Mitsubishi MELSEC FX Ethernet Connector

Mitsubishi MELSEC FX controllers over Ethernet, using MC protocol A-compatible 1E frames.

Version 10.1.6+

- Name: **MelsecFX_Ethernet**
- Version: 1.1.0.1
- Protocol: MELSEC Communication (MC) protocol, A-compatible 1E frame, binary
- Interface: TCP/IP
- Runtime: Multiplatform (.NET Standard 2.0)
- Configuration:
  - Devices / Protocols

Overview
Controller Setup
Procedure
Channel Configuration
Node Configuration
Points Configuration
Troubleshooting
Driver Revision History
Related Pages

Overview

The Mitsubishi MELSEC FX Ethernet connector reads and writes the devices of Mitsubishi MELSEC FX controllers through an FX Ethernet module, such as the FX3U-ENET or the FX3U-ENET-ADP. The connector operates as a Master over TCP/IP and sends MC protocol batch read and batch write requests in A-compatible 1E binary frames, the frame type these modules accept. It does not send 3E frames.

The connector groups points into communication blocks by device and by the polling cycle of the AccessType of each point.

For the MelsecFX driver, see Mitsubishi MELSEC FX Connector. For FX controllers on a serial line, see Mitsubishi FX Serial Connector. For general channel, node and point concepts, see Devices Module.

Communication Driver Information	
Driver name	MelsecFX_Ethernet
Assembly Name	T.ProtocolDriver.MelsecFX_Ethernet
Assembly Version	1.1.0.1
Protocol description	Mitsubishi MELSEC FX Ethernet Connector
Manufacturer	Mitsubishi Electric
Frame	MC protocol, A-compatible 1E, binary data code
Interface	TCP/IP
Multiplatform	True

Controller Setup

Configure the Ethernet module of the controller before you start the runtime:

- Open an MC protocol connection on TCP.
- Set the communication data code to **Binary**. The connector does not send ASCII frames.
- Note the port number of the connection. FX3U Ethernet modules accept MC protocol ports 1025 to 5548 and 5552 to 65534.

Procedure

1. In **Devices Channels**, create a channel and select the protocol **Mitsubishi MELSEC FX Ethernet Connector**.
2. In **Devices Nodes**, create a node on the channel and set the Station to the IP address of the Ethernet module and the port of its MC protocol connection. See Node Configuration.
3. In **Devices Points**, add one point per tag. Set Address to a device followed by its number, for example `D100`. See Points Configuration.
4. Start the runtime.

Channel Configuration

Protocol Options

- **BlockSize**: maximum size of a read block. Default: 128. Whatever the value, one request never carries more than 64 registers, 32 counters of 32 bits, or 160 bit devices.
- **Extended**: reserved. Keep the default, False.

Settings

TCP/IP:

- **NodeConnections**: maximum number of parallel requests sent to each node. Default: 1.

Timeout

The connector asks the module to wait for the controller CPU for half of the channel RxStart timeout, between 0.25 and 10 seconds. Keep RxStart at 500 ms or more.

Node Configuration

Station Configuration

Station syntax: **`<IP address>;<Port number>`**

- **IP address**: IP address of the Ethernet module of the controller.
- **Port number**: TCP port of the MC protocol connection configured in the module. The default is 5551. FX3U Ethernet modules do not accept ports 5549 to 5551, so change the default to the port set in the module.

Example: `192.168.1.10;5552`

Points Configuration

Address syntax: **`<Device><Number>`**, with no separator. Examples: `D100`, `M10`, `X17`, `Y20`, `TN0`, `CN200`, `D8000`.

Device	Memory area	Native type	Number format
X	Input	Bit	Octal
Y	Output	Bit	Octal
M	Auxiliary relay. Special relays are M8000 and up.	Bit	Decimal
S	State relay	Bit	Decimal
TS	Timer contact	Bit	Decimal
TN	Timer current value	16-bit register	Decimal
CS	Counter contact	Bit	Decimal
CN	Counter current value	16-bit register for CN0 to CN199; 32-bit counter for CN200 to CN255	Decimal
D	Data register. Special registers are D8000 and up.	16-bit register	Decimal
R	Extension register	16-bit register	Decimal

X and Y numbers are octal, as in the controller program: only the digits 0 to 7 are valid, so `X17` is the input after `X16` and `X18` is rejected. All other devices use decimal numbers.

DataType and BIT Modifier

- **Bit devices** (X, Y, M, S, TS, CS): use DataType Native. No BIT modifier, or BIT=0.
- **Register devices** (D, R, TN, CN): use DataType Native, or a numeric DataType. A DataType wider than the device reads the following registers too; for example `D100` with DataType Integer reads D100 and D101.
- **One bit of a register**: use the BIT modifier with DataType Native or another integer DataType, with the bit number inside the width of the DataType. BIT is not available with BCD or floating-point DataTypes.
- DataType Bit and the string DataTypes are not available on register devices.
- A point cannot span D7999 and D8000, or CN199 and CN200.
- The ByteSwap modifier has no effect on this connector.

A point that does not follow these rules is not used. The runtime writes one error line to the Devices trace for each such point, `Point <address> not used: <reason>`, and the reason names the supported alternative.

Writing

The protocol writes whole devices only. The connector refuses a write that covers only part of a register or of a 32-bit counter, and sends nothing to the controller:

- a point with a BIT modifier on a register device;
- a point with a DataType narrower than the device, for example DataType Byte on `D100` or DataType Short on `CN200`.

Reads of these points keep working. To write the device, use a point with DataType Native and no BIT modifier. A refused write reports error -207 and one error line in the Devices trace.

Read Blocks

A read block covers every device number from its first point to its last. If the points of one device are far apart, the block can include a device number that the controller does not have, and the controller then refuses the whole request. Keep the points of one AccessType inside ranges that exist in the controller, or use separate AccessTypes for distant ranges.

Troubleshooting

Symptom	Check
No reply, or the connection closes after each request	The Station port matches the MC protocol connection of the module; the module data code is Binary; the connection is TCP.
A point is missing at runtime	The Devices trace has a `Point <address> not used` line with the reason.
Protocol error on a read block	The controller refused the request. Verify that every device number in the block range exists in the controller model.
Write error -207	The point writes only part of a device. See Writing.

Driver Revision History

MelsecFX_Ethernet Revision History	
Version	**Notes**
1.1.0.1	The connector now starts at runtime (its driver failed to load before); communication rebuilt on the FX Ethernet 1E frame with correct device codes, octal X/Y addressing and 32-bit counters CN200 to CN255; the operand list now contains only FX devices (D, R, TN, TS, CN, CS, X, Y, M, S), so points using W, B, F, TC or CC are rejected at start-up with the reason in the trace; writes that would overwrite only part of a register are refused instead of corrupting it; errors reported by the module are surfaced for writes as well as reads; the Designer help icon now opens the connector page.
1.1	FrameworX 10.1.6: A-compatible 1E frame; MELSEC FX device set; octal X and Y numbers; 32-bit counters. Initial documentation release.

Related Pages

- Mitsubishi MELSEC FX Connector
- Mitsubishi FX Serial Connector
- Mitsubishi Q Series Connector
- Devices Module
- Devices AccessTypes Reference
- Automation Vendors Ecosystems

In this section...
