

# Mitsubishi FX Net Driver

© 2016 PTC Inc. All Rights Reserved.

# Table of Contents

|  |           |
|--|-----------|
| <b>Mitsubishi FX Net Driver</b> .....  | <b>1</b>  |
| <b>Table of Contents</b> .....   | <b>2</b>  |
| Mitsubishi FX Net Driver .....   | 3         |
| Overview .....   | 3         |
| <b>Device Setup</b> .....  | <b>4</b>  |
| Channel Properties .....   | 5         |
| Channel Properties - General .....   | 5         |
| Channel Properties - Serial Communications .....   | 6         |
| Channel Properties - Write Optimizations .....   | 8         |
| Channel Properties - Advanced .....  | 9         |
| Device Properties .....  | 10        |
| Device Properties - General .....  | 10        |
| Device Properties - Scan Mode .....  | 12        |
| Device Properties - Timing .....   | 12        |
| Device Properties - Auto-Demotion .....  | 13        |
| Device Properties - Redundancy .....   | 14        |
| Modem Setup .....  | 14        |
| <b>Data Types Description</b> .....  | <b>15</b> |
| <b>Address Descriptions</b> .....  | <b>16</b> |
| FX Addressing .....  | 16        |
| FX2C Addressing .....  | 16        |
| FX0N Addressing .....  | 17        |
| FX2N Addressing .....  | 17        |
| FX3U Addressing .....  | 18        |
| FXOpen Addressing .....  | 18        |
| <b>Event Log Messages</b> .....  | <b>20</b> |
| Device reported an invalid address in range.   Range = '<address>' to '<address>'. .....   | 20        |
| Received block length does not match expected length.   Received length = <number> (bytes),<br>Expected length = <number> (bytes). ..... | 20        |
| Error code received from device.   Error code = <code>h. ....  | 20        |
| Error Mask Definitions .....   | 20        |
| <b>Appendix: FX3U PLC Parameter Settings</b> .....   | <b>21</b> |
| <b>Index</b> .....   | <b>22</b> |

## Mitsubishi FX Net Driver

---

Help version 1.039

### CONTENTS

#### [Overview](#)

What is the Mitsubishi FX Net Driver?

#### [Device Setup](#)

How do I configure a device for use with this driver?

#### [Data Types Descriptions](#)

What data types does this driver support?

#### [Address Descriptions](#)

How do I address a data location on a Mitsubishi FX series device?

#### [Event Log Messages](#)

What messages does the Mitsubishi FX Net Driver produce?

#### [Appendix: FX3U PLC Parameters](#)

What parameter settings are necessary for an FX3U PLC to communicate with this driver?

### Overview

---

The Mitsubishi FX Net Driver provides a reliable way to connect Mitsubishi FX Net devices to OPC client applications; including HMI, SCADA, Historian, MES, ERP, and countless custom applications. It is intended for use with Mitsubishi FX series devices.

---

## Device Setup

---

### Supported Devices

FX  
FX2C  
FX0N  
FFX2N  
FFX3U

#### ● Notes:

1. FX3U is not supported in Windows CE.
2. FXOpen is a general model that supports the driver's maximum address range. It may be selected for models other than FX, FX2C, FX0N, FX2N, and FX3U. It should not be used if the device is one of the models specifically supported by this driver (such as FX, FX2C, FX0N, FX3U, and FX2N). For example, if the device is FX0N, choose the FX0N model. Selecting FXOpen as the model when the device is FX, FX2C, FX0N, FX2N, or FX3U may result in bad tag reads and incorrect values.

● **See Also:** [FX3U PLC Parameter Settings](#)

### Communication Protocol

Format 1, Checksum

### Supported Communication Parameters

Programmable

### Ethernet Encapsulation

This driver supports Ethernet Encapsulation, which allows the driver to communicate with serial devices attached to an Ethernet network using a Serial-to-Ethernet server. It may be invoked through the Communications tab in Channel Properties. For more information, refer to the OPC server's help documentation.

● **Note:** Ethernet Encapsulation is not supported for the FX3U model.

### Flow Control

When using an RS232/RS485 converter, the type of flow control that is required depends on the needs of the converter. Some converters do not require any flow control whereas others require RTS flow. Consult the converter's documentation to determine its flow requirements. An RS485 converter that provides automatic flow control is recommended.

#### ● Notes:

1. When using the manufacturer's supplied communications cable, it is sometimes necessary to choose a flow control setting of **RTS** or **RTS Always** under the Channel Properties.
2. The FX-485PC-IF RS485 Interface Unit is configured via special memory locations within the FX PLC. Users should consult the FX-485PC-IF manual for memory locations and settings to properly configure the unit (and select matching settings for the Mitsubishi FX Net Driver).

### Cable Connections

A null modem cable is required when connecting the RS-232 port of the computer to the FX-485PC-IF.

### [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.

|                         |                           |         |
|-------------------------|---------------------------|---------|
| Property Groups         | [-] <b>Identification</b> |         |
| General                 | Name                      |         |
| Ethernet Communications | Description               |         |
| Write Optimizations     | Driver                    |         |
| Advanced                | [-] <b>Diagnostics</b>    |         |
|                         | 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 - Serial Communications

Serial communication properties are available to serial drivers and vary depending on the driver, connection type, and options selected. Below is a superset of the possible properties.

Click to jump to one of the sections: [Connection Type](#), [Serial Port Settings](#) or [Ethernet Settings](#), and [Operational Behavior](#).

**Note:** With the server's online full-time operation, these properties can be changed at any time. Utilize the User Manager to restrict access rights to server features, as changes made to these properties can temporarily disrupt communications.

|                              |  |          |
|------------------------------|--|----------|
| Property Groups              | <input type="checkbox"/> <b>Connection Type</b>      |          |
| General                      | Physical Medium                                      | COM Port |
| <b>Serial Communications</b> | Shared   | No       |
| Write Optimizations          | <input type="checkbox"/> <b>Serial Port Settings</b> |          |
| Advanced                     | COM ID   | 6        |
| Communication Serialization  | Baud Rate  | 9600     |
|                              | Data Bits  | 8        |
|                              | Parity   | Even     |
|                              | Stop Bits  | 1        |
|                              | Flow Control   | None     |
|                              | <input type="checkbox"/> <b>Operational Behavior</b> |          |
|                              | Report Comm. Errors                                  | Enable   |
|                              | Close Idle Connection                                | Enable   |
|                              | Idle Time to Close (s)                               | 15       |

## Connection Type

**Physical Medium:** Choose the type of hardware device for data communications. Options include COM Port, None, Modem, and Ethernet Encapsulation. The default is COM Port.

- **None:** Select None to indicate there is no physical connection, which displays the [Operation with no Communications](#) section.
- **COM Port:** Select Com Port to display and configure the [Serial Port Settings](#) section.
- **Modem:** Select Modem if phone lines are used for communications, which are configured in the [Modem Settings](#) section.

- **Ethernet Encap.:** Select if Ethernet Encapsulation is used for communications, which displays the [Ethernet Settings](#) section.
- **Shared:** Verify the connection is correctly identified as sharing the current configuration with another channel. This is a read-only property.

## Serial Port Settings

**COM ID:** Specify the Communications ID to be used when communicating with devices assigned to the channel. The valid range is 1 to 9991 to 16. The default is 1.

**Baud Rate:** Specify the baud rate to be used to configure the selected communications port.


**Data Bits:** Specify the number of data bits per data word. Options include 5, 6, 7, or 8.

**Parity:** Specify the type of parity for the data. Options include Odd, Even, or None.

**Stop Bits:** Specify the number of stop bits per data word. Options include 1 or 2.

**Flow Control:** Select how the RTS and DTR control lines are utilized. Flow control is required to communicate with some serial devices. Options are:

- **None:** This option does not toggle or assert control lines.
- **DTR:** This option asserts the DTR line when the communications port is opened and remains on.
- **RTS:** This option specifies that the RTS line is high if bytes are available for transmission. After all buffered bytes have been sent, the RTS line is low. This is normally used with RS232/RS485 converter hardware.
- **RTS, DTR:** This option is a combination of DTR and RTS.
- **RTS Always:** This option asserts the RTS line when the communication port is opened and remains on.
- **RTS Manual:** This option asserts the RTS line based on the timing properties entered for RTS Line Control. It is only available when the driver supports manual RTS line control (or when the properties are shared and at least one of the channels belongs to a driver that provides this support).  
RTS Manual adds an **RTS Line Control** property with options as follows:
  - **Raise:** This property specifies the amount of time that the RTS line is raised prior to data transmission. The valid range is 0 to 9999 milliseconds. The default is 10 milliseconds.
  - **Drop:** This property specifies the amount of time that the RTS line remains high after data transmission. The valid range is 0 to 9999 milliseconds. The default is 10 milliseconds.
  - **Poll Delay:** This property specifies the amount of time that polling for communications is delayed. The valid range is 0 to 9999. The default is 10 milliseconds.

 **Tip:** When using two-wire RS-485, "echoes" may occur on the communication lines. Since this communication does not support echo suppression, it is recommended that echoes be disabled or a RS-485 converter be used.

## Operational Behavior

- **Report Comm. Errors:** Enable or disable reporting of low-level communications errors. When enabled, low-level errors are posted to the Event Log as they occur. When disabled, these same errors are not posted even though normal request failures are. The default is Enable.

- **Close Idle Connection:** Choose to close the connection when there are no longer any tags being referenced by a client on the channel. The default is Enable.
- **Idle Time to Close:** Specify the amount of time that the server waits once all tags have been removed before closing the COM port. The default is 15 seconds.

## Ethernet Settings

Ethernet Encapsulation provides communication with serial devices connected to terminal servers on the Ethernet network. A terminal server is essentially a virtual serial port that converts TCP/IP messages on the Ethernet network to serial data. Once the message has been converted, users can connect standard devices that support serial communications to the terminal server. The terminal server's serial port must be properly configured to match the requirements of the serial device to which it is attached. *For more information, refer to "How To... Use Ethernet Encapsulation" in the server help.*

- **Network Adapter:** Indicate a network adapter to bind for Ethernet devices in this channel. Choose a network adapter to bind to or allow the OS to select the default.  
• *Specific drivers may display additional Ethernet Encapsulation properties. For more information, refer to Channel Properties - Ethernet Encapsulation.*

## Modem Settings

- **Modem:** Specify the installed modem to be used for communications.
- **Connect Timeout:** Specify the amount of time to wait for connections to be established before failing a read or write. The default is 60 seconds.
- **Modem Properties:** Configure the modem hardware. When clicked, it opens vendor-specific modem properties.
- **Auto-Dial:** Enables the automatic dialing of entries in the Phonebook. The default is Disable. *For more information, refer to "Modem Auto-Dial" in the server help.*
- **Report Comm. Errors:** Enable or disable reporting of low-level communications errors. When enabled, low-level errors are posted to the Event Log as they occur. When disabled, these same errors are not posted even though normal request failures are. The default is Enable.
- **Close Idle Connection:** Choose to close the modem connection when there are no longer any tags being referenced by a client on the channel. The default is Enable.
- **Idle Time to Close:** Specify the amount of time that the server waits once all tags have been removed before closing the modem connection. The default is 15 seconds.

## Operation with no Communications

- **Read Processing:** Select the action to be taken when an explicit device read is requested. Options include Ignore and Fail. Ignore does nothing; Fail provides the client with an update that indicates failure. The default setting is Ignore.

## 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.



|                            |                                |                                      |
|----------------------------|--------------------------------|--------------------------------------|
| Property Groups            | [-] <b>Write Optimizations</b> |                                      |
| General                    | Optimization Method            | Write Only Latest Value for All Tags |
| Ethernet Communications    | Duty Cycle                     | 10                                   |
| <b>Write Optimizations</b> |                                |                                      |

## 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.

|                     |  |                   |
|---------------------|--|-------------------|
| Property Groups     | [-] <b>Non-Normalized Float Handling</b> |                   |
| General             | Floating-Point Values                    | Replace with Zero |
| Write Optimizations | [-] <b>Inter-Device Delay</b>            |                   |
| <b>Advanced</b>     | 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.

● *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.

## Device Properties

The maximum number of channels supported by this driver is 100. The maximum number of devices supported is 16 per channel.

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](#)

[Redundancy](#)

## Device Properties - General

|  |  |                       |  |      |                   |             |  |                    |                   |        |                   |       |    |           |         |    |   |                       |  |                 |        |           |    |
|--|--|-----------------------|--|------|-------------------|-------------|--|--------------------|-------------------|--------|-------------------|-------|----|-----------|---------|----|---|-----------------------|--|-----------------|--------|-----------|----|
| <p>Property Groups</p> <ul style="list-style-type: none"> <li>General</li> <li>Scan Mode</li> <li>Timing</li> <li>Auto-Demotion</li> <li>Redundancy</li> </ul> | <table border="1"> <tr> <td colspan="2"><b>Identification</b></td> </tr> <tr> <td>Name</td> <td>Mitsubishi FX Net</td> </tr> <tr> <td>Description</td> <td></td> </tr> <tr> <td>Channel Assignment</td> <td>Mitsubishi FX Net</td> </tr> <tr> <td>Driver</td> <td>Mitsubishi FX Net</td> </tr> <tr> <td>Model</td> <td>FX</td> </tr> <tr> <td>ID Format</td> <td>Decimal</td> </tr> <tr> <td>ID</td> <td>0</td> </tr> <tr> <td colspan="2"><b>Operating Mode</b></td> </tr> <tr> <td>Data Collection</td> <td>Enable</td> </tr> <tr> <td>Simulated</td> <td>No</td> </tr> </table> | <b>Identification</b> |  | Name | Mitsubishi FX Net | Description |  | Channel Assignment | Mitsubishi FX Net | Driver | Mitsubishi FX Net | Model | FX | ID Format | Decimal | ID | 0 | <b>Operating Mode</b> |  | Data Collection | Enable | Simulated | No |
| <b>Identification</b>  |  |                       |  |      |                   |             |  |                    |                   |        |                   |       |    |           |         |    |   |                       |  |                 |        |           |    |
| Name   | Mitsubishi FX Net  |                       |  |      |                   |             |  |                    |                   |        |                   |       |    |           |         |    |   |                       |  |                 |        |           |    |
| Description  |  |                       |  |      |                   |             |  |                    |                   |        |                   |       |    |           |         |    |   |                       |  |                 |        |           |    |
| Channel Assignment   | Mitsubishi FX Net  |                       |  |      |                   |             |  |                    |                   |        |                   |       |    |           |         |    |   |                       |  |                 |        |           |    |
| Driver   | Mitsubishi FX Net  |                       |  |      |                   |             |  |                    |                   |        |                   |       |    |           |         |    |   |                       |  |                 |        |           |    |
| Model  | FX   |                       |  |      |                   |             |  |                    |                   |        |                   |       |    |           |         |    |   |                       |  |                 |        |           |    |
| ID Format  | Decimal  |                       |  |      |                   |             |  |                    |                   |        |                   |       |    |           |         |    |   |                       |  |                 |        |           |    |
| ID   | 0  |                       |  |      |                   |             |  |                    |                   |        |                   |       |    |           |         |    |   |                       |  |                 |        |           |    |
| <b>Operating Mode</b>  |  |                       |  |      |                   |             |  |                    |                   |        |                   |       |    |           |         |    |   |                       |  |                 |        |           |    |
| 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 FX, FX2C, FX0N, FFX2N, and FFX3U.

**ID Format:** Select how the device identity is formatted. Options include Decimal, Octal, and Hex.

**ID:** the unique device number. The valid range for IDs is 0 to 15.

## 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.

- 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.

|                  |   |                                      |
|------------------|---|--------------------------------------|
| Property Groups  | <input type="checkbox"/> <b>Scan Mode</b> |                                      |
| General          | Scan Mode                                 | Respect Client-Specified Scan Rate ▾ |
| <b>Scan Mode</b> | 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. Factors such as electrically generated noise, modem delays, and poor physical connections can influence how many errors or timeouts a communications driver encounters.

Communications Timeouts properties are specific to each configured device.

|                        |   |      |
|------------------------|---|------|
| Property Groups        | <input checked="" type="checkbox"/> <b>Communication Timeouts</b> |      |
| General                | Connect Timeout (s)   | 3    |
| Scan Mode              | Request Timeout (ms)  | 5000 |
| Ethernet Encapsulation | Retry Attempts  | 3    |
| <b>Timing</b>          | <input checked="" type="checkbox"/> <b>Timing</b>                 |      |
| Auto-Demotion          | Inter-Request Delay (ms)  | 0    |

## Communications Timeouts

**Connect Timeout:** This property (which is used primarily by Ethernet based drivers) controls 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 30 seconds. The default is typically 3 seconds, but can vary depending on the driver's specific nature. If this setting is not supported by the driver, it is disabled.

● **Note:** Due to the nature of UDP connections, the connection timeout setting is not applicable when communicating via UDP.

**Request Timeout:** This property specifies an interval used by all drivers to determine how long the driver waits for a response from the target device to complete. The valid range is 50 to 9,999,999 milliseconds (167.6667 minutes). The default is usually 1000 milliseconds, but can vary depending on the driver. The default timeout for most serial drivers is based on a baud rate of 9600 baud or better. When using a driver at lower baud rates, increase the timeout to compensate for the increased time required to acquire data.

**Retry Attempts:** This property specifies 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 typically 3, but can vary depending on the driver's specific nature. The number of retries configured for an application depends largely on the communications environment.

## Timing

**Inter-Request Delay:** This property specifies how long the driver waits before sending the next request to the target device. It overrides the normal polling frequency of tags associated with the device, as well as one-time reads and writes. This delay can be useful when dealing with devices with slow turnaround times and in cases where network load is a concern. Configuring a delay for a device affects communications with all other devices on the channel. It is recommended that users separate any device that requires an inter-request delay to a separate channel if possible. Other communications properties (such as communication serialization) can extend this delay. The valid range is 0 to 300,000 milliseconds; however, some drivers may limit the maximum value due to a function of their particular design. The default is 0, which indicates no delay between requests with the target device.

● **Note:** Not all drivers support Inter-Request Delay. This setting does not appear if it is not supported by the driver.

## 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.

|                      |                               |         |
|----------------------|-------------------------------|---------|
| Property Groups      | [-] <b>Auto-Demotion</b>      |         |
| General              | Demote on Failure             | Enable  |
| Scan Mode            | Timeouts to Demote            | 3       |
| Timing               | Demotion Period (ms)          | 10000   |
| <b>Auto-Demotion</b> | 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 - Redundancy

|                   |                        |                   |
|-------------------|------------------------|-------------------|
| Property Groups   | [-] <b>Redundancy</b>  |                   |
| General           | Secondary Path         | ...               |
| Scan Mode         | Operating Mode         | Switch On Failure |
| Timing            | Monitor Item           |                   |
| <b>Redundancy</b> | 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.**

## Modem Setup

This driver supports modem functionality. For more information, please refer to the topic "Modem Support" in the OPC server help documentation.

## Data Types Description

The Mitsubishi FX Net Driver supports the following data types.

| Data Type | Description  |
|-----------|--|
| Boolean   | Single bit   |
| Word      | Unsigned 16-bit value<br><br>bit 0 is the low bit<br>bit 15 is the high bit  |
| Short     | Signed 16-bit value<br><br>bit 0 is the low bit<br>bit 14 is the high bit<br>bit 15 is the sign bit  |
| DWord     | Unsigned 32-bit value<br><br>bit 0 is the low bit<br>bit 31 is the high bit  |
| Long      | Signed 32-bit value<br><br>bit 0 is the low bit<br>bit 30 is the high bit<br>bit 31 is the sign bit  |
| Float     | 32-bit floating point value.<br><br>The driver interprets two consecutive registers as a floating point value by making the second register the high word and the first register the low word. |

## 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.

[FX Addressing](#)

[FX2C Addressing](#)

[FX0N Addressing](#)

[FX2N Addressing](#)

[FX3U Addressing](#)

[FXOpen Addressing](#)

## FX Addressing

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

| Device Type              | Range                      | Data Type                               | Access     |
|--------------------------|----------------------------|---|------------|
| Inputs                   | X000-X377*                 | <b>Boolean</b>                          | Read Only  |
| Outputs                  | Y000-Y377*                 | <b>Boolean</b>                          | Read/Write |
| Auxiliary Relays         | M0000-M1535                | <b>Boolean</b>                          | Read/Write |
| Special Aux. Relays      | M8000-M8255                | <b>Boolean</b>                          | Read/Write |
| States                   | S000-S999                  | <b>Boolean</b>                          | Read/Write |
| Timer Contacts           | TS000-TS255                | <b>Boolean</b>                          | Read Only  |
| Counter Contacts         | CS000-CS255                | <b>Boolean</b>                          | Read Only  |
| Timer Value              | T000-T255                  | Short, <b>Word</b>                      | Read/Write |
| Counter Value            | C000-C199                  | Short, <b>Word</b>                      | Read/Write |
| 32 Bit Counter Value**   | C200-C255                  | Long, <b>DWord</b>                      | Read/Write |
| Data Registers**         | D000-D999<br>D000-D998     | <b>Short</b> , Word, Long, DWord, Float | Read/Write |
| Special Data Registers** | D8000-D8255<br>D8000-D8254 | <b>Short</b> , Word, Long, DWord, Float | Read/Write |

\*Octal.

\*\*Users can specify a Long data type by appending a space and an "L" to the address. For example, "D000" would be entered as "D000 L". This does not apply to arrays or bit accessed registers.

## FX2C Addressing

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

| Device Type         | Range       | Data Type          | Access     |
|---------------------|-------------|--------------------|------------|
| Inputs              | X000-X377*  | <b>Boolean</b>     | Read Only  |
| Outputs             | Y000-Y377*  | <b>Boolean</b>     | Read/Write |
| Auxiliary Relays    | M0000-M1535 | <b>Boolean</b>     | Read/Write |
| Special Aux. Relays | M8000-M8255 | <b>Boolean</b>     | Read/Write |
| States              | S000-S999   | <b>Boolean</b>     | Read/Write |
| Timer Contacts      | TS000-TS255 | <b>Boolean</b>     | Read Only  |
| Counter Contacts    | CS000-CS255 | <b>Boolean</b>     | Read Only  |
| Timer Value         | T000-T255   | Short, <b>Word</b> | Read/Write |



| Device Type              | Range                      | Data Type                                 | Access     |
|--------------------------|----------------------------|---|------------|
| Counter Value            | C000-C199                  | Short, <b>Word</b>                        | Read/Write |
| 32 Bit Counter Value**   | C200-C255                  | Long, <b>DWord</b>                        | Read/Write |
| Data Registers**         | D000-D999<br>D000-D998     | <b>Short</b> , Word<br>Long, DWord, Float | Read/Write |
| Special Data Registers** | D8000-D8255<br>D8000-D8254 | <b>Short</b> , Word<br>Long, DWord, Float | Read/Write |

\*Octal.

\*\*Users can specify a Long data type by appending a space and an "L" to the address. For example, "D000" would be entered as "D000 L". This does not apply to arrays or bit accessed registers.

## FX0N Addressing

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

| Device Type              | Range                      | Data Type                                 | Access     |
|--------------------------|----------------------------|---|------------|
| Inputs                   | X000-X177*                 | <b>Boolean</b>                            | Read Only  |
| Outputs                  | Y000-Y177*                 | <b>Boolean</b>                            | Read/Write |
| Auxiliary Relays         | M0000-M0511                | <b>Boolean</b>                            | Read/Write |
| Special Aux. Relays      | M8000-M8255                | <b>Boolean</b>                            | Read/Write |
| States                   | S000-S127                  | <b>Boolean</b>                            | Read/Write |
| Timer Contacts           | TS00-TS63                  | <b>Boolean</b>                            | Read Only  |
| Counter Contacts         | CS00-CS31<br>CS235-CS254   | <b>Boolean</b>                            | Read Only  |
| Timer Value              | T00-T63                    | Short, <b>Word</b>                        | Read/Write |
| Counter Value            | C00-C31                    | Short, <b>Word</b>                        | Read/Write |
| 32 Bit Counter Value**   | C235-C254                  | Long, <b>DWord</b>                        | Read/Write |
| Data Registers**         | D000-D255<br>D000-D254     | <b>Short</b> , Word<br>Long, DWord, Float | Read/Write |
| Special Data Registers** | D8000-D8255<br>D8000-D8254 | <b>Short</b> , Word<br>Long, DWord, Float | Read/Write |

\*Octal.

\*\*Users can specify a Long data type by appending a space and an "L" to the address. For example, "D000" would be entered as "D000 L". This does not apply to arrays or bit accessed registers.

## FX2N Addressing

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

| Device Type         | Range       | Data Type      | Access     |
|---------------------|-------------|----------------|------------|
| Inputs              | X000-X377*  | <b>Boolean</b> | Read Only  |
| Outputs             | Y000-Y377*  | <b>Boolean</b> | Read/Write |
| Auxiliary Relays    | M0000-M3071 | <b>Boolean</b> | Read/Write |
| Special Aux. Relays | M8000-M8255 | <b>Boolean</b> | Read/Write |
| States              | S000-S999   | <b>Boolean</b> | Read/Write |
| Timer Contacts      | TS000-TS255 | <b>Boolean</b> | Read Only  |

| Device Type              | Range                      | Data Type                                 | Access     |
|--------------------------|----------------------------|---|------------|
| Counter Contacts         | CS000-CS255                | <b>Boolean</b>                            | Read Only  |
| Timer Value              | T000-T255                  | Short, <b>Word</b>                        | Read/Write |
| Counter Value            | C000-C199                  | Short, <b>Word</b>                        | Read/Write |
| 32 Bit Counter Value**   | C200-C255                  | Long, <b>DWord</b>                        | Read/Write |
| Data Registers**         | D000-D7999<br>D000-D7998   | <b>Short</b> , Word<br>Long, DWord, Float | Read/Write |
| Special Data Registers** | D8000-D8255<br>D8000-D8254 | <b>Short</b> , Word<br>Long, DWord, Float | Read/Write |

\*Octal.

\*\*Users can specify a Long data type by appending a space and an "L" to the address. For example, "D000" would be entered as "D000 L". This does not apply to arrays or bit accessed registers.

## FX3U Addressing

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

**Note:** The FX3U model is not supported in Windows CE. Also, note that the FX3U model does not support Ethernet Encapsulation.

| Device Type              | Range                      | Data Type                                 | Access     |
|--------------------------|----------------------------|---|------------|
| Inputs                   | X000-X377*                 | <b>Boolean</b>                            | Read Only  |
| Outputs                  | Y000-Y377*                 | <b>Boolean</b>                            | Read/Write |
| Auxiliary Relays         | M0000-M7679                | <b>Boolean</b>                            | Read/Write |
| Special Aux. Relays      | M8000-M8511                | <b>Boolean</b>                            | Read/Write |
| States                   | S0000-S4095                | <b>Boolean</b>                            | Read/Write |
| Timer Contacts           | TS000-TS511                | <b>Boolean</b>                            | Read Only  |
| Counter Contacts         | CS000-CS255                | <b>Boolean</b>                            | Read Only  |
| Timer Value              | T000-T511                  | Short, <b>Word</b>                        | Read/Write |
| Counter Value            | C000-C199                  | Short, <b>Word</b>                        | Read/Write |
| 32 Bit Counter Value**   | C200-C255                  | Long, <b>DWord</b>                        | Read/Write |
| Data Registers**         | D000-D7999<br>D000-D7998   | <b>Short</b> , Word<br>Long, DWord, Float | Read/Write |
| Special Data Registers** | D8000-D8511<br>D8000-D8510 | <b>Short</b> , Word<br>Long, DWord, Float | Read/Write |

\*Octal.

\*\*Users can specify a Long data type by appending a space and an "L" to the address. For example, "D000" would be entered as "D000 L". This does not apply to arrays or bit accessed registers.

## FXOpen Addressing

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

**Note:** When adding a device to the OPC server project, do not select FXOpen if the device is one of the models that is specifically supported by this driver (such as FX, FX2C, FX0N, and FX2N). For example, if the device is FX0N, select the FX0N model. Selecting FXOpen as the model when the device is FX, FX2C, FX0N, or FX2N may result in bad tag reads and incorrect values.

| Device Type              | Range                              | Data Type                                 | Access     |
|--------------------------|------------------------------------|---|------------|
| Inputs                   | X000-X777*                         | <b>Boolean</b>                            | Read Only  |
| Outputs                  | Y000-Y777*                         | <b>Boolean</b>                            | Read/Write |
| Auxiliary Relays         | M0000-M9999***                     | <b>Boolean</b>                            | Read/Write |
| Special Aux. Relays      | M0000-M9999***                     | <b>Boolean</b>                            | Read/Write |
| States                   | S000-S999                          | <b>Boolean</b>                            | Read/Write |
| Timer Contacts           | TS000-TS999                        | <b>Boolean</b>                            | Read Only  |
| Counter Contacts         | CS000-CS999                        | <b>Boolean</b>                            | Read Only  |
| Timer Value              | T000-T999                          | Short, <b>Word</b>                        | Read/Write |
| Counter Value            | C000-C999****                      | Short, <b>Word</b>                        | Read/Write |
| 32 Bit Counter Value**   | C000-C998****                      | Long, DWord                               | Read/Write |
| Data Registers**         | D000-D9999<br>D000-D9998           | <b>Short</b> , Word<br>Long, DWord, Float | Read/Write |
| Special Data Registers** | Please refer to the device manual. | <b>Short</b> , Word<br>Long, DWord, Float | Read/Write |

\*Octal.

\*\*Users can specify a Long data type by appending a space and an "L" to the address. For example, "D000" is entered as "D000 L". This does not apply to arrays or bit accessed registers.

\*\*\*For Auxiliary Relays and Special Aux. Relays, the driver can accommodate the fullest range. For information on the difference between Auxiliary Relays and Special Aux. Relays, refer to the manual.

\*\*\*\*For Counter Value and 32-Bit Counter Value, the driver can accommodate the fullest range. For information on the difference between Counter Value and 32-Bit Counter Value, refer to the manual.

# 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.

---

**Device reported an invalid address in range. | Range = '<address>' to '<address>'.**

---

**Error Type:**

Error

**Possible Cause:**

An attempt has been made to reference a non-existent location in the specified device.

**Possible Solution:**

Verify the tags assigned to addresses in the specified range on the device and eliminate those that reference invalid locations.

---

**Received block length does not match expected length. | Received length = <number> (bytes), Expected length = <number> (bytes).**

---

**Error Type:**

Warning

**Possible Cause:**

The data type maximum length or the length dictated in the address definition set a range that the results do not fit.

**Possible Solution:**

Verify the data type is correct and check the address definition for a length definition and correct or update.

---

**Error code received from device. | Error code = <code>h.**

---

**Error Type:**

Warning

---

**Error Mask Definitions**

---

**B** = Hardware break detected

**F** = Framing error

**E** = I/O error

**O** = Character buffer overrun

**R** = RX buffer overrun

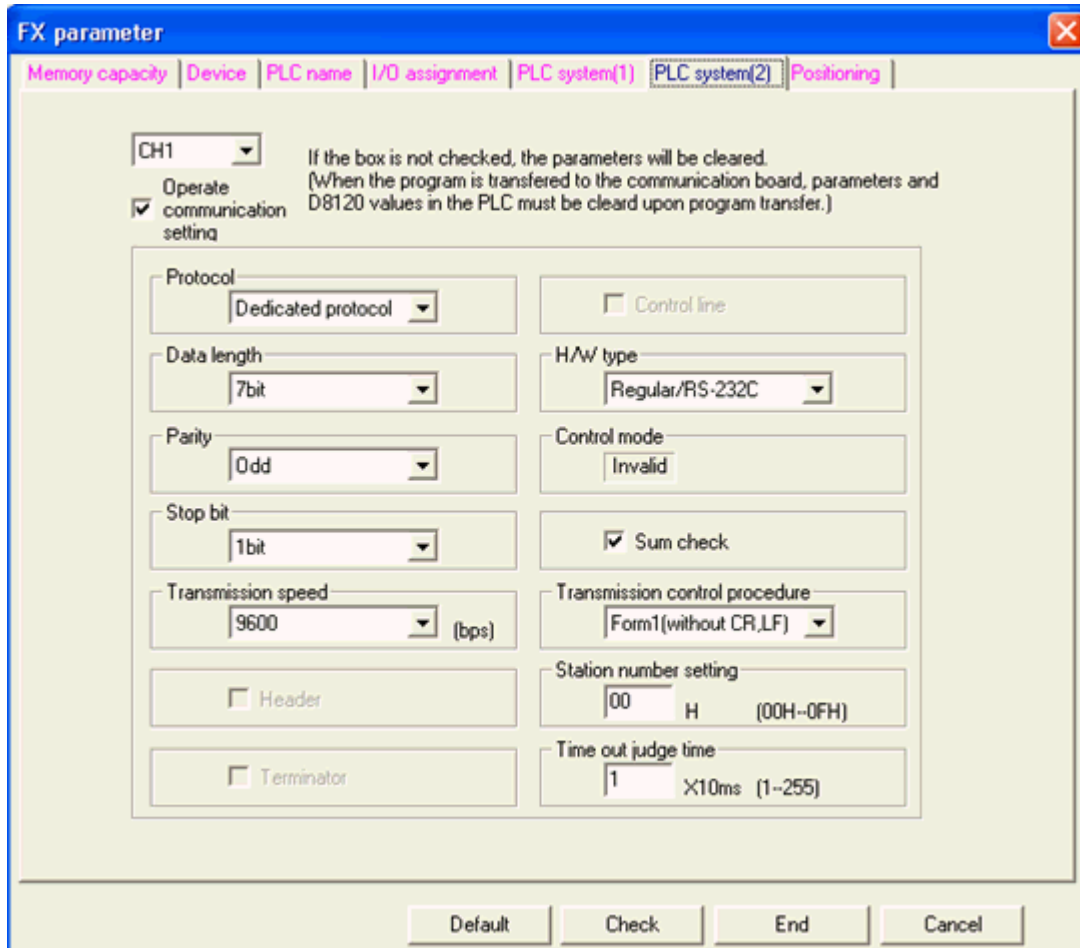
**P** = Received byte parity error

**T** = TX buffer full

## Appendix: FX3U PLC Parameter Settings

The FX3U PLC needs the following parameter settings to communicate with the Mitsubishi FX Net Driver.

1. To start, open the GX Developer-FX software application. Then, click **Parameter | PLC Parameter**.
2. Next, select the **PLC system(2)** tab. Change the settings as needed in order to match those shown in the image below.



3. Next, click **End | Online | Write to PLC**.
4. Click the **Clear PLC** memory button.

**Note:** It is important to clear the PLC memory before writing the new settings to the PLC. If the PLC memory isn't cleared, the new settings will not take effect.

5. To write the settings to the PLC, click **Execute**.
6. Once finished, cycle the power on the PLC so that it will be ready to communicate with the Mitsubishi FX Net Driver.

# Index

## A

Address Descriptions 16  
Advanced Channel Properties 9  
Auto Dial 8

## B

Baud Rate 7  
Boolean 15

## C

Cable Connections 4  
Channel Assignment 11  
Channel Properties 5  
Channel Properties - General 5  
Channel Properties - Write Optimizations 8  
Character buffer overrun 20  
Close Idle Connection 8  
COM ID 7  
Communication Parameters 4  
Communications Timeouts 12-13  
Connect Timeout 13  
Connection Type 6

## D

Data Bits 7  
Data Collection 11  
Data Types Description 15  
Demote on Failure 14  
Demotion Period 14  
Device Properties 10  
Device Properties - Auto-Demotion 13  
Device reported an invalid address in range. | Range = '<address>' to '<address>'. 20

Device Setup 4  
Diagnostics 6  
Discard Requests when Demoted 14  
Do Not Scan, Demand Poll Only 12  
Driver 5, 11  
Duty Cycle 9  
DWord 15

## **E**

Error code received from device. | Error code = <code>h. 20  
Error Mask Definitions 20  
Ethernet Encapsulation 4  
Event Log Messages 20

## **F**

Float 15  
Flow Control 4, 7  
Framing 20  
FX Addressing 16  
FX0N Addressing 17  
FX2C Addressing 16  
FX2N Addressing 17  
FX3U Addressing 18  
FX3U Parameter Settings 21  
FXOpen Addressing 18

## **G**

GX Developer-FX 21

## **H**

Hardware break 20

**I**

I/O error 20  
ID 11  
ID Format 11  
Identification 10  
Idle Time to Close 8  
IEEE-754 floating point 10  
Initial Updates from Cache 12  
Inter-Request Delay 13

**L**

Long 15

**M**

Mitsubishi FX  
    series devices 3  
Model 11  
Modem 8  
Modem Setup 14  
Modem Support 14

**N**

Network Adapter 8  
Non-Normalized Float Handling 10

**O**

Operational Behavior 7  
Optimization Method 9  
Overview 3

**P**

Parity 7, 20



Physical Medium 6

Protocol 4

## R

Read Processing 8

Received block length does not match expected length. | Received length = <number> (bytes), Expected length = <number> (bytes). 20

Redundancy 14

Report Comm. Errors 7-8

Request All Data at Scan Rate 12

Request Data No Faster than Scan Rate 12

Request Timeout 13

Respect Client-Specified Scan Rate 12

Respect Tag-Specified Scan Rate 12

Retry Attempts 13

RTS 4

RX buffer overrun 20

## S

Scan Mode 12

Serial Communications 6

Serial Port Settings 7

Short 15

Signed 15

Simulated 11

Stop Bits 7

Supported 4

## T

Timeouts to Demote 14

TX buffer 20

## U

Unsigned 15

## **W**

Word 15

Write All Values for All Tags 9

Write Only Latest Value for All Tags 9

Write Only Latest Value for Non-Boolean Tags 9

Write Optimizations 9