Mitsubishi Ethernet Driver

Table of Contents

Mitsubishi Ethernet Driver

Help version 1.083

CONTENTS

Overview

The Mitsubishi Ethernet Driver provides a reliable way to connect Mitsubishi Ethernet devices to OPC client applications; including HMI, SCADA, Historian, MES, ERP, and countless custom applications. It is intended for use with Mitsubishi A Series and Mitsubishi Q Series devices communicating via the AJ71E71, A1SJ71E71, AJ71QE71, A1SJ71QE71 or QJ71E71 Ethernet communications cards. A built-in Ethernet Port is supported for Q Series devices. This driver also supports the FX3U series PLC via the FX3U-ENET Ethernet module.

Note: Communications card model numbers listed are the base model number only. All suffixes are supported.

Setup

Supported Devices

A Series PLCs
QnA Series PLCs
Q (Q mode) Series PLCs
FX3U Series PLCs

Communication Protocol

Ethernet: using Winsock V1.1 or higher
TCP/IP, UDP

Supported Communication Parameters

Binary format only

The maximum number of channels that are supported is 256. The maximum number of devices supported is 255 per channel.

Channel Properties

Device Properties

Channel Properties

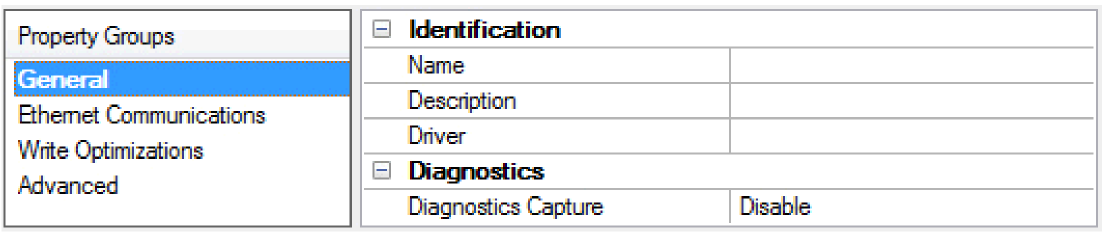
This server supports the use of simultaneous multiple communications drivers. Each protocol or driver used in a server project is called a channel. A server project may consist of many channels with the same communications driver or with unique communications drivers. A channel acts as the basic building block of an OPC link.

The properties associated with a channel are broken in to logical groupings. While some groups are specific to a given driver or protocol, the following are the common groups:

General
Ethernet or Serial Communications
Write Optimization
Advanced

Channel Properties - General

This server supports the use of simultaneous multiple communications drivers. Each protocol or driver used in a server project is called a channel. A server project may consist of many channels with the same communications driver or with unique communications drivers. A channel acts as the basic building block of an OPC link. This group is used to specify general channel properties, such as the identification attributes and operating mode.



Diagnostics Capture: Disable

Identification

Name: User-defined identity of this channel. In each server project, each channel name must be unique. Although names can be up to 256 characters, some client applications have a limited display window when browsing the OPC server's tag space. The channel name is part of the OPC browser information.
For information on reserved characters, refer to "How To... Properly Name a Channel, Device, Tag, and Tag Group" in the server help.

Description: User-defined information about this channel.
Many of these properties, including Description, have an associated system tag.

Driver: Selected protocol / driver for this channel. This property specifies the device driver that was selected during channel creation. It is a disabled setting in the channel properties.

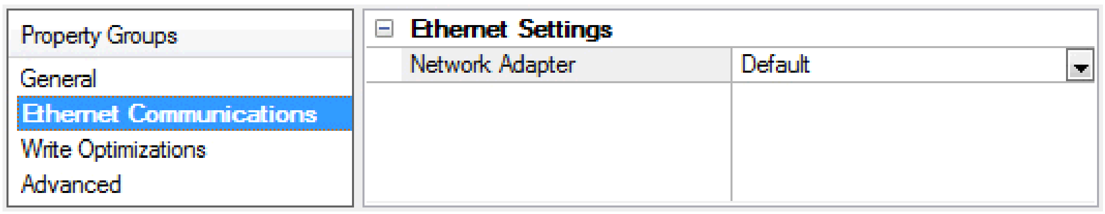
Note: With the server's online full-time operation, these properties can be changed at any time. This includes changing the channel name to prevent clients from registering data with the server. If a client has already acquired an item from the server before the channel name is changed, the items are unaffected. If, after the channel name has been changed, the client application releases the item and attempts to re-acquire using the old channel name, the item is not accepted. With this in mind, changes to the properties should not be made once a large client application has been developed. Utilize the User Manager to prevent operators from changing properties and restrict access rights to server features.

Diagnostics

Diagnostics Capture: When enabled, this option makes the channel's diagnostic information available to OPC applications. Because the server's diagnostic features require a minimal amount of overhead processing, it is recommended that they be utilized when needed and disabled when not. The default is disabled.
For more information, refer to "Communication Diagnostics" in the server help.
Not all drivers support diagnostics. To determine whether diagnostics are available for a particular driver, open the driver information and locate the "Supports device level diagnostics" statement.

Channel Properties - Ethernet Communications

Ethernet Communication can be used to communicate with devices.



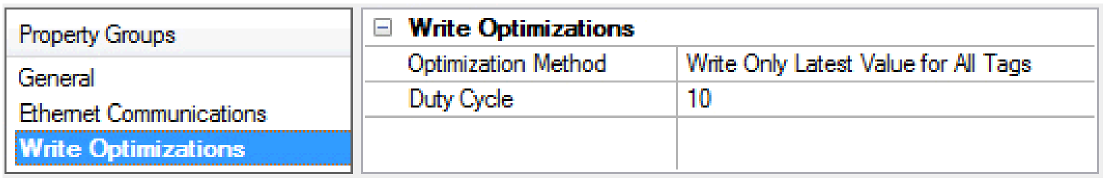
Network Adapter: Default

Ethernet Settings

Network Adapter: Specify the network adapter to bind. When Default is selected, the operating system selects the default adapter.

Channel Properties - Write Optimizations

As with any OPC server, writing data to the device may be the application's most important aspect. The server intends to ensure that the data written from the client application gets to the device on time. Given this goal, the server provides optimization properties that can be used to meet specific needs or improve application responsiveness.



Write Optimizations	
Optimization Method	Write Only Latest Value for All Tags
Duty Cycle	10

Write Optimizations

Optimization Method: controls how write data is passed to the underlying communications driver. The options are:

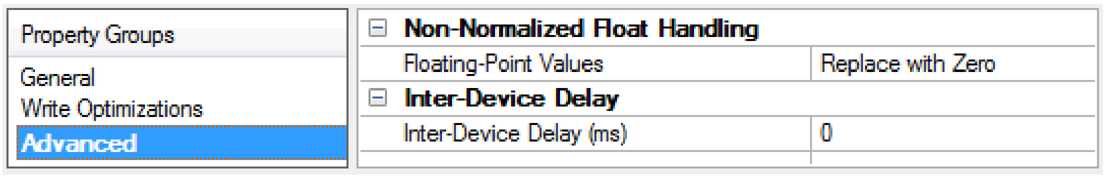
- **Write All Values for All Tags**: This option forces the server to attempt to write every value to the controller. In this mode, the server continues to gather write requests and add them to the server's internal write queue. The server processes the write queue and attempts to empty it by writing data to the device as quickly as possible. This mode ensures that everything written from the client applications is sent to the target device. This mode should be selected if the write operation order or the write item's content must uniquely be seen at the target device.
- **Write Only Latest Value for Non-Boolean Tags**: Many consecutive writes to the same value can accumulate in the write queue due to the time required to actually send the data to the device. If the server updates a write value that has already been placed in the write queue, far fewer writes are needed to reach the same final output value. In this way, no extra writes accumulate in the server's queue. When the user stops moving the slide switch, the value in the device is at the correct value at virtually the same time. As the mode states, any value that is not a Boolean value is updated in the server's internal write queue and sent to the device at the next possible opportunity. This can greatly improve the application performance.
  Note: This option does not attempt to optimize writes to Boolean values. It allows users to optimize the operation of HMI data without causing problems with Boolean operations, such as a momentary push button.
- **Write Only Latest Value for All Tags**: This option takes the theory behind the second optimization mode and applies it to all tags. It is especially useful if the application only needs to send the latest value to the device. This mode optimizes all writes by updating the tags currently in the write queue before they are sent. This is the default mode.

Duty Cycle: is used to control the ratio of write to read operations. The ratio is always based on one read for every one to ten writes. The duty cycle is set to ten by default, meaning that ten writes occur for each read operation. Although the application is performing a large number of continuous writes, it must be ensured that read data is still given time to process. A setting of one results in one read operation for every write operation. If there are no write operations to perform, reads are processed continuously. This allows optimization for applications with continuous writes versus a more balanced back and forth data flow.

Note: It is recommended that the application be characterized for compatibility with the write optimization enhancements before being used in a production environment.

Channel Properties - Advanced

This group is used to specify advanced channel properties. Not all drivers support all properties; so the Advanced group does not appear for those devices.



Non-Normalized Float Handling	
Floating-Point Values	Replace with Zero
Inter-Device Delay	
Inter-Device Delay (ms)	0

Non-Normalized Float Handling: Non-normalized float handling allows users to specify how a driver handles non-normalized IEEE-754 floating point data. A non-normalized value is defined as Infinity, Not-a-Number (NaN), or as a Denormalized Number. The default is Replace with Zero. Drivers that have native float handling may default to Unmodified. Descriptions of the options are as follows:

- **Replace with Zero**: This option allows a driver to replace non-normalized IEEE-754 floating point values with zero before being transferred to clients.
- **Unmodified**: This option allows a driver to transfer IEEE-754 denormalized, normalized, non-number, and infinity values to clients without any conversion or changes.

Note: This property is disabled if the driver does not support floating point values or if it only supports the option that is displayed. According to the channel's float normalization setting, only real-time driver tags (such as values and arrays) are subject to float normalization. For example, EFM data is not affected by this setting.lin

For more information on the floating point values, refer to "How To ... Work with Non-Normalized Floating Point Values" in the server help.

Inter-Device Delay: Specify the amount of time the communications channel waits to send new requests to the next device after data is received from the current device on the same channel. Zero (0) disables the delay.

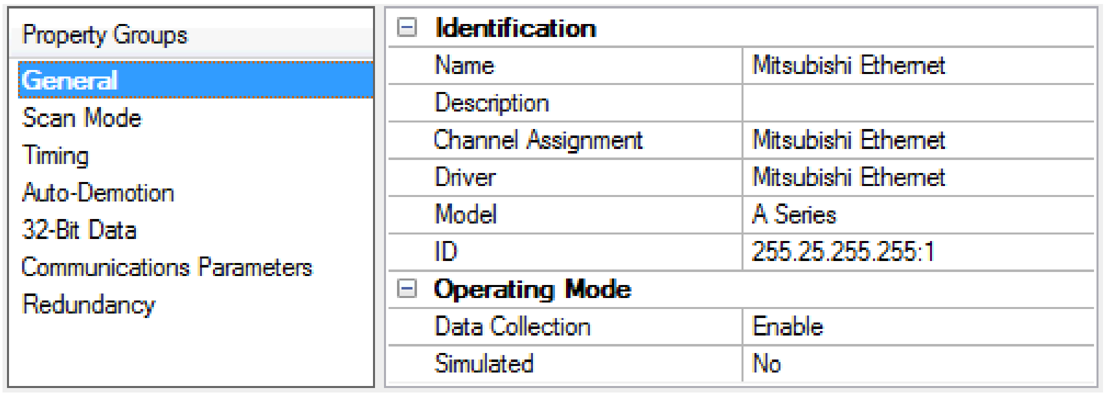
Note: This property is not available for all drivers, models, and dependent settings.

Driver Device Properties

Device properties are organized into the following groups. Click on a link below for details about the settings in that group.

General
Scan Mode
Communication Timeouts
Auto-Demotion
32-Bit Data
Communication Parameters
Time and Date Synchronization *(Q Series Only)*
Redundancy

Device Properties - General



Identification	
Name	Mitsubishi Ethernet
Description	
Channel Assignment	Mitsubishi Ethernet
Driver	Mitsubishi Ethernet
Model	A Series
ID	255.25.255.255:1
Operating Mode	
Data Collection	Enable
Simulated	No

Identification

Name: User-defined identity of this device.

Description: User-defined information about this device.

Channel Assignment: User-defined name of the channel to which this device currently belongs.

Driver: Selected protocol driver for this device.

Model: The specific version of the device. Options include A Series (for all A Series PLCs), Q Series (for all QnA and Q Series PLCs), and FX3U (for all FX3U Series PLCs).

ID (PLC Network Address): the unique device identity used to specify the IP address with a PC number (and net number if the device is a Q series PLC).

- **A Series:** Device IDs are specified as YYY.YYY.YYY.YYY:XXX. The YYY designates the device IP address (each YYY byte should be in the range of 0 to 255). The XXX designates the PC Number of the target device and can be in the range of 0 to 64 or 255 for the local PC.
- **Q Series:** Device IDs are specified as YYY.YYY.YYY.YYY:Nzzz:XXX or YYY.YYY.YYY.YYY:nzzz:XXX. The YYY designates the device IP address (each YYY byte should be in the range of 0 to 255). The zzz designates the Network Number of the target device and can be in the range of 0 to 255.
  Note: For a local connection, which is network 0, the network number can be omitted, resulting in the format YYY.YYY.YYY.YYY:XXX. The XXX designates the PC Number of the target device and can be in the range of 0 to 64 or 255 for the Local PC. *For more information, refer to **Multi-level Networks**.*
- **FX3U:** Device IDs are specified as YYY.YYY.YYY.YYY:XXX. The YYY designates the device IP address (each YYY byte should be in the range of 0 to 255). The XXX designates the PC Number of the target device and can be in the range of 0 to 15 or 255 for the local PC.

Note: The AJ71E71, A1SJ71E71, AJ71QE71, A1SJ71QE71, and QJ71E71 families of communications cards occupy ranges of X and Y memory. Writing to this memory with the Mitsubishi Ethernet Driver may disable the card and cause loss of communications. *For more information, refer to the communications card manual.*

Operating Mode

Data Collection: This property controls the device's active state. Although device communications are enabled by default, this property can be used to disable a physical device. Communications are not attempted when a device is disabled. From a client standpoint, the data is marked as invalid and write operations are not accepted. This property can be changed at any time through this property or the device system tags.

Simulated: This option places the device into Simulation Mode. In this mode, the driver does not attempt to communicate with the physical device, but the server continues to return valid OPC data. Simulated stops physical communications with the device, but allows OPC data to be returned to the OPC client as valid data. While in Simulation Mode, the server treats all device data as reflective: whatever is written to the simulated device is read back and each OPC item is treated individually. The item's memory map is based on the group Update Rate. The data is not saved if the server removes the item (such as when the server is reinitialized). The default is No.

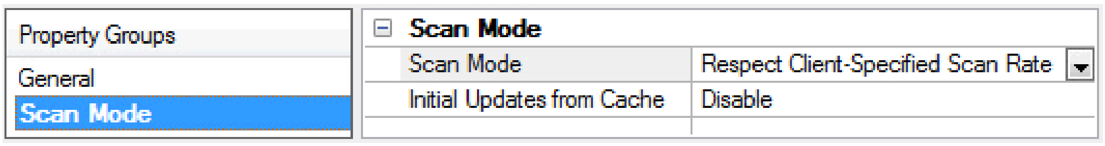
Notes:

1. This System tag (_Simulated) is read only and cannot be written to for runtime protection. The System tag allows this property to be monitored from the client.
2. In Simulation mode, the item's memory map is based on client update rate(s) (Group Update Rate for OPC clients or Scan Rate for native and DDE interfaces). This means that two clients that reference the same item with different update rates return different data.

Simulation Mode is for test and simulation purposes only. It should never be used in a production environment.

Device Properties - Scan Mode

The Scan Mode specifies the subscribed-client requested scan rate for tags that require device communications. Synchronous and asynchronous device reads and writes are processed as soon as possible; unaffected by the Scan Mode properties.



Scan Mode	
Scan Mode	Respect Client-Specified Scan Rate
Initial Updates from Cache	Disable

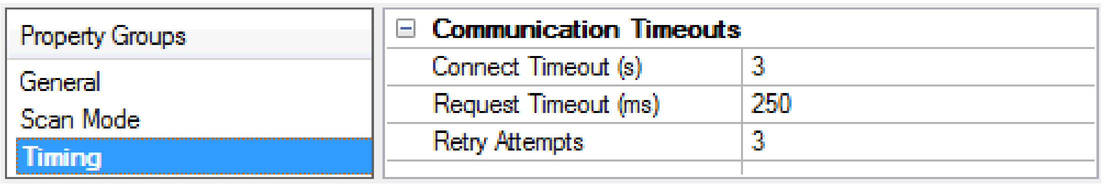
Scan Mode: specifies how tags in the device are scanned for updates sent to subscribed clients. Descriptions of the options are:

- **Respect Client-Specified Scan Rate**: This mode uses the scan rate requested by the client.
- **Request Data No Faster than Scan Rate**: This mode specifies the maximum scan rate to be used. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.
  Note: When the server has an active client and items for the device and the scan rate value is increased, the changes take effect immediately. When the scan rate value is decreased, the changes do not take effect until all client applications have been disconnected.
- **Request All Data at Scan Rate**: This mode forces tags to be scanned at the specified rate for subscribed clients. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.
- **Do Not Scan, Demand Poll Only**: This mode does not periodically poll tags that belong to the device nor perform a read to get an item's initial value once it becomes active. It is the client's responsibility to poll for updates, either by writing to the _DemandPoll tag or by issuing explicit device reads for individual items. *For more information, refer to "Device Demand Poll" in server help*.
- **Respect Tag-Specified Scan Rate**: This mode forces static tags to be scanned at the rate specified in their static configuration tag properties. Dynamic tags are scanned at the client-specified scan rate.

Initial Updates from Cache: When enabled, this option allows the server to provide the first updates for newly activated tag references from stored (cached) data. Cache updates can only be provided when the new item reference shares the same address, scan rate, data type, client access, and scaling properties. A device read is used for the initial update for the first client reference only. The default is disabled; any time a client activates a tag reference the server attempts to read the initial value from the device.

Device Properties - Timing

The device Communications Timeouts properties allow the driver's response to error conditions to be tailored to fit the application's needs. In many cases, the environment requires changes to these properties for optimum performance. Communications Timeouts properties are specific to each configured device.



Communication Timeouts	
Connect Timeout (s)	3
Request Timeout (ms)	250
Retry Attempts	3

Communications Timeouts

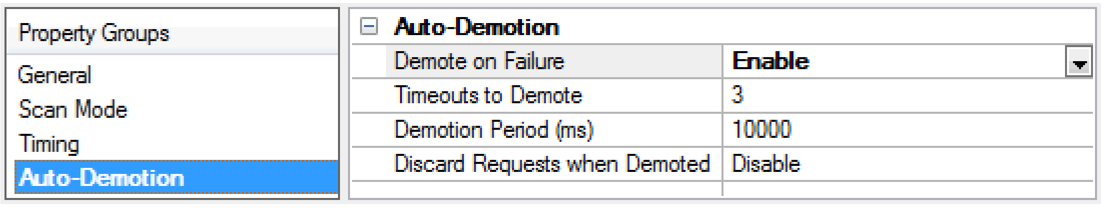
Connect Timeout: Specify the amount of time required to establish a socket connection to a remote device. The device's connection time often takes longer than normal communications requests to that same device. The valid range is 1 to 60 seconds. The default is typically 3 seconds.

Request Timeout: Specify how long the driver waits for a response from the target device to complete before moving on to the next request. The valid range is 50 to 30,000 milliseconds. The default is 1000 milliseconds.

Retry Attempts: Specify how many times the driver retries a communications request before considering the request to have failed and the device to be in error. The valid range is 1 to 10. The default is 3.

Device Properties - Auto-Demotion

The Auto-Demotion properties can temporarily place a device off-scan in the event that a device is not responding. By placing a non-responsive device offline for a specific time period, the driver can continue to optimize its communications with other devices on the same channel. After the time period has been reached, the driver re-attempts to communicate with the non-responsive device. If the device is responsive, the device is placed on-scan; otherwise, it restarts its off-scan time period.



Auto-Demotion	
Demote on Failure	Enable
Timeouts to Demote	3
Demotion Period (ms)	10000
Discard Requests when Demoted	Disable

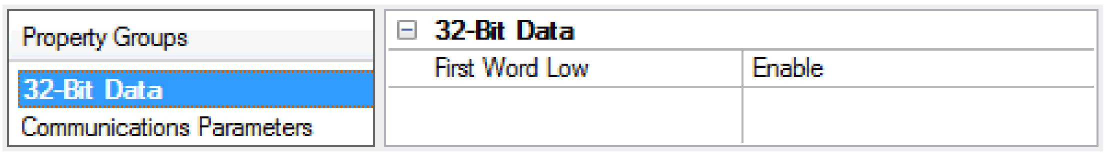
Demote on Failure: When enabled, the device is automatically taken off-scan until it is responding again.
Tip: Determine when a device is off-scan by monitoring its demoted state using the _AutoDemoted system tag.

Timeouts to Demote: Specify how many successive cycles of request timeouts and retries occur before the device is placed off-scan. The valid range is 1 to 30 successive failures. The default is 3.

Demotion Period: Indicate how long the device should be placed off-scan when the timeouts value is reached. During this period, no read requests are sent to the device and all data associated with the read requests are set to bad quality. When this period expires, the driver places the device on-scan and allows for another attempt at communications. The valid range is 100 to 3600000 milliseconds. The default is 10000 milliseconds.

Discard Requests when Demoted: Select whether or not write requests should be attempted during the off-scan period. Disable to always send write requests regardless of the demotion period. Enable to discard writes; the server automatically fails any write request received from a client and does not post a message to the Event Log.

Device Properties - 32-Bit Data

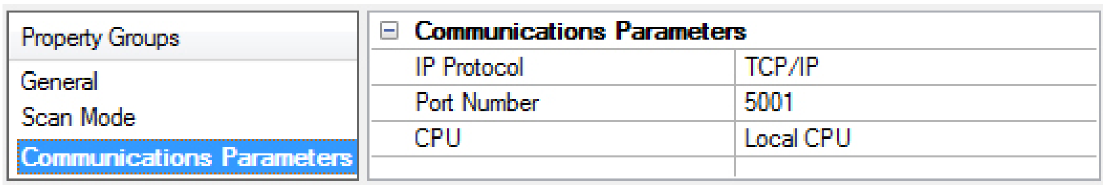


32-Bit Data	
First Word Low	Enable

First Word Low: In a Mitsubishi Ethernet device, the addresses of two consecutive registers are used for 32-bit data types. When this option is enabled, the driver assumes the first word is low for the 32-bit value. When this option is disabled, the driver assumes that the first word is high for the 32-bit value. The default setting is enabled.

Note: This property can't be changed while there are active references on the device.

Device Properties - Communications Parameters



Communications Parameters	
IP Protocol	TCP/IP
Port Number	5001
CPU	Local CPU

IP Protocol: Select the IP protocol: TCP/IP or UDP. TCP/IP is less efficient than UDP and requires a special ladder for network error recovery in the A and QnA series PLCs. Furthermore, Q series users planning to communicate with devices on a remote network must configure multiple ports in the relay device when using TCP/IP. As such, UDP is recommended wherever possible.
For more information, refer to ***Multi-level Networks***.

Port Number: Specify the port number to use when communicating with the device. The default for UDP is 5000. The default for TCP is 5001.

Notes:

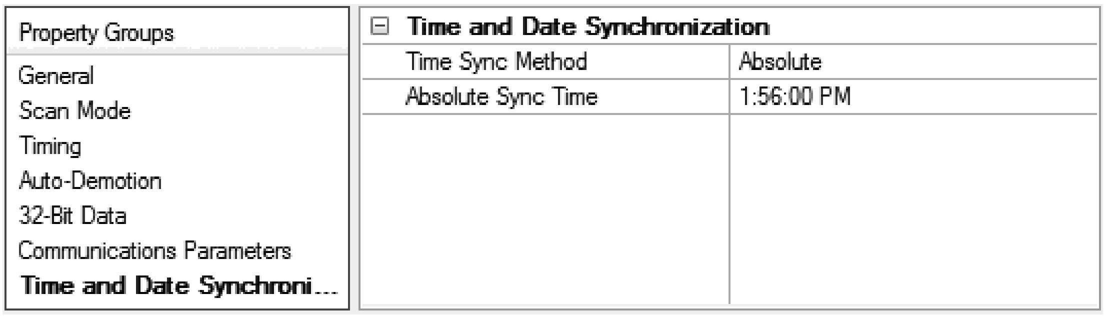
1. The default settings are based on GX Developer version 8.25B.

2. This property can't be changed while there are active references to the device.

CPU: Choose the CPU for communications with the target. For a single CPU system, select Local. The default is Local.

Device Properties - Time and Date Synchronization

Time and Date Synchronization properties are only available to the Q Series Model PLCs.



Property Groups	Time and Date Synchronization	
General	Time Sync Method	Absolute
Scan Mode	Absolute Sync Time	1:56:00 PM
Timing		
Auto-Demotion		
32-Bit Data		
Communications Parameters		
Time and Date Synchroni...		

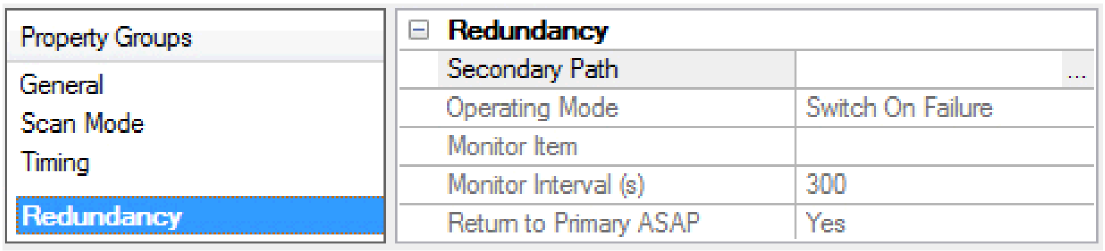
Time Sync Method: Choose the synchronization method to define how the time and date are reconciled between the host system and the device. Options include Disabled, Interval, and Absolute. The default setting is Disabled.

Absolute Sync Time: Specify an exact hour and minute of each day to synchronize time between the server and the device when the synchronization method is Absolute. The default value is set to the local PC time when the device was created. Only the hour and minute of the day are used to determine if synchronization is required. The seconds are ignored. As an example, if this property displays as 3:52:00 PM, time synchronization will occur each day at 57120 seconds after midnight.

Sync Interval: Specify the time, in minutes, between synchronizations - how often time and date reconciliation should occur when the synchronization method is Interval. The driver can periodically synchronize the time and date of the PLC with the time and date of the host computer. The valid range is 5 to 1440 minutes (24 hours). The default setting is 5 minutes.

Note: For example, if 240 minutes is entered, the driver sets the PLC date and time every 4 hours.

Device Properties - Redundancy



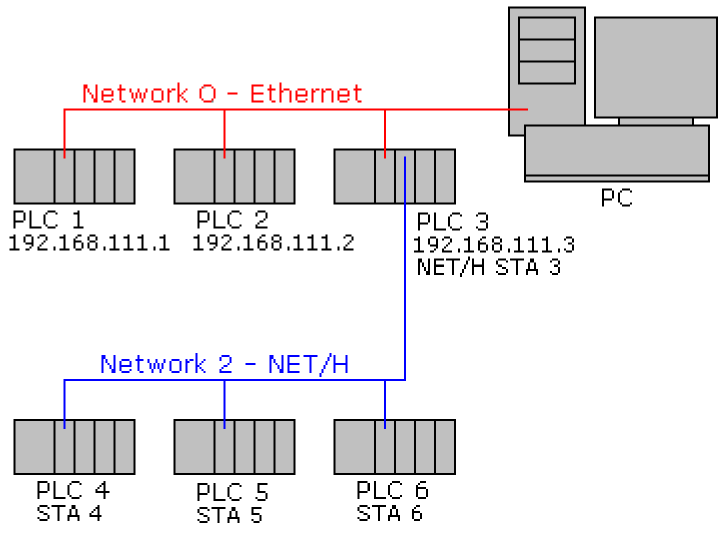
Property Groups	Redundancy	
General	Secondary Path	...
Scan Mode	Operating Mode	Switch On Failure
Timing	Monitor Item	
Redundancy	Monitor Interval (s)	300
	Return to Primary ASAP	Yes

Redundancy is available with the Media-Level Redundancy Plug-in.

Consult the website, a sales representative, or the user manual for more information.

Multi-Level Networks

The Q Series model is used to communicate with devices on remote networks. In the example shown below, PLC 1, PLC 2 and PLC 3 are on the local Ethernet network (Network 0). PLC 4, PLC 5 and PLC 6 are on a remote NET/H network. PLC 3 serves as a relay device connecting the two networks.



If PLC 1, PLC 2 and PLC 3 have QJ71E71-100 Ethernet modules configured with IPs 192.168.111.1, 192.168.111.2 and 192.168.111.3 respectively. In addition to the Ethernet module, PLC 3 also has a QJ71BR11 NET/H module configured as station 3. Assume that PLC 4, PLC 5 and PLC 6 have NET/H modules configured as stations 4, 5 and 6 respectively.

To communicate with all six PLCs, six devices would need to be created in the server project. The Device IDs would be as follows:

PLC	Device ID	Comment
1	192.168.111.1:N0:255*	Local network, local PC
2	192.168.111.2:N0:255*	Local network, local PC
3	192.168.111.3:N0:255*	Local network, local PC
4	192.168.111.3:N2:4	Network 2, PC 4, via PLC 3
5	192.168.111.3:N2:5	Network 2, PC 5, via PLC 3
6	192.168.111.3:N2:6	Network 2, PC 6, via PLC 3

*This example shows :N0 as the network number for the local network. It is also possible to omit the network number when it is Network 0 (local network), thus, the Device ID 192.168.111.1:255 would also be valid in this case.

Notes:

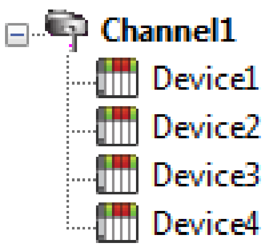
1. For performance and reliability, the driver is designed to use a separate socket for each device. Thus, if TCP/IP is used, the relay device in this example would need to have at least 4 ports configured - one to connect to each of the driver's sockets for PLC 3, PLC 4, PLC 5 and PLC 6. However, only a single port needs to be configured in the relay device if UDP and the "unspecified" destination IP (255.255.255.255) and port number (0xFFFF) are being used. Therefore, UDP is generally recommended for this type of application. For more information, refer to **PLC Setup**.

2. A relay device may take 5 or more seconds to report a failed read and write to a remote device. It is recommended that the request timeout for remote devices be set accordingly. For more information, refer to **Device Setup**.

Optimizing Communications

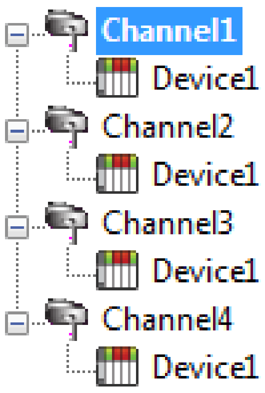
The Mitsubishi Ethernet Driver has been designed to provide the best performance with the least amount of impact on the system's overall performance. While the Mitsubishi Ethernet Driver is fast, there are a couple of guidelines that can be used to control and optimize the application and gain maximum performance.

This server refers to communications protocols like Mitsubishi Ethernet Device as a channel. Each channel defined in the application represents a separate path of execution in the server. Once a channel has been defined, a series of devices must then be defined under that channel. Each of these devices represents a single Mitsubishi Ethernet device from which data should be collected. While this approach to defining the application provides a high level of performance, it doesn't take full advantage of the Mitsubishi Ethernet Driver or the network. An example of how the application may appear when configured using a single channel is shown below.



Each device appears under a single Mitsubishi Ethernet device channel. In this configuration, the driver must move from one device to the next as quickly as possible in order to gather information at an effective rate. As more devices are added or more information is requested from a single device, the overall update rate begins to suffer.

If the Mitsubishi Ethernet Driver could only define one single channel, then the example shown above would be the only option available; however, the driver can define up to 256 channels. Using multiple channels distributes the data collection workload by simultaneously issuing multiple requests to the network. An example of how the same application may appear when configured using multiple channels to improve performance is shown below.



Each device has now been defined under its own channel. In this new configuration, a single path of execution is dedicated to the task of gathering data from each device. If the application has 256 or fewer devices, it can be optimized exactly how it is shown here.

The performance improves even if the application has more than 256 devices. While 256 or fewer devices may be ideal, the application still benefits from additional channels. Although by spreading the device load across all channels causes the server to move from device to device again, it can do so with far less devices to process on a single channel.

● **Tip**: An additional performance gain can be achieved by using UDP instead of TCP/IP. *For more information, refer to* ***Device Setup*** *and* ***PLC Setup****.*

Data Types Description

The Mitsubishi Ethernet Driver supports the following data types.

Data Type	Description
Boolean	Single bit
Word	Unsigned 16-bit value bit 0 is the low bit bit 15 is the high bit
Short	Signed 16-bit value bit 0 is the low bit bit 14 is the high bit bit 15 is the sign bit
DWord	Unsigned 32-bit value Bit 0 is the low bit. Bit 31 is the high bit.
Long	Signed 32-bit value Bit 0 is the low bit. Bit 30 is the high bit. Bit 31 is the sign bit.
Float	32-bit floating point value
String	Null-terminated ASCII string support includes HiLo and LoHi byte order selection and string lengths up to 128 bytes.
BCD	Two byte packed BCD Value range is 0-9999. Behavior is undefined for values beyond this range.
LBCD	Four byte packed BCD Value range is 0-99999999. Behavior is undefined for values beyond this range.
Date	32-bit value
Date Example	Date format: YYYY-MM-DDTHH:MM:SS.000 2000-01-01T12:30:45.000
Double*	64-bit floating point value The driver interprets four consecutive registers as a Double precision value by making the first two registers the low DWord and the last two registers the high DWord.
Double Example*	If register D0000000 is specified as a Double, bit 0 of register D0000000 would be bit 0 of the 64-bit data type. Bit 15 of register D0000003 would be bit 63 of the 64-bit data type.

*The descriptions above assume the default first word low data handling of 32-bit data types.

Address Descriptions

Address specifications vary depending on the model in use. Select a link from the following list to obtain specific address information for the model of interest.

A Series
Q Series
FX3U Series

Mitsubishi A Series Address Descriptions

The default data types for dynamically defined tags are shown in **bold**.

Device Type	Range	Data Type	Access
Inputs*	X000-X1FFF (Hex) X000-X1FF0 (Hex) X000-X1FE0 (Hex)	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Outputs*	Y000-Y1FFF (Hex) Y000-Y1FF0 (Hex) Y000-Y1FE0 (Hex)	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Link Relays*	B000-B1FFF (Hex) B000-B1FF0 (Hex) B000-B1FE0 (Hex)	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Internal Relays*	M0000-M8191 M0000-M8176 M0000-M8160	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Special Int. Relays*	M9000-M9255 M9000-M9240 M9000-M9224	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read Only
Latch Relays*	L0000-L8191 L0000-L8176 L0000-L8160	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Annunciator Relays*	F0000-F2047 F0000-F2032 F0000-F2016	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Timer Contacts*	TS0000-TS2047 TS0000-TS2032 TS0000-TS2016	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Timer Coils*	TC0000-TC2047 TC0000-TC2032 TC0000-TC2016	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Counter Contacts*	CS0000-CS1023 CS0000-CS1008 CS0000-CS0992	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Counter Coils*	CC0000-CC1023 CC0000-CC1008 CC0000-CC0992	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write

*Users can specify a Long data type by appending a space and an "L" to the address. For example, "CS0000" would be entered as "CS0000 L". This does not apply to arrays or bit accessed registers.

Note: All Boolean device types can be accessed as Short, Word, BCD, Long, DWord and LBCD; however, the device must be addressed on a 16-bit boundary.

Device Type	Range	Data Type	Access
Timer Value	TN0000-TN2047	**Short**, Word, BCD	Read/Write
Counter Value	CN0000-CN1023	Short, **Word**, BCD	Read/Write
Data Registers***	D0000-D8191 D0000-D8190 D0000-D8188	**Short**, Word, BCD Long, DWord, LBCD, Float, Date Double	Read/Write
Data Register Bit Access	D0000.00-D8191.15* D0000.00-D8190.31*	**Short**, Word, BCD, Boolean Long, DWord, LBCD	Read/Write
Data Registers String Access HiLo Byte Ordering	DSH00000.002-DSH08190.002 DSH00000.128-DSH08127.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write
Data Registers String Access LoHi Byte Ordering	DSL00000.002–DSL08190.002 DSL00000.128-DSL08127.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write
Special Data Registers***	D9000-D9255 D9000-D9254 D9000-D9252	**Short**, Word, BCD Long, DWord, LBCD, Float, Date Double	Read Only
Data Register Bit Access	D9000.00-D9255.15* D9000.00-D9254.31*	**Short**, Word, BCD, Boolean** Long, DWord, LBCD	Read Only
Link Registers***	W0000-W1FFF (Hex) W0000-W1FFE (Hex) W0000-W1FFC (Hex)	**Short**, Word, BCD Long, DWord, LBCD, Float, Date Double	Read/Write
Link Register Bit Access	W0000.00-W1FFF.15* W0000.00-W1FFE.31*	**Short**, Word, BCD, Boolean** Long, DWord, LBCD	Read/Write
Link Registers String Access HiLo Byte Ordering	WSH0000.002-WSH1FFE.002 WSH0000.128-WSH1FBF.128 The string length may also be specified using a	**String**	Read/Write

Device Type	Range	Data Type	Access
	colon. The string length must be between 2-128 bytes and even.		
Link Registers String Access LoHi Byte Ordering	WSL0000.002-WSL1FFE.002 WSL0000.128-WSL1FBF.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write
File Register***	R0000-R8191 R0000-R8190 R0000-R8188	**Short**, Word, BCD Long, DWord, LBCD, Float, Date Double	Read/Write
File Register Bit Access	R0000.00-R8191.15* R0000.00-R8190.31*	**Short**, Word, BCD, Boolean** Long, DWord, LBCD	Read/Write
File Registers String Access HiLo Byte Ordering	RSH00000.002-RSH08190.002 RSH00000.128-RSH08127.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write
File Registers String Access LoHi Byte Ordering	RSL00000.002-RSL08190.002 RSL00000.128-RSL08127.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write

*For register memory, the data types Short, Word, BCD, DWord, Long, LBCD and Boolean may append an optional ".bb" (dot bit) or ":bb" (colon bit) to the address in order to reference a bit in a particular value. The valid ranges for the optional bit are 0-15 for Short, Word, BCD, Boolean; and 0-31 for Long, DWord and LBCD. Strings use the bit number to specify length. The valid length of a string in D memory is 2 to 128 bytes. The string length must be an even number. Float types do not support bit operations. The bit number is always in decimal notation.
**When accessing register memory as Boolean, a bit number is required.
***Users can specify a Long data type by appending a space and an "L" to the address. For example, "CS0000" would be entered as "CS0000 L". This does not apply to arrays or bit accessed registers.

Array Access

All device types can be accessed as arrays. The default array tag for all device types is Word. The size of the array depends on both the data type and the device type. All Register device types can access up to the following: 254 elements for Short, Word and BCD; 127 elements for Long, DWord, LBCD and Float; and 63 elements for Double. All Bit memory types can access up to the following: 127 elements for Short, Word and BCD; and 63 elements for Long, DWord and LBCD. Arrays may be 1 or 2 dimensions, but the array size may not exceed the limits stated above.

Note: An array is created when array notation is appended onto a normal device reference.

Examples

1. D100 [4] Single dimension includes the following register addresses: D100, D101, D102, D103.
2. M016 [3][4] Two Dimensions includes the following device addresses as words: M016, M032, M048, M064, M080, M096, M112, M128, M144, M160, M176, M192 3 rows x 4 columns = 12 words 12 x 16 (word) =192 total bits.

Additional Device Examples

1. Access X device memory as Word: X??? where the ??? is a hex number on 16-bit boundaries such as 010, 020, 030, and so forth.
2. Access M device memory as Long: M???? where the ???? is a decimal number on 16-bit boundaries such as 0, 16, 32, 48, and so forth.

Mitsubishi Q Series Address Descriptions

The default data types for dynamically defined tags are shown in **bold**.

Device Type	Range	Data Type	Access
Inputs*	X0000-X3FFF (Hex) X0000-X3FF0 (Hex) X0000-X3FE0 (Hex)	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Direct Inputs*	DX0000-DX3FFF (Hex) DX0000-DX3FF0 (Hex) DX0000-DX3FE0 (Hex)	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Outputs*	Y0000-Y3FFF (Hex) Y0000-Y3FF0 (Hex) Y0000-Y3FE0 (Hex)	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Direct Outputs*	DY0000-DY3FFF (Hex) DY0000-DY3FF0 (Hex) DY0000-DY3FE0 (Hex)	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Link Relays*	B0000-BEA60 (Hex) B0000-BEA50 (Hex) B0000-BEA40 (Hex)	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Special Link Relays*	SB0000-SB7D00 (Hex) SB0000-SB7CF0 (Hex) SB0000-SB7CE0 (Hex)	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Internal Relays*	M0000-M60000 M0000-M59984 M0000-M59968	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write

Device Type	Range	Data Type	Access
Special Int. Relays*	SM0000-SM2047 SM0000-SM2032 SM0000-SM2016	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Latch Relays*	L0000-L32000 L0000-L31984 L0000-L31968	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Annunciator Relays*	F0000-F32000 F0000-F31984 F0000-F31968	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Edge Relays*	V0000-V32000 V0000-V31984 V0000-V31968	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Step Relays*	S0000-S16383 S0000-S16368 S0000-S16352	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Timer Contacts*	TS0000-TS32000 TS0000-TS31984 TS0000-TS31968	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Timer Coils*	TC0000-TC32000 TC0000-TC31984 TC0000-TC31968	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Integrating Timer Contacts*	SS0000-SS2047 SS0000-SS2032 SS0000-SS2016	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Integrating Timer Coils*	SC0000-SC2047 SC0000-SC2032 SC0000-SC2016	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Counter Contacts*	CS0000-CS32000 CS0000-CS31984 CS0000-CS31968	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write
Counter Coils*	CC0000-CC32000 CC0000-CC31984 CC0000-CC31968	**Boolean**, Short, Word, BCD, Long, DWord, LBCD	Read/Write

*Users can specify a Long data type by appending a space and an "L" to the address. For example, "CS0000" would be entered as "CS0000 L". This does not apply to arrays or bit accessed registers.

Note: All Boolean device types can be accessed as Short, Word, BCD, Long, DWord and LBCD; however, the device must be addressed on a 16-bit boundary.

Device Type	Range	Data Type	Access
Timer Value	TN0000-TN32000	**Short**, Word, BCD	Read/Write
Integrating Timer Value	SN0000-SN2047	**Short**, Word, BCD	Read/Write
Counter Value	CN0000-CN32000	Short, **Word**, **BCD**	Read/Write

Device Type	Range	Data Type	Access
Data Registers***	D0000000-D4184063 D0000000-D4184062 D0000000-D4184060 See Also: Extended Registers	**Short**, Word, BCD, Long, DWord, LBCD, Float, Date, Double	Read/Write
Data Register Bit Access	D0000000.00–D4184063.15* D0000000.00–D4184062.31* See Also: Extended Registers	**Short**, Word, BCD, Boolean**, Long, DWord, LBCD	Read/Write
Data Registers String Access HiLo Byte Ordering	DSH00000.002-DSH4184062.002 DSH00000.128-DSH4183999.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write
Data Registers String Access LoHi Byte Ordering	DSL00000.002-DSL4184062.002 DSL00000.128-DSL4183999.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write
Special Data Registers***	SD0000-SD2047 SD0000-SD2046 SD0000-SD2044	**Short**, Word, BCD, Long, DWord, LBCD, Float, Date, Double	Read/Write
Data Register Bit Access	SD0000.00-SD2047.15* SD0000.00-SD2046.31*	**Short**, Word, BCD, Boolean**, Long, DWord, LBCD	Read/Write
Link Registers***	W0000-W3FD7FF (Hex) W0000-W3FD7FE (Hex) W0000-W3FD7FC (Hex) See Also: Extended Registers	**Short**, Word, BCD, Long, DWord, LBCD, Float, Date, Double	Read/Write
Link Register Bit Access	W0000.00-W3FD7FF.15* W0000.00-W3FD7FE.31* See Also: Extended Registers	**Short**, Word, BCD, Boolean**, Long, DWord, LBCD	Read/Write
Link Registers String Access HiLo Byte Ordering	WSH0000.002-WSH3FD7FE.002 WSH0000.128-WSH3FD7BF.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write
Link Registers String Access	WSL0000.002-WSL3FD7FE.002 WSL0000.128-WSL3FD7BF.128	**String**	Read/Write

Device Type	Range	Data Type	Access
LoHi Byte Ordering	The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.		

Special Link Registers***	SW0000-SW7D00(Hex)
	SW0000-SW7CFF(Hex)
	SW0000-SW7CFD (Hex)
		Short, Word, BCD, Long, DWord, LBCD, Float, Date, Double	Read/Write

Link Register Bit Access	SW0000.00-SW7D00.15* SW0000.00-SW7CFF.31*	**Short**, Word, BCD, Boolean**, Long, DWord, LBCD	Read/Write
File Register***	R00000-R32767 R00000-R32766 R00000-R32764 ZR0000-ZR3FD7FF (Hex) ZR0000-ZR3FD7FE (Hex) ZR0000-ZR3FD7FC (Hex)	**Short**, Word, BCD, Long, DWord, LBCD, Float, Date, Double **Short**, Word, BCD Long, DWord, LBCD, Float, Date Double	Read/Write
File Register Bit Access	R00000.00-R32767.15* R00000.00-R32766.31* ZR0000.00-ZR3FD7FF.15* ZR0000.00-ZR3FD7FE.31*	**Short**, Word, BCD, Boolean** Long, DWord, LBCD **Short**, Word, BCD, Boolean** Long, DWord, LBCD	Read/Write
File Registers String Access HiLo Byte Ordering	RSH00000.002-RSH32766.002 RSH00000.128-RSH32703.128 ZRSH0000.002-ZRSH3FD7FE.002 ZRSH0000.128-ZRSH3FD7BF.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String** **String**	Read/Write
File Registers String Access LoHi Byte Ordering	RSL00000.002-RSL32766.002 RSL00000.128-RSL32703.128 ZRSL0000.002-ZRSL3FD7FE.002 ZRSL0000.128-ZRSL3FD7BF.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String** **String**	Read/Write
Index Registers***	Z00-Z20 Z00-Z19	**Short**, Word, BCD Long, DWord, LBCD, Float,	Read/Write

Device Type	Range	Data Type	Access
	Z00-Z17	Date Double	
Index Register Bit Access	Z00.00-Z20.15* Z00.00-Z19.31*	**Short**, Word, BCD, Boolean** Long, DWord, LBCD	Read/Write

*For register memory, the data types Short, Word, BCD, DWord, Long, LBCD and Boolean may append an optional ".bb" (dot bit) or ":bb" (colon bit) to the address in order to reference a bit in a particular value. The valid ranges for the optional bit are 0-15 for Short, Word, BCD and Boolean; and 0-31 for Long, DWord and LBCD. Strings use the bit number to specify length. The valid length of a string in D memory is 2 to 128 bytes. The string length must be an even number. Float types do not support bit operations. The bit number is always in decimal notation.
**When accessing register memory as Boolean, a bit number is required.
***Users can specify a Long data type by appending a space and an "L" to the address. For example, "CS0000" would be entered as "CS0000 L". This does not apply to arrays or bit accessed registers.

Extended Registers

The extended range for Data Registers is D12288 to D4184063. The extended range for Link Registers is W3FFF (Hex) to W3FD7FF (Hex). These must be configured on the device.

Array Access

All device types can be accessed as arrays. The default array tag for all device types is Word. The size of the array depends on both the data type and the device type. All Register device types can access up to the following: 254 elements for Short, Word and BCD; 127 elements for Long, DWord, LBCD and Float; and 63 elements for Double. All Bit memory types can access up to the following: 127 elements for Short, Word and BCD; and 63 elements for Long, DWord and LBCD. Arrays may be 1 or 2 dimensions, but the array size may not exceed the limits stated above.

Note: An array is created when array notation is appended onto a normal device reference.

Examples:

1. D100 [4] Single dimension includes the following register addresses: D100, D101, D102, D103.
2. M016 [3][4] Two Dimensions includes the following device addresses as words: M016, M032, M048, M064, M080, M096, M112, M128, M144, M160, M176, M192 3 rows x 4 columns =12 words 12 x 16 (word) =192 total bits.

Additional Device Examples

1. Access X device memory as Word: X??? where the ??? is a hex number on 16-bit boundaries such as 010, 020, 030, and so forth.
2. Access M device memory as Long: M???? where the ???? is a decimal number on 16-bit boundaries such as 0, 16, 32, 48, and so forth.

Mitsubishi FX3U Series Address Descriptions

The default data types for dynamically defined tags are shown in **bold**.

Device Type	Range	Data Type	Access
Inputs*	X000-X377 (Oct) X000-X360 (Oct) X000-X340 (Oct)	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Outputs*	Y000-Y377 (Oct) Y000-Y360 (Oct) Y000-Y340 (Oct)	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Internal Relays*	M0000-M7679 M0000-M7664 M0000-M7648	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Special Int. Relays*	M8000-M8511 M8000-M8496 M8000-M8480	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Step Relays*	S0000-S4095 S0000-S4080 S0000-S4064	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Timer Contacts*	TS000-TS511 TS000-TS496 TS000-TS480	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write
Counter Contacts*	CS000-CS255 CS000-CS240 CS000-CS224	**Boolean** Short, Word, BCD Long, DWord, LBCD	Read/Write

*Users can specify a Long data type by appending a space and an "L" to the address. For example, "CS0000" would be entered as "CS0000 L". This does not apply to arrays or bit accessed registers.

Note: All Boolean device types can be accessed as Short, Word, BCD, Long, DWord and LBCD; however, the device must be addressed on a 16-bit boundary.

Device Type	Range	Data Type	Access
Timer Value	TN000-TN511	**Short**, Word, BCD	Read/Write
Counter Value***	CN000-CN199 CN200-CN255	Short, **Word**, BCD Long, **DWord**, LBCD	Read/Write
Data Registers***	D0000-D7999 D0000-D7998 D0000-D7996	**Short**, Word, BCD Long, DWord, LBCD, Float, Date Double	Read/Write
Data Register Bit Access	D0000.00-D7999.15* D0000.00-D7998.31*	**Short**, Word, BCD, Boolean Long, DWord, LBCD	Read/Write
Data Registers String Access HiLo Byte Ordering	DSH0000.002-DSH7998.002 DSH0000.128-DSH7935.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write

Device Type	Range	Data Type	Access
Data Registers String Access LoHi Byte Ordering	DSL0000.002-DSL7998.002 DSL0000.128-DSL7935.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write
Special Data Registers***	D8000-D8511 D8000-D8510 D8000-D8508	**Short**, Word, BCD Long, DWord, LBCD, Float, Date Double	Read/Write
Special Data Register Bit Access	D8000.00-D8511.15* D8000.00-D8510.31*	**Short**, Word, BCD, Boolean** Long, DWord, LBCD	Read/Write
File Register***	R00000-R32767 R00000-R32766 R00000-R32764	**Short**, Word, BCD Long, DWord, LBCD, Float, Date Double	Read/Write
File Register Bit Access	R00000.00-R32767.15* R00000.00-R32766.31*	**Short**, Word, BCD, Boolean** Long, DWord, LBCD	Read/Write
File Registers String Access HiLo Byte Ordering	RSH00000.002-RSH32766.002 RSH00000.128-RSH32703.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write
File Registers String Access LoHi Byte Ordering	RSL00000.002-RSL32766.002 RSL00000.128-RSL32703.128 The string length may also be specified using a colon. The string length must be between 2-128 bytes and even.	**String**	Read/Write

*For register memory, the data types Short, Word, BCD, DWord, Long, LBCD and Boolean append an optional ".bb" (dot bit) or ":bb" (colon bit) to the address in order to reference a bit in a particular value. The valid ranges for the optional bit are 0-15 for Short, Word, BCD and Boolean; and 0-31 for Long, DWord and LBCD. Strings use the bit number to specify length. The valid length of a string is 2 to 128 bytes. The string length must be an even number. Float types do not support bit operations. The bit number is always in decimal notation.
**When accessing register memory as Boolean, a bit number is required.
***Users can specify a Long data type by appending a space and an "L" to the address. For example, "CS0000" would be entered as "CS0000 L". This does not apply to arrays or bit accessed registers.

Array Access

All device types can be accessed as arrays. The default array tag for all device types is Word, excepting CN200-255 (which is DWord). The size of the array depends on both the data type and the device type. All Register device types can access up to the following: 64 elements for Short, Word and BCD; 32 elements for

Long, DWord, LBCD and Float; and 16 elements for Double. All Bit memory types can access up to the following: 32 elements for Short, Word and BCD; and 16 elements for Long, DWord and LBCD. Arrays may be 1 or 2 dimensions, but the array size may not exceed the limits stated above.

Notes:

1. An array is created when array notation is appended onto a normal device reference.
2. Due to a limit of the protocol, the largest bit memory array that can be written to is 10 Words/Shorts/BCDs (or 5 DWords/Longs/LBCDs). Although this limit differs from the largest bit memory array that can be read (32 words), the maximum Read/Write array size for register memory type is the same (64 words).

Examples

1. D100 [4] Single dimension includes the following register addresses: D100, D101, D102, D103.
2. M016 [3][4] Two Dimensions includes the following device addresses as words: M016, M032, M048, M064, M080, M096, M112, M128, M144, M160, M176, M192 3 rowsx4 columns=12 words 12 x 16 (word) = 192 total bits.

Additional Device Examples

1. Access M device memory as Long: M???? where the ???? is a decimal number on 16-bit boundaries such as 0, 16, 32, 48, and so forth.
2. Access Y device memory as Short: Y??? where the ??? is an Octal number on 16-bit boundaries such as 020, 040, 060, and so forth.

Event Log Messages

The following information concerns messages posted to the Event Log pane in the main user interface. Consult the server help on filtering and sorting the Event Log detail view. Server help contains many common messages, so should also be searched. Generally, the type of message (informational, warning) and troubleshooting information is provided whenever possible.

Unable to read from address block on device. The device reported an invalid address or an error. | Address block = '<address>' to '<address>'.

Error Type:
Error

Possible Cause:

1. An attempt has been made to read a non-existent location in the specified device.
2. An attempt has been made to read from an address in a device that is not located on the specified network node.

Possible Solution:

1. Verify or correct the tags assigned to addresses in the specified range on the device. Eliminate or update any that reference invalid locations.
2. Verify that the node ID referenced in the device address is correct.

Unable to read from device. The device returned a PC number error.

Error Type:
Error

Possible Cause:
The PC number entered for the device ID is invalid. This may occur if the MelsecNet station is not available.

Possible Solution:

1. If attempting to communicate with a PC located on MelsecNet, verify the PC number of the target PC.
2. If intending to communicate directly with the local PC with the Ethernet connection, specify a PC number of 255.

Note:
All tag reads fail until the PC number is corrected.

Unable to write to address on device. The device returned a PC number error. | Address = '<address>'.

Error Type:
Error

Possible Cause:
The PC number entered for the device ID is invalid. This may occur if the MelsecNet station is not available.

Possible Solution:
1. If attempting to communicate with a PC located on MelsecNet, verify the PC number of the target PC.
2. If intending to communicate directly with the local PC with the Ethernet connection, specify a PC number of 255.

Unable to write to address on device. The device reported an invalid address or an error. | Address = '<address>'.

Error Type:
Error

Possible Cause:
1. An attempt has been made to write a non-existent location in the specified device.
2. An attempt has been made to write from an address in a device that is not located on the specified network node.

Possible Solution:
1. Verify or correct the tags assigned to addresses in the specified range on the device. Eliminate or update any that reference invalid locations.
2. Verify that the node ID referenced in the device address is correct.

Unable to read from address on device. The device reported an invalid address or an error. | Address = '<address>'.

Error Type:
Error

Possible Cause:
1. An attempt has been made to read a non-existent location in the specified device.
2. An attempt has been made to read from an address in a device that is not located on the specified network node.

Possible Solution:
1. Verify or correct the tags assigned to addresses in the specified range on the device. Eliminate or update any that reference invalid locations.
2. Verify that the node ID referenced in the device address is correct.

Unable to read from address on device. Device returned an error. | Address = '<address>', Error code = <code>.

Error Type:

Error

Possible Cause:
Communication with the device succeeded, but the device reported a problem.

Possible Solution:
Consult the device documentation for information about the error code provided.

Unable to read from address block on device. Device returned an error. | Address block = '<address>' to '<address>', Error code = <code>.

Error Type:
Error

Possible Cause:
Communication with the device succeeded, but the device reported a problem.

Possible Solution:
Consult the device documentation for information about the error code provided.

Unable to write to address on device. Device returned error. | Address = '<address>', Error code = <code>.

Error Type:
Error

Possible Cause:
Communication with the device succeeded, but the device reported a problem.

Possible Solution:
Consult the device documentation for information about the error code provided.

Unable to read from address block on device. | Address block = '<address>' to '<address>'.

Error Type:
Error

Possible Cause:
The driver could not allocate the resources required to read from the device.

Possible Solution:
Shut down unnecessary applications and try again.

Unable to read from address on device. | Address = '<address>'.

Error Type:
Error

Possible Cause:
The driver could not allocate the resources required to read from the device.

Possible Solution:
Shut down unnecessary applications and try again.

Unable to write to address on device. Device must be configured to allow writes while in RUN mode. | Address = '<address>'.

Error Type:
Warning

Possible Cause:
The device is not configured to allow transactions during RUN mode.

Possible Solution:
1. For A-Series and QnA-Series PLCs, configure the AJ71E71 card to allow writes to occur during RUN by setting DIP switch 7 to the ON position.
2. For Q-Series PLCs, use GX Developer to enable the setting "Enable Write at RUN time" in Ethernet Operations settings.

See Also:
1. A-Series PLC Setup
2. QnA-Series PLC Setup
3. Q-Series PLC Setup

Failed to synchronize time and date for device. | Retry interval = <number> (minutes).

Error Type:
Warning

Possible Cause:
The driver failed to write time and date data to the PLC.

Possible Solution:
1. Verify the cabling between the PC and the PLC device.
2. Verify that the specified communications parameters match those of the device.
3. Verify that the network ID given to the named device matches that of the actual device.

Note:
The driver automatically retries after the indicated time interval.

Appendix: PLC Setup

The hardware must be configured for Ethernet communications. For information on a specific hardware series, select a link from the list below.

The following is provided for convenience only. Refer to the manufacturer's documentation for current and official instructions.

A Series PLC Setup

Note: The following is provided for convenience only. Refer to the manufacturer's documentation for current and official instructions.

Hardware Settings

The DIP switches on the AJ71E71 Ethernet interface card must be set as follows.

- DIP switches 1-6 must be set to OFF.
- DIP switch 7 must be set to ON.
- DIP switch 8 must be set to OFF.

Ladder Program

The Mitsubishi A Series PLC requires that a ladder program be used to initialize the AJ71E71 or A1SJ71E71 Ethernet interface card and define the desired open system. TCP/IP and UDP open systems may be used with this driver. In the case of TCP/IP, error handling code should also be implemented.

Note: TCP/IP is less efficient than UDP and requires special ladder to handle network error recovery. Also, if planning to communicate with devices on a remote network, TCP/IP requires that multiple ports be configured in the relay device. Thus, UDP is recommended wherever possible. For more information, refer to Multi-level Networks.

Initialization Ladder

The following initialization code sets the IP address of the device and triggers execution of the open code. For this example, an IP of 192.168.111.123 (C0.A8.6F.7B Hex) is assumed.

```
| M9038                                 H             |
+-| |--+--------------------------[DMOV C0A86F7B D100]|
|      |                              H    K        K |
+      +--------------------------[TO 0000 0  D100 2 ]|
|      |                                              |
+      +------------------------------------[SET M40 ]|
| M40                                                 |
+-| |------------------------------------------<Y0019>|
| X0019  Y0019                                        |
+-| |----| |--------------------------------[PLS M41 ]|
| M41                                                 |
+-| |---------------------------------------[SET M42 ]|
|                                                     |
```

Open and Error Handling Ladder for TCP/IP

The following open and error handling code assumes TCP/IP communications, unpassive mode, on port 5001 (1389 Hex).

This code is for the first communications buffer of the AJ71E71 card. Similar code must be implemented for each additional buffer needed. Simply ensure that the proper interface bits are used as well as separate error handling bits and timers for each buffer.

Note: It is strongly recommended that users follow the code fragment as closely as possible. Without proper error handling and recovery on the PLC side of the connection, communications with the PLC may not be reestablished after a physical error, such as a cable break, occurs. Without the error handling represented here, PLC might have to be reset in order to reestablish communications.

```
| M42   X0010  Y0008                  H    K  H    K  |
+-| |---|/|----|/|-----+----------[TO 0000 16 8002 1 ]|
|                      |               H    K  H    K  |
+                      +----------[TO 0000 24 1389 1 ]|
|                      |                              |
+                      +-------------------[SET Y0008]|
| X0010                                               |
+-| |---------------------------------------[PLF M50 ]|
| M50                                                 |
+-| |--+-----------------------------------[RST Y0008]|
|      |                                              |
+      +------------------------------------[RST M42 ]|
|      |                                              |
+      +------------------------------------[SET M51 ]|
| M51                                             K20 |
+-| |--------------------------------------------<T0 >|
| T0                                                  |
+-| |--+------------------------------------[RST M51 ]|
|      |                                              |
+      +------------------------------------[SET M42 ]|
```

Given the ladder fragment shown here for TCP/IP port operation, the AJ71E71 will be forced to close and re-enable the port for a connection if the current connection is lost. This will occur 2 seconds after the error is detected as controlled by T0. Reloading the port mode and port number and the set of Y008 resets the port.

Open Ladder for UDP

The following open code assumes UDP communications on port 5000 (1388 Hex). The UDP open system requires that the destination address be specified. This would be the IP and port that the driver will use to communicate with the PLC. To prevent issues with conflicting port usage, the Mitsubishi Ethernet Driver allows Windows to assign any unused UDP port to each device configured in the driver on startup. Thus, the port that the driver will use is not predictable. Therefore, the destination port must be configured in the PLC as "unspecified". This is done by entering FFFF (Hex) as shown below. The exact IP address that the driver will use may be specified. This example assumes 192.168.111.24 (C0.A8.6F.18 Hex). However, the destination may also be left as "unspecified" with 255.255.255.255 (FF.FF.FF.FF Hex).

Note: If a specific IP address is put into the ladder code, only the machine with that IP address will be able to communicate with the PLC via UDP. If the IP address is left as "unspecified," then any IP address can communicate with the PLC.

```
| M42   X0010  Y0008                   H    K  H    K  |
+-|  |---|/|----|/|-----+----------[TO 0000 16 110  1 ]|
|                       |              H    K  H    K  |
+                       +----------[TO 0000 24 1388 1 ]|
|                       |              H    K  H    K  |
+                       +----------[TO 0000 25 6F18 1 ]|
|                       |              H    K  H    K  |
+                       +----------[TO 0000 26 C0A8 1 ]|
|                       |              H    K  H    K  |
+                       +----------[TO 0000 27 FFFF 1 ]|
|                       |                              |
+                       +------------------[SET Y0008]|
```

QnA Series PLC Setup

The following is provided for convenience only. Refer to the manufacturer's documentation for current and official instructions.

Hardware Settings

The DIP switches on the A1SJ71QE71 Ethernet interface card must be set as follows:

- DIP switches 1-2 must be set to OFF.
- DIP switch 3 must be set to ON.
- DIP switches 4-6 must be set to OFF.
- DIP switch 7 must be set to ON.
- DIP switch 8 must be set to OFF.

Ladder Program

The Mitsubishi QnA Series PLC requires that a ladder program be used to initialize the AJ71QE71 or A1SJ71QE71 Ethernet interface card and define the desired open system. TCP/IP and UDP open systems may be used with this driver. In the case of TCP/IP, error handling code should also be implemented. Note that TCP/IP is less efficient than UDP and requires a special ladder to handle network error recovery. Also, if planning to communicate with devices on a remote network, TCP/IP requires that multiple ports be configured in the relay device. Thus, UDP is recommended wherever possible. For more information, refer to Multi-level Networks.

Note: Power must be cycled to the PLC in order for any network configuration to take effect.

Initialization Ladder

The following initialization code sets the IP address of the device and triggers execution of the open code. For this example, an IP of 192.168.111.123 (C0.A8.6F.7B Hex) is assumed.

```
| SM1038                              H             |
+-| |--+--------------------------[DMOV C0A86F7B D100]|
|      |                            H     K       K  |
+      +--------------------------[TO 0000 0   D100 2 ]|
|      |                                             |
+      +------------------------------------[SET M40 ]|
| M40                                                |
+-| |-------------------------------------------<Y0019>|
| X0019   Y0019                                      |
+-| |----| |----------------------------------[PLS M41 ]|
| M41                                                |
+-| |---------------------------------------[SET M42 ]|
|                                                    |
```

Open and Error Handling Ladder for TCP/IP

The following open and error handling code assumes TCP/IP communications, unpassive mode, on port 5001 (1389 Hex).

This code is for the first communications buffer of the A1SJ71QE71 card. Similar code must be implemented for each addition buffer needed. Simply ensure that the proper interface bits are used as well as separate error handling bits and timers for each buffer.

Note: It is strongly recommended that users follow the code fragment as closely as possible. Without proper error handling and recovery on the PLC side of the connection, communications may not able to be reestablished with the PLC after a physical error, such as a cable break, occurs. Without the error handling represented here, the PLC might need to be reset in order to reestablish communications.

```
| M42    X0010   Y0008                H     K  H     K  |
+-| |---|/|----|/|-----+----------[TO 0000 32 8000 1 ]|
|                      |            H     K  H     K  |
+                      +----------[TO 0000 40 1389 1 ]|
|                      |                              |
+                      +------------------[SET Y0008]|
| X0010                                              |
+-| |---------------------------------------[PLF M50 ]|
| M50                                                |
+-| |--+------------------------------------[RST Y0008]|
|      |                                             |
+      +------------------------------------[RST M42 ]|
|      |                                             |
+      +------------------------------------[SET M51 ]|
| M51                                          K20  |
+-| |-------------------------------------------<T0 >|
| T0                                                 |
+-| |--+------------------------------------[RST M51 ]|
|      |                                             |
+      +------------------------------------[SET M42 ]|
```

Given the ladder fragment shown here for TCP/IP port operation, the A1SJ71QE71 will be forced to close and re-enable the port for a connection if the current connection is lost. This will occur 2 seconds after the error is detected as controlled by T0. Reloading the port mode and port number and the set of Y008 resets the port.

Open Ladder for UDP

The following open code assumes UDP communications on port 5000 (1388 Hex). The UDP open system requires that the destination address be specified. This would be the IP and port that the driver will use to communicate with the PLC. To prevent issues with conflicting port usage, the Mitsubishi Ethernet Driver allows Windows to assign any unused UDP port to each device configured in the driver on startup. Thus, the port that the driver will use is not predictable. Users must configure the destination port in the PLC as "unspecified". This is done by entering FFFF (Hex) as shown below. The exact IP address the driver will use may be specified. This example assumes 192.168.111.24 (C0.A8.6F.18 Hex). However, The destination may also be left as "unspecified" with 255.255.255.255 (FF.FF.FF.FF Hex).

Note: If a specific IP address is put into the ladder code, only the machine with that IP address will be able to communicate with the PLC via UDP. If the IP address is left as "unspecified," then any IP address can communicate with the PLC.

```
| M42    X0010   Y0008                  H     K  H     K  |
+-|  |---|/|----|/|-----+-----------[TO 0000 32 110   1 ]|
|                       |               H     K  H     K  |
+                       +-----------[TO 0000 40 1388 1 ]|
|                       |               H     K  H     K  |
+                       +-----------[TO 0000 41 6F18 1 ]|
|                       |               H     K  H     K  |
+                       +-----------[TO 0000 42 C0A8 1 ]|
|                       |               H     K  H     K  |
+                       +-----------[TO 0000 43 FFFF 1 ]|
|                       |                                |
+                       +-------------------[SET Y0008]|
```

Q Series PLC Setup

The following is provided for convenience only. Refer to the manufacturer's documentation for current and official instructions.

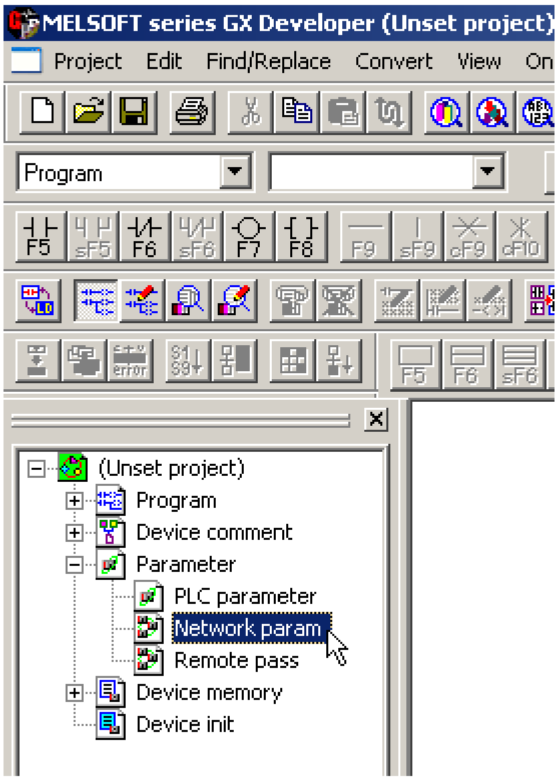
Unlike the A and QnA series, the newest Q Series Ethernet modules (QJ71E71-100) do not have DIP switches that need to be set. Furthermore, special ladder logic to enable Ethernet communications is not required. Users must set network related parameters in the controller, however, using the Mitsubishi GX Developer software. Ports may be configured to use TCP/IP or UDP.

Note: TCP/IP is less efficient than UDP. Users planning to communicate with devices on a remote network should note that TCP/IP requires multiple ports be configured in the relay device. UDP is recommended wherever possible. For more information, refer to **Multi-level Networks**.

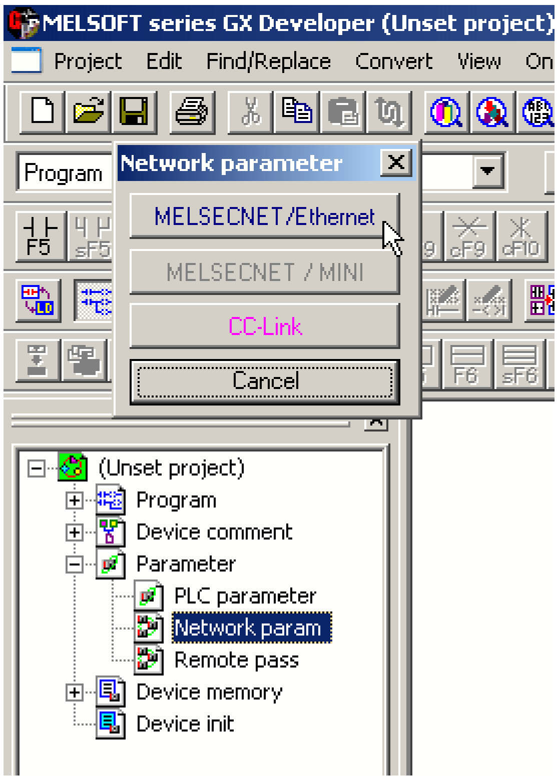
Device Configuration

1. To start, create a new GX Developer project for a Q Series (Q mode) PLC. Alternatively, open and edit an existing project.

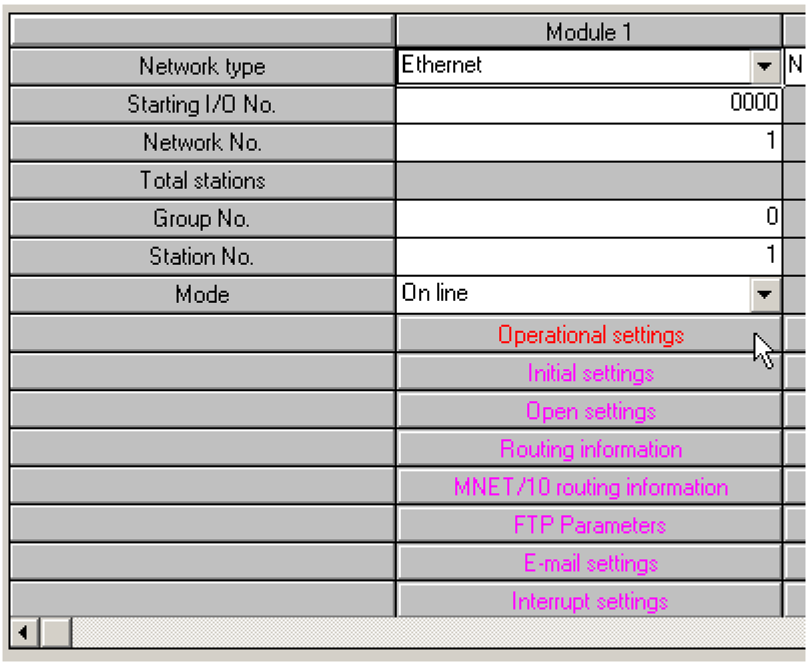
2. Next, select **Network Param**.



3. In **Network Parameter**, click **MELSECNET/Ethernet**.



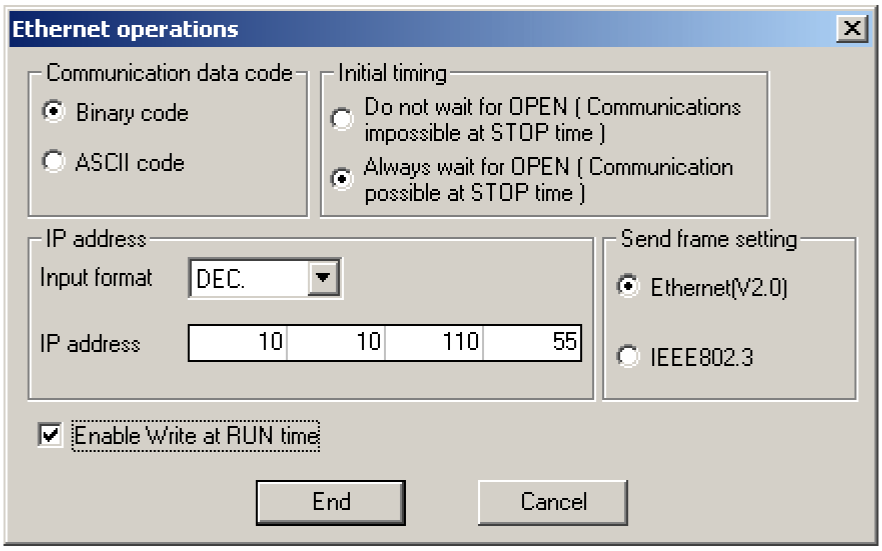
4. Fill in the required information for the Ethernet module. Although the network type must be Ethernet, other settings will depend on the particular application. The example below is for station 1 on network 1. The starting I/O No. is 0 in this case because the QJ71E71 Ethernet module is installed in the slot adjacent to the CPU. If there are other modules between the CPU and Ethernet unit, determine the total I/O mapped to those and set the starting I/O of the Ethernet unit accordingly. Once these basic network settings are specified, click on **Operational Settings**.



	Module 1
Network type	Ethernet
Starting I/O No.	0000
Network No.	1
Total stations	
Group No.	0
Station No.	1
Mode	On line
	Operational settings
	Initial settings
	Open settings
	Routing information
	MNET/10 routing information
	FTP Parameters
	E-mail settings
	Interrupt settings

5. The **Ethernet Operations** dialog is used to define the device's IP address. Except for the IP address, the settings should be as shown below.

Note: Unless security or safety concerns require otherwise, make sure "Enable Write at RUN time" is checked. If this is left unchecked, all writes will fail when the PLC is in Run mode.



Ethernet operations

Communication data code: [✓ Binary code] ASCII code

Initial timing: Do not wait for OPEN (Communications impossible at STOP time) [✓ Always wait for OPEN (Communication possible at STOP time)]

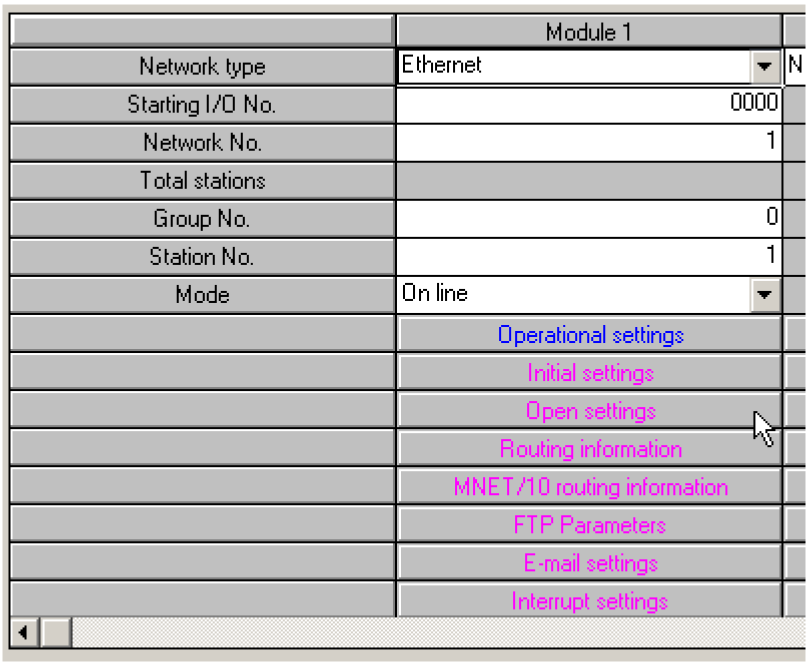
IP address — Input format: DEC.; IP address: 10 10 110 55

Send frame setting: [✓ Ethernet(V2.0)] IEEE802.3

[✓ Enable Write at RUN time]

6. Click **End**.

7. Upon returning to the basic network parameters dialog, click **Open settings**.

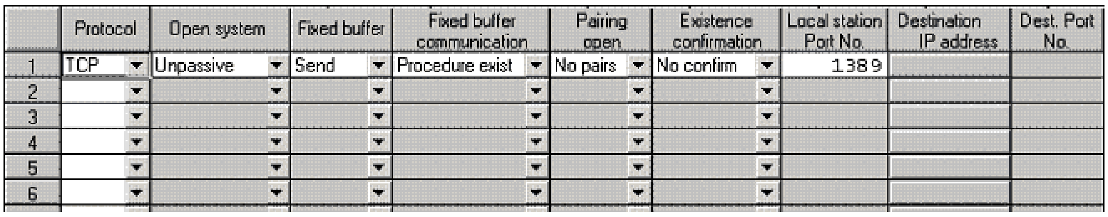


	Module 1
Network type	Ethernet
Starting I/O No.	0000
Network No.	1
Total stations	
Group No.	0
Station No.	1
Mode	On line
	Operational settings
	Initial settings
	Open settings
	Routing information
	MNET/10 routing information
	FTP Parameters
	E-mail settings
	Interrupt settings

8. Specify the desired open settings. These depend on the chosen IP protocol, which may be TCP or UDP.

Open Settings for TCP

Enter **TCP** for the protocol. For simplicity, the **Unpassive** open system is recommended. By using the unpassive open system, users will not have to configure the IP and port that the driver will use. In the example below, the local port number 5001 (1389 Hex) is specified.



	Protocol	Open system	Fixed buffer	Fixed buffer communication	Pairing open	Existence confirmation	Local station Port No.	Destination IP address	Dest. Port No.
1	TCP	Unpassive	Send	Procedure exist	No pairs	No confirm	1389		
2									
3									
4									
5									
6									

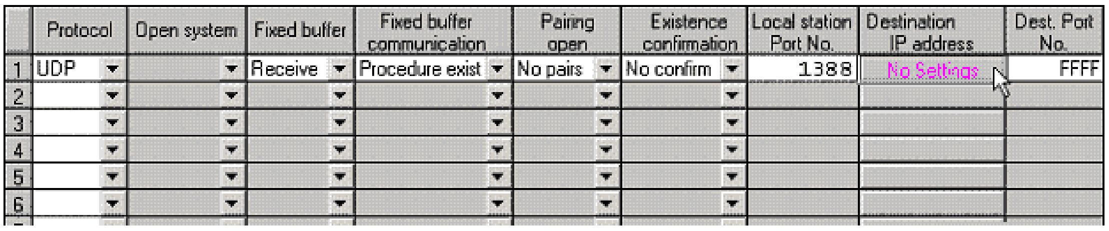
Tip: Consult the Knowledge Base and the Mitsubishi Technical Bulletin "Existence Confirmation Configuration using Fixed Buffer Communications with a QJ71E71-100 Ethernet Module" for detailed information about device configuration for TCP communications.

Open Settings for UDP

1. Enter **UDP** for the protocol. There are no open system options for UDP. In the example below, the local port number 5000 (1388 Hex) is specified.

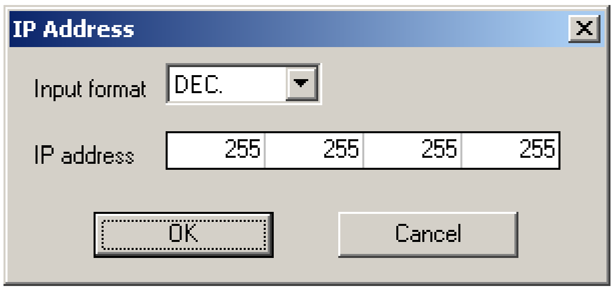
2. Next, specify the destination IP and port. This would be the IP and port that the driver will use to communicate with the PLC. To prevent issues with conflicting port usage, the Mitsubishi Ethernet Driver allows Windows to assign any unused UDP port to each device configured in the driver on startup. Thus, the port that the driver will use is not predictable. Users must configure the destination port in the PLC as "unspecified". This is done by entering FFFF (Hex) as shown below.

3. Finally, click on the Destination IP address button.



	Protocol	Open system	Fixed buffer	Fixed buffer communication	Pairing open	Existence confirmation	Local station Port No.	Destination IP address	Dest. Port No.
1	UDP		Receive	Procedure exist	No pairs	No confirm	1388	No Settings	FFFF
2									
3									
4									
5									
6									

4. Either specify the IP address that the driver will be using or leave it at the "unspecified" address of 255.255.255.255 as shown below.



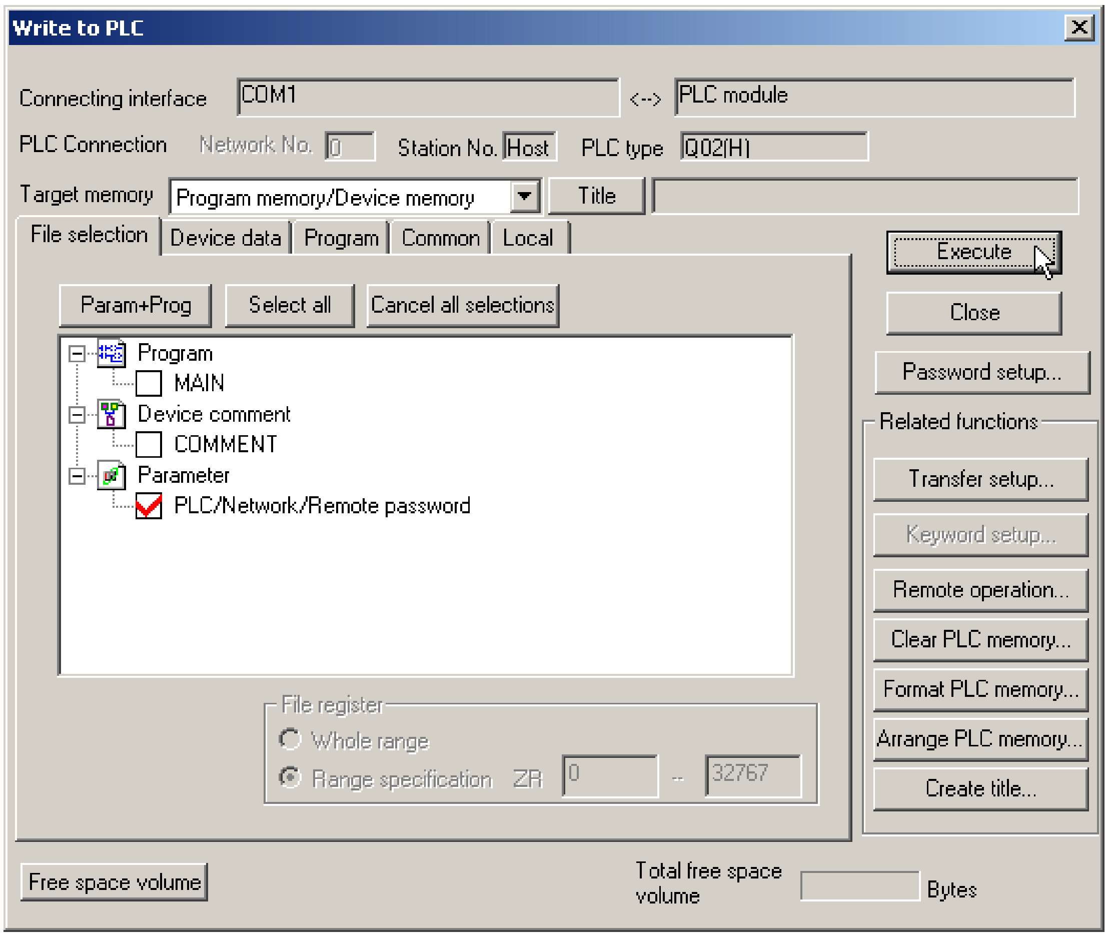
IP Address

Input format: DEC.

IP address: 255 255 255 255

Write Network Parameters to PLC

After all of the network parameters have been specified, they must be written to the PLC. This can be done by selecting the **Online** | **Write To PLC...** menu option. Check the network parameters file selection and then click **Execute**.


Write to PLC

Connecting interface: COM1 <--> PLC module

PLC Connection — Network No. 0 — Station No. Host — PLC type Q02(H)

Target memory: Program memory/Device memory — Title

File selection | Device data | Program | Common | Local

- Program
  - ☐ MAIN
- Device comment
  - ☐ COMMENT
- Parameter
  - ✓ PLC/Network/Remote password

File register: ○ Whole range ● Range specification ZR 0 -- 32767

Total free space volume: Bytes

Note: Users must cycle the power on the PLC in order for the network parameter changes to take effect.

Q Series Built-in Ethernet Port PLC Setup

The following is provided for convenience only. Refer to the manufacturer's documentation for current and official instructions.

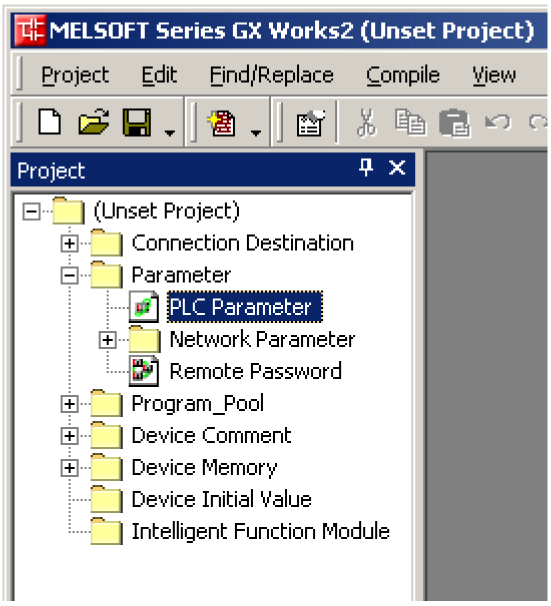
For the Mitsubishi Ethernet Driver to communicate with the Mitsubishi Q Series CPU's built-in Ethernet port, some network parameters must be configured in the PLC.

Device Configuration

The following instructions were created using Mitsubishi GX Works2 software.

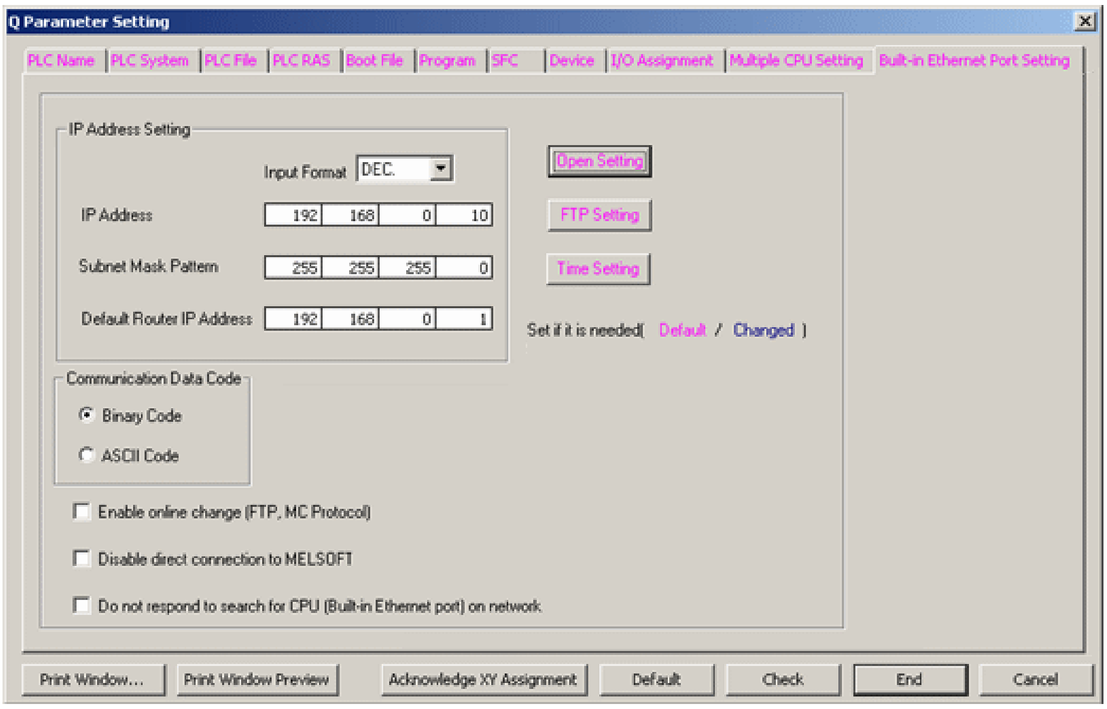
1. To start, create a new project for a Q Series (Q mode) PLC. Alternatively, open and edit an existing project.

2. Next, select **PLC Parameter**.



3. Open the **Built-in Ethernet Port Setting** tab, and then make the following changes:

- Beneath **IP Address Setting**, fill in all required information.
- Beneath **Communication Data Code**, select **Binary Code**.



IP Address Setting

Input Format: DEC.

IP Address	192	168	0	10
Subnet Mask Pattern	255	255	255	0
Default Router IP Address	192	168	0	1

Set if it is needed(Default / Changed)

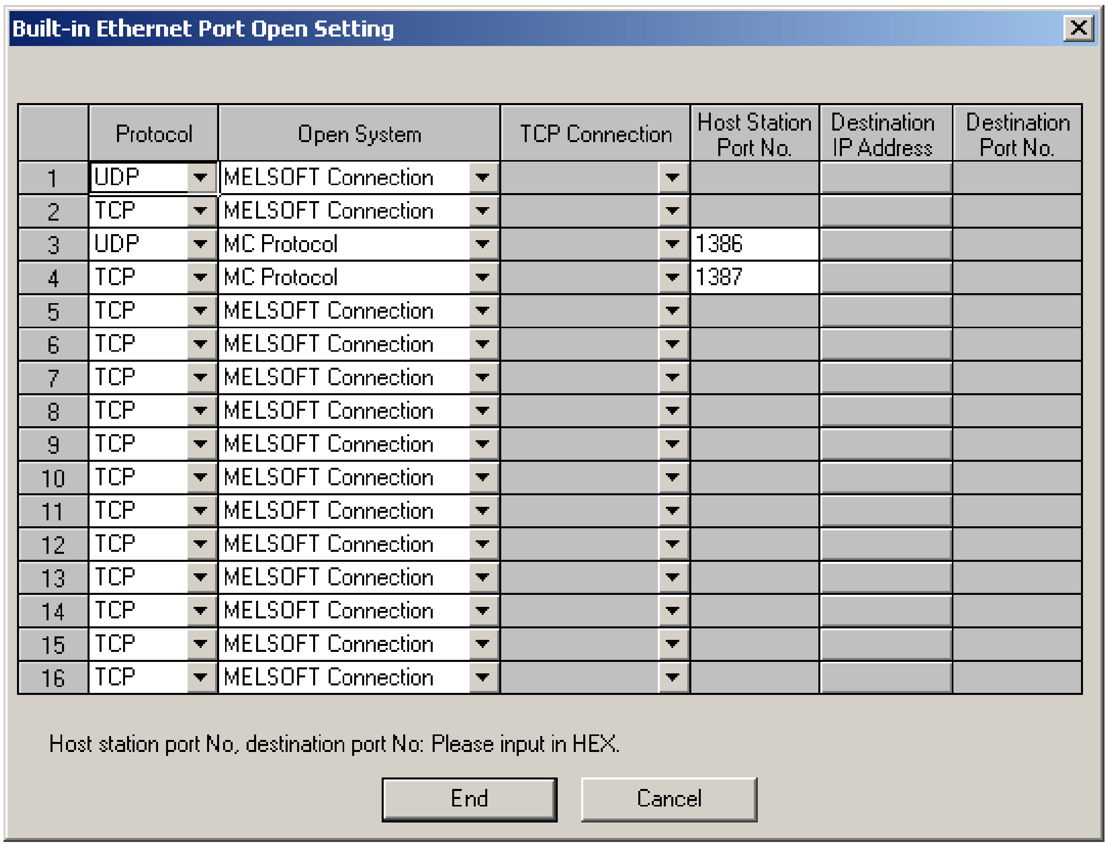
Communication Data Code

(●) Binary Code
( ) ASCII Code

☐ Enable online change (FTP, MC Protocol)
☐ Disable direct connection to MELSOFT
☐ Do not respond to search for CPU (Built-in Ethernet port) on network

4. Next, click **Open Setting**, and then make the following changes:

- Specify the **Protocol**. Options include **UDP** or **TCP**.
- Specify the **Open System** as **MC Protocol**.
- Specify the **Host Station Port No**.



	Protocol	Open System	TCP Connection	Host Station Port No.	Destination IP Address	Destination Port No.
1	UDP	MELSOFT Connection				
2	TCP	MELSOFT Connection				
3	UDP	MC Protocol		1386		
4	TCP	MC Protocol		1387		
5	TCP	MELSOFT Connection				
6	TCP	MELSOFT Connection				
7	TCP	MELSOFT Connection				
8	TCP	MELSOFT Connection				
9	TCP	MELSOFT Connection				
10	TCP	MELSOFT Connection				
11	TCP	MELSOFT Connection				
12	TCP	MELSOFT Connection				
13	TCP	MELSOFT Connection				
14	TCP	MELSOFT Connection				
15	TCP	MELSOFT Connection				
16	TCP	MELSOFT Connection				

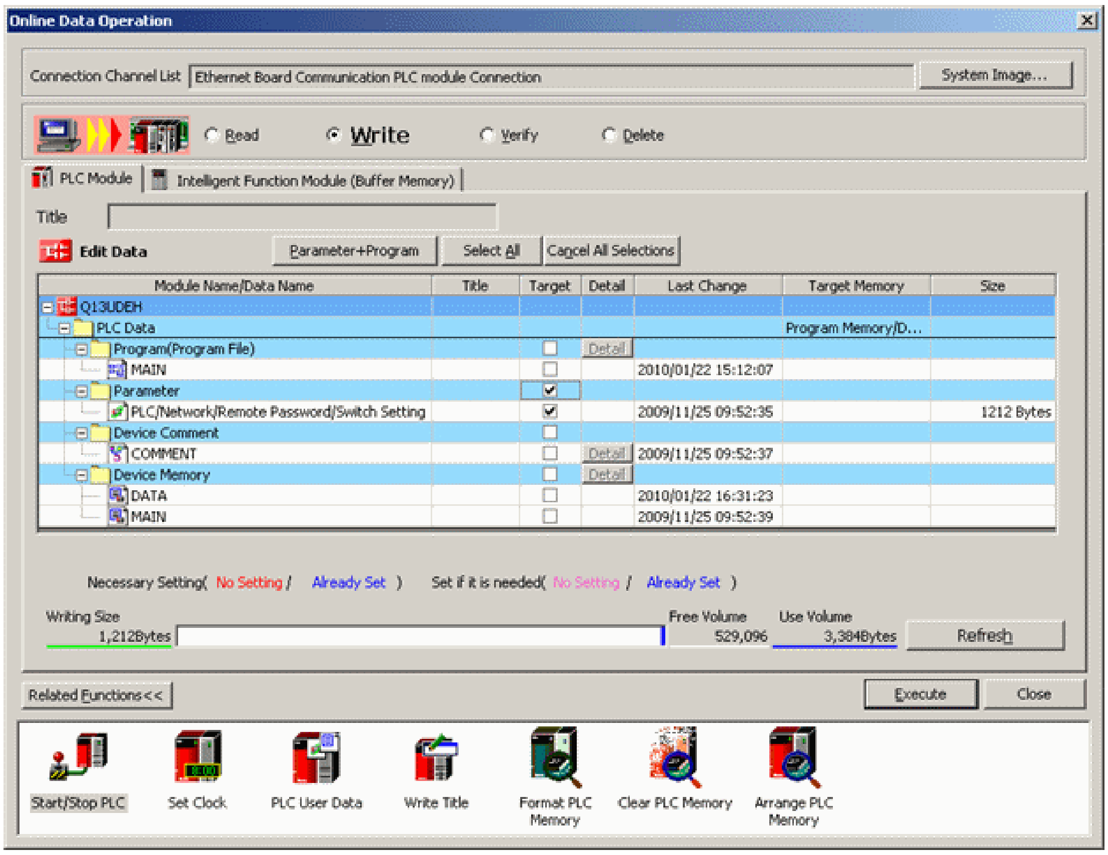
Host station port No, destination port No: Please input in HEX.

Note: In the example above, the local port numbers 4998 (1386H) and 4999 (1387H) are used.
Important: The driver's default port settings of 5000 UDP and 5001 TCP are not valid port numbers for the built-in Ethernet port. The driver uses decimal numbers for the port number; GX Works2 uses hexadecimal number for the port numbers. Valid port number setting ranges are 0401H (1025) to 1387H (4999), and 1392H (5010) to FFFEH (65534).

5. Click **End**.

Writing the Network Parameters to the PLC

After all network parameters have been specified, they must be written to the PLC. To do so, click **Online** | **Write To PLC...**. Then, check **Parameter** (located beneath **Target**) and then click **Execute**.



Online Data Operation

Connection Channel List: Ethernet Board Communication PLC module Connection

○ Read [✓ Write] ○ Verify ○ Delete

PLC Module | Intelligent Function Module (Buffer Memory)

Title:

Edit Data

Module Name/Data Name	Title	Target	Detail	Last Change	Target Memory	Size
Q13UDEH						
PLC Data					Program Memory/D...	
Program(Program File)		☐	Detail			
MAIN		☐		2010/01/22 15:12:07		
Parameter		✓				
PLC/Network/Remote Password/Switch Setting		✓		2009/11/25 09:52:35		1212 Bytes
Device Comment		☐				
COMMENT		☐	Detail	2009/11/25 09:52:37		
Device Memory		☐	Detail			
DATA		☐		2010/01/22 16:31:23		
MAIN		☐		2009/11/25 09:52:39		

Necessary Setting(No Setting / Already Set) Set if it is needed(No Setting / Already Set)

Writing Size: 1,212Bytes

Free Volume: 529,096

Use Volume: 3,384Bytes

Note: Users must cycle the power on the PLC in order for the network parameter changes to take effect.

Index