the processor has the following format:

Offset	Description	Length
0	Reserved	1
1	Write Block ID	1
2	Number of commands processed	1
3-248	Spare	246
249	3000 or 3100	1

Block 3001 and 3101 requests are used to change the commands listed in the block to the enable type (type = 1). Block 3001 is used for Port 0 and 3101 is used for Port 1 commands. The format for the request block from the ladder logic is as follows:

Offset	Description	Length
0	3001 or 3101	1
1	Number of command indexes	1
2 to 62	Command index list	60
63-247	Spare	186

The response block from the module to the processor has the following format:

Offset	Description	Length
0	Reserved	1
1	Write Block ID	1
2	Number of commands processed	1
3-248	Spare	246
249	3001 or 3101	1

Block 3002 and 3102 requests are used to change the commands listed in the block to the conditional type (type = 2). Block 3002 is used for Port 0 and 3102 is used for Port 1 commands. This type code should only be utilized for write commands. The format for the request block from the ladder logic is as follows:

Offset	Description	Length
0	3002 or 3102	1
1	Number of command indexes	1
2 to 62	Command index list	60
63-247	Spare	186

The response block from the module to the processor has the following format:

Offset	Description	Length
0	Reserved	1
1	Write Block ID	1
2	Number of commands processed	1
3-248	Spare	246
249	3002 or 3102	1

2.3.2.3 Warm Boot Block

This block is sent from the ControlLogix processor to the module (output image) when the module is required to perform a warm boot (software reset operation). This block is commonly sent to the module any time the configuration file is modified. This will force the module to read the new configuration information and to restart. The structure of the control block is shown below:

Offset	Description	Length
0	9998	1
1-247	Spare	247

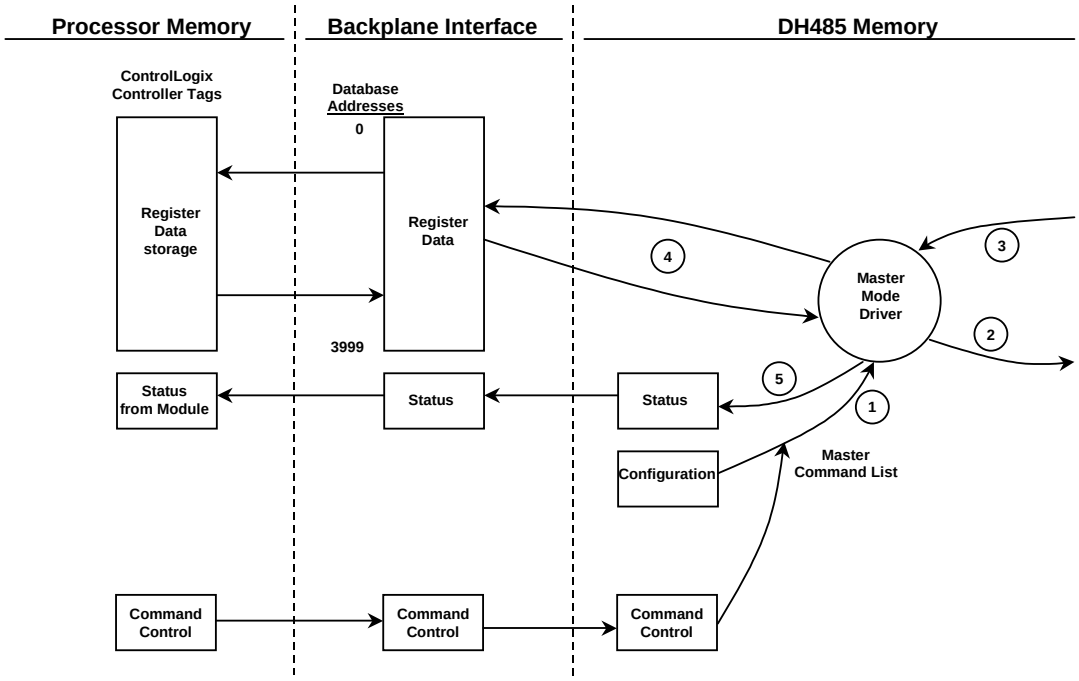
2.3.2.4 Cold Boot Block

Block 9999 is used to perform a cold-boot operation on the module. The format of the block constructed by the processor is as follows:

Offset	Description	Length
0	9999	1
1-247	Spare	247

2.4 Master Driver

Master Mode of operation of the MVI56-DH485 module is responsible for issuing read or write commands to other remote devices on the DH485 network. These commands are user configured in the module via the Master Command List received from the configuration file. Command status is returned to the processor for each individual command in the command list status data area. The location of this status block in the module's internal database is user defined. The following flow chart and associated table detail the flow of data into and out of the module.

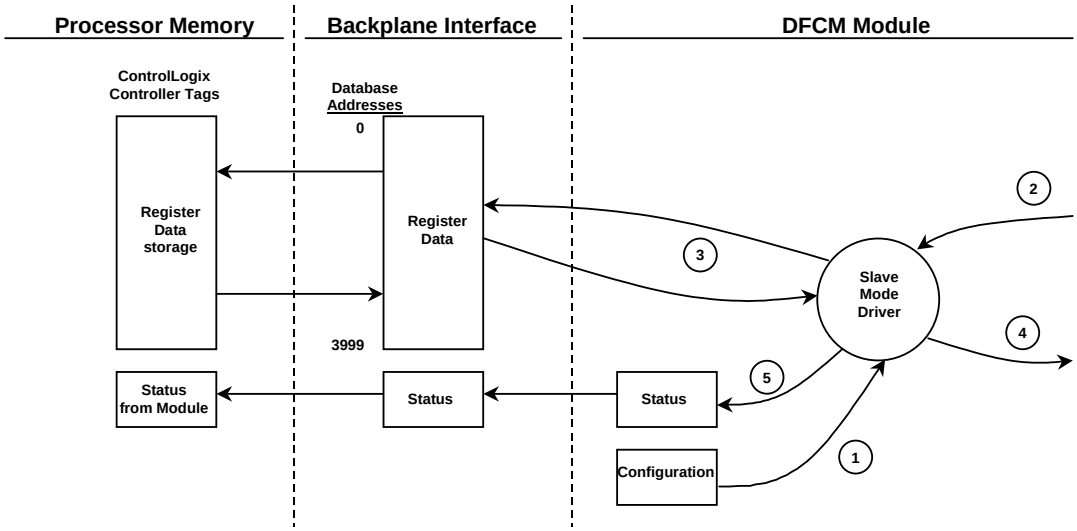


Step	Description
1	The Master driver obtains configuration data from the configuration file on the compact flash disk in the module. This information is used by the Master driver to determine the type of commands to be issued to the other nodes on the DH485 network.
2	Once configured, the Master driver begins transmitting read and/or write commands to the other nodes on the network. If writing data to another node, the data for the write command is obtained from the module's internal database to build the command.
3	Presuming successful processing by the node specified in the command, a response message is received into the Master driver for processing.
4	Data received from the node on the network is passed into the module's internal database, assuming a read command.
5	Status is returned to the ControlLogix processor for each command in the Master Command List.

Refer to **Modifying the Configuration Data** section for a complete description of the parameters required to define the virtual DH485 master port. Care must be taken in constructing each command in the list for predictable operation of the module. If two commands write to the same internal database address of the module, the results will not be as desired. All commands containing invalid data will be ignored by the module.

2.5 Slave Driver

The Slave Driver Mode allows the MVI56-DH485 module to respond to CIF and data read and write commands issued by a remote node on the DH485 network. The following flow chart and associated table detail the flow of data into and out of the module.



Step	Description
1	The DH485 slave port driver receives the configuration information from the configuration file on the compact flash disk. This information is used to configure the serial port and define the slave node characteristics. The module simulates N-files and CIF to permit remote access of the database.
2	A Host device, such as an Allen-Bradley SLC or an MMI package issues a read or write command to the module's node address. The port driver qualifies the message before accepting it into the module.
3	Once the module accepts the command, the data is immediately transferred to or from the internal database in the module or the CIF data area. If the command is a read command, the data is read out of the database and a response message is built. If the command is a write command, the data is written directly into the database and a response message is built.
4	Once the data processing has been completed in Step 3, the response is issued to the originating master node.
5	Counters are available in the Status Block that permit the ladder logic program to determine the level of activity of the Slave Driver.

Review the **Modifing The Configuration Data** section for a complete list of the parameters that must be defined for a slave port. The slave driver supports the following DH485 command set:

TYPE	ACCESS	Description
CIF	Read	485CIF, Peer-to-Peer, Read MSG requests
CIF	Write	485CIF, Peer-to-Peer, Write MSG requests
Data Table	Read	500CPU, Peer-to-Peer, Read MSG requests
Data Table	Write	500CPU, Peer-to-Peer, Write MSG requests

The Data table commands require the use of files. These files are emulated in the module. The user configuration of the module defines how these files are emulated in the module. Two file mappings are available. The first sets the first file number and file size for each file. Using this mapping, the files are set as contiguous files overlaying the database. For example, if the user sets the file size to 200 and sets the first file number as 7, the files will overlay the database as shown in the following diagram:

		Database Register
N7:0	---------->	0
N8:0	---------->	200
N9:0	---------->	400
N10:0	---------->	600
N11:0	---------->	800
N12:0	---------->	1000
N13:0	---------->	1200
N14:0	---------->	1400
N15:0	---------->	1600
N16:0	---------->	1800
N17:0	---------->	2000
N18:0	---------->	2200
N19:0	---------->	2400
N20:0	---------->	2600
N21:0	---------->	2800
N22:0	---------->	3000
N23:0	---------->	3200
N24:0	---------->	3400
N25:0	---------->	3600
N26:0	---------->	3800

In addition to the fixed file emulation, the module also supports user defined mapping of files. With this mapping, the user defines the database offset, file number, the starting element number and the length. This data is entered in the [DH485 Port x Maps] section of the configuration file for each port (x=0 for Port 0 and 1 for Port 1. The following is an example for Port 0:

```
[DH485 Port 0 Maps]
```

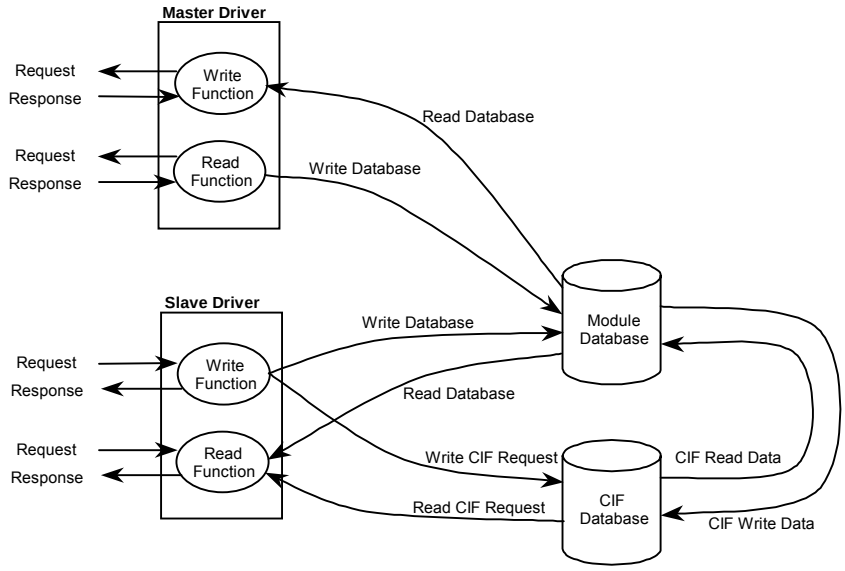
# DB_Address	File_Number	Element	Length
START			
0	10	0	100
100	12	100	100
END			

With the configuration displayed above, requests for file 10 elements 0 to 99 will be associated with the module's internal data registers 0 to 99 and file 12 elements 100 to 199 will be associated with the data base registers 100 to 199. The module supports up to 50 of these data mappings. When a request is made of the slave driver, the mapping defined in this override section will first be searched first. If the data area in the request is found in the list, that data mapping will be utilized. If the data area in the request is not found in the list, the fixed data area mapping will be used.

2.6 CIF Data

The module supports the common interface file (CIF) of the DH485 protocol. A separate data area can be defined for each application port. This data area is divided into read and write data and is mapped into the module's internal database. The module's application constantly transfers the read CIF data from the CIF database to the module database and the write CIF data from the module to the CIF database. The parameters used to define the CIF database are contained in the configuration file. This optional feature of the module should only be utilized if required by a remote master on the network. Otherwise, the data file functions (500CPU message) instructions should be used to access and control the module's data. The following diagram illustrates the relationship of the DH485 drivers, the CIF database and the module's database:

DH485 Network



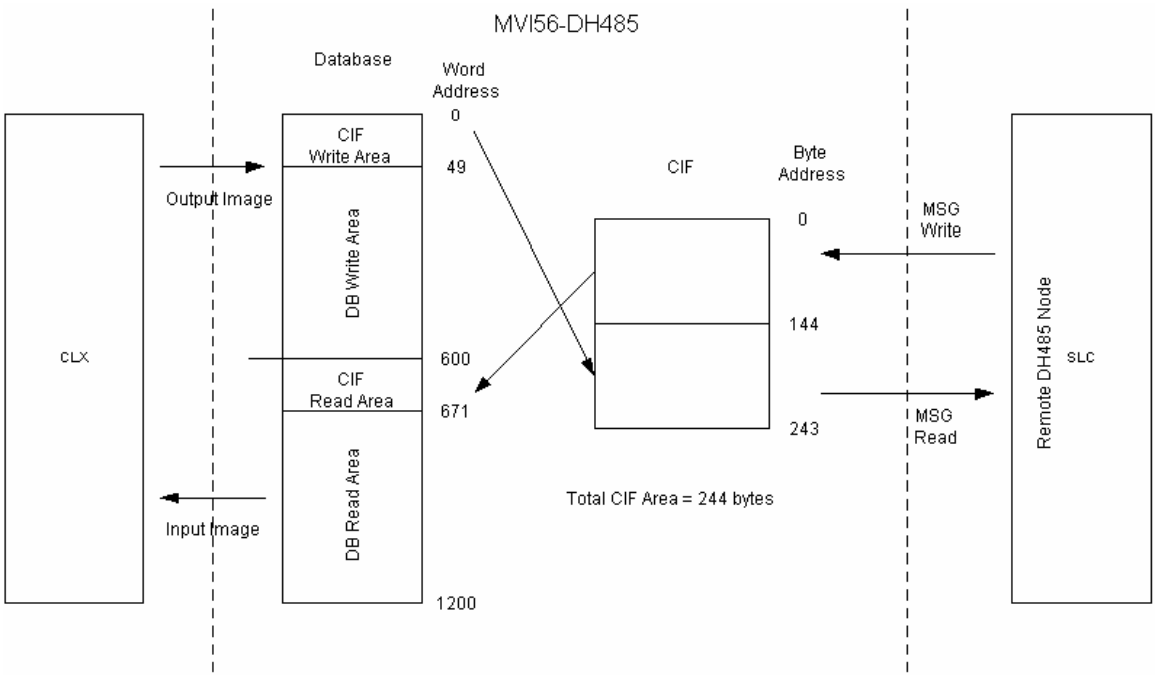
Important: The user can configure the size of the CIF Read Area using the CIF Read Count Parameter. The size of the CIF Write area is calculated using the following formula:

CIF Write Count = 244 – CIF Read Count.

In order to clarify the direction of data flow, the following shows an example:

Parameter	Value	Format
Read Register Start	600	Word
Read Register Count	600	Word
Write Register Start	0	Word
Write Register Count	600	Word
CIF Read DB Offset	1200	Byte
CIF Read Count	144	Byte
CIF Write DB Offset	0	Byte

This configuration will imply that the data will be transferred as shown in the following diagram:

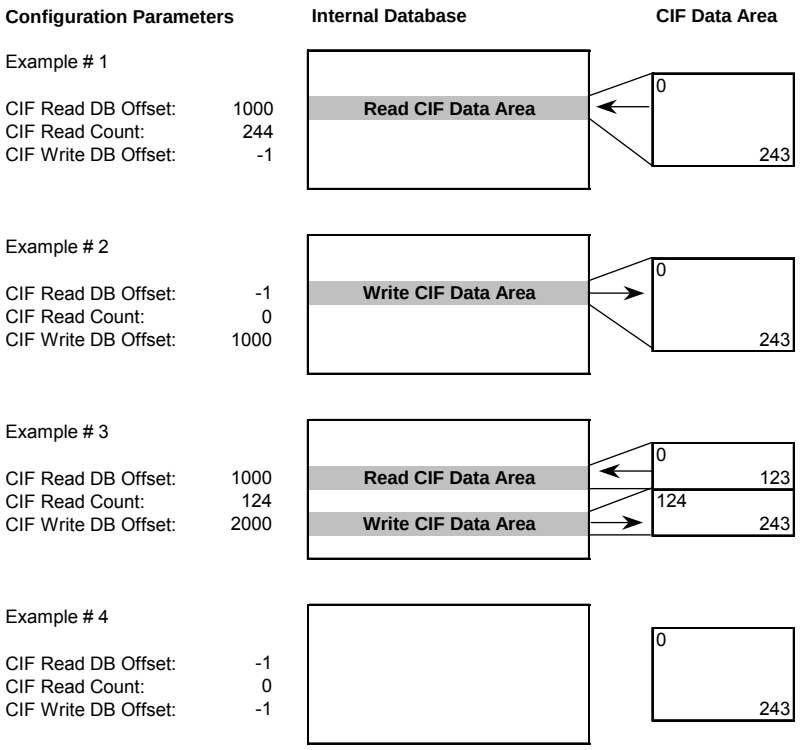


As shown in the diagram, the CIF area is split into the Read and Write areas. The CIF Read area always starts at an offset in the CIF file, except when the CIF Read Count is equal to 0. In this case, the CIF Read Area will not exist.

The diagram shows how the CIF area should interact with the backplane read and write area in order to achieve the correct flow of data.

If a Write MSG from a remote SLC processor is sent to byte addresses 0 and 1 in the module's CIF area, the data would be copied to word address 600 in the database and would then be read to the ControlLogix processor.

The following diagram displays four different uses and configurations of the CIF data area:



Example 1 only uses CIF read data and utilizes the maximum read data area. Example 2 only uses the CIF write data and utilizes the maximum write data area. Example 3 uses both CIF read and write data. 124 bytes of read data are used and 120 bytes of write data are used. Example 4 does not use the CIF data in the application.

2.7 SLC Interface

The DH485 driver is written for peer-to-peer operation on the DH485 network. Therefore, the driver can both execute commands (master role) and respond to commands (slave role). If the driver contains no commands, ladder logic is required in the other DH485 nodes in order to control or access the data in the module containing the driver. If the driver is configured with user defined commands, these commands will control or monitor data in other nodes on the network without the ladder logic requirement. The driver is developed to support both modes of operation simultaneously.

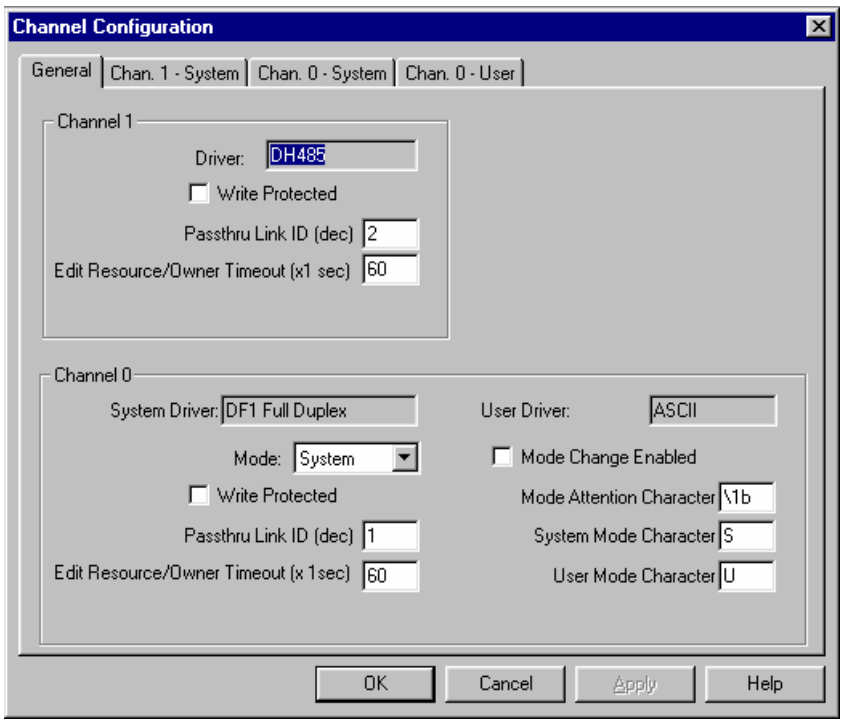
The section first describes the SLC configuration required for the DH485 protocol and the cabling required. Following this discussion, the section contains example ladder logic to interface with the module's database.

The ladder logic required in an SLC to interface with the DH485 Driver simply requires the use of the MSG instruction. The DH485 driver supports two data access options. You can interface to a module containing the DH485 driver using the CIF (Common Interface File) or normal SLC data file access. Use of the CIF access is limited to a data area of 244 bytes and is offered for backwards compatibility to older processors and should be used only when the data access method is not available.

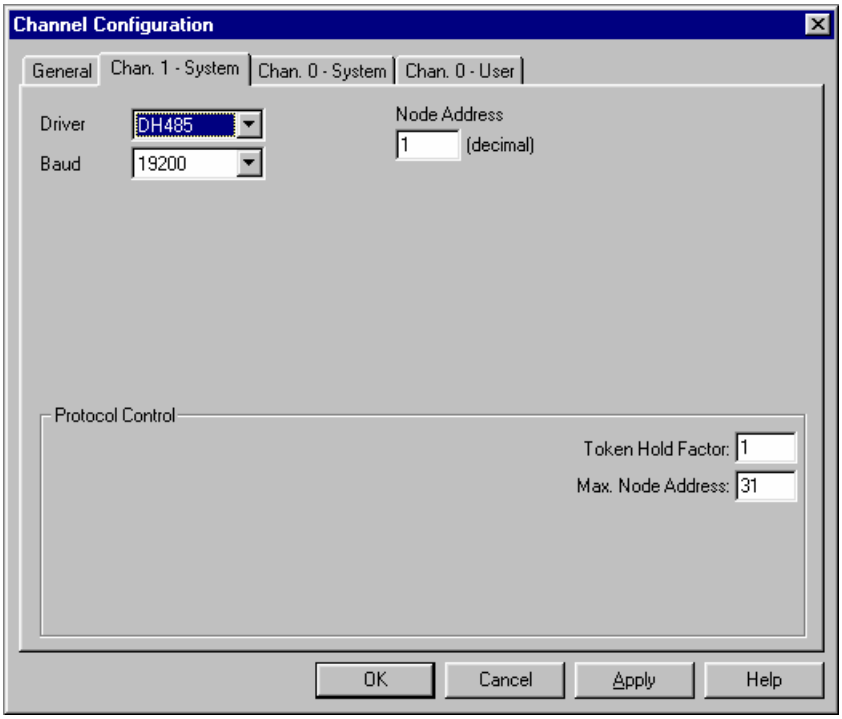
The preferred method of data access to the driver is through the use of the SLC data files. This access method permits full access to the data contained in the module's internal database.

2.7.1 SLC Communication Set Up

In order to use the DH485 port on the SLC, channel 0 or 1 must be configured as a DH485 port. Select Channel Configuration from the RSLogix 500 software to display the following dialog box.



Select one of the tabs in the dialog box to select the channel to configure with the DH485 protocol. The following figure displays an example configuration for channel 1.



Select the correct baud rate used on the DH485 network. Next select the node address for the SLC on the network. Each node on the network must have a unique node address. Next set the Token Hold Factor and Maximum Node Address for the SLC. Select the Apply command button to apply the settings configured. If you require help setting up the channel, select the Help command button on the form. After configuring the channel, download the new configuration to the SLC processor.

2.7.2 Ladder Logic

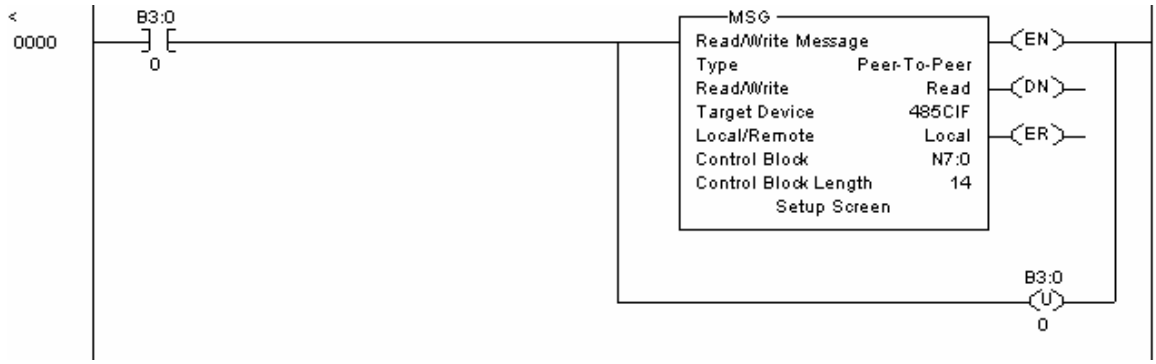
If the SLC will be used to monitor and control data in the DH485 driver device, ladder logic is required. No ladder logic is required if user commands are defined in the DH485 driver module to handle all data requirements. The examples below describe the two data access methods: CIF and Data File.

2.7.2.1 CIF Data Access

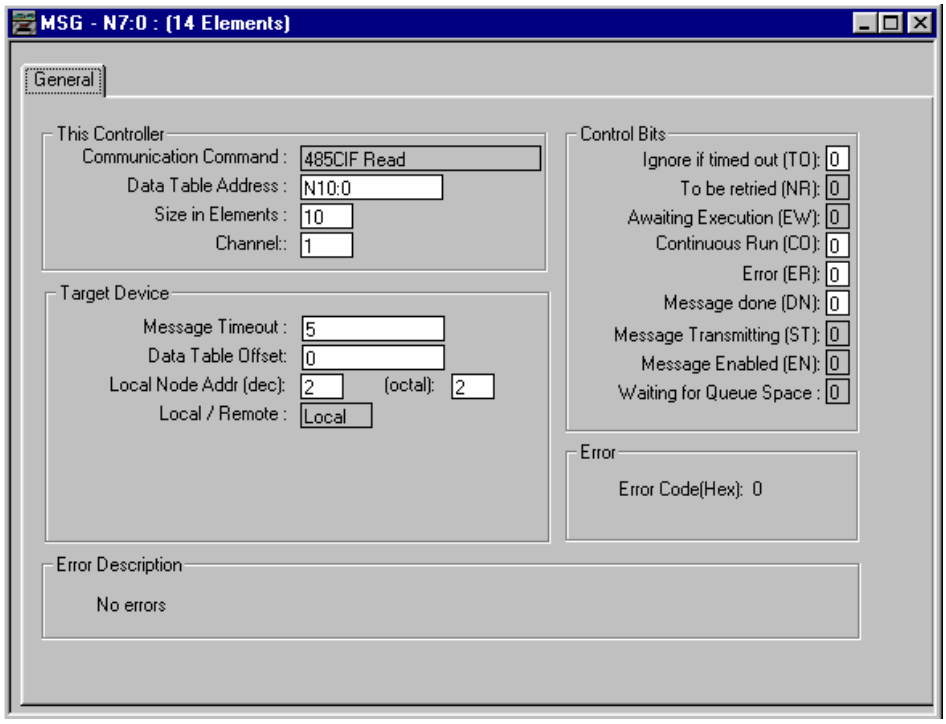
CIF data access permits up to 244 bytes of data in the DH485 driver module to be monitored and controlled for each port. Due to the extreme limit on the data size, this method should only be used if the data file access method is not offered on the monitoring and controlling device. The driver supports both read and write access to this data area.

2.7.2.1.1 Read Access

In order to monitor data contained in the DH485 driver module, a read command message must be executed from the ladder logic. To access the CIF data area in the module, select the 485CIF option for the Target Device in the MSG instruction. The following is an example rung that displays a CIF method MSG instruction:



Be certain to allocate a unique area in the SLC's data file area for the Control Block area used by the MSG instruction. Failure to provide this area will result in unpredictable behavior. Select the Setup Screen area in the MSG instruction to display the following dialog box:

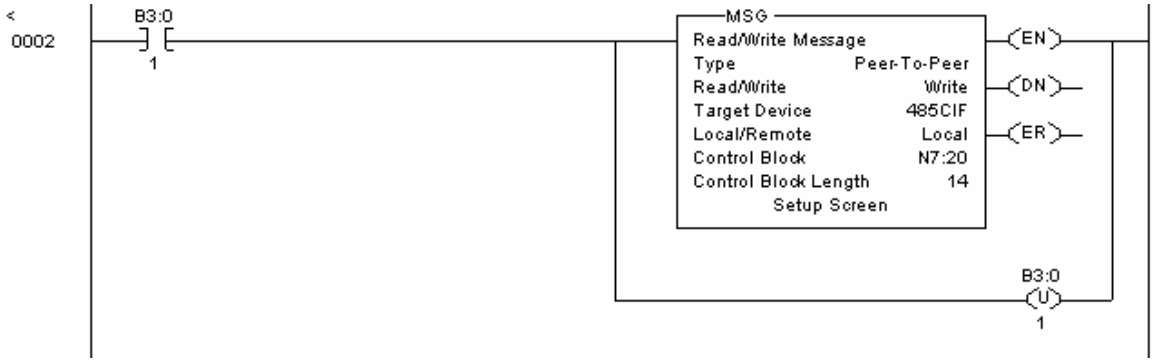


This example reads 10 words of data from the CIF data file in the module starting at byte offset 0 in node 2 and places the response data in the data file N10:0 to 9. Note that the Data Table Offset value is a byte offset and not a word offset. The size of data read is determined by the number of elements times the size of each element. For a N-type file, the size is 1 word (2 bytes). For a F-type file, the size is 2 words (4-bytes).

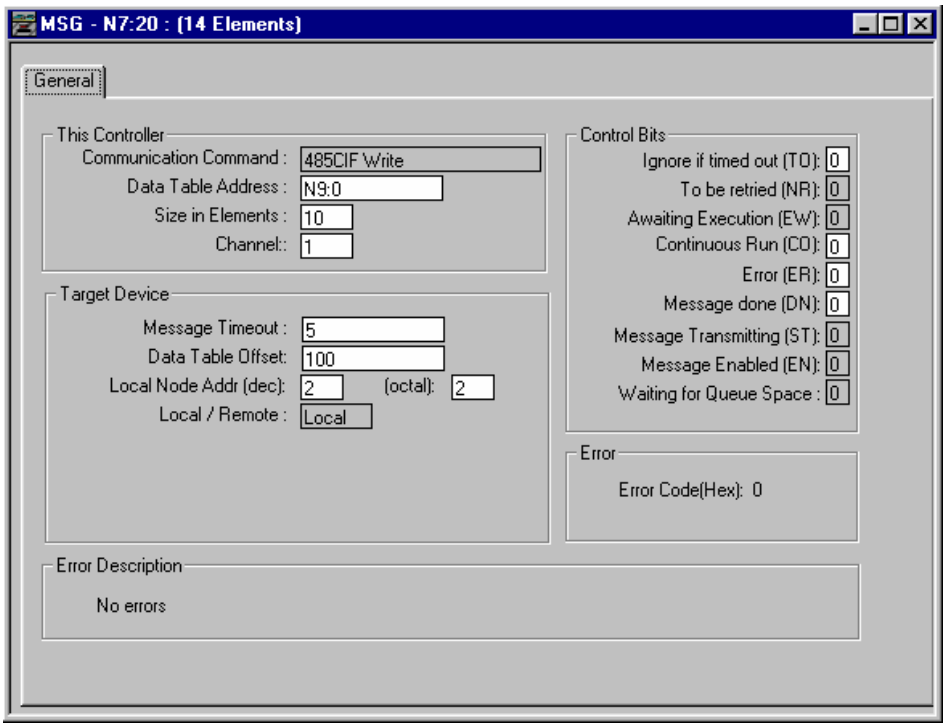
2.7.2.1.2 Write Access

In order to control data contained in the DH485 driver module, a write command message must be executed from the ladder logic. Two Hundred and Fourty-Four (244) bytes are available in the CIF Read Data Area (Byte Addresses 0 to 241). To

alter the CIF data area in the module, select the 485CIF option for the Target Device in the MSG instruction. The following is an example rung that displays a CIF method MSG instruction:



Be certain to allocate a unique area in the SLC's data file area for the Control Block area used by the MSG instruction. Failure to provide this area will result in unpredictable behavior. Select the Setup Screen area in the MSG instruction to display the following dialog box:



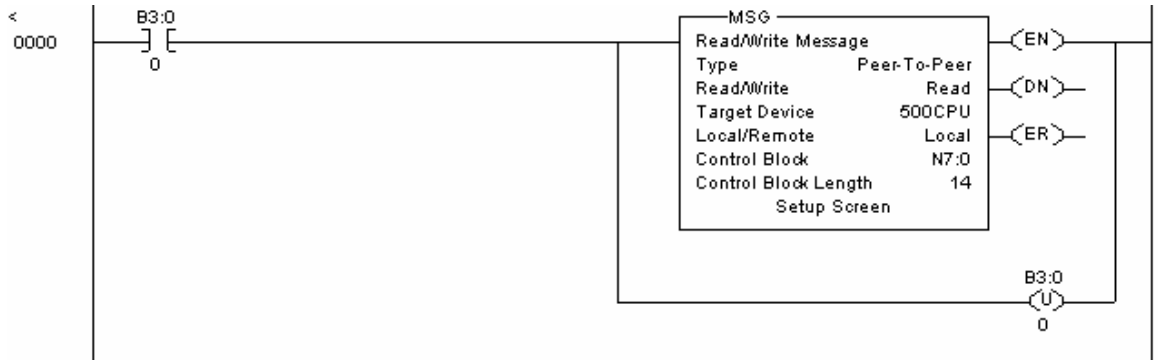
This example will place the 10 words from the SLC data file N9:0 to 9 in the CIF data area starting at byte offset 100 in node 2. Note that the Data Table Offset value is a byte offset and not a word offset. The size of data written is determined by the number of elements times the size of each element. For a N-type file, the size is 1 word (2 bytes). For a F-type file, the size is 2 words (4-bytes).

2.7.2.2 Data File Access

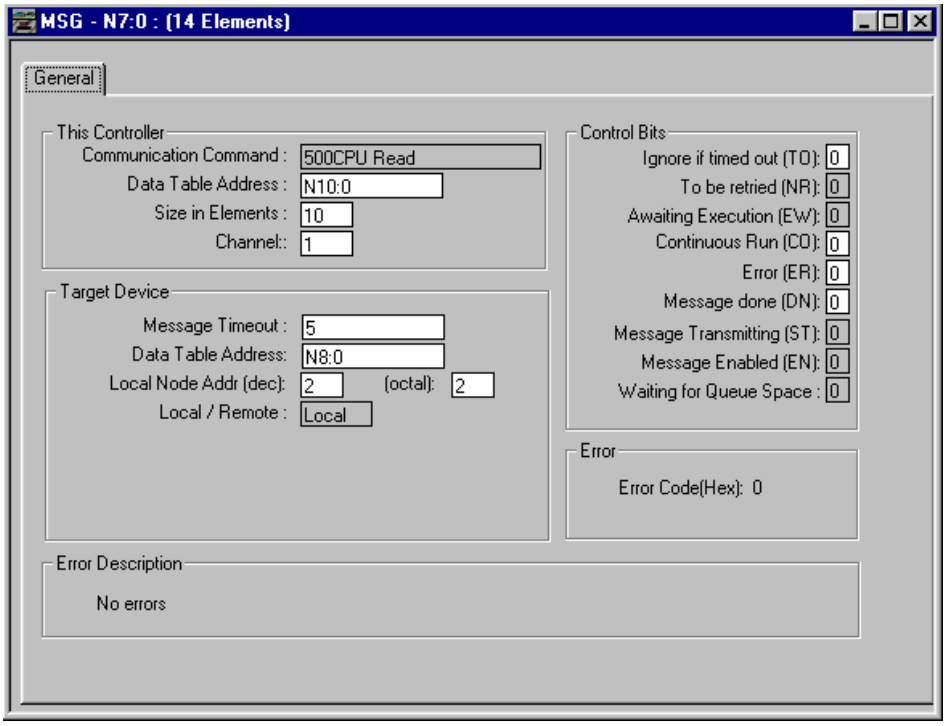
The data file access method permits full access to the full internal database of the DH485 driver module. This access method supports normal SLC file read and write access to the data in the module.

2.7.2.2.1 Read Access

In order to monitor data contained in the DH485 driver module, a read command message must be executed from the ladder logic. To read the data area in the module, select the 500CPU option for the Target Device in the MSG instruction. The following is an example rung that displays a data file access method MSG instruction:



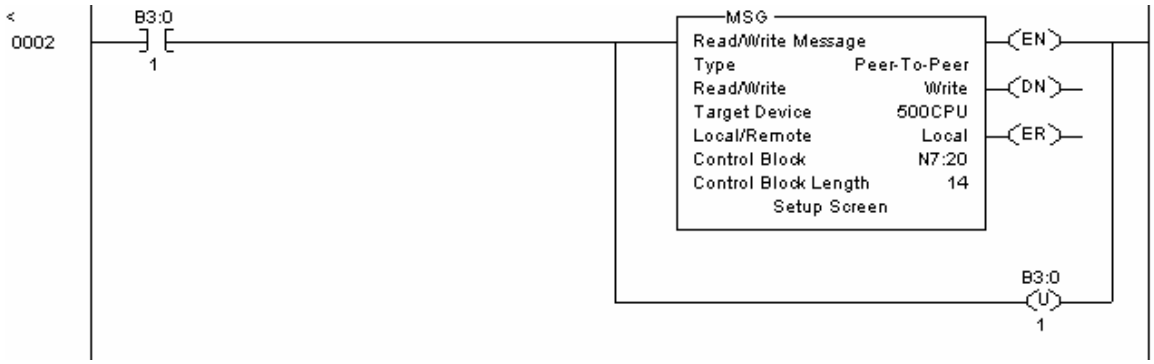
Be certain to allocate a unique area in the SLC's data file area for the Control Block area used by the MSG instruction. Failure to provide this area will result in unpredictable behavior. Select the Setup Screen area in the MSG instruction to display the following dialog box:



This example will read 10 words from the module and place the data in the SLC data file area N10:0 to 9. N-files are emulated on the module containing the DH485 driver. If the module is configured for the first N file of N7, a file size of 200 and a file offset value of 0, the module will respond with the words held in the module's internal database registers 200 to 209 for this command. Refer to the DH485 driver database documentation for a full explanation of the module's internal database.

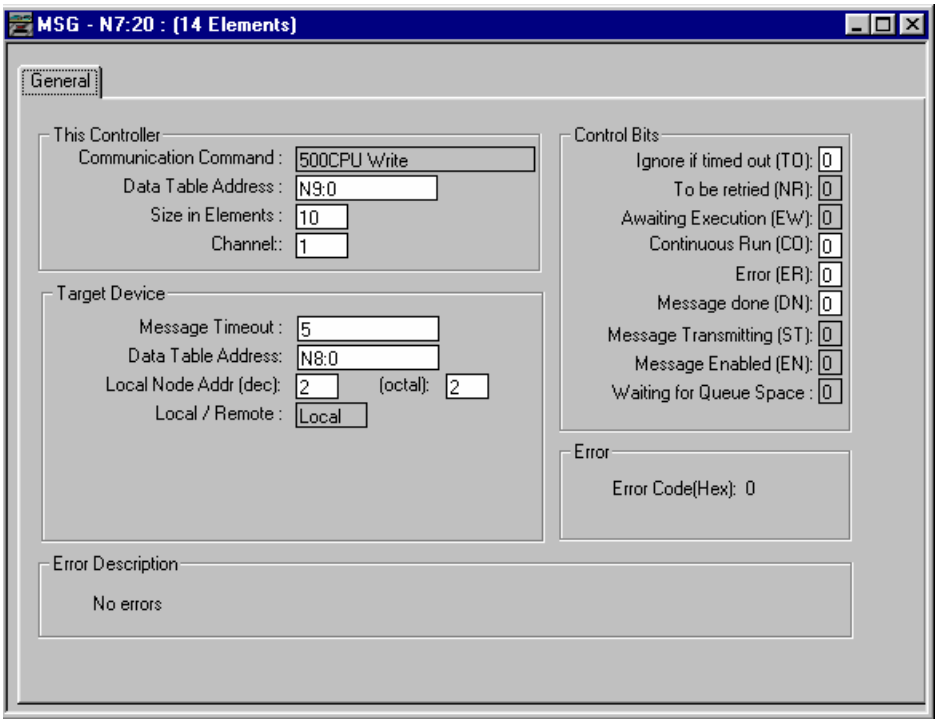
2.7.2.2.2 Write Access

In order to control data contained in the DH485 driver module, a write command message must be executed from the ladder logic. To alter the data area in the module, select the 500CPU option for the Target Device in the MSG instruction. The following is an example rung that displays a data file access method MSG instruction:



Be certain to allocate a unique area in the SLC's data file area for the Control Block area used by the MSG instruction. Failure to provide this area will result in

unpredictable behavior. Select the Setup Screen area in the MSG instruction to display the following dialog box:



This example will write 10 words from the SLC data file area N9:0 to 9 to the module. N-files are emulated on the module containing the DH485 driver. If the module is configured for the first N file of N7, a file size of 200 and a file offset value of 0, the module will place the 10 words of data received in the module's internal database registers 200 to 209. Refer to the DH485 driver database documentation for a full explanation of the module's internal database.

3 Module Configuration

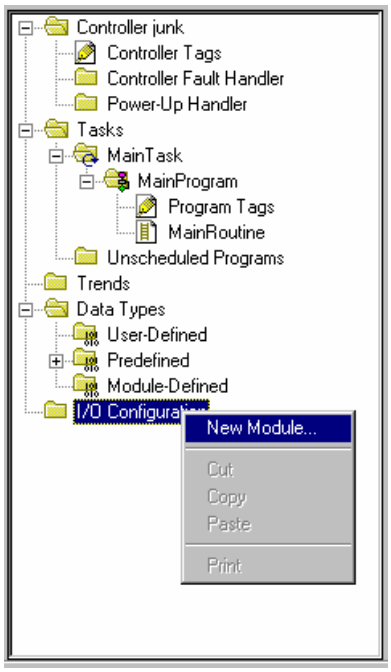
This section contains the setup procedure, data, and ladder logic for successful application of the MVI56-DH485 module. Each step in the setup procedure is defined in order to simplify the use of the module.

3.1 Setting Up the Module

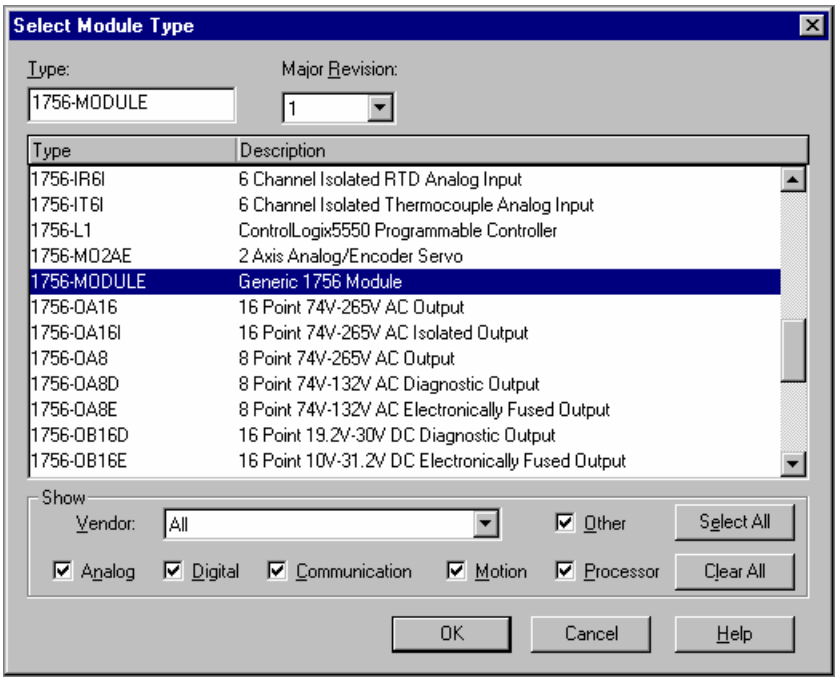
Set up of the MVI56-DH485 module only requires software configuration using the RSLogix 5000 program and the DH485.CFG file on the Compact Flash Disk in the module. The easiest method to implement the module is to start with the appropriate example provided with the module MVI56DH485.ACD. If you are installing this module in an existing application, you can simply copy the elements required from the example ladder logic to your application.

Note: This module can only be added to a project using the software in offline mode.

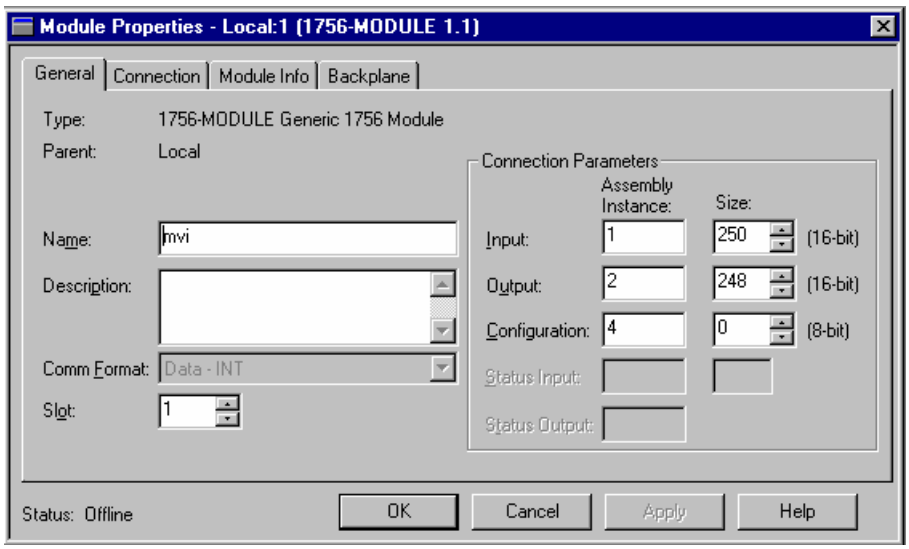
The first step in setting up the module is to define the module to the system. Right-click the mouse button on the I/O Configuration option in the Controller Organization window to display a pop-up menu. Select the **New Module** option from the I/O Configuration menu.



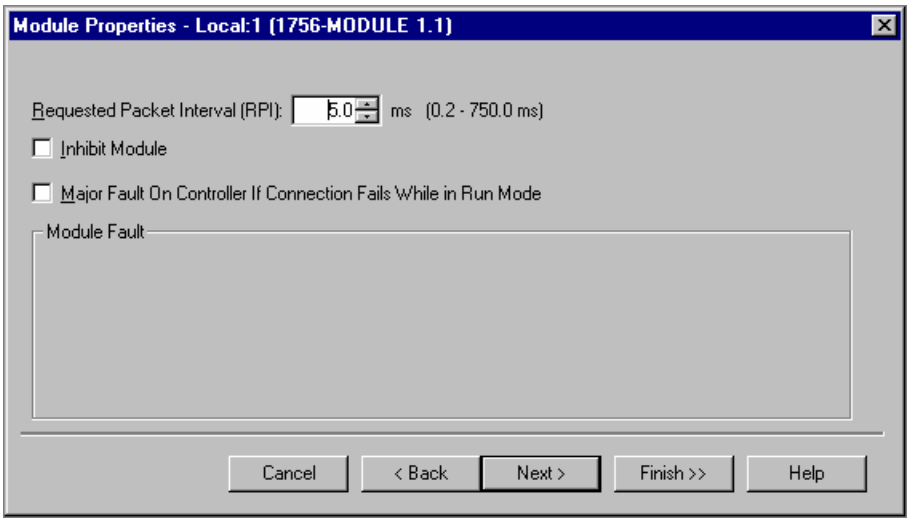
This causes the program to display the following dialog box.



Select the **1756-Module (Generic 1756 Module)** from the list and click the **OK** button. The following dialog box appears:

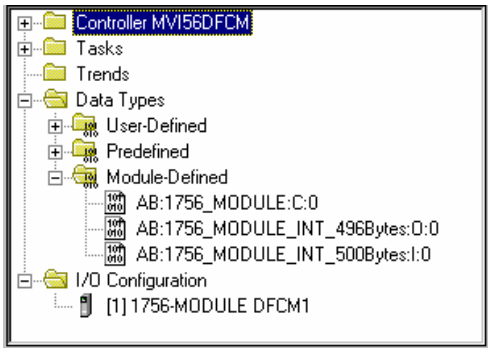


Fill in the dialog box as shown adjusting the Name, Description and Slot options for your application. Be certain to select the **Comm Format** as **Data - INT** in the dialog box. Failure to set the **Assembly Instance** and **Size** values correctly will result in a module that will not communicate over the backplane of the ControlLogix rack. Click the **OK** command button to display the next dialog box.

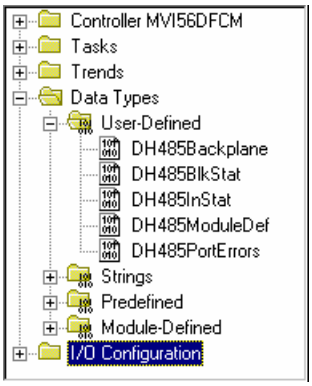


Select the Request Packet Interval value for scanning the I/O on the module. This value represents the minimum frequency that the module will handle scheduled events. This value should not be set to less than 5 milliseconds. Values between 5 and 10 milliseconds should work with most applications.

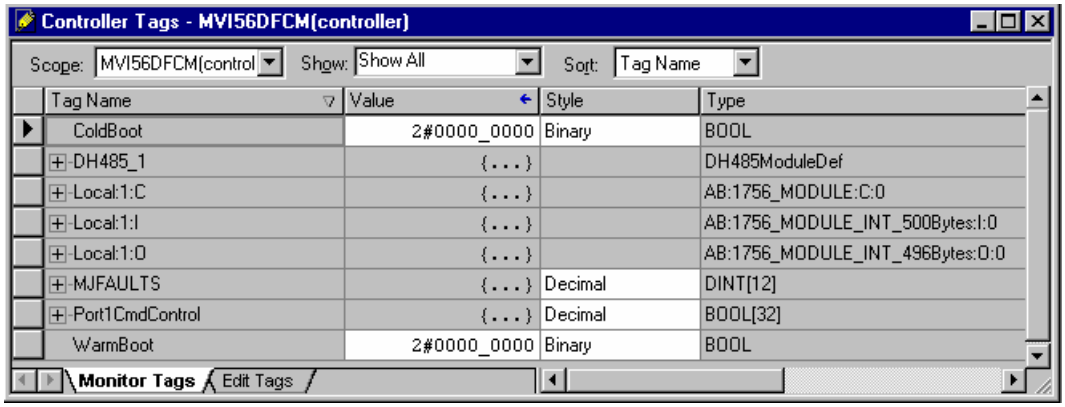
After completing the module setup, the Controller Organization window displays the module's presence. The data required for the module will be defined to the application, and objects will be allocated in the Controller Tags data area.



The next step in the module's setup is to define the User Defined Data Types to be used with the module. Copy these data types from the example ladder logic if you are not using the example. They will be defined if you are starting from the example ladder logic. The Controller Organization window should display the User Defined Data Types shown in the following example:

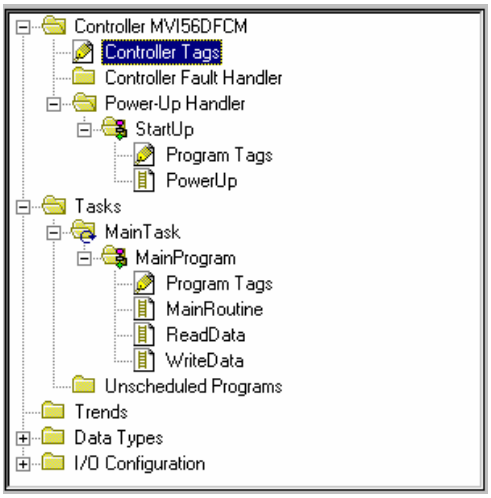


The next step in module setup is to define the data tag to be used to interface with the module and the ladder logic. Open the Controller Tags Edit Tags dialog box and enter the values shown in the following example. The MVI56-DH485 module is defined in the example as DH485_1. You can set the tag name to any valid tag name you desire. If you are using the example ladder logic, this step has already been performed.



At this point, take the time to adjust array sizes for the read and write data in the DH485_1 tag. Also, edit the DH485.CFG file to match the backplane data transfer parameters required for the application. Add the other information to the configuration file for the specific application. Use the forms in Appendix B to aid in building the configuration file.

The last step in the module setup is to add the ladder logic. If the example ladder logic is used, adjust the ladder to fit the application. When the ladder example is not used, copy the ladder logic shown in the Controller Organization window below to the application.

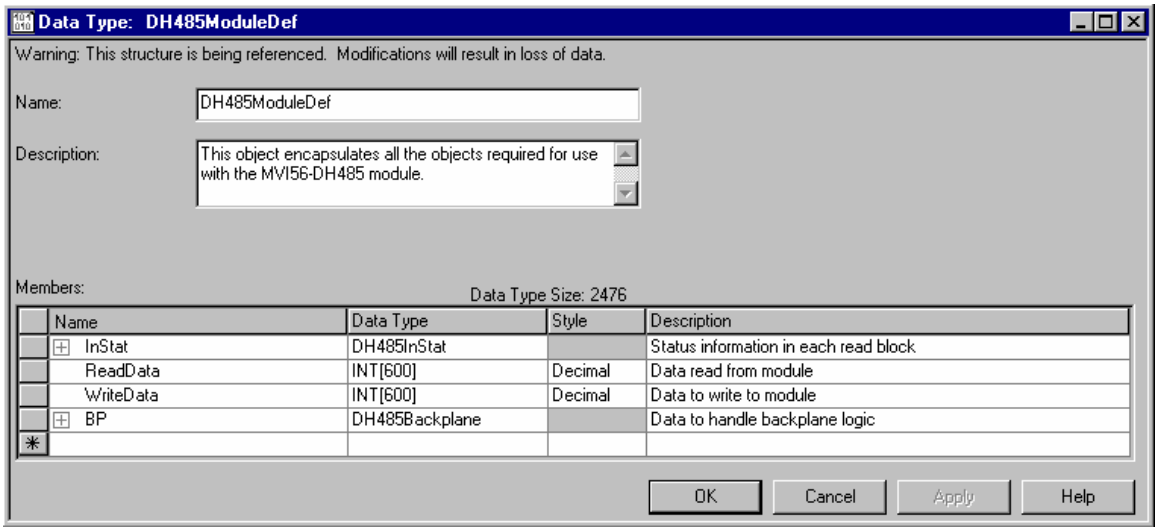


The module is now set up and ready to be used with your application. Insert the module in the rack and attach the DH485 serial communication cables. Download the new application to the controller and places the processor in run mode. Download the DH485.CFG file to the module using the configuration/debug port. If all the configuration parameters are set correctly and the module is attached to a DH485 networks, the module's Application LED should remain off and the backplane activity LED (BP ACT) should blink very rapidly. Refer to the Diagnostics and Troubleshooting section if you encounter errors.

Attach a computer or terminal to the configuration/debug port on the module and check the status of the module using the resident debugger in the module.

3.2 Module Data Object (DH485ModuleDef)

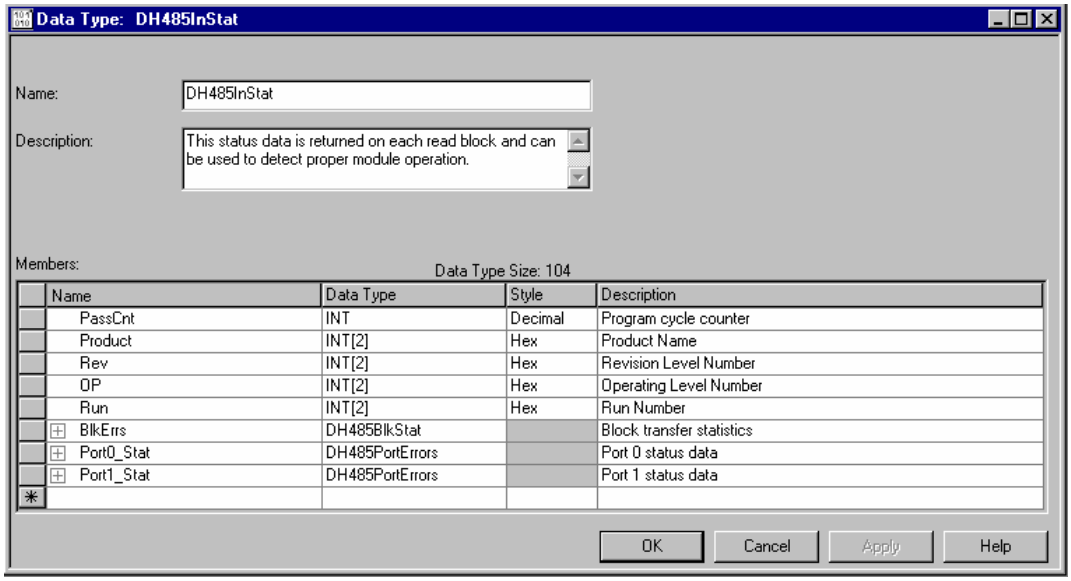
All status and variable data related to the MVI56-DH485 is stored in a user defined data type. An instance of the data type is required before the module can be used. This is done by declaring a variable of the data type in the Controller Tags Edit Tags dialog box. The structure of the object is displayed in the following figure:



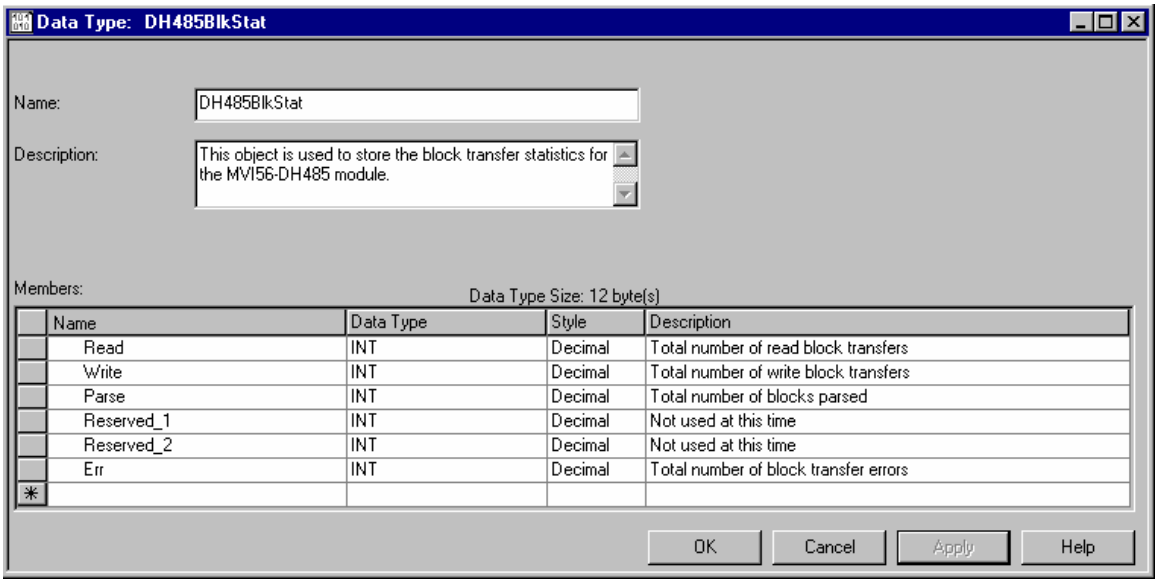
This object contains objects that define variables to be used with the module and status data related to the module. Each of these object types is discussed in the following sections of the document.

3.2.1.1 Status Object (DH485InStat)

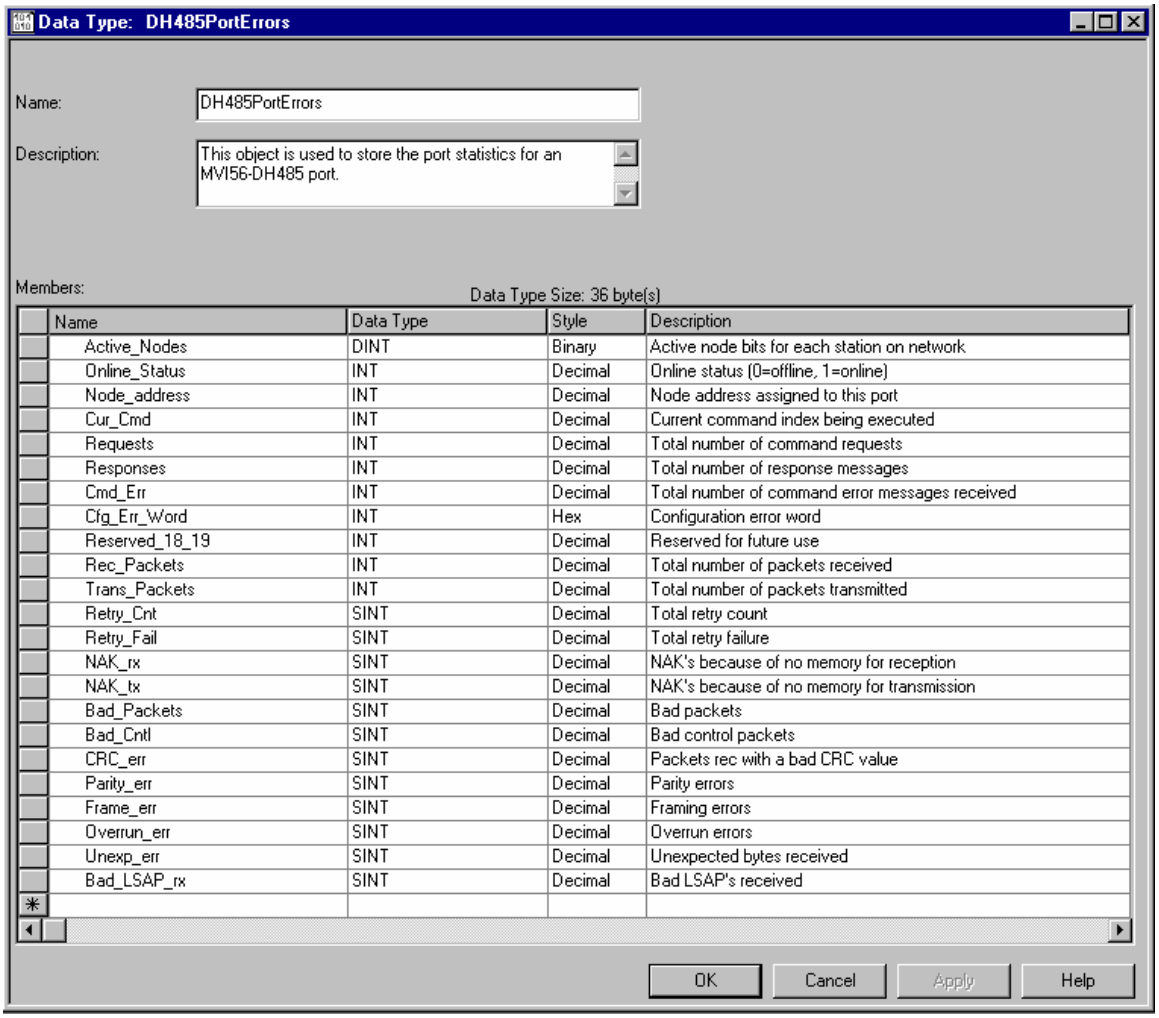
This object is used to store the status data of the module. The DH485InStat object shown below is updated each time a read block is received by the processor. The status data for each port is passed in blocks with identification codes of 0 and –1. This data can be used to monitor the state of the module at a "real-time rate".



The structure of the DH485BlkStat is shown in the following diagram and is used to store the status information for the backplane data transfer operation:



The structure of the DH485PortErrors is displayed in the following diagram:



Name	Data Type	Style	Description
Active_Nodes	DINT	Binary	Active node bits for each station on network
Online_Status	INT	Decimal	Online status (0=offline, 1=online)
Node_address	INT	Decimal	Node address assigned to this port
Cur_Cmd	INT	Decimal	Current command index being executed
Requests	INT	Decimal	Total number of command requests
Responses	INT	Decimal	Total number of response messages
Cmd_Err	INT	Decimal	Total number of command error messages received
Cfg_Err_Word	INT	Hex	Configuration error word
Reserved_18_19	INT	Decimal	Reserved for future use
Rec_Packets	INT	Decimal	Total number of packets received
Trans_Packets	INT	Decimal	Total number of packets transmitted
Retry_Cnt	SINT	Decimal	Total retry count
Retry_Fail	SINT	Decimal	Total retry failure
NAK_rx	SINT	Decimal	NAK's because of no memory for reception
NAK_tx	SINT	Decimal	NAK's because of no memory for transmission
Bad_Packets	SINT	Decimal	Bad packets
Bad_Cntl	SINT	Decimal	Bad control packets
CRC_err	SINT	Decimal	Packets rec with a bad CRC value
Parity_err	SINT	Decimal	Parity errors
Frame_err	SINT	Decimal	Framing errors
Overrun_err	SINT	Decimal	Overrun errors
Unexp_err	SINT	Decimal	Unexpected bytes received
Bad_LSAP_rx	SINT	Decimal	Bad LSAP's received

Refer to Appendix A for a complete list of the data stored in this object. This data is passed in blocks 0 and –1 for each port on the module.

3.2.2 User Data Objects

These objects are used to hold data to be transferred between the processor and the MVI56-DH485 module and to aid in the generation of the ladder logic. The user data is the read and write data transferred between the processor and the module as "pages" of data up to 200 words long.

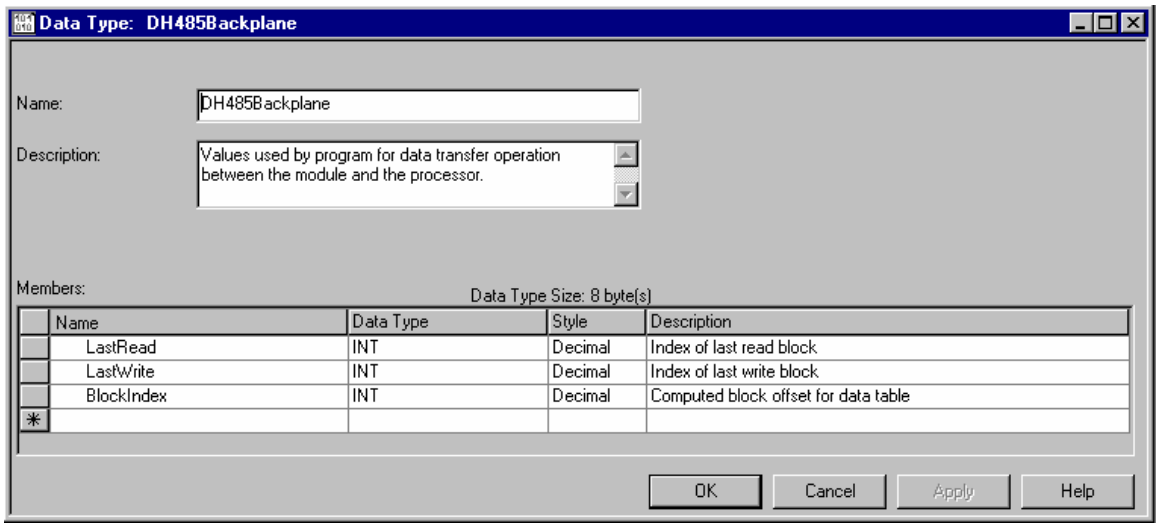
ReadData	INT[600]	Decimal	Data read from module
WriteData	INT[600]	Decimal	Data to write to module

The read data (**ReadData**) is an array set to match the value entered in the **Read Register Count** parameter of the DH485.CFG file. For ease of use, this array should be dimensioned as an even increment of 200 words. This data is paged up to 200 words at a time from the module to the processor. The ReadData task is responsible

for placing the data received into the proper position in the read data array. This data can be used for status and control in the ladder logic of the processor.

The write data (**WriteData**) is an array set to match the value entered in the **Write Register Count** parameter of the DH485.CFG file. For ease of use, this array should be dimensioned as even increments of 200 words. This data is paged up to 200 words at a time from the processor to the module. The WriteData task is responsible for placing the write data into the output image for transfer to the module. This data is passed from the processor to the module for status and control information for use in other nodes on the network.

The DH485Backplane object is utilized by the ladder logic to store the variables used in the backplane transfer operation. The structure of the object is shown in the following diagram:

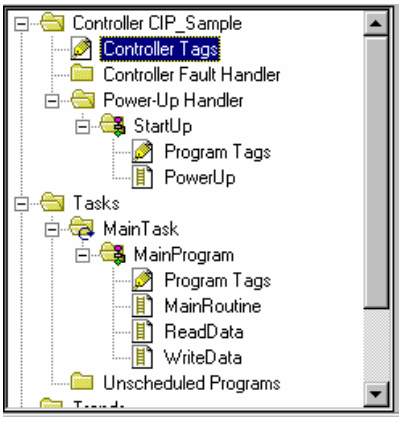


Other data objects can be added to the DH485ModuleDef object as required for specific applications. It is recommended to only add objects or members to the DH485ModuleDef object that relate to the module.

4 Modifying the Sample Ladder Logic

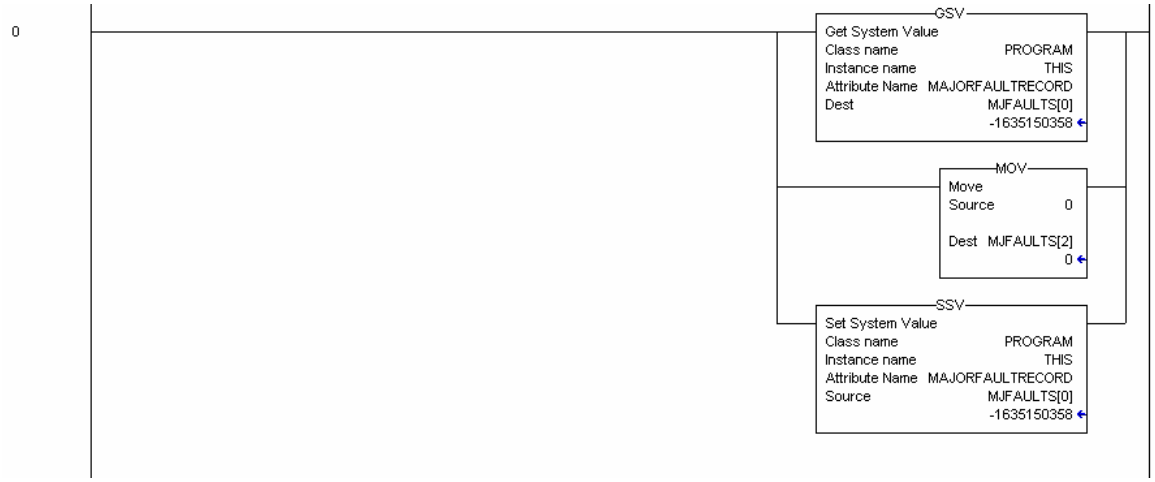
Ladder logic is required for application of the MVI56-DH485 module. Tasks that must be handled by the ladder logic are module read data initialization (optional), data transfer, special block handling, and status data receipt. Additionally, a power-up handler should be written to handle the initialization of the module's data and to clear any processor fault conditions.

The Controller Organization window for the example ladder logic is shown in the following window:



4.1 Power Up

The PowerUp ladder logic is used to initialize the data objects used by the MVI56-DH485 module and to recover controller faults on the initial power-up of the processor. The ladder logic required to perform these tasks is shown in the following rungs.



This rung is used to recover from a processor fault condition due to power-loss and restart when the processor is in run mode. You may also have to handle other fault conditions. Additionally, a fault handler can be written for the processor to handle other faults. The MJFAULTS object must be defined in the Controller Tags before it can be used in this logic.

This rung is used to initialize the last read and write values, the output image for the MVI56-DH485 module and the write data area to zero. The last read (**DH485_1.BP.LastRead**) and write (**DH485_1.BP.LastWrite**) values are used in the data transfer logic. The output image for the MVI56-DH485 module (**Local:1:O.Data[]**) is used to transfer data from the processor to the module. The write data area (**DH485_1.WriteData[]**) is used to store the processor data to be written to the module using the output image.

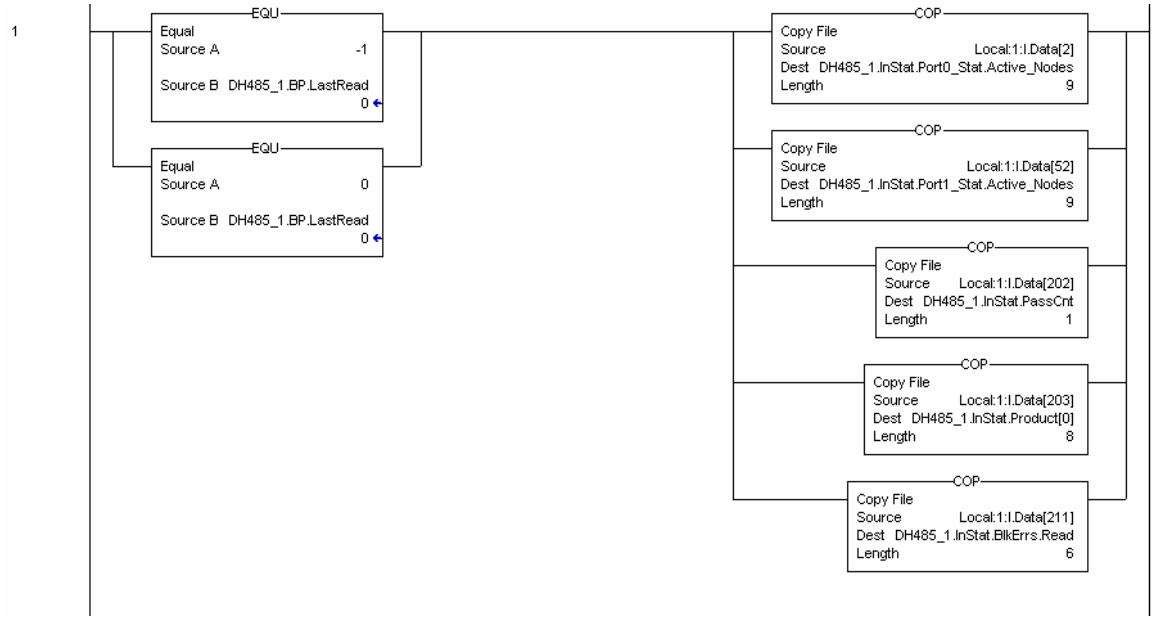
4.2 MainRoutine

The MainRoutine is used to recognize the presence of new read data from the module for the processor. The module will cycle through its list of read blocks to transfer data from the module to the processor. Whenever new data is available, the module will set the value for the block in the module's input image (**Local:1:I:Data[249]**). The ladder logic must constantly scan this input word for a new value. When a new value is present, the ladder logic should perform the ReadData and WriteData tasks in that order.

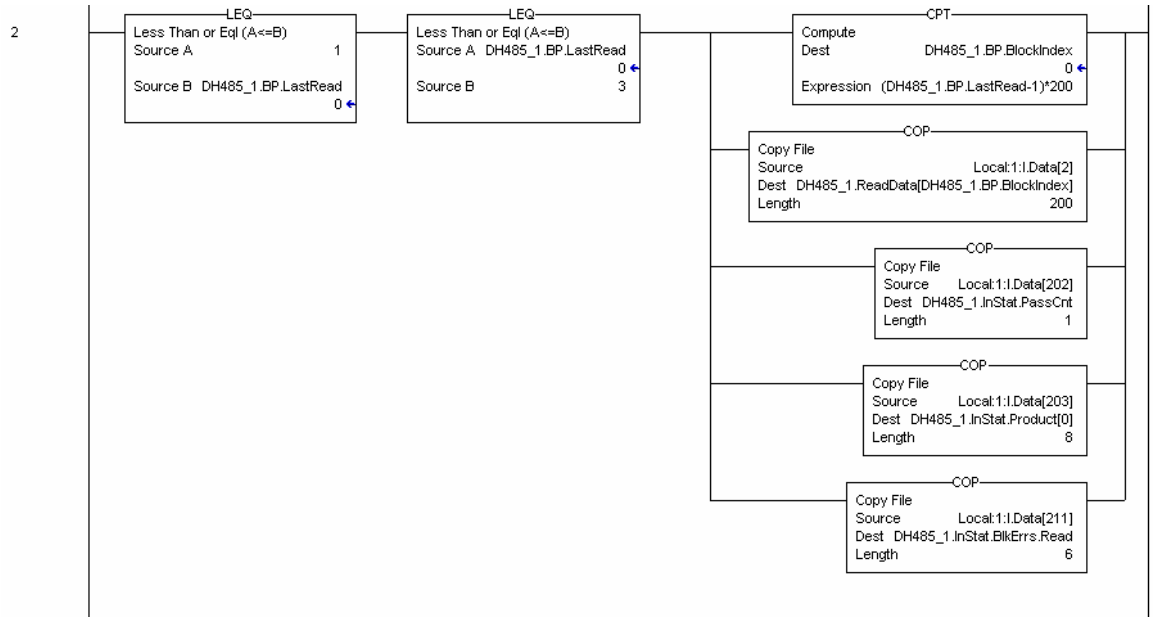
4.3 ReadData

The ReadData task is responsible for handling all new data received from the module and placing it in the proper location in the processor. Data is transferred from the module to the processor using the module's input image (**Local:1:I:Data[]**). The first rung of the task sets the last read block number (**DH485_1.BP.LastRead**) to the current block number sent from the module (**Local:1:I:Data[249]**).

The module will send block identification code 0 (and possibly –1 if not configured for any read blocks) to the processor. These blocks will only contain the status data. The following rung shown displays logic to handle these blocks:



The last rung of the ladder logic determines if the new data received in the input image is user data. If user data is present, the ladder logic will place the data in the correct location in the processor's read data area (**DH485_1.ReadData[]**). Up to 200 data words can be transferred in each block transfer. In addition to the user data, the block also contains general status data. This data should be copied to the correct data area in the module (**DH485_1.InStat**). This status data can be used to determine the "health" of the MVI56-DH485 module.

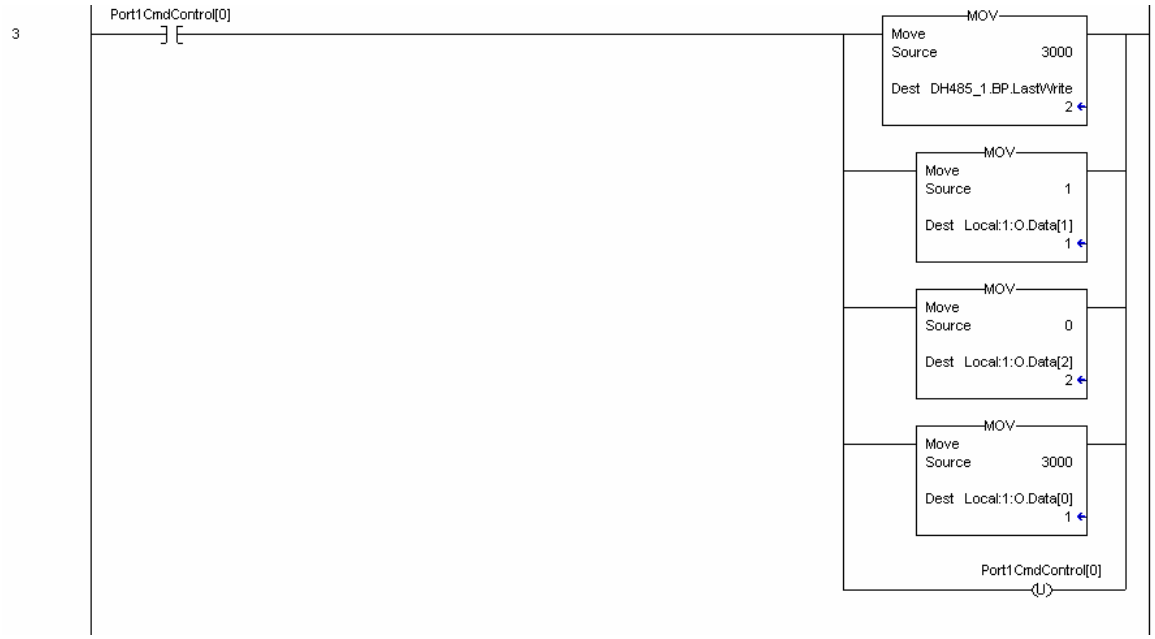


4.4 WriteData

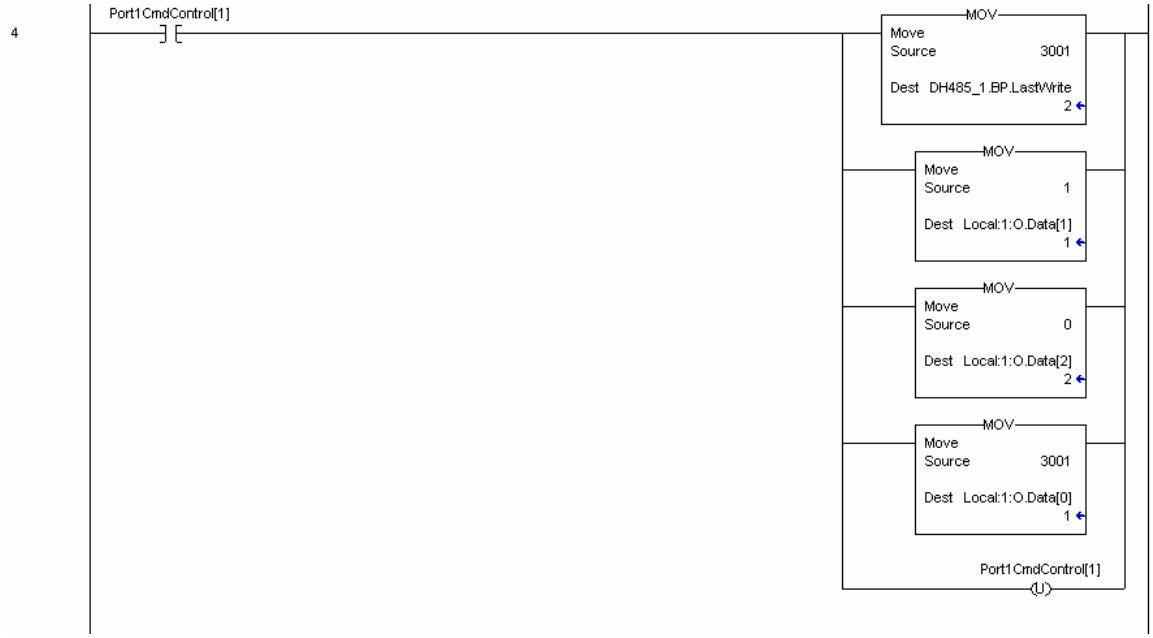
The WriteData task is responsible for sending data from the processor to the MVI56-DH485 module and for build control blocks. Data is transferred from the processor to the module using the module's output image (**Local:1:O:Data[]**). The first rung is used to store the currently requested data set in the module's **DH485_1.BP.LastWrite** data object. This object is used in all subsequent ladder logic in case the input word (**Local:1:I:Data[1]**) changes during processing.

The next two rungs are used to handle processor control of the module using the warm- and cold- boot control block numbers. When the processor requires the module to perform one of these operations, it simply copies the block number into the output image of the module and the module will perform the operation. Be certain to set the required block number in the last write object to prevent further processing in the WriteData task. Examples of each control block is given below:

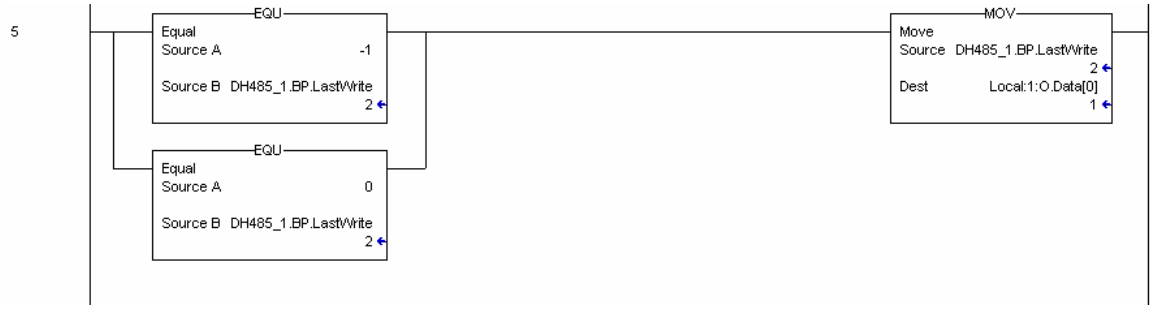
The next rung displays an example of command control using block identification code 3000 to disable commands in the command list. The following example rung will disable one command (command index 0).



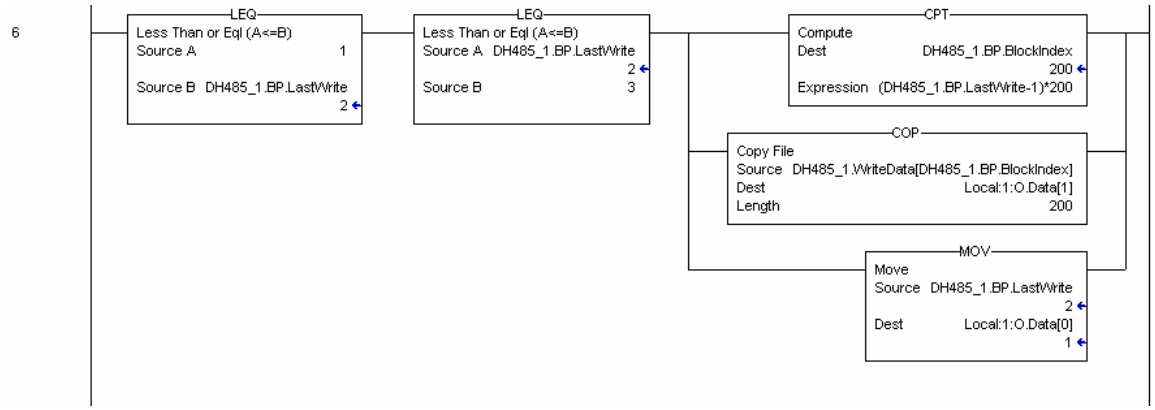
The next rung displays an example of command control using block identification code 3001 to enable commands in the command list. The following example rung will enable one command (command index 0).



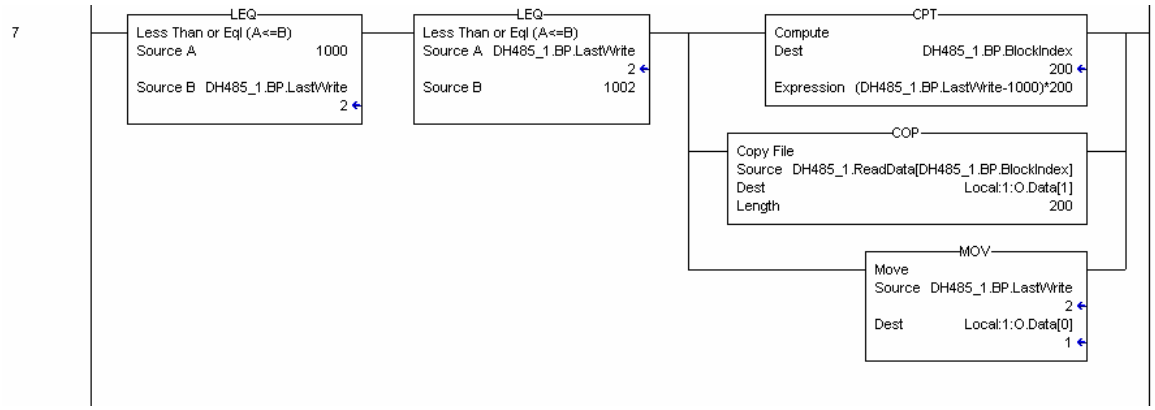
If the module is configured for no or only one block transfer, special processing is required. The module must observe the first word of the module's output image changing in order to recognize the receipt of new data. If the value never changes, the module will not process the data. This presents a problem when fewer than two blocks are to be transferred to the module from the processor. To overcome this problem, the module will send -1 and 0 in the input word. When the module is configured for zero write blocks, the following block request sequence will be present: -1, 0, -1, 0... When the module is configured for one write block, the following block request sequence will be present: 1, 0, 1, 0, 1, 0... The rung below is required to handle these conditions.



The next rung in the ladder logic is the most important. It handles the transfer of processor data to the module. Up to 200 words of user data held in the processor (**DH485_1.WriteData[]**) can be transferred to the module at one time.



If the module is configured to use the optional initialization of the read data area, ladder logic is required. The following rung demonstrates how to process these request blocks from the module:



This option is used to initialize the read data of the module to a known state before the module starts the normal data transfer between the module and the processor.

5 Modifying The Configuration Data

In order for the module to operate, a configuration file (DH485.CFG) is required. This configuration file contains information to set the data transfer characteristics between the module and the processor, to configure the module's peer-to-peer DH485 port characteristics, file mapping and user defined command list. Each parameter in the file must be set carefully in order for the application to be implemented successfully. Before editing the file, design your system using the forms located in Appendix B of this document. Appendix B contains a configuration form to be used to construct the DH485.CFG file. Appendix C contains an example listing of a DH485.CFG file.

The text file is separated into four sections with topic header names enclosed in the **[ ]** characters. The sections present in the file are as follows:

Section Name	Description
[Module]	Module parameters and backplane configuration information.
[DH485 Port x]	General parameters for a specific DH485 port on the module.
[DH485 Port x Commands]	Commands to be generated by the specific port driver
[DH485 Port x Maps]	Override file maps for the specific port driver. These maps are used when data file requests are made of the module.

After each section header, the file contains a set of parameters. Unique labels are used under each section to specify a parameter. Each label in the file must be entered exactly as shown in the file for the parameter to be identified by the program. If the module is not considering a parameter, check the label for the data item. Each parameter's value is separated from the label with the '**:**' character. This character is used by the program to delimit the position in the data record where to start reading data. All data for a parameter must be placed after the '**:**' character. For numeric parameter values any text located after the value will not be used. There must be at least one space character between the end of the parameter value and the following text. An example of a parameter entry is given below:

Error/Status Offset : 3000 #Database location for Error/Status Data

The parameter label is "Error/Status Offset" and the parameter value is 3000. The characters after the parameter value are ignored and are used for internal documentation of the configuration file.

Any record that begins with the '#' character is considered to be a comment record. These records can be placed anywhere in the file as long as the '**#**' character is found in the first column of the line. These lines are ignored in the file and can be used to provide documentation within the configuration file. Liberal use of comments within the file can ease the use and interpretation of the data in the file.

The command list and file mapping definition sections are formatted differently than the other sections. These sections contain lists of parameters to be used. Each list begins with the label **START** and ends when the **END** label is reached. When

entering the records into the list, make certain that the first character in each line is left blank.

The [DH485 Port x Commands] section for each port is used to define the commands to be issued by the module to other devices on the network. These commands can be used for data collection and/or control.

5.1 Command List Overview

In order to interface the MVI56-DH485 module to act as a master device, the user must construct a command list for each port. The commands in the list specify the node to be addressed, the function to be performed (read or write), the data area in the device to interface with and the registers in the internal database to be associated with the device data. The command list supports up to 100 commands. The command list is processed from top (command #0) to bottom. A poll interval parameter is associated with each command to specify a minimum delay time in seconds between the issuance of a command. If the user specifies a value of 10 for the parameter, the command will be executed no more frequently than every 10 seconds.

Write commands have a special feature, as they can be set to execute only if the data in the write command changes. If the register data values in the command have not changed since the command was last issued, the command will not be executed. If the data in the command has changed since the command was last issued, the command will be executed. Use of this feature can lighten the load on the network. In order to implement this feature; set the enable code for the command to a value of 2.

5.1.1 Command Entry Format

Each command entered in the command list section has the same format. The following is an example section for Port 0:

```
[DH485 Port 0 Commands]                    Command list for first DH485 port
# Swap Codes : 0=None, 1=Swap Words, 2=Swap bytes and words, 3=Swap bytes
# FileType Codes : 0=status, 1=bit, 2=timer, 3=counter, 4=control, 5=int, 6=float, 10=CIF
#  EnableCode  DB_Address  Count  Swap  Poll_Int  Node  Func  FileType  File#  Element#
START
           1        1000     10     0         0     1     1         5     10         0
           1        1010     10     0         0     1     0         5     11         0
END
```

The first part of each record in the section relates to the module interface and the last part relates to the node to be interfaced with. The following table details the definition of each field required for a user command:

Field	Definition
Enable Type Code	This field defines if the command is enabled and when it should be executed. The following codes are recognized by the application: 0 = Command is disabled 1 = Command is executed at the polling interval specified 2 = The write command is only executed when data changes

Field	Definition
Database Start Address	This field defines the starting address in the module's internal database to associate with the command. This field can have a value from 0 to 3999. The address supplied is a word address in the database.
Element Count	This field defines the number of elements to be used with the command. If the command is interfacing with CIF data, this parameter represents a byte count. For data file access, the data size utilized is dependent on the file type used.
Swap Code	This field is used to change the order of the bytes and/or words used when sending or receiving the data. The following codes are utilized: 0 = No data swapping 1 = Swap words in the data buffer 2 = Swap words and bytes in the data buffer 3 = Swap bytes in the data buffer
Poll Interval	This field is used to set the time interval between successive execution of the command. This parameter is specified in seconds. If the field is set to 10, the command will not be executed more frequently than every 10 seconds.
Node Number	This field is used to define the node address of the DH485 node to send the command request. This field should be set to a value from 0 to 31.
Function Code	This field defines the function to be executed by the command. The module uses the following codes: 0 = Read 1 = Write
File Type	This field is used to define the file type to be interfaced with in the other DH485 node. The program utilizes the following codes for this field: 0 = Status File (2 bytes per element) 1 = Bit File (2 bytes per element) 2 = Timer File (6 bytes per element) 3 = Counter File (6 bytes per element) 4 = Control File (6 bytes per element) 5 = Integer File (2 bytes per element) 6 = Floating-point File (4 bytes per element) 10 = CIF File (1 byte per element)
File Number	This field defines the file number to access. This field is ignored for CIF file access and should be set to 0. For Bit, Timer, Counter, Control, Integer, and Float data types, a maximum value of 255 is valid.
Element Number	This field defines the first element in the file specified to be associated with the command. For a CIF file, this parameter is given as the byte location. For Bit, Timer, Counter, Control, Integer, and Float data types, the maximum value is 255. For CIF data types the maximum value is 510.

The maximum number of elements requested from a remote node is determined by the file type and function code utilized in the command. The following table lists the maximum element count for each file type:

Maximum Element Counts for Read/Write Commands

File Type	Bytes/ Elements	Read Result	Write Result
Status	2	83	83
Bit	2	118	115
Timer	6	39	38
Counter	6	39	38
Control	6	39	38
Integer	2	118	115
Float	4	59	57
CIF	1	236	234

5.2 File Override Mapping

The [DH485 Port x Maps] section for each port is used to define file override mappings for the module. Use of these maps provides flexibility defining the file emulation supported in the slave driver of the module. Up to 50 maps can be defined in the module to override the fixed file-mapping feature of the module.

5.2.1 File Override Entry Format

Each file map entered in the configuration file has the same format. The following is an example section for Port 0:

```
[DH485 Port 0 Maps]                    Override file maps for first DH485 port
#  DB_Address    File_Number   Element    Length
START
          0            10         0       100
        100            12         0       100
END
```

The following table details the definition of each field required for an override map:

Field	Definition
Database Start Address	This field defines the starting address in the module's database for the file emulation. This parameter can be assigned a value of 0 to 3999.
File Number	This field defines the file number to be emulated at the database location specified.
Element	This field specifies the first element in the file to be emulated. This element number corresponds to the database start address set for the record.
Word Count (Length)	This field defines the number of word registers to be emulated in the file.

These file maps are searched first when a node on the network makes a request. If the requested data area is found in the map list, the database area associated with the command will be utilized. If the requested data area is not found in the map list, the fixed mapping data configuration will be used.

6 Diagnostics and Troubleshooting

The module provides diagnostic information in three forms to the user. 1) Status Data values are transferred from the module to the ControlLogix processor. 2) All data contained in the module can be viewed through the configuration/debug port to an attached terminal emulator. 3) LED status indicators on the front of the module yield information on the modules status.

The following sections explain how to obtain the Status Data from the module and the meaning of the individual LED's on the module.

6.1 Status Data From the Module's Input Image

The MVI56-DH485 module returns a 15-word status data area in each normal read block (input image) that can be used to determine the module's operating status. This data is transferred to the ControlLogix processor continuously. On each cycle of data transfer to the processor, the module will pass block 0 and/or –1. These blocks contain the status data for each port and the general module status data. For a complete listing of the status data object, refer to Appendix A.

6.2 Configuration/Debug Port

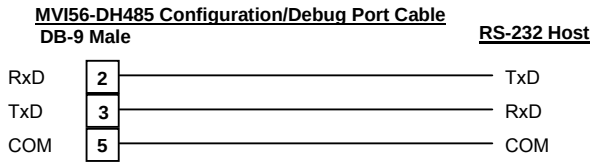
The module contains a configuration/debug port that is used to transfer the configuration between the module and a remote PC and to view all the configuration and status data of the module. The sections below discuss the facilities offered on this interface.

6.2.1 Required Hardware

The hardware requirements to interface with the configuration/debugger port are not too stringent. A personal computer with a standard serial port should suffice. For optimal performance, the minimum is required:

- 80468 based processor (Pentium preferred)
- 1 megabyte of memory
- At least one serial communications port available

Additionally, a null-modem cable is required between your PC and the port. The module's port has a DB-9 male connector at the end of a RJ-45 to DB-9 pigtail. The RJ-45 end of the cable is to be placed in the MVI56-DH485 port 1 connector (top port). The cable required is displayed in the following diagram:



6.2.2 Required Software

The software required on your personal computer to interface with the configuration/debugger port is operating system dependent. Tested software includes the following:

DOS	ProComm, PS-Term and several other terminal emulation programs
Windows 3.1	Terminal
Windows 95/98	HyperTerminal and PS-Term
Windows NT/2000/XP	HyperTerminal
Linux	Minicom

Any ASCII terminal emulation software package provided with your operating system should work as long as it can be configured as follows:

Baud Rate	57,600
Parity	None
Data Bits	8
Stop Bits	1
File Transfer Protocol	Zmodem

6.2.3 Using the Port

The following steps are required to interface with the configuration/debugger port:

1. Connect your computer to the module's port using a null-modem cable.
2. Start the terminal emulation program on your computer and configure the communication parameters to those shown in the Required Software section (57,600, N, 8, 1).
3. Enter the '?' character on your computer. If everything is set up correctly, the port's menu will be displayed.

If there is no response from the module, check the communication setup and the cable. In addition, make sure you are connected to the correct port on your computer and the module.

6.2.4 Menu Options

Features available through the use of the configuration/debug port on the MVI56-DH485 module are all reached using single keystrokes on your computer. There is a single main menu and several sub-menus presented on the port. To view the current selections available, press the '?' key on your computer. If you are in main menu mode, the following menu will be displayed:

```
DH485 APPLICATION MENU
  ?=Display Menu
  B=Block Transfer Statistics
  C=Module Configuration
  D=Database View
  E=DH485 Port 0 Menu
  F=DH485 Port 1 Menu
  R=Transfer Configuration from PC to MVI Unit
  S=Transfer Configuration from MVI Unit to PC
  V=Version Information
  W=Warm Boot Module
  Esc=Exit Program
```

If this menu is not displayed, press the 'M' key to display the main menu. All facilities offered by the configuration/debugger are shown on the main menu. Each option is discussed below:

Special characters used in the display are as follows:

B = Block Transfer Statistics

This menu option is used to display the configuration and statistics of the backplane data transfer operations. After selecting this option, the following will be displayed. Selecting this option at one-second intervals can be used to determine the number of blocks transferred each second.

```
BACKPLANE STATISTICS:

DATA TRANSFER CONFIGURATION:
  WRITE DATA TRANSFER:
        Start : 0       Count : 600     Max Blocks : 3       Last : 2
  READ DATA TRANSFER:
        Start : 600     Count : 600     Max Blocks : 3       Last : 0
  BLOCK COUNTS:
        Retry : 0       Failed: 0       Fail Cnt: 0
        Read  : 64384   Write : 64384   Parsing : 64383
        Error : 0       Event : 0       Command : 0
```

C = Module Configuration

This option displays the general module configuration information for the MVI56-DH485 module. After selecting the option, the following screen will be displayed:

```
MODULE CONFIGURATION:

MVI56-DH485 Communication Module Test

  DATABASE:
  Err/Stat Blk Pointer : 3000
  Initialize Outputs   : 0
  BLOCK TRANSFER:
  READ  -- Start : 400      Count : 400      Max : 2
  WRITE -- Start : 0        Count : 400      Max : 2
  FAIL COUNT     : 0
```

D = Database View

Selection of this menu option places the program in database view menu mode. This mode of operation is used to display the module's internal database values. To view the menu options available in this mode, press the '?' key and the following menu will be displayed:

```
DATABASE VIEW MENU
 ?=Display Menu
 0-3=Pages 0 to 3000
 S=Show Again
 -=Back 5 Pages
 P=Previous Page
 +=Skip 5 Pages
 N=Next Page
 D=Decimal Display
 H=Hexadecimal Display
 F=Float Display
 A=ASCII Display
 M=Main Menu
```

All data contained in the module's database is available for viewing using the menu options. Each option available on the menu is discussed in the sections below:

0-3 = Register pages 0-3000

This menu option is used to jump to a specific set of registers in the database and display the data. The keys perform the following functions:

Key	FUNCTION
0	Display registers 0 to 99
1	Display registers 1000 to 1099
2	Display registers 2000 to 2099
3	Display registers 3000 to 3099

S = Show Again

This menu option is used to display the current page of 100 registers in the database. Example output of the database display is shown below:

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

- = Back 5 Pages

This menu option is used to skip the previous 500 registers of data for viewing and display the data.

P = Previous Page

This menu option is used to select and display the previous 100 registers of data.

+ = Skip 5 Pages

This menu option is used to skip 500 registers of data and to display the new page of data.

N = Next Page

This menu option is used to select the next 100 registers of data for viewing and displays the data.

D = Decimal Display

This menu option is used to display the data on the current page in decimal format.

H = Hexadecimal Display

This menu option is used to display the data on the current page in hexadecimal format.

F = Float Display

This menu option is used to display the data on the current page in floating-point format. The program assumes that the values are aligned on even register boundaries. If floating-point values are not aligned as such, they will not be displayed properly.

A = ASCII Display

This menu option is used to display the data on the current page in ASCII format. This is useful for regions of the database that contain ASCII data.

M = Main Menu

This menu option is used to return to the main menu mode.

6.2.4.1 E and F = DH485 Port x Menu

These menu options are used to view the configuration and status data related to the DH485 driver for each of the application ports. After selecting one of the menu options, press the '**?**' key to display the following menu:

```
DH-485 DRIVER MENU (PORT 0)
  ?=Display Menu
  C=Configuration
  E=Command Error List
  L=Command List
  O=Override File Map List
  S=Status Data
  V=Version Information
  M=Return to Main Menu
```

Select one of the options from the menu to perform one of the operations available on the menu. Each menu option is discussed in the following sections:

C = Configuration

This menu option is selected to view the configuration information for the DH485 port. After selecting the option, the following will be displayed:

```
DH-485 DRIVER CONFIGURATION DATA (PORT 0):
My Node Number : 2         Maximum Nodes : 2         Baudrate :  19200
TokenHoldFactor: 1         Resp Timeout  : 10
Status Offset  : 1000      Err Offset    : 1100
CIF FILE PARAMETERS:
  CIF Read Offset : 4000      CIF Read Count : 100
  CIF Write Offset: 4100      CIF Write Count: 144
FILE EMULATION PARAMETERS:
  Database Offset : 0         First File #   : 7         File Size : 200
  Override Map Cnt: 2
Command Count : 4              Min Cmd Delay : 50
```

E = Command Error List

This menu option is selected to view the command error list for the user-defined commands. Select the '**?**' key to view the following menu:

```
COMMAND ERROR LIST MENU (DH485 Port 0)
 ?=Display Menu
 S=Show Again
 -=Back 2 Pages
 P=Previous Page
 +=Skip 2 Pages
 N=Next Page
 D=Decimal Display
 H=Hexadecimal Display
 M=Main Menu
```

The following is an example command error list display:

```
DH485 COMMAND ERROR LIST FOR PORT 0, COMMANDS 0 TO 19 (DECIMAL)
      0      0      0      0      0      0      0      0      0      0
      0      0      0      0      0      0      0      0      0      0
```

L = Command List

This menu option is selected to view the user-defined commands for the port. Select the '**?**' key to view the following menu:

```
DH485 MASTER COMMAND LIST MENU (Port 0)
 ?=Display Menu
 S=Show Again
 -=Back 5 Pages
 P=Previous Page
 +=Skip 5 Pages
 N=Next Page
 M=Main Menu
```

The following is an example command list display:

```
DH485 COMMAND LIST FOR PORT 0, COMMANDS 0 TO 9

ENB MBREG POLLINT COUNT SWAP LSTERR NODE FUNC KIND FILE# ELEM
  1   400      0    10    0   0000    1    0    5     9    0
  1   400      0    10    0   0000    1    1    5     9   10
  1   410      0    10    0   0000    1    0   10     0   10
  1   410      0    10    0   0000    1    1   10     0   20
  0     0      0     0    0   0000    0    0    0     0    0
  0     0      0     0    0   0000    0    0    0     0    0
  0     0      0     0    0   0000    0    0    0     0    0
  0     0      0     0    0   0000    0    0    0     0    0
  0     0      0     0    0   0000    0    0    0     0    0
  0     0      0     0    0   0000    0    0    0     0    0
```

O = Override File Map List

This menu option is selected to view the user defined file map configuration data. After selecting the option, press the '**?**' key to view the following menu:

```
DH485 OVERRIDE FILE MAP LIST MENU (Port 0)
 ?=Display Menu
 S=Show Again
 P=Previous Page
 N=Next Page
 M=Main Menu
```

The following is an example override file map list display:

```
DH485 OVERRIDE FILE MAP LIST FOR PORT 0, (0 TO 9)

 DB ADDR   FILE#   ELEMENT   LENGTH
      0      10         0      100
    100      12         0      100
      0       0         0        0
      0       0         0        0
      0       0         0        0
      0       0         0        0
      0       0         0        0
      0       0         0        0
      0       0         0        0
      0       0         0        0
```

S = Status Data

This menu option is selected to view the status data for the DH485 port. After selecting the option, the following will be displayed:

```
DH-485 DRIVER STATUS DATA (PORT 0):
Node : 2    Status: ONLINE   Cfg Error: 0x0000
MASTER COMMAND COUNTERS:
  Requests: 58604    Responses   : 58603
  Errors  : 0        Current Cmd#: 5
DIAGNOSTIC COUNTERS:
  Packets Rx    : 62930    Packets Tx   : 62930
  Retry Count   : 209      Retry Failed : 5
  NAK No Mem Rx : 0        NAK No Mem Tx: 0
  Total Bad Pkts: 0        Bad Ctrl's   : 0
  Bad CRC's     : 0        Bad Parity   : 0
  Bad Framing   : 0        Overrun Error: 0
  Unexpected Rx : 0        Bad LSAP Rx  : 0

ACTIVE NODES:
 [00..15]   . 01 02 03  .  .  .  .  .  .  .  .  .  .  .  .
 [16..31]   .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
```

The Cfg Error: parameter displayed will have a value of 0x0000 if there are no configuration errors for the driver. This is a bit mapped value (displayed in hexadecimal format) with each bit representing a configuration error. The following table lists the bits used by the module:

Bit	Code	Description
0	0x0001	Invalid baud rate
1	0x0002	Invalid node address
2	0x0004	Invalid maximum node address
3	0x0008	Invalid token hold factor
4	0x0010	Invalid response timeout
5	0x0020	Invalid status or command error DB offset
6	0x0040	Invalid CIF read count or DB offset
7	0x0080	Invalid CIF write DB offset
8	0x0100	Invalid file size
9	0x0200	Invalid file offset
10	0x0400	
11	0x0800	
12	0x1000	
13	0x2000	
14	0x4000	
15	0x8000	

V = Version Information

This menu option is selected to the version information for the DH485 driver. After selecting the option, the following will be displayed:

```
DH485 DRIVER VERSION INFORMATION:

  DH485 DRIVER VERSION : 1.05
  FAR CORE LEFT        : 266960
```

M = Return to Main Menu

This menu option is selected to return to the main program menu.

R = Transfer Configuration from PC to MVI Unit

This menu option is selected to transfer a new DH485.CFG from a remote PC to the module. This operation is required when the changes in the configuration of the module are made. After selecting the option, follow the instructions displayed on the terminal. The ZModem protocol is required to transfer the file from the PC to the module. The name of the configuration file must be DH485.CFG. After the download process succeeds, the module will perform a warm boot operation to implement the new configuration.

S = Transfer Configuration from MVI Unit to PC

This menu option is used to transfer the configuration file present on the module's compact flash disk to a remote PC. After selecting the option, follow the instructions presented on the terminal. The ZModem file-transfer protocol is required for this operation.

V = Version Information

This option is used to view the current version of the software for the module and other important values. After selecting the option, the following will be displayed:

```
VERSION INFORMATION:

  DH485 MASTER/SLAVE COMMUNICATION MODULE (MVI56-DH485)
  (c) 1999-2003, ProSoft Technology, Inc.

  PRODUCT NAME CODE          : DH45
  SOFTWARE REVISION LEVEL    : 1.01
  OPERATING SYSTEM REVISION  : 0203
  RUN NUMBER                 : 0601
  PROGRAM SCAN COUNTER       : 37833
  FREE MEMORY                : 272000

  BACKPLANE DRIVER VERSION   : 1.03
  BACKPLANE API VERSION      : 1.01
  MODULE NAME : MVI56 ProSoft Technology, Inc
  VENDOR ID   : 309       DEVICE TYPE    : 12
  PRODUCT CODE: 83        SERIAL NUMBER  : 00000001
  REVISION    : 1.01

  MODULE NAME:
MVI56-DH485 Communication Module Test
```

This information may be requested when calling for technical support on the product. Values at the bottom of the display are important in determining module operation. The **Program Scan Counter** value is incremented each time a module's program cycle is complete. This value can be used to determine the frequency of program execution by pressing the 'V' key at one-second intervals.

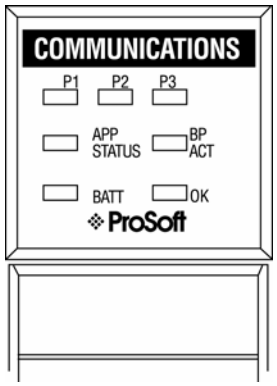
W = Warm Boot Module

This option is selected when a warm-boot operation is required of the module. This request is automatically made after a new configuration file is downloaded to the module. This operation can be performed to initialize the database and all module status data.

Esc = Exit Program

This option is used to exit the program and perform a reboot of the module. This option should only be selected if instructed by the ProSoft Technical Support Group. If you select the option, the module will cease operation. Data will no longer be transferred between the ports and the module and between the ControlLogix processor and the module. This might cause an upset to a currently running process.

6.3 LED Status Indicators



The LED's indicate the module’s operating status as follows:

ProSoft Module	Color	Status	Indication
P1	Green	On	Data is being transferred between the module and a remote terminal using the Configuration/Debug port.
		Off	No data is being transferred on the Configuration/Debug port.
P2	Green	On	Data is being transferred between the module and the DH485 network on Port 0.
		Off	No data is being transferred on the port.
P3	Green	On	Data is being transferred between the module and the DH485 network on Port 1.
		Off	No data is being transferred on the port.
APP	Amber	Off	The MVI56-DH485 is working normally.
		On	The MVI56-DH485 module program has recognized a communication error.
BP ACT	Amber	On	The LED is on when the module is performing a write operation on the backplane.
		Off	The LED is off when the module is performing a read operation on the backplane. Under normal operation, the LED should blink rapidly on and off.
OK	Red/ Green	Off	The card is not receiving any power and is not securely plugged into the rack.
		Green	The module is operating normally.
		Red	The program has detected an error or is being configured. If the LED remains red for over 10 seconds, the program has probably halted. Remove the card from the rack and re-insert the card to restart the module's program.
BAT	Red	Off	The battery voltage is OK and functioning.
		On	The battery voltage is low or the battery is not present. Replace the battery on the module.

During module configuration, the OK will be red and the APP and BP ACT LED's are on.

If the APP, BP ACT and OK LED's blink at a rate of every one-second, call ProSoft Technology, Inc. support. There is a serious problem with the module, and it may have to be sent back to ProSoft.

6.3.1 Clearing a Fault Condition

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

1. Remove the card from the rack and re-insert the card in the rack.
2. Verify the configuration data being transferred to the module from the ControlLogix processor.

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

6.3.2 Troubleshooting

The following table is designed to assist you in troubleshooting the module. Please use this table to attempt to correct a problem. However, if you have additional questions or problems, please do not hesitate to contact us.

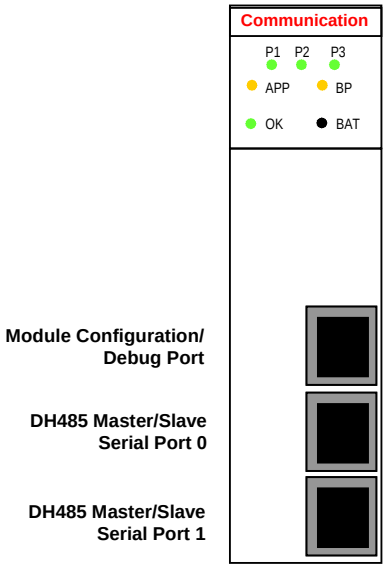
The entries in this section have been placed in the order in which the problems would most likely occur after powering up the module.

Problem Description	Steps to take
Processor Fault	Be sure that the module is plugged into the slot that has been configured for the MVI56-DH485 module. Assure that the slot in the rack configuration has been set up correctly:
Processor I/O LED flashes	This indicates there is a problem with backplane communications. Be certain this and all modules in the rack are configured in the processor.
BP ACT LED remains off or blinks slowly	This indicates that backplane transfer operations are failing. Use the Configuration/Debug port facility to check this. To establish backplane communications make sure of the following: The backplane driver is loaded in the module. The module is configured for read and write block data transfer. The ladder logic handles all read and write block situations. The module is configured in the processor.
OK LED remains red	The program has halted or a critical error has occurred. Connect to the Configuration/Debug port to see if the module is running. If the program has halted, remove the card from the rack and re-insert the card in the rack.

7 Cable Connections

The MVI56-DH485 module has the following communication connections on the module:

- Two DH485 communication port (RJ45 connector)
- One RS-232 Configuration/Debug port (RJ45 connector)



7.1 DH485 Communication Ports

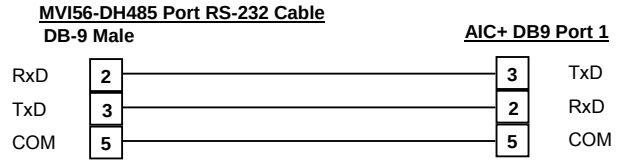
This section describes the different methods to connect a DH485 driver enabled device to a DH485 network. The Allen-Bradley AIC+ Advanced Interface Converter User Manual (Catalog Number 1761-NET-AIC) should be consulted for cabling requirements and configurations for a DH485 network using an AIC+. The MVI56-DH485 module has two physical DH485 connectors with a RJ45 plug located on the front of the module.

7.1.1 Connecting the Cable to the Connector

ProSoft provides a single RJ45 to male DB-9 adapter to permit simpler interfacing to other devices for each port. The module's DH485 ports can be configured to operate in RS-232 or RS-485 mode. The interface to be associated with each port is set with jumpers on the module. The following sections describe each interface.

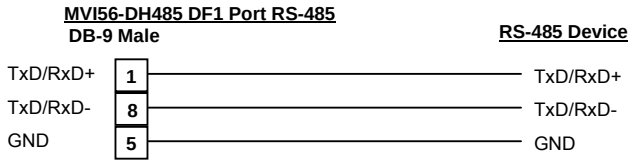
7.1.1.1 RS-232

When the RS-232 interface is selected, the port can be connected to a DH485 network using an AIC+. The cable required for this connection is shown in the following display:

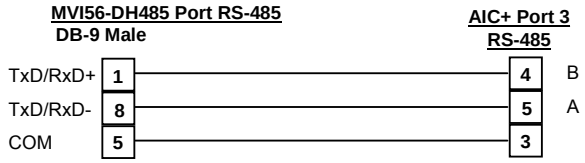


7.1.1.2 RS-485

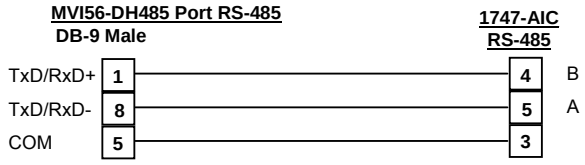
When the RS-485 interface is used, a single two or three wire cable is required. The use of the ground is optional and dependent on the RS-485 network. The cable required for this interface is shown in the following diagram:



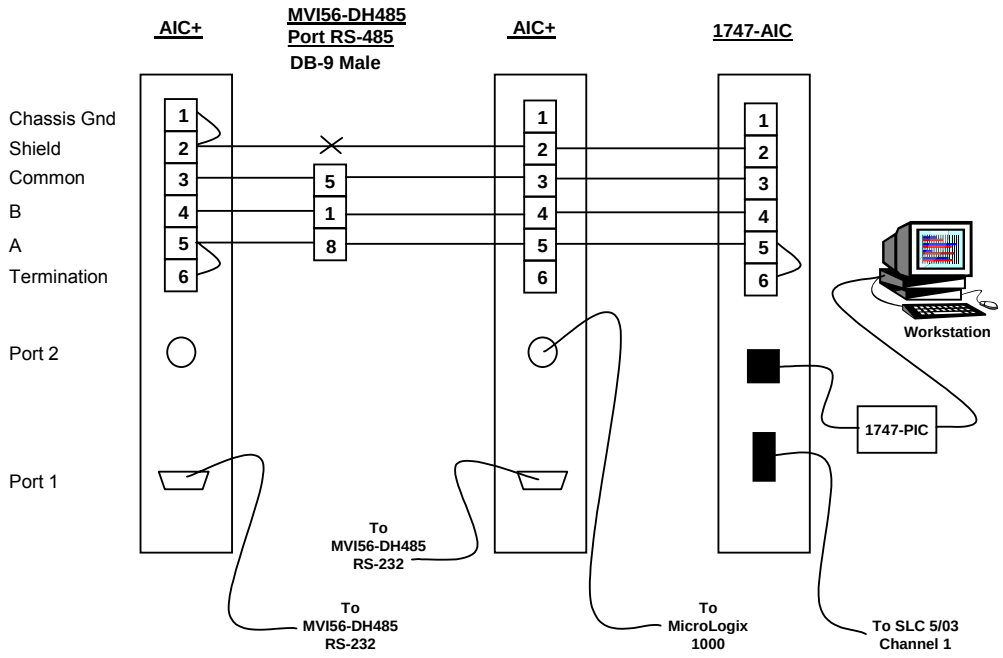
When connecting to port 3 of an AIC+ module, the following is the correct wiring:



When connecting to a 1747-AIC module, the following is the correct wiring:



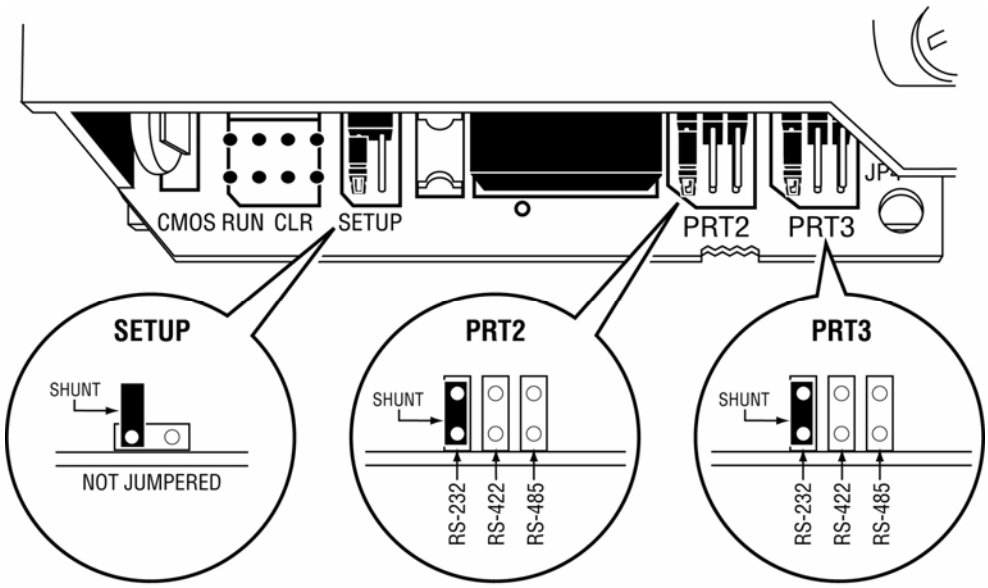
An example DH485 network is displayed in the following diagram:



This network displays the two different methods to configure the module for a DH485 network. Please note there is no place on the module's RS-485 to land the shield, and when used in the configuration shown, it must be wired externally. Make certain the RS interface jumper on the module is set to the correct position: RS-232 or RS-485.

7.1.1.2.1 Jumper Configurations

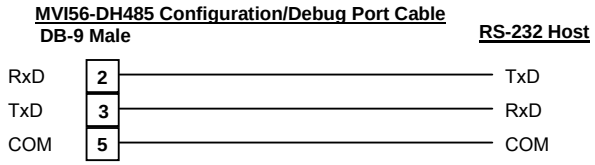
Changing from RS-232 to RS-485 requires that you change the jumper on the module to the appropriate settings. To do so, use a pair of needle-nose pliers to move the jumper shunt to the appropriate pins as shown in the following diagram:



Note: The RS-422 jumper setting shown in the diagram is not used with this module configuration.

7.2 RS-232 Configuration/Debug Port

This port is physically an RJ-45 connection. An RJ-45 to DB-9 pigtail cable is shipped with the module. This port permits a PC based terminal emulation program to view configuration and status data in the module and to control the module. The cable for communications on this port is shown in the following diagram:



Appendix A – MVI56-DH485 Status Data

This appendix contains a listing status data presented to the processor by the module. The following table lists the data returned with each normal read block:

WORD#	BYTE#	DESCRIPTION
202		Scan Counter
203-204		Product Name (ASCII)
205-206		Revision (ASCII)
207-208		Operating System Revision (ASCII)
209-210		Production Run Number (ASCII)
211		Read Block Count
212		Write Block Count
213		Parse Block Count
214		Reserved
215		Reserved
216		Number of Block Errors

The user can also copy this data block to the module using the Error/Status Pointer parameter in the configuration file.

The following table lists the status data passed from the module to the ControlLogix processor in blocks 0 and –1:

WORD#	BYTE#	DESCRIPTION	
2-3	0-3	Active node bits for stations 0 to 31	Port 0
4	4-5	Online status (0=Offline, 1=Online)	
5	6-7	Node address of the unit/port emulated	
6	8-9	Current command index being issued	
7	10-11	Total number of request messages	
8	12-13	Total number of response messages received	
9	14-15	Total number of command list errors	
10	16-17	Configuration error word. Each bit represent a configuration error condition.	
11	18-19	Reserved for future use.	
12	20-21	Total number of packets received	
13	22-23	Total number of packets transmitted	
14	24	Total retry count	
	25	Retry failure counter	
15	26	Total number of NAK's because of no memory for reception	
	27	Total number of NAK's because of no memory for transmission	
16	28	Total number of bad packets	
	29	Total number of bad control packets	
17	30	Total number of packets received with a bad CRC value	
	31	Total number of parity errors	
18	32	Total number of framing errors	
	33	Total number of overrun errors	
19	34	Total number of unexpected bytes received	
	35	Total number of bad LSAP's received	
20-51		Reserved for future use.	
52-53	0-3	Active node bits for stations 0 to 31	Port 1
54	4-5	Online status (0=Offline, 1=Online)	
55	6-7	Node address of the unit/port emulated	
56	8-9	Current command index being issued	
57	10-11	Total number of request messages	
58	12-13	Total number of response messages received	
59	14-15	Total number of command list errors	
60	16-17	Configuration error word. Each bit represent a configuration error condition.	
61	18-19	Reserved for future use.	
62	20-21	Total number of packets received	
63	22-23	Total number of packets transmitted	
64	24	Total retry count	
	25	Retry failure counter	
65	26	Total number of NAK's because of no memory for reception	
	27	Total number of NAK's because of no memory for transmission	
66	28	Total number of bad packets	
	29	Total number of bad control packets	
67	30	Total number of packets received with a bad CRC value	
	31	Total number of parity errors	
68	32	Total number of framing errors	
	33	Total number of overrun errors	
69	34	Total number of unexpected bytes received	
	35	Total number of bad LSAP's received	
102-201		Reserved for future use.	
202		Scan Counter	
203-204		Product Name (ASCII)	
205-206		Revision (ASCII)	
207-208		Operating System Revision (ASCII)	
209-210		Production Run Number (ASCII)	
211		Read Block Count	
212		Write Block Count	
213		Parse Block Count	
214		Reserved	
215		Reserved	
216		Number of Block Errors	

The data in words 10 and 60 (Configuration Error Word) are bit-mapped values with the following definition:

Bit	Code	Description
0	0x0001	Invalid baud rate
1	0x0002	Invalid node address
2	0x0004	Invalid maximum node address
3	0x0008	Invalid token hold factor
4	0x0010	Invalid response timeout
5	0x0020	Invalid status or command error DB offset
6	0x0040	Invalid CIF read count or DB offset
7	0x0080	Invalid CIF write DB offset
8	0x0100	Invalid file size
9	0x0200	Invalid file offset
10	0x0400	
11	0x0800	
12	0x1000	
13	0x2000	
14	0x4000	
15	0x8000	

When no configuration errors are present, the words will have a value of 0x0000 (hexadecimal). Configuration errors should be corrected to have the module perform as required by the application.

Additionally, each command in the user command list contains an error status data area. This data can be viewed through the debug/configuration port. This data can be configured to be placed in the module's database with each register containing an error value for each command for each port as shown in the following diagram:

WORD#	DESCRIPTION
0	Error code for command index 0.
1	Error code for command index 1.
---	---
99	Error code for command index 99.

The error codes placed in this data area have the following definitions:

DH485 GENERAL ERROR CODES

Error #	Description
0	Operation successful
1	Invalid parameter
2	Device is already open
3	Device is not present
4	Invalid access
5	The function has timed out
6	
7	Unable to configure the requested port
8	Unable to allocate memory for DH485 driver

DH485 API SPECIFIC ERROR CODES

Error #	Description
0x0800	Command only permitted in master mode
0x0801	Command already active on the port
0x0802	Response to request timed out
0x0803	Unable to allocate memory for the request
0x0804	Illegal command or format
0x0805	Host could not complete request (hardware fault)
0x0806	Out of memory, file or rung does not exist
0x0807	Field has an illegal value
0x0808	Not enough fields in request message
0x0809	Too many fields in request message
0x080A	Symbol not found
0x080B	Symbol 0 or greater than maximum characters permitted in message
0x080C	Does not exist, illegal size
0x080D	File wrong size, address past end of file
0x080E	Data or file too large (memory not available)
0x080F	Request too large to transmit message (size+address > max message)
0x0810	Access denied
0x0811	Command cannot be executed
0x0812	Illegal data type information
0x0813	Illegal parameter, invalid data in search or command block
0x0814	File open by another node
0x0815	Program owned by another node
0x0816	Unknown error returned from host
0x0817	No message active on the port

Appendix B – MVI56-DH485 Configuration

This appendix contains a listing of the data required in the DH485.CFG file in order to configure the module. The table below lists the sections and items for each section that can be used in the module configuration file (DH485.CFG):

[Section]/Item	Range	Description	IF Error	Config. Value
[MODULE]		Configuration header for general module information		
Module Name:	Up to 80 chars	Name of the module for use on reports. Use this parameter to identify your module in your system.		
Error/Status Pointer:	-1 to 3955	Starting register location in virtual database for the error/status table. If a value of -1 is entered, the error/status data will not be placed in the database. All other valid values determine the starting location of the data. This data area includes the module version information. Please refer to Appendix A for information about this data block.	-1	
Write Register Start:	0 to 3999	This parameter specifies the starting register in the module where the data transferred from the processor will be placed. Valid range for this parameter is 0 to 3999.	0	
Write Register Count:	0 to 4000	This parameter specifies the number of registers to transfer from the processor to the module. Valid entry for this parameter is 0 to 4000.	0	
Read Register Start:	0 to 3999	This parameter specifies the starting register in the module where data will be transferred from the module to the processor. Valid range for this parameter is 0 to 3999.	0	
Read Register Count:	0 to 4000	This parameter specifies the number of registers to be transferred from the module to the processor. Valid entry for this parameter is 0 to 4000.	0	
Failure Flag Count:	0 to 1000	This parameter specifies the number of successive transfer errors that must occur before the communication ports are shut down. If the parameter is set to 0, the communication ports will continue to operate under all conditions. If the value is set larger than 0 (1-1000), communications will cease if the specified number of failures occur.	0	

[Section]/Item	Range	Description	IF Error	Config. Value
Initialize Output Data:	0 or 1	This parameter is used to determine if the output data for the module should be initialized with values from the processor. If the value is set to 0, the output data will be initialized to 0. If the value is set to 1, the data will be initialized with data from the processor. Use of this option requires associated ladder logic to pass the data from the processor to the module.	0	

Variable Name	Data Range	Description	IF Error	Config. Value
[DH485 PORT x]		General configuration information for the specified DH485 port on the module.		
Baud Rate:	9600 or 19200	This is the baud rate to use for the DH485 network. Select one of the listed baud rates.	19200	
Node Address:	0 to 31	This is the node address to be utilized by the DH485 driver for this port on the network. Enter a value not already used on the network in the range of 0 to 31. If a value of 255 is utilized or set by the module, the port is disabled. NOTE: All nodes on the network should be set to the lowest set of values in the range of 1 to 4).	255	
Maximum Node Address:	0 to 31	Enter the maximum address that the initiator searches for before wrapping to zero. The default is 31. This parameter should be set to the maximum node address set in the DH485 network.	31	
Token Hold Factor:	1 to 4	Enter the number of transmissions (plus retries) that a node holding a token can send onto the data link each time that it receives the token. Enter a value between 0-31. The default is 1.	1	
Response Timeout:	1 to 50	This parameters sets the number of 100 millisecond time intervals to wait for a response to a request from the module. If the module does not receive the response with in the time period specified, a timeout condition will be set for the command.	10	

[Section]/Item	Range	Description	IF Error	Config. Value
Status DB Offset:	-1 and 0 to 3980	This parameter sets the location of the status data for the port in the module's internal database. If the parameter is set to -1, the data is not placed in the database. If a valid value is entered, the module's status data will be placed in the database starting at the location indicated.	-1	
Command Error DB Offset:	-1 and 0 to 3900	This parameter sets the location of the command error list data for the port in the module's internal database. If the parameter is set to -1, the data is not placed in the database. If a valid value is entered, the module's error list data will be placed in the database starting at the location indicated.	-1	
CIF Read DB Offset:	-1 and 0 to 7500 (only even values)	This parameter sets the starting byte location in the module's database where the CIF file Read will be placed. This data is passed from CIF memory area to the set location in the module's database. If this parameter is set to -1, no CIF read data will be utilized. When the CIF Read Area is disabled (CIF Read DB Offset = -1) or CIF Read Count = 0, the Debug menu will show this parameter as 65535.	-1	
CIF Read Count:	0 to 242	This parameters sets the number of bytes to transfer from the CIF file to the database. The CIF write count will be calculated as (244 - CIF Read Count)	0	
CIF Write DB Offset:	-1 and 0 to 7500 (only even values)	This parameter sets the starting byte location in the module's database where the CIF file Write data will be read from. This data is passed to the CIF memory area from the set location in the module's database. If this parameter is set to -1, no CIF write data will be utilized. When the CIF Write Area is disabled (CIF Write DB Offset = -1) or CIF Write Count = 0, the Debug menu will show this parameter as 65535.	-1	
First File:	0 to 255	This parameter sets the file number for the first file to be emulated by the module.	0	
File Size:	1 to 1000	This parameter sets the word size of all the files emulated in the module.	1000	
File Offset:	0 to 999	This parameter sets the word offset into the module's database where the file emulation will start.	0	

[Section]/Item	Range	Description	IF Error	Config. Value
Min Command Delay:	0 to 10000	This parameter sets the minimum number of milliseconds to wait before issuing each command. This parameter is utilized to keep the network from being flooded with requests from the module.	0	

[DH485 Port x Commands]		Command list for specified DH485 port

```
# Swap Codes : 0=None, 1=Swap Words, 2=Swap bytes and words,
3=Swap bytes
# FileType Codes : 0=status, 1=bit, 2=timer, 3=counter, 4=control, 5=int, 6=float,
10=CIF
#  EnableCode  DB_Address  Count  Swap  Poll_Int  Node  Func  FileType  File#
Element#
START
            1        1000     10     0         0     1     1         5
10         0
            1        1010     10     0         0     1     0         5
11         0
END
```

[DH485 Port x Maps]		Override file maps for specified DH485 port

```
#  DB_Address    File_Number   Element    Length
START
            0             10         0       100
          100             12         0       100
END
```

The following table lists the fields required for each command list entry:

Field	Definition
Enable Type Code	This field defines if the command is enabled and when it should be executed. The following codes are recognized by the application: 0 = Command is disabled 1 = Command is executed at the polling interval specified 2 = The write command is only executed when data changes (The enable code only applies to write commands)
Database Start Address	This field defines the starting address in the module's internal database to associate with the command. This field can have a value from 0 to 3999. The address supplied is a word address in the database.

Field	Definition
Element Count	This field defines the number of elements to be used with the command. If the command is interfacing with CIF data, this parameter represents a byte count. For data file access, the data size utilized is dependent on the file type used. Please refer to the Maximum Element Counts table.
Swap Code	This field is used to change the order of the bytes and/or words used when sending or receiving the data. The following codes are utilized: 0 = No data swapping 1 = Swap words in the data buffer 2 = Swap words and bytes in the data buffer 3 = Swap bytes in the data buffer
Poll Interval	This field is used to set the time interval between successive execution of the command. This parameter is specified in seconds. If the field is set to 10, the command will not be executed more frequently than every 10 seconds.
Node Number	This field is used to define the node address of the DH485 node to send the command request. This field should be set to a value from 0 to 31.
Function Code	This field defines the function to be executed by the command. The module uses the following codes: 0 = Read 1 = Write
File Type	This field is used to define the file type to be interfaced with in the other DH485 node. The program utilizes the following codes for this field: For further detail on Timer and Counter Data types, see the section entitled Timer and Counter Data Types following this table. 0 = Status File (2 bytes per element) 1 = Bit File (2 bytes per element) 2 = Timer File (6 bytes per element) 3 = Counter File (6 bytes per element) 4 = Control File (6 bytes per element) 5 = Integer File (2 bytes per element) 6 = Floating-point File (4 bytes per element) 10 = CIF File (1 byte per element)
File Number	This field defines the file number to access. This field is ignored for CIF file access and should be set to 0.
Element Number	This field defines the first element in the file specified to be associated with the command.

MAXIMUM ELEMENT COUNTS FOR READ/WRITE COMMANDS

File Type Code	File Type	Bytes/ Element	Max Read Elements	Max Write Elements
0	Status	2	83	83
1	Bit	2	118	115
2	Timer	6	39	38
3	Counter	6	39	38

MAXIMUM ELEMENT COUNTS FOR READ/WRITE COMMANDS				
4	Control	6	39	38
5	Integer	2	118	115
6	Float	4	59	57
10	CIF	1	236	234

Timer, Counter, and Control Data Types

Timer Data Type

The Timer data type uses its 3 words (6 bytes) as described below (e.g., T4:0):

Word 0:

Bit 9 – timebase selection

Bit 13 – T4:0.DN

Bit 14 – T4:0.TT

Bit 15 – T4:0.EN

Word 1:

T4:0.PRE

Word 2:

T4:0.ACC

Counter Data Type

The Counter data type uses its 3 words (6 bytes) as described below (e.g., C5:0)

Word 0:

Bit 10:UA

Bit 11:UN

Bit 12:OV

Bit 13:DN

Bit 14:CD

Bit 15:CV

Word 1: C5:0.PRE

Word 2: C5:0.ACC

Control Data Type

The Control Data Type uses its three (3) words (6 bytes) as shown in the following example. For example, R6:0.

Word 0:

Bit 8: FD

Bit 9: IN

Bit 10: UL

Bit 11: ER

Bit 12: EM

Bit 13: DN

Bit 14: EU

Bit 15: EN

Word 1: R6:0.LEN

Word 2: R6:.0.POS

The following is a form for setting up the command list:

DH485 COMMAND DEFINITION FORM

Enable Type Code	Database Start Address	Element Count	Swap Code	Poll Interval	Node Number	Function Code	File Type	File Number	Element Number

The following table lists the fields required for each record in the [DH485 Port x Maps] section:

Field	Definition
Database Start Address	This field defines the starting address in the module's database for the file emulation. This parameter can be assigned a value of 0 to 3999.
File Number	This field defines the file number to be emulated at the database location specified.
Element	This field specifies the first element in the file to be emulated. This element number corresponds to the database start address set for the record.
Word Count (Length)	This field defines the number of word registers to be emulated in the file.

The following is a form for setting up the override map list:

File Override Map Definition Form

Database Start Address	File Number	Element Number	Word Count (Length)

Appendix C – MVI56-DH485 Configuration File Example

This appendix contains an example DH485.CFG file listing:

```
#  DH485.CFG
#
# Example file for the DH485 driver. This file contains a configuration to
# support a peer node on a DH485 network. It will respond to request messages
# for both data file and CIF access. The configuration also supports a
# command list that will read and write data from another node.
#
#  DATE    : 02/03/2003
#  LOCATION: Test Bench
#  MODIFIED: RAR
#
#


#
# This section of the file describes the database setup and module level
# parameters.

[Module]
Module Name : MVI56-DH485 Communication Module Test

Error/Status Pointer : 3000

Read Register Start  :  400 #Starting DB address where read data stored
Read Register Count  :  400 #Number of regs to read from module by processor
Write Register Start :    0 #Starting DB address where write data accessed
Write Register Count :  400 #Number of regs to write to module from processor

Failure Flag Count   :    0 #Determines if BP failure will cause protocol to be
                            #disabled (0=Ignore, >0 = failure count to disable)

Initialize Output Data : No  #Read output values from controller (Yes or No)


# This section is used to define the communication characteristics of the
# first DH485 driver port (Port 0).

[DH485 Port 0]
Baud Rate              : 19200  #Baud rate of 9600 or 19200
```

```
Node Address            :     2  #My node address of 0 to 31
Maximum Node Address    :     2  #Maximum node address in network (1-31)
Token Hold Factor       :     1  #Value from 0 to 31 for this node to hold token
Response Timeout        :    10  #Response timeout in 100mSec increments
Status DB Offset        :  1000  #DB loc for port status data (-1=ignore)
Command Error DB Offset :  1100  #DB loc for cmd list error data (-1=ignore)
CIF Read DB Offset      :  4000  #DB byte loc <- CIF read data (-1=ignore)
CIF Read Count          :   100  #Numb of bytes to read from CIF data area (0-244)
CIF Write DB Offset     :  4100  #DB byte loc -> CIF write data (-1=ignore)
First File              :     7  #First file number to emulate in DB
File Size               :   200  #Number of words in each file emulated
File Offset             :     0  #DB start address for file emulation
Min Command Delay       :    50  #Min number of milliseconds between commands

# This section contains the list of commands to execute on the DH485 driver
# port.

[DH485 Port 0 Commands]
#
# Enable Types   : 0=Disable, 1=Poll at interval, 2=Conditional Poll
# Swap Codes     : 0=None, 1=Swap Words, 2=Swap bytes and words, 3=Swap bytes
# Function Codes : 0=Read, 1=Write
# FileType Codes : 0=status, 1=bit, 2=timer, 3=counter, 4=control, 5=int,
#                  6=float, 10=CIF
#
# Enbl        DB   Elem    Swap   Poll   Node   Func   File   File   Element
# Type   Address   Count   Code    Int      #   Code   Type      #    Number
START
     1       400      10      0      0      1      0      5      9         0
     1       400      10      0      0      1      1      5      9        10
     1       410      10      0      0      1      0     10      0        10
     1       410      10      0      0      1      1     10      0        20
END


# This section contains a list of file override values to overlay the database.
# The list will be searched by the driver for each data request made of the
# driver on the specified port. If the file:element value in the request
# matches an entry in the list, the database offset assigned to the entry
# will be used in conjunction with the other data in the entry to return
# or set the data.

[DH485 Port 0 Maps]
#  DB_Address    File_Number   Element    Length
START
```

```
             0             10         0       100
           100             12         0       100
END


# This section is used to define the communication characteristics of the
# second DH485 driver port (Port 1).

[DH485 Port 1]
Baud Rate               : 19200  #Baud rate of 9600 or 19200
Node Address            :     3  #My node address of 0 to 31
Maximum Node Address    :     5  #Maximum node address in network (1-31)
Token Hold Factor       :     1  #Value from 0 to 31 for this node to hold token
Response Timeout        :    10  #Response timeout in 100mSec increments
Status DB Offset        :  2000  #DB loc for port status data (-1=ignore)
Command Error DB Offset :  2100  #DB loc for cmd list error data (-1=ignore)
CIF Read DB Offset      :    -1  #DB byte loc <- CIF read data (-1=ignore)
CIF Read Count          :     0  #Numb of bytes to read from CIF data area (0-244)
CIF Write DB Offset     :    -1  #DB byte loc -> CIF write data (-1=ignore)
First File              :    10  #First file number to emulate in DB
File Size               :   100  #Number of words in each file emulated
File Offset             :     0  #DB start address for file emulation
Min Command Delay       :    50  #Min number of milliseconds between commands


# This section contains the list of commands to execute on the DH485 driver
# port.

[DH485 Port 1 Commands]
#
# Enable Types   : 0=Disable, 1=Poll at interval, 2=Conditional Poll
# Swap Codes     : 0=None, 1=Swap Words, 2=Swap bytes and words, 3=Swap bytes
# Function Codes : 0=Read, 1=Write
# FileType Codes : 0=status, 1=bit, 2=timer, 3=counter, 4=control, 5=int,
#                  6=float, 10=CIF
#
# Enbl        DB   Elem    Swap   Poll   Node   Func   File   File   Element
# Type   Address   Count   Code    Int      #   Code   Type      #    Number
START
END


# This section contains a list of file override values to overlay the database.
# The list will be searched by the driver for each data request made of the
# driver on the specified port. If the file:element value in the request
```

```
# matches an entry in the list, the database offset assigned to the entry
# will be used in conjunction with the other data in the entry to return
# or set the data.

[DH485 Port 1 Maps]
#  DB_Address    File_Number   Element    Length
START
END
```

Appendix D – Product Specifications

The MVI56-DH485 (DH485 Communication Module) allows Allen-Bradley ControlLogix I/O compatible processors to easily interface with other DH485 protocol compatible devices. Compatible devices include not only Allen-Bradley SLCs and PLCs (which support the DH485 protocol), but also a wide assortment of end devices.

General Specifications

The MVI56-DH485 module acts as a gateway between the DH485 network and the Allen-Bradley backplane. The data transfer from the ControlLogix processor is asynchronous from the actions on the DH485 network. A 4000-word register space in the module is used to exchange data between the processor and the DH485 network.

Some of the general specifications include:

- Support for the storage and transfer of up to 4000 registers to/from the ControlLogix processor's controller tags
- Module memory usage that is completely user definable
- Two ports to emulate a DH485 master/slave device (peer-to-peer device)
- All configuration information contained in a single, user defined, text file (DH485.CFG)

Functional Specifications

The MVI56-DH485 module accepts DH485 commands from an attached DH485 master unit (i.e., SLC 5/03 processor ladder logic MSG instruction). The DH485 driver permits a remote master to interact with all data contained in the module. This data can be derived from other DH485 devices on the network through a master port or from the ControlLogix processor. The MVI56-DH485 module driver actively issues DH485 commands to other nodes on the DH485 network. One hundred, user defined commands are supported by the driver on each port. The ControlLogix processor can be programmed to control the activity on the port by actively selecting command type for each command directly from the ladder logic.

Physical

This module is designed by ProSoft Technology and incorporates licensed technology from Allen-Bradley (ControlLogix backplane technology).

- ControLogix Form Factor - Single Slot
- Connections:

2– RJ45 connectors for support of RS-232 or RS-485 interfaces for the DH485 interface drivers

1– RJ45 RS-232 Configuration Tool Connector

ControlLogix Interface

- Operation via simple ladder logic
- Complete set up and monitoring of module through RSLogix 5000 software, user define configuration file and configuration/debug port
- ControlLogix backplane interface via I/O access
- All data related to the module is contained in a single controller tag with defined objects to ease in the monitoring and interfacing with the module

Hardware Specifications

The MVI56-DH485 module is designed by ProSoft Technology and incorporates licensed technology from Allen-Bradley (ControlLogix backplane technology).

- Current Loads: 800 mA @ 5V (from backplane)
- Operating Temperature: 0 to 60 Deg C (32 to 140 Deg F)
- Storage Temperature: -40 to 85 Deg C (-40 to 185 Def F)
- Relative Humidity: 5-95% (w/o condensation)
- DH485 Port Connectors: Two RJ45 Connectors (RJ45 to DB9 cables shipped with unit) supporting RS-232, RS-422 and RS-485 interfaces
- Configuration Connector: RJ45 RS-232 Connector (RJ45 to DB9 cable shipped with unit)

Support, Service, and Warranty

Technical Support

ProSoft Technology survives on its ability to provide meaningful support to its customers. Should any questions or problems arise, please feel free to contact us at:

Factory/Technical Support

ProSoft Technology, Inc.

1675 Chester Avenue, Second Floor

Bakersfield, CA 93301

(661) 716-5100

(661) 716-5101 (fax)

E-mail address: prosoft@prosoft-technology.com

Web Site: http://www.prosoft-technology.com

Before calling for support, please prepare yourself for the call. In order to provide the best and quickest support possible, we will most likely ask for the following information (you may wish to fax it to us prior to calling):

1. Product Version Number
2. System hierarchy
3. Module configuration and contents of DH485.CFG file
4. Module Operation
    - Configuration/Debug status information
    - LED patterns
5. Information about the processor and data areas as viewed through RSLogix 5000 and LED patterns on the processor
6. Details about the DH485 network

An after-hours answering system (on the Bakersfield number) allows pager access to one of our qualified technical and/or application support engineers at any time to answer the questions that are important to you.

Module Service and Repair

The MVI56-DH485 card is an electronic product, designed and manufactured to function under somewhat adverse conditions. As with any product, through age, misapplication, or any one of many possible problems the card may require repair.

When purchased from ProSoft Technology, the module has a one-year parts and labor warranty according to the limits specified in the warranty. Replacement and/or

returns should be directed to the distributor from whom the product was purchased. If you need to return the card for repair, obtain an RMA number from ProSoft Technology. Please call the factory for this number and display the number prominently on the outside of the shipping carton used to return the card.

General Warranty Policy

ProSoft Technology, Inc. (Hereinafter referred to as ProSoft) warrants that the Product shall conform to and perform in accordance with published technical specifications and the accompanying written materials, and shall be free of defects in materials and workmanship, for the period of time herein indicated, such warranty period commencing upon receipt of the Product.

This warranty is limited to the repair and/or replacement, at ProSoft's election, of defective or non-conforming Product, and ProSoft shall not be responsible for the failure of the Product to perform specified functions, or any other non-conformance caused by or attributable to: (a) any misapplication or misuse of the Product; (b) failure of Customer to adhere to any of ProSoft's specifications or instructions; (c) neglect of, abuse of, or accident to, the Product; or (d) any associated or complementary equipment or software not furnished by ProSoft.

Limited warranty service may be obtained by delivering the Product to ProSoft and providing proof of purchase or receipt date. Customer agrees to insure the Product or assume the risk of loss or damage in transit, to prepay shipping charges to ProSoft, and to use the original shipping container or equivalent. Contact ProSoft Customer Service for further information.

Limitation of Liability

EXCEPT AS EXPRESSLY PROVIDED HEREIN, PROSOFT MAKES NO WARRANT OF ANY KIND, EXPRESSED OR IMPLIED, WITH RESPECT TO ANY EQUIPMENT, PARTS OR SERVICES PROVIDED PURSUANT TO THIS AGREEMENT, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANT ABILITY AND FITNESS FOR A PARTICULAR PURPOSE. NEITHER PROSOFT OR ITS DEALER SHALL BE LIABLE FOR ANY OTHER DAMAGES, INCLUDING BUT NOT LIMITED TO DIRECT, INDIRECT, INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES, WHETHER IN AN ACTION IN CONTRACT OR TORT (INCLUDING NEGLIGENCE AND STRICT LIABILITY), SUCH AS, BUT NOT LIMITED TO, LOSS OF ANTICIPATED PROFITS OR BENEFITS RESULTING FROM, OR ARISING OUT OF, OR IN CONNECTION WITH THE USE OR FURNISHING OF EQUIPMENT, PARTS OR SERVICES HEREUNDER OR THE PERFORMANCE, USE OR INABILITY TO USE THE SAME, EVEN IF PROSOFT OR ITS DEALER'S TOTAL LIABILITY EXCEED THE PRICE PAID FOR THE PRODUCT.

Where directed by State Law, some of the above exclusions or limitations may not be applicable in some states. This warranty provides specific legal rights; other rights that vary from state to state may also exist. This warranty shall not be applicable to the extent that any provisions of this warranty are prohibited by any Federal, State or Municipal Law that cannot be preempted.

Hardware Product Warranty Details

Warranty Period: ProSoft warranties hardware product for a period of one (1) year.

Warranty Procedure: Upon return of the hardware Product ProSoft will, at its option, repair or replace Product at no additional charge, freight prepaid, except as set forth below. Repair parts and replacement Product will be furnished on an exchange basis and will be either reconditioned or new. All replaced Product and parts become the property of ProSoft. If ProSoft determines that the Product is not under warranty, it will, at the Customer's option, repair the Product using current ProSoft standard rates for parts and labor, and return the Product freight collect.

- - - - - - - - - - - - - - - - END OF MANUAL - - - - - - - - - - - - - - - -